Adaptive resource provisioning for multi-tier applications using V-cache in cloud computing

A Dissertation submitted

By

Gurjot Balraj Singh

To

Department of Computer Science and Engineering

In partial fulfillment of the Requirement for the

Award of the Degree of

Master of Technology in CSE

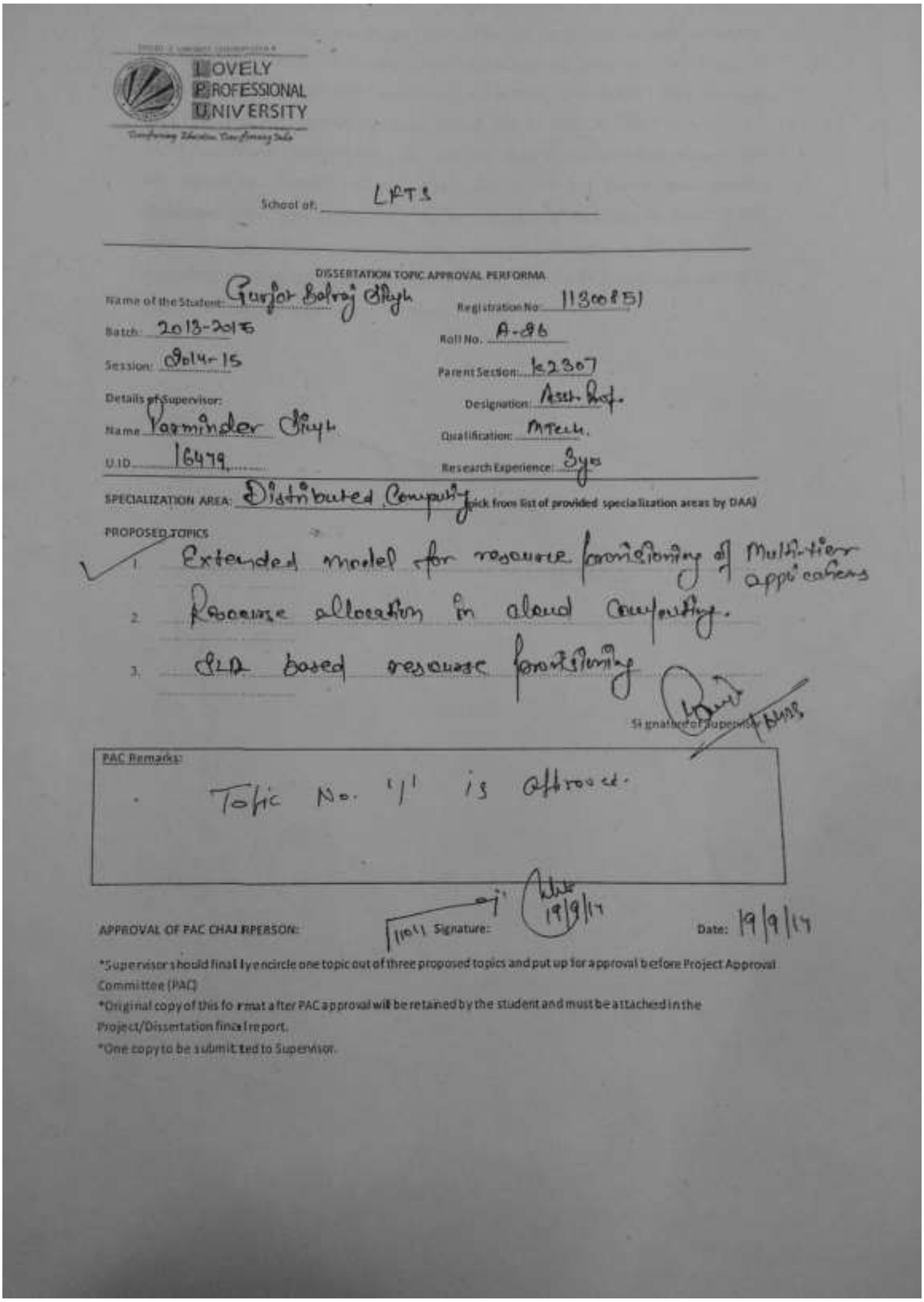
Under the guidance

Of

Mr. Parminder Singh

(May 2015)

PAC FORM



LOVELY PROFESSIONAL UNIVERSITY

School of: LFTS

DISSERTATION TOPIC APPROVAL PERFORMA

Name of the Student: Gurjot Balraj Singh Registration No: 11300851

Batch: 2013-2015 Roll No. A-26

Session: 2014-15 Parent Section: K2307

Details of Supervisor: Designation: Asst. Prof.

Name Parminder Singh Qualification: M.Tech.

U.ID 16479 Research Experience: 3yrs

SPECIALIZATION AREA: Distributed Computing (pick from list of provided specialization areas by DAA)

PROPOSED TOPICS

✓ 1. Extended model for resource provisioning of Multi-tier applications

2. Resource allocation in cloud computing.

3. SLA based resource provisioning

Signature of Supervisor

PAC Remarks:

Topic No. '1' is approved.

APPROVAL OF PAC CHAIRPERSON: Signature: 19/9/14 Date: 19/9/14

*Supervisor should finally encircle one topic out of three proposed topics and put up for approval before Project Approval Committee (PAC)

*Original copy of this fo rmat after PAC approval will be retained by the student and must be attached in the Project/Dissertation final report.

*One copy to be submitted to Supervisor.

ABSTRACT

Web applications these days are mostly multi-tier for the sake of software reusability and flexibility. However it is very difficult to estimate the behavior of workloads in these tiers due to the fact that resource demand at each tier is distinct. So it becomes very confronting to allocate resources at each tier. In order to fulfill SLA (Service Level Agreement) requirements it becomes necessary to provision resources at each tier separately. Recently various approaches on control theory and dynamic approaches have been proposed for the provisioning of resources in cloud. In this paper we deploy adaptive technique using v-cache for multi-tier application in cloud computing so as to enhance the efficiency of provisioning of resources as well as to combat with resource contention problems. We present an extended v-cache model where heterogeneity of the users is considered and cache size is determined dynamically on the basis of SLA signed as well as according to the priority of the user. Also every application has certain peak hours where the workload is maximum. So we are considering those peak load hours to determine the size of cache so as to accommodate maximum requests while the workload is high. Also the size of cache is reduced during off-peak load hours where the workload is minimum. Extended policy generator is presented where decision about size of the cache is made on the basis of current server time and Type of SLA signed. Although this provisioning comes under static resource provisioning but introduction of dynamic cache has made it dynamic.

CERTIFICATION

This is to certify that Gurjot Balraj Singh has prorating M.Tech dissertation "**Adaptive resource provisioning for multi-tier applications using V-cache in cloud computing**" under my guidance and supervision. To the best of my knowledge, the present work is the result of his original investigation and study. No part of the dissertation proposal has ever been submitted for any other degree or diploma. The dissertation is fit for the submission and the partial fulfillment of the conditions for the award of M.Tech Computer Science & Engineering.

Date: Signature of Advisor

ACKNOWLEDGEMENT

The authors are highly grateful to the University, Lovely Professional University (LPU), Phagwara, for providing this opportunity to carry out the present thesis/ work. The constant guidance and encouragement received from **Mr. Parminder Singh** Assistant Prof., LPU has been of great help in carrying out the present work and is acknowledged with reverential thanks. Without the wise counsel and able guidance, it would have been impossible to complete the thesis in this manner. The author expresses gratitude to other faculty members of Computer Engineering Department, LPU for their intellectual support throughout the course of this work. Finally, the authors are indebted to all those whosoever have contributed in this thesis work and friendly stay at Lovely Professional University.

DECLARATION

I hereby declare that the dissertation entitled, *Adaptive resource provisioning for multi-tier applications using V-cache in cloud computing* submitted for the M.Tech degree is entirely my original work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree or diploma.

Date:___________

Investigator

Regn. No.___________

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Cloud Computing

Cloud Computing refers to on demand, Self Service Internet Infrastructure that enables the users to access the resources from anywhere and anytime. Cloud Computing is aimed to implement Scalability and Reliability in the networked environment. In Cloud Computing environment the resources can be allocated according to the needs of the user.

There are 10 basic Principles of Cloud computing namely User centric system, Friendliness to users, Openness, Transparency, Interoperability, Construction of task representation, and Division of task in accordance with their specifics, Evolution, Balance and Security. [5]

Although there are multiple benefits so there exist some security Problems as well.

- **Loss of Control**: - Loss of control occurs because in cloud database a huge voluminous data is being stored whose location cannot be predicted by users.
- **Loss of Trust: -** If the data is being lost then data replacement cannot provide the trust to the user so there is loss of trust.
- **Multi-Tenancy**:- Using single server to serve different client servers may lead to amalgamation of data. [5]

Characteristics of Cloud Computing.

- Clients have instant access to the resources.
- Client can access the services using service discovery from any place.
- Client pays for only those resources which are used by him.
- If the request Processing requires more resources then it is not up to the client to go and search them but it is the system that deals with it.

- Client need not to install Web Services discovery software on his personal computer. It is already deployed on the cloud.

- Reduced cost implementation with the use of Virtualization. [6]

Cloud computing consist of mainly three types of services that are:-

- Saas (Software as a service). It delivers the application as a service. We can hire the application and pay per use rather than owning it.

- Paas (platform as a service). It provides computing platform where we can develop our own application. It consists of support for building and delivering the complete life cycle of the web based applications.

- Iaas (Infrastructure as a service). It delivers the computing resources that can be CPU cycles or Memory or Database capacity. If we are developing the application by using Paas then Iaas is automatically included. [5]

1.2 Cloud delivery models.

Mainly there are three Cloud delivery models namely

1. Public Clouds.

2. Private Clouds.

3. Hybrid Clouds.

1. Public Clouds: -

In public clouds the infrastructure and services are provided basically on-the-shelf over the internet. These types of clouds acquire the greatest level of efficiency in sharing the resources. We can deploy public clouds when:-

- When our application is being used by number of peoples. For ex. Gmail.

- When we need to develop and test application code.

- When we need to access saas from a vendor who has implemented strong security strategies.

- When we are working over collaborating projects.
- When we work under fluctuating workload environments.

2. Private Cloud: -

In a private cloud infrastructure and services are maintained over a private network. It is basically used by an organization but it can also be hosted over other networks. However this type of cloud scenarios have well implemented security and control but you still need to purchase infrastructure and services which further enhances the cost. We can deploy private clouds when:-

- Our business is dedicated to the industry where security and control are the prominent challenges.
- Our organization is competent enough that it can run its own cloud data center effectively and efficiently.

3. Hybrid Cloud: -

Hybrid cloud is a computing environment where an organization manages and provides some in-hour resources and others are provided externally. It uses a mixture of on-premises public cloud and private cloud services with orchestration between the two platforms. It provides the facility by allowing workloads to move between public and private clouds with the change in computation needs and cost. So hybrid cloud gives our business more flexibility and greater data deployment options. For example a company may deploy an on-premises private cloud in order to host critical or sensitive data/workloads but can also use a third party public cloud provider ex. Google compute engine to host less critical data or resources such as testing and development of workloads. Hybrid cloud is best suited for dynamic and highly changeable workloads. For ex. Transaction orders entry system which has high workload over working days and totally contrary during holidays. [8]

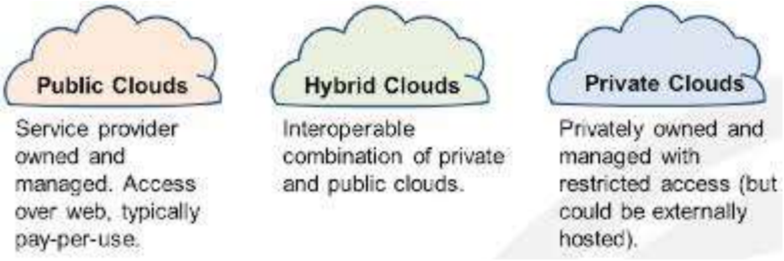

Public Clouds
Service provider owned and managed. Access over web, typically pay-per-use.
Hybrid Clouds
Interoperable combination of private and public clouds.
Private Clouds
Privately owned and managed with restricted access (but could be externally hosted).

Figure 1.1 Cloud delivery models. [8]

1.3 Web Services

Web services are basically web application components that run on web. We can either Publish, Find or Use the web services to include various components that together build a web service.

Basic components of Web services are:-

- **WSDL**

  WSDL stands for web service description language and it is used to describe our web services. It is a XML based language.

- **SOAP**

  SOAP stands for simple object access Protocol and this protocol is used to access the web services. SOAP is also based on XML.

- **UDDI**

  UDDI stands for universal description, discovery and integration and it is basically a directory service where companies can search for the desired web service. UDDI is described in Web Service Description Language (WSDL).

- **RDF**

  RDF stands for Resource description Framework and is used to describe the resources on the web. It is also described in XML.

Now Web Services communicate using open protocols and these are self-contained as well as self describing. We can discover the web services with the help of common directory service UDDI. HTTP and XML are the basis for the web services. [7]

Web services are of two types:-

- **Reusable Application Components: -** Web services can contain Reusable codes for Instance Currency conversion, Language translation or Weather reports.

- **Connect Existing Software: -** Web services can also be used to solve the problem of interoperability by giving different applications a way to link them. [7]

1.4 Applications of Web Services.

We have created an application and we want that other applications can be able to communicate with it. For example if we are creating a Java application for Stock Information and it is updated after every 5 min and we want that other applications can access its data.

There can be many traditional ways to accomplish this but with certain drawbacks:-

- We can serialize our java objects and then send to the application desiring that data. Now problem is that if other application is using C# then it would not be able to read the data.

- Other approach is that we are sending text file to the application that needs data. Now C# application can read this data. If C# application wants to attract with another application other than stock application say weather report application if this application uses its own format then it will take tremendous amount of time for the C# application to work. [3]

So we need a Standard file format i.e. Web Services is a solution. To send a Web service message a WSS standard is predefined.

- Ways to add security headers.
- Attachment of security tokens and credentials to message.
- Inserting a timestamp.
- Signing the message.
- Encrypting.
- Extensibility.

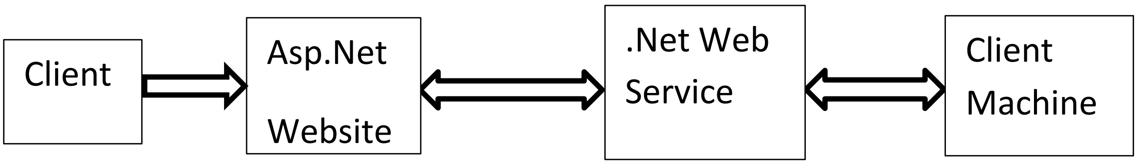


Figure 1.2 Web service chain. [4]

Problems with web services.

- Immaturity of Standards.
- Performance
- Complexity and Interoperability.
- Key Management.
- XML Encryption. [4]

1.5 Emerging Trends in Web Services.

Mash Ups

It is the new breed to Web based applications created by programmers. The mash up mixes two or more Existing different web services from competing and disparate web sites. For ex. A mash up could overlay Traffic from one source and could mash up and display over maps from Google, Yahoo etc. Ex. Wikimap. [2]

1.6 Web Services in Cloud Computing.

Now combining different data centers creates a cloud. For example Google has a cloud of 12 Data centers with about 55,600 Servers. In these data centers Physical Machines are virtualized as instances. Service providers deploy there services on these instances. An end user usually connects to the cloud and accesses the data and runs the applications or services. Then user request is redirected to the particular service instance.

Connection information i.e. RTT (Round Trip Time) between a user and an instance is kept by the cloud provider. [1]

1.7 Provisioning of resources in Multi-tier applications.

As all the tiers have different set of requirements of computing resources so it becomes a challenge to monitor both Quality of service (QOS) and resource cost. To overcome this difficulty generally various QOS requirements are used (i.e. throughput, delay, latency) and are generally stated in SLA. When defined QOS satisfies SLA then service providers get revenue otherwise they have to pay penalty to the users. So ultimately our objective is to provide appropriate resources to the users at each tier so as to reduce the overall cost. According to Jiang et.al. to provide optimal resource provisioning in cloud service providers often impose as a SLA (Service Level Agreement) which defines various parameters that application must offer i.e. Maximum average response time, Latency etc. [2]

Now capacity planning is a classical method to estimate the amount of resource usage. It is determined by maximum request rate to a system in a given target period. We can estimate using either periodic data or historic data. However maximum request rate occurrence is very rare. Figure 1 shows the demonstration of 24 hours of working of system with CPU utilization and disk utilization. We can clearly conclude that CPU usage for most of the time is less than 50% and disk utilization is below 20% for about 70% of the time. [24]

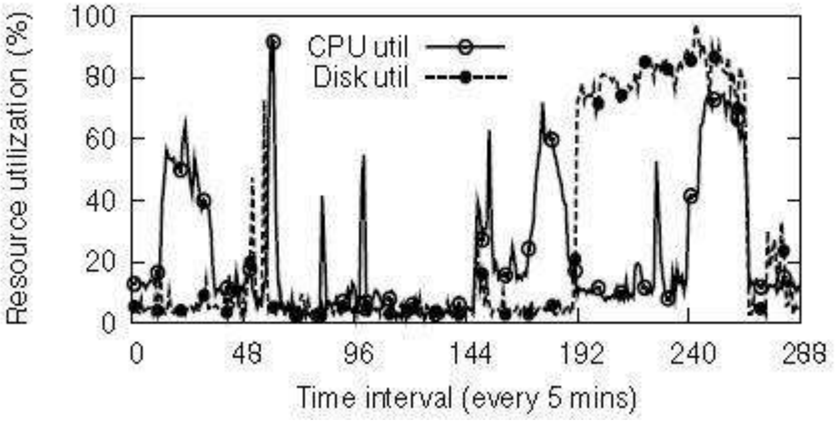


Figure 1.3 Demonstration of 24 hours of working of system. [24]

However with emerging virtualization technology it becomes feasible to allocate the resources according to the fluctuations in resource demand. Particularly cloud paradigm is concerned with “Pay as you go”. For example amazon charges $0.1 per hour per virtual instance [24].

1.8 Challenges in Multi-tier scenario.

Although there are tremendous advantages of cloud computing but it comes with many challenges which are discussed below.

1. First challenge is that each class of resource has different impact on QOS. So it becomes very challenging to predict the allocation of resources to users.
2. Secondly in multi-tier applications each tier has substantially different set of requirements. So it becomes much more difficulty to cope with this problem dynamically.
3. No performance guarantees are given by the service providers with regard to application level performance.

A general scenario of 3 tier application is shown in figure 1.2. Although single tier is easier to implement than multi-tier and has simple architecture but it lacks functionality. Moreover most modern web sites these days use multi-tier architecture. In multi-tier architecture each tier has certain functionality. In the given figure basically three tiers are used namely presentation tier, application tier and data tier and these tiers are implemented as Web server, application server and data server respectively. [24]

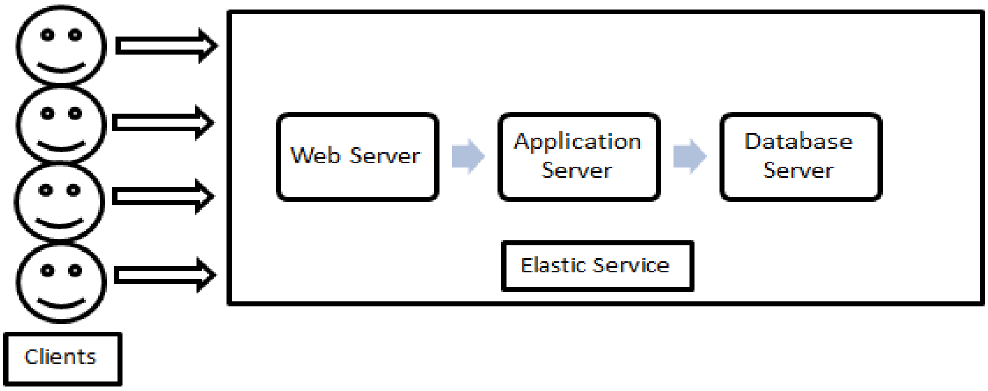


Figure 1.4 General Scenario of multi-tier applications. [3]

Although various techniques are available for the provisioning of resources we will take a look over them one by one. Some techniques have disadvantages as well as advantages but optimal resource provisioning technique is not yet available. It is very difficult to understand the overall good understanding of the behavior of whole system.

1.9 Different types of Resource provisioning.

According to the needs of the application these can be classified as:-

1) Static Provisioning: This type of provisioning is best suited for the applications that are generally predictable and have unchanging demands or workloads; this static provisioning can be applied there constructively. In this scenario of advance provisioning, initially the contract is being signed between the customer and services provider and agreed amount of resources are prepared in advance of the start of service. There are basically two options for customer either flat free charge can be applied or customer can be billed on monthly basis.

2) Dynamic Provisioning: This type of provisioning is best suited where there are fluctuating workloads. Where the demand of resources may vary by the applications dynamic provisioning is best suited there. We basically migrate VMs on-the-fly to compute new nodes within the cloud. In this scenario additional numbers of resources are allocated when they are needed and are removed when the workload decreases. Customer is charged on pay-per-use basis.

3) User Self-provisioning: In user self- provisioning, resources are purchased by the customer from provider via web form by creating an account of the customer and paying for the resources via credit card. The provider provides the resources for use of customer for hours if not minutes. [26]

1.10 Parameters for Resource provisioning.

i) Response time: Response time taken by the designed resource provisioning algorithm should be minimum while executing the task.

ii) Minimize Cost: The cost incurred should also be minimum form the cloud user point of view.

iii) Revenue Maximization: this must be achieved from the Cloud Service Provider's view.

iv) Fault tolerant: Despite of failure of nodes our algorithm should continue to provide services.

v) Reduced SLA Violation: SLA violations should be reduced to minimum by our designed algorithm.

vi) Reduced Power Consumption: VM placement & migration techniques should lower power consumption. [26]

CHAPTER 2

REVIEW OF LITERATURE

S. No	Author Name	Title	Year
1	Swaminathan sivasubramanian et.al.	SLA-driven resource provisioning of multi-tier internet applications	2007
2	Evangelia Kalyvianaki et.al.	Applying Kalman filters to dynamic resource provisioning of virtualized server applications	2008
3	Jiang Dejun et.al.	Autonomous resource provisioning for multi-service web application	2010
4	Sireesha Muppala et.al.	Regression based multi-tier resource provisioning for session slowdown guarantees	2010
5	Yee Ming Chen et.al.	Optimal provisioning of resource in a cloud service	2010
6	Jing Bi et.al.	Dynamic provisioning modeling for virtualized multi-tier application in cloud data center	2010
7	Saurabh kumar garg et.al	SLA-Based resource provisioning for heterogeneous workloads in virtualized cloud data centers	2011
8	Anshul gandhi et.al.	Hybrid resource provisioning for minimizing data center SLA violations and power consumptions	2011
9	Riccardo Lancellotti et.al.	Dynamic Request Management Algorithms for Web-Based Services in Cloud Computing	2011
10	Wesam dewoud et.al.	Dynamic Provisioning of multi-tier applications	2012

11	Rui Han et.al.	Lightweight resource scaling for cloud applications	2012
12	Saouli Hazma et.al.	A Cloud computing approach based on mobile agents for web services discovery	2012
13	Jiang Dejun et.al.	Resource provisioning of web applications in heterogeneous clouds	2013
14	Pooja V et.al.	Mobile Computation Dynamic Offloading Using Cloud	2013
15	Yanfei Guo et.al.	V-cache :- Towards flexible resource provisioning for multi-tier application in Iaas clouds	2013

SLA-driven resource provisioning of multi-tier internet applications.

Swaminathan sivasubramanian et.al. (2007)"SLA-driven resource provisioning of multi-tier internet applications". This paper focuses on end to end performance instead of dealing with individual tier separately. In this approach run time caches are used which further enhances the efficiency and performance. With the use of caches similar requests can be processed faster and efficiently but keeping in mind the consistency problems. This depends on cache hit ratio. Various advantages of this approach is it deals with end to end performance. Moreover SLA of service can be maintained with minimum number of servers. However some disadvantages also persist with this approach that is it is very difficult to maintain consistency of caches. If data in caches it too old then it needs to be updated in order to maintain consistency. Moreover it does not take performance capability of each tier individually as each tier has its own processing requirements. However many algorithms are available to tackle with this problem. [11]

Applying Kalman filters to dynamic resource provisioning of virtualized server applications.

Evangelia Kalyvianaki et.al. (2008). "Applying Kalman filters to dynamic resource provisioning of virtualized server applications". In this paper they have proposed two controllers based on kalman filters which adapt according to changing workloads.

Moreover we can make this mechanism to adapt more quickly or slowly to workload changes and also parameters for these controllers can be computed online. In this method controllers are used to allocate the resources to the CPU. In each CPU there is manager that periodically sends CPU usage to controller and controller provides resources accordingly. In this approach they are using two controllers. 1. Basic controller. 2. Process noise covariance controller. Although in previous works only basic controllers were used but with the addition of PNCC further improvement is achieved by considering resource coupling tiers. Moreover PNCC adaptive is used to take non-stationary noises under consideration. Although kalman uses an efficient approach but this technique is suitable with small servers only maintaining controller with each CPU and linking those controllers is a difficult task to achieve and makes system complex. [14]

Autonomous resource provisioning for multi-service web application.

Jiang Dejun et.al. (2010). "Autonomous resource provisioning for multi-service web application". In this paper SLA (API) is applied only to the front end service and rest of the services are automated. Using this approach various services can be provisioned whether an application or data service. Each service may consist of number of instances with replicated code and various caches in order to improve performance. In this paper they are creating acyclic graph in which services are adjusted. Each service computes its "What if analysis" according to number of available machines i.e. predicting future performance and these performance promises are reported to their parent services. Each service consists of aggregated performance values of their child nodes. Responsibility is totally of the intermediate service for provisioning of resources locally. One advantage of this approach is that it is a decentralized approach where each service is responsible autonomously for its own provisioning. Limitation also persists that they are assuming that the availability of the resources is infinite. So this technique is only subjected to data centers and clouds. [12]

Regression based multi-tier resource provisioning for session slowdown guarantees.

Sireesha Muppala et.al. (2010). "Regression based multi-tier resource provisioning for session slowdown guarantees". In this paper statistical machine learning is used. According to this approach firstly offline training is provided to machines and tested under diverse workloads offline. When the system is used in real-time workloads regression analysis is used to model the internet service behavior pattern whenever the service is violated then system uses predefined metrics and act accordingly. Pros of using this technique are that once the system has cleared training process it is fully automated. However some cons also exist 1. Training should be efficient. If training is not up to the mark then overall system will be affected. 2. Caches or queuing model is not deployed so this approach is not suitable for heavy workloads as it will compromise with the efficiency. [13]

Optimal provisioning of resource in a cloud service.

Yee Ming Chen et.al. (2010). "Optimal provisioning of resource in a cloud service". This paper uses discrete particle swarm optimization algorithm (DSPO). In this method they are creating t*p matrix and each row representing task allocation and column representing allocated tasks in a processor. With the use of this method we are able to achieve the speed of convergence and ability to obtain faster and feasible allocation. However speed of convergence and obtaining faster and feasible allocation are the advantages of this approach. But cons are that dynamic workloads are not taken into the consideration. Moreover this approach is not suitable for large systems. [15]

Dynamic provisioning modeling for virtualized multi-tier application in cloud data center.

Jing Bi et.al.(2010). "Dynamic provisioning modeling for virtualized multi-tier application in cloud data center". In this paper they presented a novel technique for dynamic resource provisioning under cluster based virtualized multi-tier applications. It consists of resource pool where the physical machines are present in data center along with virtualized applications running on them. Advantage of using virtualized machines is that the demand of the resources increases or decreases with respect to time so in order to cope with this problem virtualized environment is used. It consists of resource pool and self-management community. In self-management community basically four functions are performed. 1. Monitor. 2. Analyzer. 3. Resource scheduler. 4. Virtualized application executor. Resource computational pool sends delegate to self-management community and SMC generates the response accordingly. Under virtualized multi-tier application queuing model we are having analytic performance models. In this model they are using ODS (On Demand Scheduler) for the first tier only to schedule the request. Once the request is scheduled it only needs to be processed and passed to successor tiers. In open queuing model first step is to estimate the capacity of VM's at each tier in accordance to the rate of request they can handle. Next step is to determine the number of VM's required to process the request and satisfy the requirements. Advantage of using this approach is increased efficiency and flexibility for cloud data centers. To pursue further research we can use load prediction method technique. [16]

SLA-Based resource provisioning for heterogeneous workloads in virtualized cloud data centers.

Saurabh kumar garg et.al. (2011). "SLA-Based resource provisioning for heterogeneous workloads in virtualized cloud data centers". In this paper they are proposing admission control and scheduling mechanisms to tackle with the dynamic workload nature of cloud computing. Process of scheduling used is,

1. Admission control: - Admission control decides whether a VM can be allocated to the request to serve and if accepted then it signs SLA with the user.

2. VM manager: - VM manager initiates a VM and allocate it to the request then job scheduler schedules applications on this VM.

3. Job scheduler: - Job scheduler assigns jobs to initiated VM's.

4. SLA manager: - SLA monitors each accepted application. SLA enforcement and rescheduling algorithm is used. It performs following functions. 1. Enforce SLA. 2. Schedule jobs from batch job. 3. Consolidation. [17]

Hybrid resource provisioning for minimizing data center SLA violations and power consumptions.

Anshul gandhi et.al. (2011). "Hybrid resource provisioning for minimizing data center SLA violations and power consumptions". In this paper they are using the historical data and on the basis of that data they allocate the resources i.e. 1. Predictive. 2. Reactive. In predictive they are using historical data and on the basis of that data they allocate the resources but with the use of reactive technique we can recompute the demand of resource at finer times. Advantage of this approach is that this model is simple and efficient for small centers. However there are also some disadvantages that queuing model is not used and this approach is not suitable for large centers where multiple requests arrive in a small time span. [18]

Dynamic Request Managemennt Algorithms For Web-Based Services in Cloud Computing.

Riccardo Lancellotti et.al. (2011). “Dynamic Request Management Algorithms for Web-Based Services in Cloud Computing”. In this Paper they presented an algorithm ‘Performance Gain Prediction’ which decides whether to process a request locally or to redirect to other server on the basis of some predefined parameters i.e. (Server load, Computational Cost, User session migration and redirection Delay). However this approach increased the efficiency to 25% from the traditional algorithms. But limitation is if the request requires very less computation so Applying the algorithm to such a request will only decrease the latency and response time and Computing this algorithm for each of the thousand requests in a minute is very complex. However for servers with very minimum number of requests this algorithm is worth Functioning. [19]

Dynamic Provisioning of multi-tier applications.

Wesam dewoud et.al. (2012). “Dynamic Provisioning of multi-tier applications”. This paper is enhancement to the previous paper discussed i.e. “hybrid resource provisioning for minimizing data center SLA violations and power consumption”. In this paper along with predictive and reactive techniques a flexible queuing model is also incorporated for determining how many resources are needed to each tier. Two basic things which we need to consider while provisioning resources are, 1. How much to provision and 2. When to provision. In this paper the required capacity at each tier is evaluated using the queuing model thus allocating the desired capacities at various tiers all at once. Various advantages of this approach are efficient and flexible model. Also queuing model along with predictive and reactive techniques further enhances the capability of model. [20]

Lightweight resource scaling for cloud applications.

Rui Han et.al.(2012). “Lightweight resource scaling for cloud applications”. This paper introduces a new approach of adding or removing lightweight resources (i.e. CPU’s memory, I/O etc.) rather than adding or removing whole virtual machine instances because latter bears more cost and overhead. They are using LS algorithm. According to it if value of t^0 > tu (LSU) then we need to scale up resources and if the value of $t^0 < t^l$ (LSD) then we need to scale down resources. Now t^l = Lower bound and t^u = Upper bound. To use this approach effectively we need to check the performance periodically.

Various advantages of this approach are firstly this approach is simple and efficient and secondly it adapts cost effective scaling. We can extend this approach by scheduling the resources between application and also understanding tradeoffs between reservation cost and risk of high running cost. [21]

A Cloud computing approach based on mobile agents for web services discovery

Saouli Hazma et.al. (2012) “A Cloud computing approach based on mobile agents for web services discovery”. In this paper they presented the approach to discover the services over the web. First the search is initiated on the key words provided by the client and then the filtering algorithm is applied to select the appropriate service among the discovered. Cloudsim is used as a simulator in this research work. Two cloud Regions are considered and when the request arrives the services are searched in the local cloud which initiates the searching on the other cloud and filtering is performed there and the result is returned to the local cloud and then to the client. The problem is that this approach is beneficial for limited number of clouds whereas if the large number of clouds are considered in the semantic search then the problem arises of the absence of single tool. [28]

Resource provisioning of web applications in heterogeneous clouds.

Jiang Dejun et.al. (2013). “Resource provisioning of web applications in heterogeneous clouds”. In this paper they calculated the performance profile of each individual instance. Now these parameters help us to balance the load more efficiently and more accurately rather than allocating the load on the basis of parameters calculated from overall tier. They controlled workload by applying a load balancer in front of provisioned instance. Then load balancer calculates weighted workload distribution. Now whenever a new instance is added in any tier load balancer calculates its performance profile in order to balance the workload intensities. In this paper heterogeneous instances are taken into account. So it accommodates various advantages firstly heterogeneous nature is considered and secondly better allotment of instances. [22]

Mobile Computation Dynamic Offloading Using Cloud

Pooja V. et.al. (2013). “Mobile Computation Dynamic Offloading Using Cloud“. This paper presents an approach of Dynamic offloading the application to the cloud to maximize the efficiency and increase battery life and to meet the unpredictable user requirements. The choice of offloading depends on both the application characteristics as well as the Current environment. Decision is taken on the basis of five parameters i.e. Input data size, CPU usage, Memory Usage, Execution Time and Power consumption. On the basis of these parameters three Execution modes are considered i.e. Local (Executed on mobile), Remote Wi-Fi (Outsource to the remote server through Wi-Fi), and Remote 3G (Outsource to the remote server through Sprint 3G network). The offloading Decision can also be taken on the basis of user preference. So a group of Predefined Preference policies can be stored at the mobile device which can be selected by the user. Limitation of this approach is that offloading requires large amount of data to be transmitted over the network and the communication overhead so Fast network must be available at the time User want to use the application. As 3G network is not available all over the locations so it becomes difficult to apply this approach. [27]

V-cache :- Towards flexible resource provisioning for multi-tier application in Iaas clouds.

Yanfei Guo et.al. (2013). “V-cache :- Towards flexible resource provisioning for multi-tier application in Iaas clouds”. In this paper they presented a machine learning approach in which caches are deployed in front of the applications. These caches use a genetic algorithm that identifies the request that can be most benefited from caches and dynamically resizes itself so as to accommodate all requests. In architecture of V-cache there are –

1. Workload analyzer – It performs following functions.

- Clustering of requests according to their size, types and processing costs.
- It also maintains statistics of completed requests.

2. Policy generator – It performs following functions.

- Identifies the request that benefits the most from the caches. It takes cluster of requests as input and request redirection map to request redirectors as output.

- It also determines the size of cache to accommodate those requests.

3. Request predictor – It performs following functions.

- It determines whether to send or forward the request to cache tier or to forward the request to web tier on the basis of URL and host sent by policy generator.

We can extend work further for heterogeneous applications and integrating admission control for overload control and performance guarantees. [23]

CHAPTER 3
PRESENT WORK

The Enhanced V-Cache resource provisioning technique will enhance the efficiency by considering the fluctuating workloads. The workload is never constant under real time situations. So, fluctuating workloads should be considered in order to provision the resources more effectively and efficiently. In this approach we are working on cache so that we can filter the requests and those requests which can be served through cache could be served locally and this will eventually increase the throughput, less memory usage and less CPU utilization. At the end we can save the resources from unwanted wastage. However many techniques are available static as well as dynamic but we lack with optimal resource provisioning mechanism. So by increasing efficiency of caching policy we can increase the efficiency locally.

3.1 PROBLEM FORMULATION

3.1.1 Existing system

First let us consider the existing V-cache architecture.

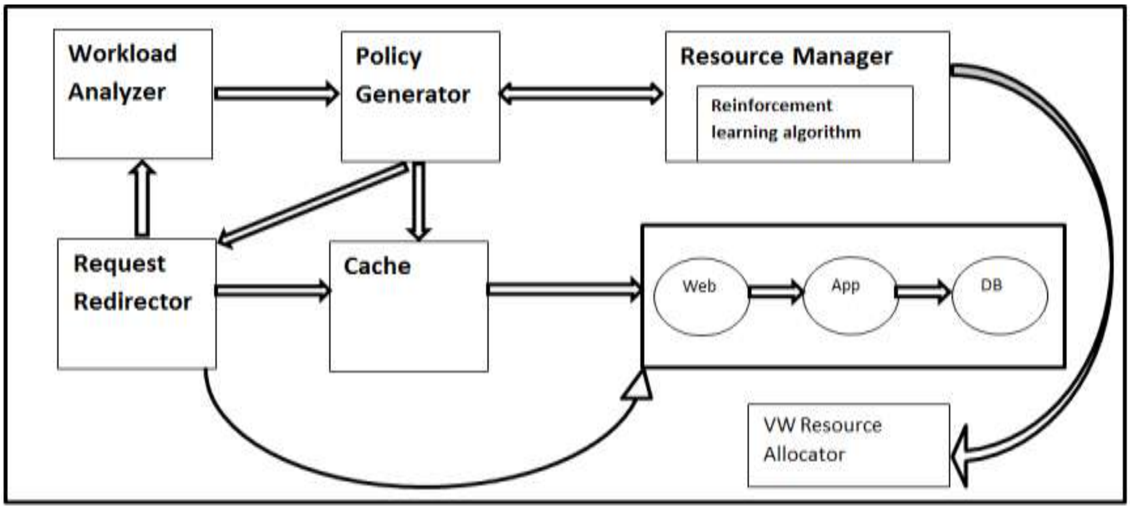


Figure 3.1 Architecture of v-cache. [20]

Various algorithms have been proposed both static and dynamic for provisioning of resources in a cloud. However none of them is able to achieve ideal efficiency due to fluctuating nature of workloads. Workloads changes according to the needs of user so there is a need of scale-up and scale-down the resources in order to maximize the proper utilization of resources. Another challenge is that demand of each tier is separate and should be dealt separately. So we are proposing the method which takes into consideration both the fluctuating workloads as well as resource contention problems.

In figure 3.1 we are deploying v-caches so as to predict the nature of the request with the use of URL and Host data and process accordingly. We predict the requests that can be benefited the most from caches. This work is done by Policy generator. It also maintains statistics of completed requests. Then the request is passed to Request redirector which further determines whether to forward the request to caches tier of to pass the request to web-tier. In this system they have fixed the size of cache and fluctuating workload has not been taken into consideration. We cannot guarantee that the workload will remain same for 24 hours. As it is clear in fig 1.3 that the workload has a certain peak hours where we need additional resources in order to combat with resource contention problem so we decide some peak hours that may vary from application to application and the size of cache is determined through that policy. When the application is running under peak hours then size of dynamic cache is increased so that maximum requests can be accommodated and benefitted from cache. On the other hand when application is running under normal hours we decrease the size of dynamic cache in order to save the resources.

In the existing system when the request arrives it is first processed by workload analyzer where the type of request is determined and size to store that request into cache is determined. After that the request is passed to policy generator where we determine whether to process the request through cache or to process the request through server. It also compares TTL property of the dynamic requests. The request whose TTL has been expired are sent directly to the server for processing which further increases the cost. So after the policy generator determines the overall situation the request is passed to request redirector which redirects the request either to cache or server for processing according to the results provided by policy generator. Now in the existing system size of cache are fixed which do not take into consideration fluctuating workloads. So size of cache should be determined with utmost care. During peak hours when the workload is maximum cache cannot be able to accommodate all the requests whereas during normal working hours there may be a lot of space unused in these caches which is leading to wastage of resources. So size of cache should be fixed carefully. So we are proposing a model which takes into account the fluctuating workloads and changes the size of cache according to the type off application because different applications may have different peak hours. So heterogeneity of the applications is also considered.

They have used optimal caching policy in which it is the mapping of request either to cache or to multi-tier application on the basis of minimum cost incurred in either case.

For n request cluster there can be $2^{n.}$ Suppose system workload has n clusters so for each cluster it maps 1 to p requests to cache and p+1 to n directly to the application. We define the cost incurred in mapping M as

$$y = \sum_{i=1}^{p} [c_i n_i h_i + a_i n_i (1 - h_i)] + \sum_{j=p+1}^{n} a_j n_j$$

Where *ci* and *ai* are the processing cost of the request cluster i from cache and application respectively. Also *ni* is the total number of requests present in the cluster and *hi* is the hit ratio of the cache. Using these 2^{n} mappings we can find optimal solution with least processing cost. This system uses genetic algorithm which records response time of each request and when the same request appears again then it checks the data from database for that request and makes the decision accordingly.

3.1. Proposed system

The Extended V-Cache resource provisioning technique will enhance the efficiency by considering the fluctuating workloads. As the request arrives we determine if it can be served by cache and concurrently our system estimates the workload conditions if possible the request is served locally thus saving the resources and scaling of resources works concurrently. Architecture of extended v-cache is shown below.

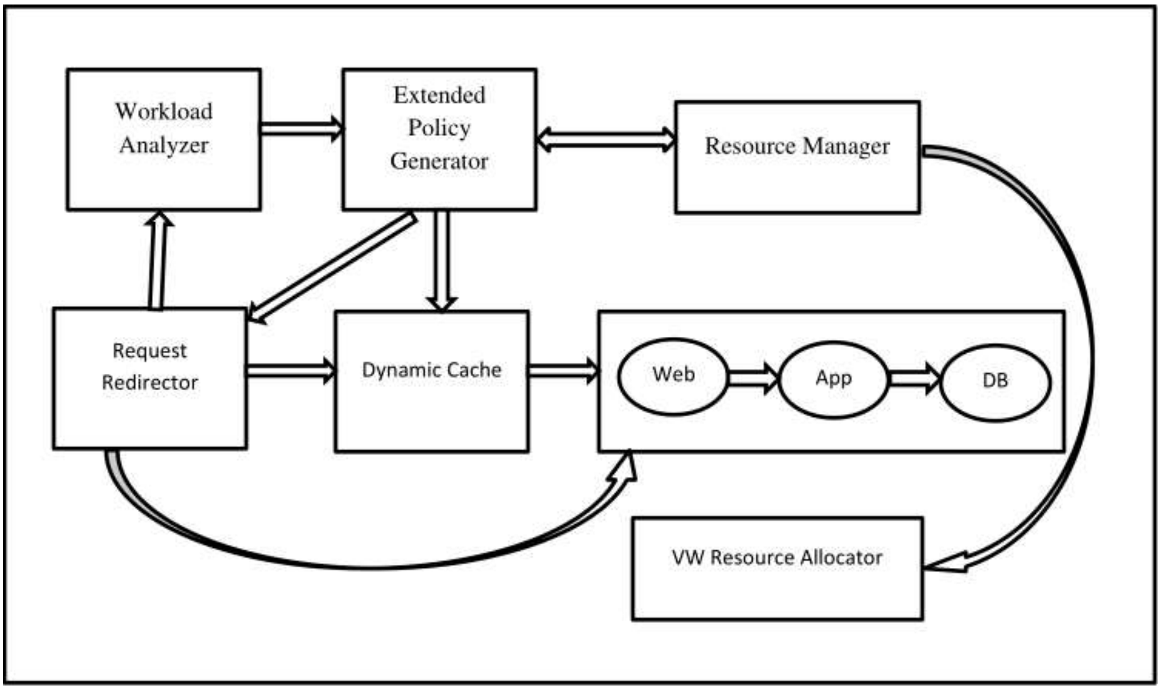


Figure 3.2 Architecture of extended v-cache.

In extended v-cache we consider the changing pattern of workload and size of cache is determined accordingly in order to accommodate maximum request into cache at the time of peak hours to increase the efficiency, throughput, minimize CPU utilization and reduce RAM consumption. Keeping the same size of cache during normal working hours is also wasting the resources unnecessarily so under normal working hours size of the cache is reduced. Now when the request arrives at workload analyzer it first determines the type of the request and also maintains the statistics of completed requests. After that request is passed to extended policy generator where decision whether to process the request through cache or through web application is taken. Also it checks if the application is running under peak hours then size of dynamic cache is increased and if application is running under normal hours them size of the cache remains unchanged. After processing through extended policy generator the request is passed to request redirector which redirects the request according to the results provided by the extended policy generator.

Various parameters through which we will conclude the efficiency of our algorithm are

1. Throughput: -Throughput is the total number of request served from cache under a given interval of time. Throughput should be maximum in order to attain ideal efficiency and to provision the resources effectively.

2. CPU usage: -CPU usage should be minimum and maximum number of requests should be served through cache. The size of cache should be chosen very carefully as large size may lead to wastage of resources and small size may lead to poor provisioning.

3. Memory usage: -Memory usage is the memory i.e. RAM used to process the request. If the request is processed through cache then the memory is saved and there is low consumption of memory. Memory usage should be minimum.

The pseudo code for this whole system is given below: -

3.1.3 Pseudo code

```
START
Application $A$ starts
Peak hours for application $A = P_k$
Request arrives with id $Ai$
Workload analyzer identifies type of request and passes to policy generator
Policy generator determines whether request $Ai$ has been served earlier
Policy generator also checks whether application $A$ is running under peak hours $P_k$
IF      application A running under peak hours $P_k$
        Increase the size of dynamic cache $C_d$
Else
        Size of $C_d$ remains unchanged
ENDIF
IF      request $Ai$ arrived for the first time
        Response=Process the request to web-tier
Else
        Response=Process the request $Ai$ through cache
```

```
ENDIF
Request is passed to request redirector
IF      Response= Process the request to web-tier
        Redirect the request to multi-tier application
ELSE    Redirect the request to Cache tier.
ENDIF
STOP
I=i+1
```

Repeat this for every request.

Once the application starts accepting the requests

3.2 Objectives

1. To analyze resource provisioning algorithms available in multi-tier applications.
2. To identify the challenges and problems of resource provisioning in multi-tier applications.
3. To enhance the efficiency of V-Cache algorithm using the adaptive techniques.
4. To implement proposed and existing algorithm and analyze performance in terms of accuracy and efficiency.

3.3 Research methodology

Research methodology used in this approach is simple yet efficient. In this approach we adopted the concept of dynamic caching policy according to the type of user. Rather than allocating same storage space we define storage space according to our SLA defined and according to the user accessing our services.

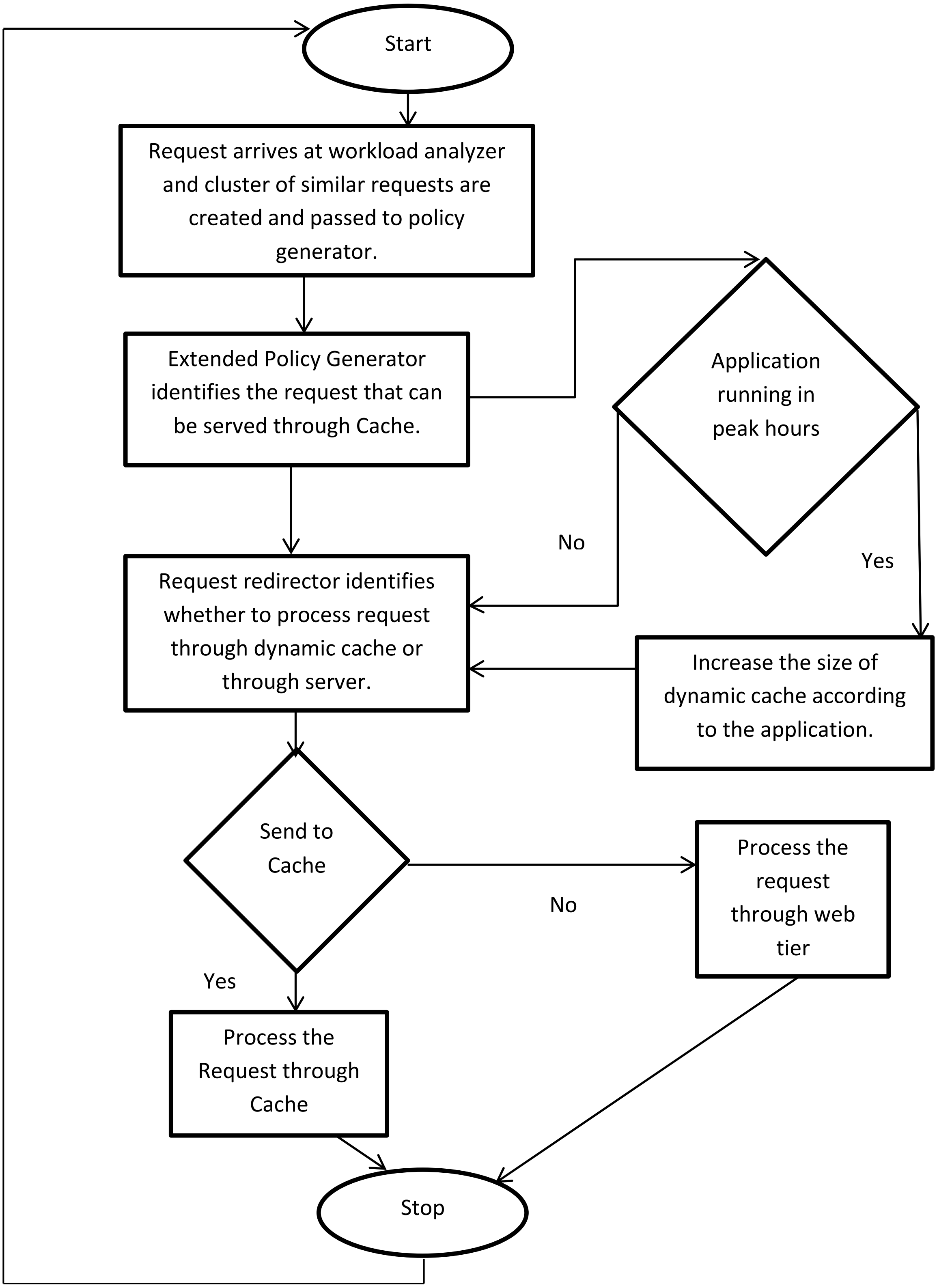

Start
Request arrives at workload analyzer and cluster of similar requests are created and passed to policy generator.
Extended Policy Generator identifies the request that can be served through Cache.
Application running in peak hours
No
Yes
Request redirector identifies whether to process request through dynamic cache or through server.
Increase the size of dynamic cache according to the application.
Send to Cache
No
Process the request through web tier
Yes
Process the Request through Cache
Stop

Fig. 3.3 Flow chart of research methodology.

Workload analyzer: - Workload analyzer identifies the type of request i.e. (Static or Dynamic) and the SLA agreements between the service provider and client. After that the request is passed to policy generator to act according to the type of request.

Policy Generator: - It generates policy according to SLA, Peak Hours and type of the user and other information provided by the workload analyzer. We have set policies according to the various SLA categories and type of user requesting for services. Size of cache is determined dynamically according to the policies defined. After this the request is passed to request redirector.

Request Redirector: - Request redirector redirects the request either to cache server or to the web server according to the decision taken by policy generator.

Database Tables

Various tables have been designed to store the results and policies to take effective decisions at finer times. Different tables used are DynamicCachePolicies, Policies, RedirectionDatabases.

Dynamic Cache Policies: - This table is used to verdict the size of cache dynamically according to the type of user requesting for services. Various columns used in this table are ID, Peakstart, Peakstop. According to the type of application with unique ID peaks hours are defined and cache size is determined accordingly.

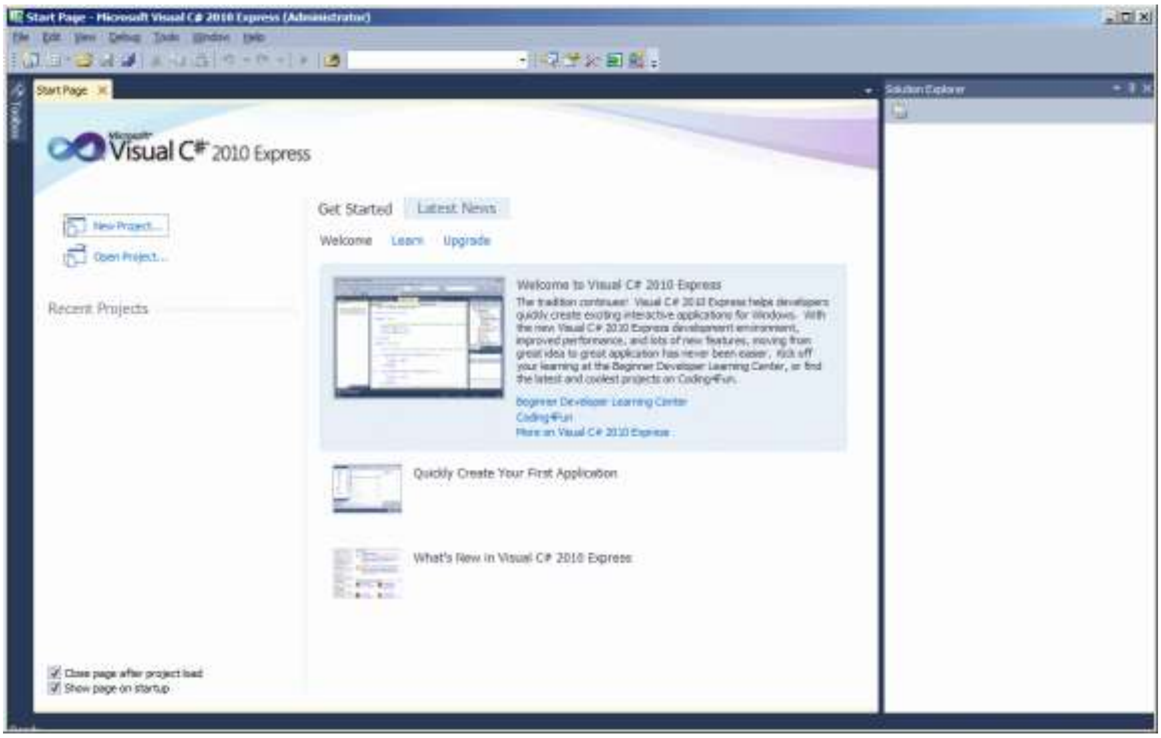
Policy: - This table is designed for the sake of storing and retrieving the information of every unique request as well as defining the policies for that request. ID is used as unique identification number for every request appearing for first time. Its URL is stored and according to specific URL number of caches, SLA for that URL, Size of cache, Peak hour size and off peak hour size is defined in this table.

Redirection Database: - Using this table we are storing the address of every request in cache according to the type of request and according to the peak hours and off peak hours. For every request with different ID we are storing the URL of server and appropriate cache address is stored in Cache URL for that request.

CHAPTER 4

RESULTS AND DISSCUSSION

In this chapter results of proposed system are discussed. We have used visual studio .NET platform for implementation as well as for simulation. Microsoft visual studio is an IDE (Integrated Development Environment) by Microsoft. The main use of visual studio is for developing computer programs for Microsoft as well as web services, web sites and web applications. Microsoft development programs are used by visual studio such as windows presentation foundation, windows forms, windows applications, windows API, windows store and Microsoft Silverlight. The integrated debugger present in visual studio works both as machine-level debugger as well as source-code debugger. Many other built-in tools are available such as forms for building interactive GUI applications, class designer, web designer and database schema designer. [9]



Welcome to Visual C# 2010 Express

The tradition continues! Visual C# 2010 Express helps developers quickly create exciting interactive applications for Windows. With the new Visual C# 2010 Express development environment, improved performance, and lots of new features, moving from great idea to great application has never been easier. Kick off your learning at the Beginner Developer Learning Center, or find the latest and coolest projects on Coding4Fun.

Beginner Developer Learning Center
Coding4Fun
More on Visual C# 2010 Express
Quickly Create Your First Application
What's New in Visual C# 2010 Express
☑ Close page after project load
☑ Show page on startup

Figure 5.1 Visual Studio running on windows. [10]

Visual studio enhances its functionality by accepting plugins at almost every level. It also supports many programming languages and allows the debugger and code editor to support nearly any programming language. Languages that pre-exist in visual studio are

C++, C, C++/CLI, VB.Net, F#. Other languages are also supported such as Python, M and Ruby. Other markup languages are also supported like HTML/XHTML, XML/XSLT, JavaScript and CSS. [9]

In this section we are going to discuss about the working of our proposed system i.e. extended v-cache. Provisioning of resources is very crucial in order to maintain SLA and QoS as well as to reduce the cost both for client as well as for service provider. Also it is very necessary to attain efficient and effective results. Now when the request arrives at workload analyzer it first determines the type of request according to the URL and host.

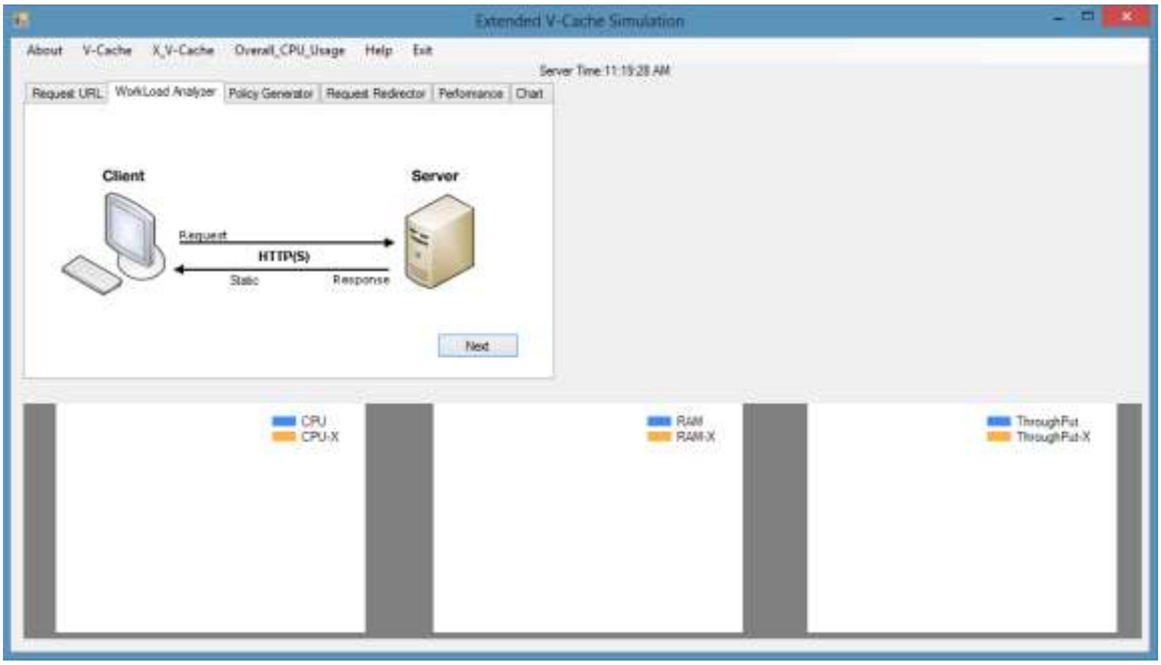
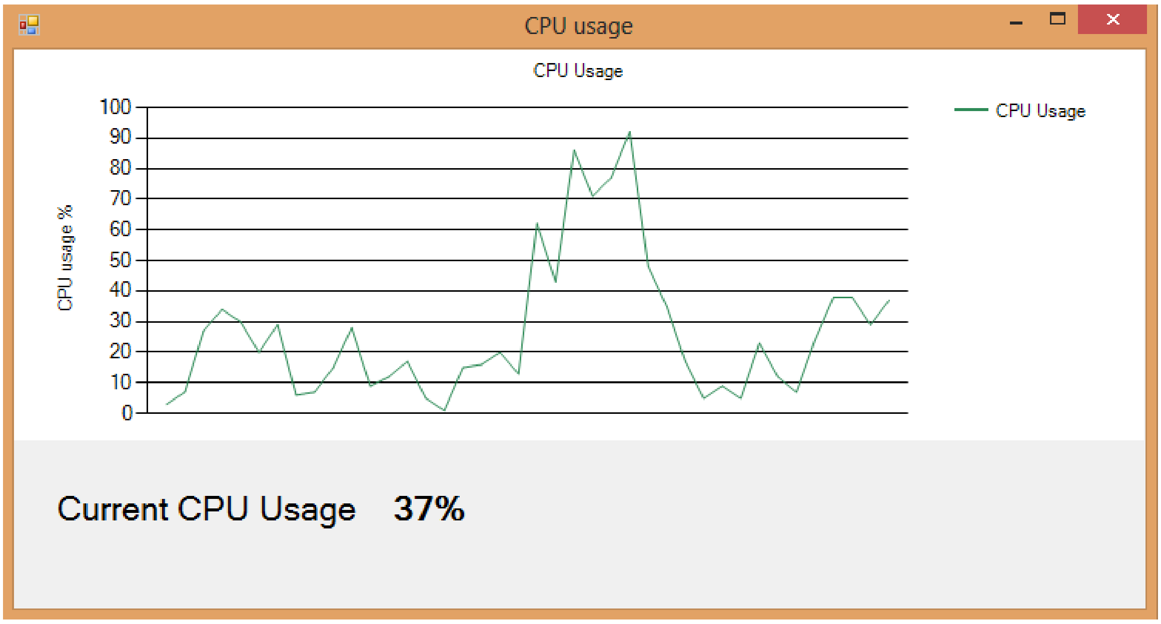


Figure 5.2 Working of workload analyzer.

Working of workload analyzer is shown in above figure where the request is static. We are considering web pages as request parameter. So we have designed three websites namely one tier, two tier and three tier. One-tier website consists of static webpages and static content. Two-tier website consists of business logic of currency converter and it is a dynamic website however the change in content of two-tier website is not frequent. Third website is three-tier where all the three tiers are included i.e. presentation tier, application tier and database tier. Now when we enter the address into the URL box the request is passed to workload analyzer and type of request is identified and after that request is passed to policy generator. These defined policies are implemented on the request according to the SLA and appropriate cache size is determined. After that the request is redirected to web-tier. Now we can also check the CPU overall usage and depict the pattern of request arrival and service.



Current CPU Usage 37%

Figure 5.3 Effect on overall CPU usage.

CPU usage, Memory consumption and throughput are measured using this approach. Although our results are dynamic as we are taking the results from real working server but it depicts the overall pattern of results. Now for one tier website we first deploy and run the website on our server and then that server address is copies and given to our application URL. Results or performance is calculated from that server hosting our website and then results are compared. For one tier application where we are having only the static pages is easy to calculate the results because we can store maximum static pages into our cache and efficiency can be gradually increased. However storing dynamic pages is not a good idea because TTL of dynamic pages expires very frequently and maintain the cache in that case becomes confronting and also more the number of cache miss occurs more degradation is seen in our performance. So for one-tier web application the results are shown below.

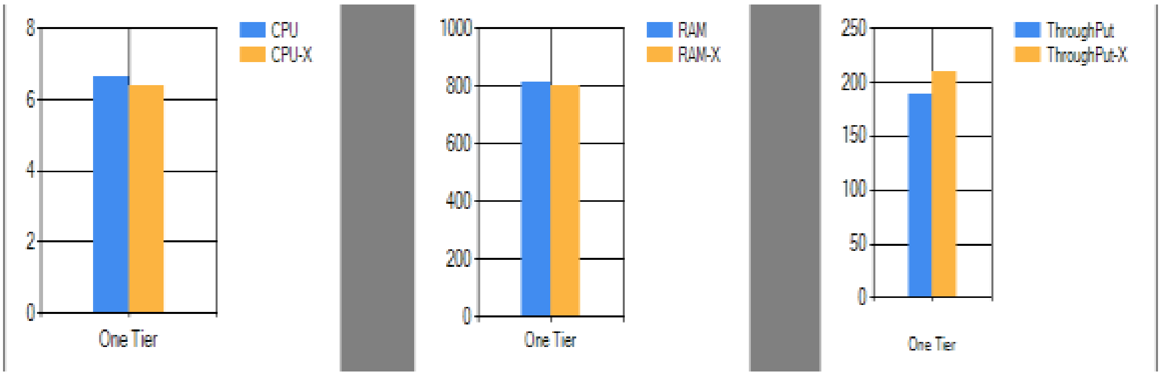


Figure 5.4 Results for one-tier application.

Above shown results are for one tier application where blue bars are for V-cache and orange bars are for extended V-cache. We can see that when more requests are served through cache in peak hours our CPU utilization is decreased and ram consumption is also decreased. Throughput is increased because now with the concept of dynamic cache more requests are served through caches during peak hours so throughput is increased gradually. Whereas this pattern does not fit well for dynamic websites however there is some increase in throughput but due to TTL of pages stored it is very confronting to take decision regarding service of dynamic pages.

For two-tier applications TTL property is to be taken into consideration. TTL is different for different pages and it is not possible to determine in advance. So cache miss ratio in case of dynamic pages is very high. So we are working on static pages. Various tables are used to store the results as well as for storing the policies defined. According to the values stored in table we reach to the decision of whether to increase or decrease the size of dynamic cache. Although in V-cache policy generator was used only to determine whether to serve the request through cache or through web-tier but in extended V-cache extended policy generator is used which also takes the decision about the size of cache on the basis of peak hours or off-peak hours. We want that at whatsoever SLA should not be violated. Type of SLA signed with individual user is also stored in database.

Results of two-tier applications conclude that there is decrease in RAM and CPU usage which is good in saving the resources. Due to the TTL property of dynamic pages there is a slight increase in throughput.

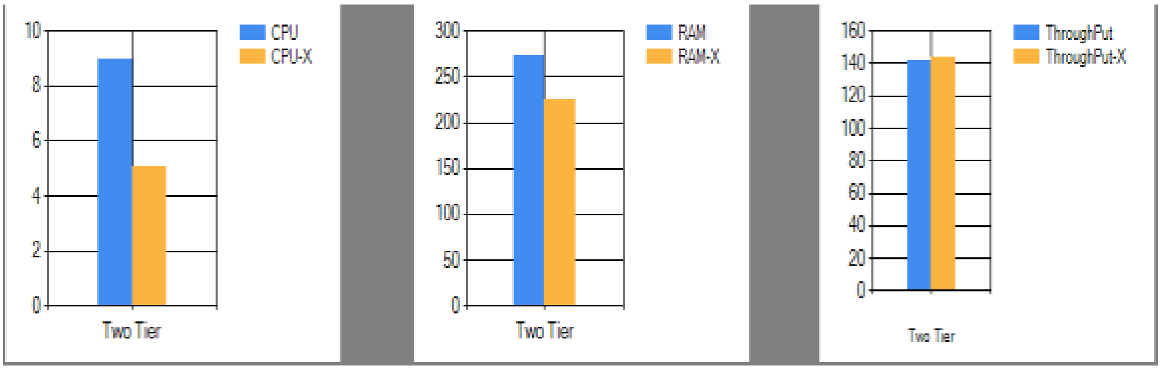
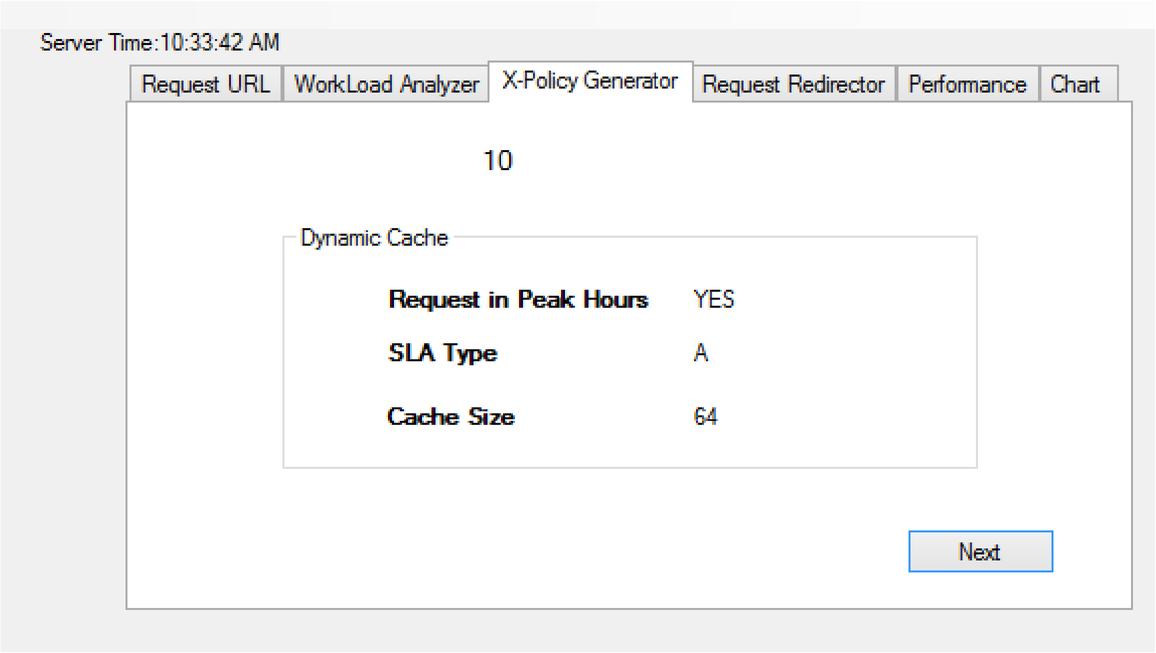


Figure 5.5 Results for two-tier application.

Above shown results are for two-tier application where there is gradual decrease in CPU usage and RAM thus saving resources and using cache efficiently. Results of two-tier application are taken in peak hours where the size of dynamic cache is increased to 64 MB thus storing maximum pages.



10

Dynamic Cache

Request in Peak Hours	YES
SLA Type	A
Cache Size	64

Figure 5.6 Request in peak hours.

Results of dynamic cache are shown above. Three parameters are considered where request in peak hours determine the size of cache. Every application has different peak hours. For example if we consider Facebook application then according to the workload of that application it can be depicted that application is accessed maximum from 18:00 hours to 23:50 hours.

So this value is given in peak hours and off-peak hours respectively. Different applications have different SLA signed between customer and service provider. We must take care that in any case SLA should not be violated.

Under three tier applications the request includes database tier also. Now when in peak hours we are increasing the size of cache it is like we are storing more requests in our cache for particular applications and that leads to increased throughput. Suppose we signed SLA with three users i.e. user A, user B and user C. We have to provide them their desired results in 5, 10 and 15 days respectively. So rather than serving the request of user under same priority i.e. 10 days we serve the request according to the timelines of different users keeping in mind SLA of each and every user. If in case we serve the request with same priority than we are violating the SLA of user A. so keeping in mind all these things we proceed with serving the request according to the priority of the user.

Also if we are storing the same number of pages statically then we are wasting our resources. Under peak hours more pages can be stored and served concurrently thus saving the resources and utilizing the cache efficiently. So suppose we have three caches for different users User A, B and C has been allocated 16 MB, 32 MB and 64 MB cache memory respectively. Suppose every website consist of 10 pages and each page is of 4 MB so ultimately we are only able to store 4 pages for user A, 8 pages for user B and all the 10 pages for user C.

So if every user accesses all the pages of its website during peak hours and only most frequently 4 pages are accessed during off-peak hours then using extended V-cache maximum utilization is achieved. If we consider peak and off-peak hours then in existing system SLA will be violated during peak hours. However using the proposed approach we are effectively maintaining the size of cache during peak and off-peak hours. So dynamic cache and extended policy generator helps in proper, effective and efficient utilization of cache. Results for three-tier application are shown below.

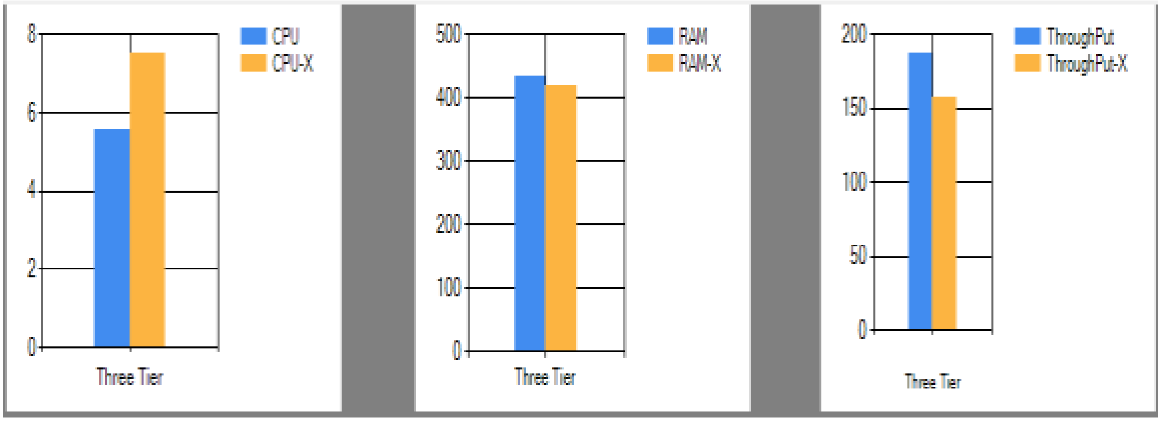


Figure 5.7 Results for three- tier application.

From the above results we can easily depict that RAM consumption is reduced. Although the results are dynamic and so there are chances that sometimes our CPU is busy in other tasks so throughput decreases. Overall results for all the tiers are shown below.

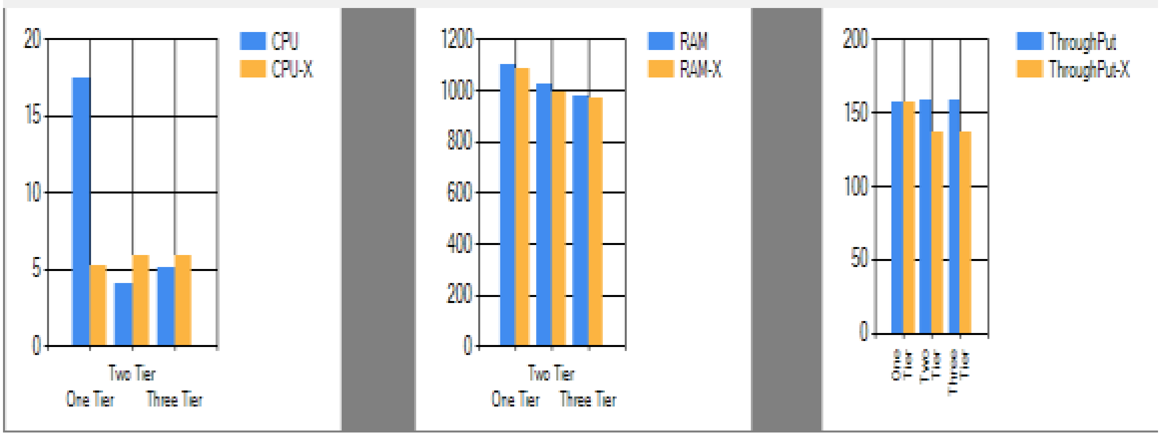


Figure 5.8 Results for all three-tiers.

At the end this research concludes that there is gradual increase in throughput for static web pages. Also the RAM and CPU usage is decreased to huge extent but three-tier website which uses dynamic web pages are hard to store and manage due to TTL property of pages. Also using the cache for dynamic web pages decreases our efficiency with increase in cache miss rate. So dynamic web pages should only be stored during peak hours for short duration.

CHAPTER 5
CONCLUSION AND FUTURE SCOPE

As each tier in the cloud has its own processing requirements so it becomes very challenging to cater appropriate resources at each tier. Moreover cloud should be flexible as most of the webpages used these days are dynamic so adaptive nature of the cloud is as influential as other challenges discussed. So in this paper we are working on heterogeneity of user as well as combating with resource contention problem by using extended v-cache approach. At the end we conclude that storing maximum static web pages are far more efficient and effective than storing and managing dynamic web pages. Cache hit ratio is maximum for static web pages. During peak hours maximum numbers of pages are stored in our dynamic cache with the increase in size of cache which results in increased throughput. Also using this approach we are assuring that SLA is not violated which further enhance the efficiency of this approach. During off-peak hours size of the cache is reduced thus saving the resources. Extended policy generator assures that there is no violation of SLA and Qos is increased. It is easily depict able from the results that using this approach consumption of costly resources is decreased gradually and cache is used efficiently and effectively. In future we will be focusing on managing the resources by developing minimum share algorithm. This work can further be improved by considering heterogeneous application. In this approach homogeneous applications are considered. Moreover this approach defines different SLA agreement with different users.

CHAPTER 6

REFERENCES

Web References

[1] Retrieved March 23, 2014, from the opentutorials.com http://theopentutorials.com/tutorials/web-services/types-of-web-services-big-and-restful/

[2] Retrieved February 24, 2014, from webopedia.org: www.webopedia.com/TERM/W/Web_Services.html

[3] Retrieved April 12, 2014, from tutorialspoint.com: http://www.tutorialspoint.com/webservices/what_are_web_services.htm

[4] Retrieved February 5, 2014, from owasp.org. http://www.owasp.org/index.php.web_services

[5] Retrieved February 25, 2014 from ibm.com: http://www.ibm.com/cloud-computing/in/en/what-is-cloud-computing.html

[6] Retrieved April 14, 2014 from infoworld.com: http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031

[7] Retrieved April 23, 2014 from w3.org: http://www.w3.org/web _services

[8] Retrieved January 12, 2015 from applications2u.com http://www.applications2u.com/cloud-delivery-models/

[9] Retrieved March 12, 2015 from wikipedia.com
http://en.wikipedia.org/wiki/Microsoft_Visual_Studio

[10] Retrieved March 18,2015 from functionx.com
http://www.functionx.com/csharp/windows/ide1.gif

References

[11] Sivasubramanian, Swaminathan, Guillaume Pierre, Maarten van Steen, and Sandjai Bhulai. *Sla-driven resource provisioning of multi-tier internet applications*. Technical report,Department of Mathematics and Computer Science, Vrije Universiteit, Amsterdam, 2006.

[12] Jiang, Dejun, Guillaume Pierre, and Chi-Hung Chi. "Autonomous resource provisioning for multi-service web applications." In *Proceedings of the 19th international conference on World wide web*, pp. 471-480. ACM, 2010.

[13] Muppala, Sireesha, Xiaobo Zhou, and Liqiang Zhang. "Regression based multi-tier resource provisioning for session slowdown guarantees." In *Performance Computing and Communications Conference (IPCCC), 2010 IEEE 29th International*, pp. 198-205. IEEE, 2010.

[14] Kalyvianaki, Evangelia, and Steven Hand. "Applying Kalman filters to dynamic resource provisioning of virtualized server applications." In *Proc. 3rd Int. Workshop Feedback Control Implementation and Design in Computing Systems and Networks (FeBid)*, p. 6. 2008.

[15] Tsai, Yee Ming Chen1 Shin-Ying. "Optimal provisioning of resource in a cloud service." *IJCSI* (2010): 95.

[16] Bi, Jing, Zhiliang Zhu, Ruixiong Tian, and Qingbo Wang. "Dynamic provisioning modeling for virtualized multi-tier applications in cloud data center." In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pp. 370-377. IEEE, 2010.

[17] Garg, Saurabh Kumar, Srinivasa K. Gopalaiyengar, and Rajkumar Buyya. "SLA-based resource provisioning for heterogeneous workloads in a virtualized cloud datacenter." In *Algorithms and Architectures for Parallel Processing*, pp. 371-384. Springer Berlin Heidelberg, 2011.

[18] Gandhi, Anshul, Yuan Chen, Daniel Gmach, Martin Arlitt, and Manish Marwah. "Hybrid resource provisioning for minimizing data center SLA violations and power consumption." *Sustainable Computing: Informatics and Systems* 2, no. 2 (2012): 91-104.

[19] Lancellotti, Riccardo, et al. "Dynamic request management algorithms for Web-based services in cloud computing." *Computer Software and Applications Conference (COMPSAC), 2011 IEEE 35th Annual*. IEEE, 2011.

[20] Wesam dewoud, Prashant Shenoy, Abhishek Chandra, and Pawan Goyal. "Dynamic provisioning of multi-tier internet applications." In *Autonomic Computing, 2005. ICAC 2012. Proceedings. Second International Conference on*, pp. 217-228. IEEE, 2012.

[21] Han, Rui, Li Guo, Moustafa M. Ghanem, and Yike Guo. "Lightweight resource scaling for cloud applications." In *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, pp. 644-651. IEEE, 2012.

[22] Dejun, Jiang, Guillaume Pierre, and Chi-Hung Chi. "Resource provisioning of web applications in heterogeneous clouds." In *Proceedings of the 2nd USENIX conference on Web application development*, pp. 5-5. USENIX Association, 2011.

[23] Guo, Yanfei, Palden Lama, Jia Rao, and Xiaobo Zhou. "V-cache: Towards flexible resource provisioning for multi-tier applications in iaas clouds." In*Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*, pp. 88-99. IEEE, 2013.

[24] Huang, Dong, Bingsheng He, and Chunyan Miao. "A Survey of Resource Management in Multi-Tier Web Applications." (2014): 1-17.

[25] Iqbal, Waheed, Matthew N. Dailey, David Carrera, and Paul Janecek. "Adaptive resource provisioning for read intensive multi-tier applications in the cloud."*Future Generation Computer Systems* 27, no. 6 (2011): 871-879.

[26] Nagesh, Bhavani B. "Resource Provisioning Techniques in Cloud Computing Environment-A Survey." *IJRCCT* 3.3 (2014): 395-401.

[27] Patil, Pooja V. Bhokare1 Chitra J. "Mobile Computation Dynamic Offloading using Cloud."

[28] Hamza, Salma, et al. "A Cloud computing approach based on mobile agents for Web services discovery." *Innovative Computing Technology (INTECH), 2012 Second International Conference on*. IEEE, 2012.

CHAPTER 7

APPENDIX

Abbreviations	**Meaning**
SLA	Service Level Agreement
QoWS	Quality of Web service
LSD	Level Scale Down
LSU	Level Scale Up
Qos	Quality of service
UWS	University Web Services
TTL	Time to live
VMs	Virtual Machines
SAAS	Software as a service
PAAS	Platform as a service
IAAS	Infrastructure as a service
LOC	Loss of control
LOT	Loss of trust
CPU	Central processing unit
RAM	Random access memory
WSDL	Web service description language
SOAP	Simple object access protocol

UDDI	Universal description, discovery and integration
XML	Extensible markup language
HTTP	Hypertext transfer protocol
CSS	Cascading style sheet