International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:9, 2007

Estimation of Buffer Size of Internet Gateway Server via G/M/1 Queuing Model

Dr. L.K. Singh, Dr. R. M. L, and Riktesh Srivastava

***Abstract*—**How to efficiently assign system resource to route the Client demand by Gateway servers is a tricky predicament. In this paper, we tender an enhanced proposal for autonomous recital of Gateway servers under highly vibrant traffic loads. We devise a methodology to calculate Queue Length and Waiting Time utilizing Gateway Server information to reduce response time variance in presence of bursty traffic.

The most widespread contemplation is performance, because Gateway Servers must offer cost-effective and high-availability services in the elongated period, thus they have to be scaled to meet the expected load. Performance measurements can be the base for performance modeling and prediction. With the help of performance models, the performance metrics (like buffer estimation, waiting time) can be determined at the development process.

This paper describes the possible queue models those can be applied in the estimation of queue length to estimate the final value of the memory size. Both simulation and experimental studies using synthesized workloads and analysis of real-world Gateway Servers demonstrate the effectiveness of the proposed system.

.

***Keywords*—**Gateway Server, G/M/1 Queuing Model.

I. Introduction

AS E-Commerce continues to grow swiftly [2], the original architecture providing the service of E-Commerce is becoming more and more important. According to [3], slow performance of Internet Server will cost as much as $4.35 billion annually in lost revenues.

The architecture of Gateway Server is in general referred to as "Web services". The term "Web services" describes specific business functionality through Internet connections, for the purpose of providing a way for another entity to use the services it provides [4]. Web services are the building blocks for the future generation of applications and solutions on the Internet.

The services provided through Internet Server(s) are colossal. It transfers information in form of text, images and voice. The whiz of information technology is because of consistent and steady functioning of Internet Server(s) [13],[14],[15].

However, Web traffic is highly dynamic and volatile [4], [5]. The data arrives and departs from different nodes randomly. Thus, we can envisage that, "*number of channels"* for arrival and *"number of channels"* for departing must be identical. The incoming data can be stochastically treated as a *"process"* and so will be the case of departing from the memory of Web Servers. These situations make the working of Memory of Web Servers - a typical case of - ***"Queuing Process"***[1]. During the last 40 years, research has shown that queuing models serve as a fundamental tool to model computing systems [6], [8]. In fact, queuing models have been successfully applied to areas such as capacity planning [7] and performance analysis [7] etc.

The layout of Gateway Server is depicted in Figure 1, in which data arriving and data departing at a node has been revealed:

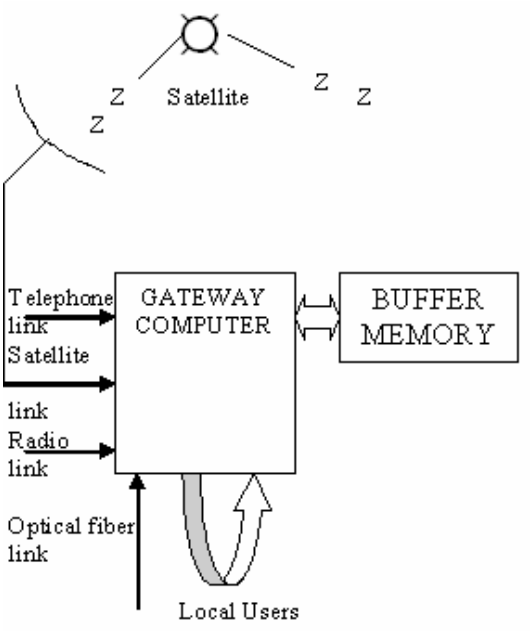

Fig. 1 Gateway Computer-Data Arrival / Data departure

It has been implicated that memory passes through diverse situations of Queue Models, i.e., *M/M/1, M/G/1, G/M/1* and *G/G/1*. In all these four models, *M/M/1* model is most disciplined and can be analyzed analytically to estimate the queuing parameters, i.e., *"Queue Length"* and *"Waiting time"* [11],[12]. These are derived as given below:

Dr. L.K. Singh is Director at I.E.T.
Dr. R.M.L. is with Avadh University, Faizabad, INDIA
Riktesh Srivastava is Project Head – Internet Technologies, at Academecia Software Solutions, Dehradun, INDIA
(e-mail: rikteshsri1@rediffmail.com)..

$$\overline{Q(L)} = \frac{\lambda/\mu}{(1-\lambda/\mu)} \quad (1)$$

$$\overline{W(t)} = 1/\mu \left\{ \frac{\lambda/\mu}{(1-\lambda/\mu)} \right\} \quad (2)$$

Amongst these, G/M/1 model is one of most undisciplined and provides situations for calculating queuing parameter's analytically incredibly tough. One can relatively envision that this will be case, which requires bigger value of memory size, thereby, making task a challenging application. However, to get all cases verified, we propose to carry out the study for all the four cases employing stimulation techniques on workstation. Simulation technique is a viable method when problem arises to computer analytically. Simulation techniques for queuing models is possible if arrival instances and departing instances are generated having pre-defined rate of arrivals and rate of departures for a given distribution. After gazing the models, it is mandatory to delineate two types of strings, for the given average rate of arrivals and departures, a string of departure or arrival for pre-assigned average values having negative exponential distribution. The other set of strings having pre-assigned average rate of arrivals/average rate of departure having general distribution. General distribution is a case when there is no distribution for which the string belongs more than, say, 50. The computer for estimation of *Queue Length* and *Waiting time* can employ the two strings. For stable maneuver of the system, it is to be ensured that processing speed of workstation be larger than the arrival rates. This condition when convene, is called ***"functioning of the system in ergodic state"***.

The reminder of this paper is planned as follows. Section 2 gives the background of Computer Simulation study for Memory Size and Waiting time. Section 3 provides the formal description of formal of departure instances at Gateway Server. Section 4 illustrated the simulation study for estimation of Buffer Size and Waiting time for G/M/1 model. Based on the description of Section 2, 3 and 4, Section 5 reviews the Mathematical study of waiting time and Queue Length w.r.t. 4 queuing models. Section 6 abridges our conclusions and offers probable guidelines for prospect work. Appendix [A] gives the source code developed using VC++ for the estimation of Buffer Size for G/M/1 model.

II. Simulation Analysis for Memory Size and Waiting time Estimations

The analytical study for estimation of Queue Length and Waiting time has been accepted out in M/M/1 model. But, when arrival instances are modified to General Distributions, it's very intricate and convoluted to estimate ***Average Queue Length*** and ***Waiting Time***. It becomes uncomplicated with stanch results to compute Queue parameters by simulating the model on Digital Computer. The amount of trials, for arrival instances used to be very large. In real analysis, these ought to be in stipulations of para-instances. Many computers accessible in Personal Computer (PC) variety, cannot be used directly to lodge such large statistics of arrival and departure instances. It is consequently premeditated in the study to engender lower numbers of arriving instances and to have identical quantity of departing instances. The computed Queue Length and Waiting time at lower rate are extrapolated for large quantity of arrival instances. The consequence so acquired can be used for estimating the ***"Buffer Size***".

Customary software's accessible for spawning arrival instances endowed with equiprobable distribution of random number between *0* and *1*. Five sets of random numbers are generated. Then each set is transformed to a desired distributions using under given equations:

1) Conversion of Equiprobable to Negative Exponential distribution

$$y = 1/k(\log(1-x)) \quad (3)$$

where,

y=Negative-Exponential data

x=Equiprobable random between 0 and 1

k= appropriate constant

2) Bernaulli Distribution

The equation for the transformation of equiprobable to Bernaulli distribution ss:

$$f(y(i))=a+by(i) \quad \text{for } i \in 1 \text{ to } N \quad (4)$$

Equiprobable Distribution = x(i)
Bernaulli Distribution = y(i)
then,

$y(i) = -a \pm \sqrt{a^2 + 2bx(i)} \ /b$
where,
$b=1/\lambda$ in arrival process
$b=1/\mu$ in departure process
$a=1-b$

3) Geometric Distribution

$$f(y(i))=a/(1-by(i)) \text{ for } i \in 1 \text{ to } N \quad (5)$$

Equiprobable Distribution= x(i) for i= 1 Є 1 to N
Geometric Distribution = y(i) for i= 1 Є 1 to N
then,
$y(i) = 1/b\ (1-e^{-\delta x(i)})$
where,
$\delta=b/a$
$b=1-a$
$b=\lambda/\mu$ for arrival/departure.

4) Gaussian Distribution

There is varied procedure of generating the Gaussian Distribution. The accepted practice is

by intriguing mean of 12 Equiprobable numbers and arranges them in a String.

Mathematical expression is specified beneath:

Let Equiprobable number are $x_1, x_2, \ldots\ldots\ldots\ldots x_{12}$.

Then, Gaussian distribution

$$y=1/12\left[\sum_{i=1}^{12} x(i)\right] \quad (6)$$

5) Equiprobable distribution

Equiprobable numbers are spawned between 0 and 1 for the identical range using command offered with the system.

III. Formation of Departure States

As epitomized in arrival process, Departure process will also have two elongated strings for departure instances are to be stimulated for:

1) Negative-Exponential Distribution
2) Average rate of departure, $\mu=\lambda+1$

The identical approach is employed, as been depicted in the arrival process. The other string is made of, for $\mu=\lambda+1$, having general distributions with combination of 5 distributions as conferred on arrival process. What is done actually, 5 sets of N/5 numbers are generated having distributions – Bernaulli, Gaussian, Negative-Exponential, Geometric, Equiprobable and are merged randomly.

From these two strings, two departure chains are computed using formulae:

$$D_n = D_{n-1} + d_n \quad (7)$$

where, n=0 to N and d=n^{th} departure duration. To guarantee that departure doesn't take situate before arrival, the departure chain is shifted by 1 second.

IV. Estimation of Queue Length and Waiting Time

In the segment, we have computed *"Queue Length"* and *"Waiting Time"* for G/M/1 model. The source code for the same is developed using VC++. To measure the Queue Length at each departing instances, i.e., $t_1,t_2,\ldots\ldots,t_n$. Let the Queue Size at $t_1,t_2,\ldots\ldots,t_n$ be $q_1,q_2,\ldots\ldots,q_n$.

Then, Average Queue Length

$$\overline{Q(L)} = 1/n\sum_{i=1}^{n} Q(i) \quad (8)$$

$$\overline{W(t)} = 1/\mu[Q(i)] \quad (9)$$

V. Simulation Results

The program developed to compute the value of Queue Length and to estimate the size of Memory at each serving node has been given in Appendix [A].

The arrival rate for the sake of computation conveniences are taken as λ=5000, 7500, 10000, 12500, 15000 and for μ=5001, 7501, 10001, 12501, 15001. The computed results are revealed in the subsequent table:

TABLE I
Estimation of Queue Length

Rate of arrival (λ)	Rate of departure (μ)	Q(L) for G/M/1
5000	5001	5098
7500	7501	7647
10000	10001	10194
12500	12501	12740
15000	15001	15283

The size of memory to provide minimal data flow is given as:

Memory Size:

$$\hat{M} = Q(L)+Variance \quad (10)$$

In case of M/M/1 Queue model,

$$Variance = Q(L) \quad (11)$$

Hence, the memory size,

$$\hat{M} = 2.Q(L) \quad (12)$$

VI. Extrapolation for the Higher Rate of Arrivals

We have calculated Queue Length for 5 rate of arrivals. These 5 points are on the curve of Queue Lengths. Queue Length at very high rate will be also on the curve of Queue Length.

Let the equation of the curve is represented as:

$$y_\phi = a_0 + a_1\lambda + a_2\lambda^2 \quad (13)$$

Now, using Non-linear regression techniques, we have to calculate the value of a_0, a_1, and a_2, for the minimal error. These leads to develop formation equations as given below:

$$\left.\begin{aligned} na_0 + a_1\Sigma\lambda_i + a_2\Sigma\lambda_i^2 &= \Sigma y_i \\ a_0\Sigma\lambda_i + a_1\Sigma\lambda_i^2 + a_2\Sigma\lambda_i^3 &= \Sigma\lambda_i y_i \\ a_0\Sigma\lambda_i^2 + a_1\Sigma\lambda_i^3 + a_2\Sigma\lambda_i^4 &= \Sigma\lambda_i^2 y_i \end{aligned}\right\} \quad (14)$$

There are three unknown variables, i.e., a_0,a_1 and a_2 and three equations for given values of λ.

These values are computed and given as:

$a_0=1484.5$

$a_1=0.873201$

$a_2=0.0000051$

Based on the obtained value,

$$Q(L)= 1484.5 + 0.873201\lambda + 0.0000051\lambda^2 \qquad (14)$$

where, λ is the rate of arrival and Q(L) is the Queue Length.

The results for the Queue Length and hence estimated Memory Size are as given Table II:

TABLE II
COMPUTATION OF QUEUE LENGTH W.R.T G/M/1 MODEL

Rate of arrivals	Q(L) for G/M/1	Memory Size [G/M/1]
10^{10}	6.00008 x 10^{14}	2[6.00008x10^{14}]
10^{11}	6.00008 x 10^{16}	2[6.00008x10^{16}]
10^{12}	6.00008 x 10^{18}	2[6.00008x10^{18}]

VII. CONCLUSION AND FUTURE WORK

We proposed that for Internet server(s), the buffer estimation is contemporary need in crafting the blueprint of future Internet system. The study delineates that for stable functioning of Internet Server, each node of the Internet should work in ergodic environment. The other issue which damages the implementation of Internet system is freezing of data in some node because of inadequate size of memory. In circumstances of multi-channel arrivals and single channel departure, akin to Queue Model, G/M/1 has been studied in facet by simulating queues at lower rate of arrivals. The consequence so attained is extrapolated to compute the size of memory at very towering rate. The outcome demonstrates that we require few hundred terra bytes for arrival rate of 10^{12} bytes. The results endow with procedure for absolute Internet design and its realization in ergodic conditions. Extensive simulation study illustrates pioneering method that can provide smooth performance control and better track in Internet server systems.

With respect of contemporary progress, further research also includes the implementation of Computation Grid, using the same technology.

APPENDIX [A]

Source code developed for G/M/1 Queuing Model using Visual C++ programming language. The code has been effectively experienced on Compaq Presario System (Model M2000) with 256 MB RAM with 1.30 GHz Processor Speed.

```
#include <iostream.h>
#include <fstream.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <common.h>
#include <random.h>
#include <Barnulli.h>
#include <Geometric.h>
#include <equiprobable.h>
#include <exp.h>

int totalnumber=0; // total random numbers
void main()
{
char ch='n';
totalnumber=60000;  //for 5 second
cout<<"Enter the total number:";
cin>>totalnumber;
cout<<"Ready to generate random
number……..."<<endl;
generateRandom();
cout<<"File generated.."<<endl;
cout<<"*****************"<<endl;
cout<<"Arrival started……....."<<endl;
cout<<"*****************"<<endl;
bar_arr();
cout<<"Bar arrival completed"<<endl;
eqp_arr();
cout<<"Eqp. completed"<<endl;
exp_arr();
cout<<"Exponential completed"<<endl;
gaus_arr();
cout<<"Gaussian completed."<<endl;
geo_arr();


cout<<"Geometric completed."<<endl;

//Now departure called
cout<<"****************************************
*************"<<endl;
cout<<"Departure started……"<<endl;
cout<<"****************************************
*************"<<endl;
exp_dep();
cout<<"Exponential  completed."<<endl;
merge_arr();
merge_dep();
cout<<"Files merged…."<<endl;
queue();
}
```

```
//Bernaulli distribution (arrival) function
void bar_arr()
{
int i=0;
double b=0.0, a=0.0, x=0.0, s=0.0, sumy=0.0, arr=0.0;
FILE *fp,*fp1;
b=1.0/double(totalNumber);
a=1-b;
fp1=fopen("bar_arr.xls","w");
clrscr();
for(i=0;i<totalNumber;i++)
{
x=data[i];
arr=(-a+(sqrt(a*a+2*b*x)))/b;;
sumy=sumy+arr;
}
for(i=0;i<totalNumber;i++)
{
x=data[i];
arr=((-a+(sqrt(a*a+2*b*x)))/b)*VALUEARR/sumy;
fprintf(fp1,"1.9lf\n",arr);
}
fclose(fp1);
}

//Equiprobable Distribution (arrival)
void eqp_arr()
{
int i;
double x;
double s=0, sumy=0;arr;
FILE * fp,*fp1;
fp1=fopen("eqp_arr.xls","w");
for(i=0;i<totalNumber;i++)
{
x=data[i];
sumy=sumy+x;
}
for(i=0;i<totalNumber;i++)
{
x=data[i];
arr=x*VALUEARR/sumy;
fprintf(fp1,"%1.9lf\n",arr);
s=s+arr;
}
fclose(fp1);
}

//Exponential Arrival
void exp_arr()
{
int i;
float b=1/3000.0,x,s=0, sumy=0.0,arr;
FILE *fp, *fp1;
fp1=fopen("exp_arr.xls","w");
for(i=0;i<totalNumber;i++)
{
x=data[i];
arr=-(log(1-x))/b;
sumy=sumy+arr;
}
for(i=0;i<totalNumber;i++)
{
x=data[i];
arr=(-(log(1-x))/b)*.2/sumy;
s=a+arr;
fprintf(fp1, "%1.9f\n",arr);
}
fclose(fp1);
}

//Exponential Distribution (departure)
void exp_dep()
{
int i;
float b,x,s,sumy=0.0,a,arr;
FILE *fp,*fp1;
b=1.0/double(totalNumber+1);
fp1=fopen(exp_dep.xls","w");
for(i=0;i<totalNumber+1;i++)
{
x=data[i];
arr=-(log(1-x))b;
sumy=sumy+arr;
}
for(i=0;i<totalNumber+1;i++)
{
x=data[i];
arr=(-(log(1-x))/b)*VALUEDEP/sumy;
s=s+arr;
if(i!=totalNumber)
fprintf(fp1, "%1.9f\n",arr);
}
fclose(fp1);
}
//Gaussian Distribution (Arrival)
void gaus_arr()
{
FILE *fp;
fp=fopen("Gauss_arr.xls","w");
double *rdata;
rdata=new double[totalNumber+12];
double gsum=0,psum=0;
int i;
for(i=0;i<totalNumber;i++)
{
double sum=0;
for(int j=i;j<i+12;j++)
sum+=data[j];
sum/=12;
gsum+=sum;
rdata[i]=sum;
}
for(i=0;i<totalNumber;i++)
{
rdata[i]=rdata[i]*VALUEARR/gsum;
psum+=rdata[i];
fprintf(fp,"1.9f\n",rdata[i]);
```

```
}
fclose(fp);
delete[ ] rdata;
}

//Modified geometric distribution (arrival)
void geo_arr()
{
int i;
double b,a,delta,x,s,sumy=0.0,ar,arr;
FILE *fp,*fp1;
fp1=fopen("Geo_arr.xls","w");
b=1.0/double(totalNumber);
a=1-b;
delta=b/a;
for(i=0;i<totalNumber;i++)
{
x=data[i];
arr=(1-exp(delta*x))/b;
if(arr<0)
arr=(-1)*arr;
sumy=sumy+arr;
}
for(i=0;i<totalNumber;i++)
{
x=data[i];
arr=((1-exp(delta*x))/b))*VALUEARR/sumy;
if(arr<0)
arr=(-1)*arr;
s=s+arr;
fprintf(fp1,"%1.9f\n",arr);
}
fclose(fp1);
}

//Random number Generation
double *data;
extern int totalNumber;
void sort()
{
int i,j;
for(i=0;i<totalNumber;i++)
{
for(j=0;j<i;j++)
{
if(data[i]<data[j])
{
double k=data[i];
data[i]=data[j];
data[j]=k;
}
}
}
}
int find(int s, int n)
{
int i;
for(i=0;i<n;i++)
if(data[i]==s)
return 1;
return 0;
}

void generateRandom()
{
int size=totalNumber+12;
//12 added for gaussian
int i;
data=new double[size];
for(i=0;i<size;i++)
{
        int search=rand();
        if(find(search,i)==1)
        i--;
        else
        data[i]=double(search);
}
sort();
double max=data[0];
for(i=0;i<size;i++)
{
if(data[i]>max)
max=data[i];
}

for(i=0;i<size;i++)
{
data[i]/=max+1;
}
FILE *fp;
fp=fopen("random.xls","w");
for(i=0;i<size;i++)
{
fprintf(fp, "1.9f\n",data[i]);
}
}

//define VALUEARR 2
//define VALUEDEP 2
extern int totalNumber;
int check(char *filename)
{
ifsteam in(filename);
int i,sum=0;
double *arr;
arr=new double[5*totalNumber];
for(i=0;i<5*totalNumber;i++)
{
in>>arr[i];
}
for(i=0;i,5*totalNumber;i++)
{
for(int j=i+1;j<5*totalNumber;j++)
{
if(arr[i]>arr[j])
{
double temp=arr[i];
arr[i]=arr[j];
```

```
arr[j]=temp;
}
}
}
FILE *fp;
fp=fopen("my.xls","w");
for(i=0;i<5*totalNumber;i++)
{
fprintf(fp,"%1.9f\n",arr[i]);
}
fclose(fp);
for(i=0;;)
{
double s=arr[i];
i++;
if(i>=5*totalNumber)
break;
while(1)
{
if(i>=5*totalNumber)
break;
if(s==arr[i])
{
sum++;
i++;
continue;
}
break;
}
}
cout<<"Total="<<sum;
delete[ ] arr;
return 1;
}

void merge()
{
system("type exp_arr.xls>>merge_arr.xls");
system("type bar_arr.xls>>merge_dep.xls");
system("type eqp_arr.xls>>merge_dep.xls");
system("type Gauss_arr.xls>>merge_dep.xls");
system("type Geo_arr.xls>>merge_dep.xls");
system("type exp_dep.xls>>merge_dep.xls");
ifstream in;
int i;
double *arr;
arr=new double[5*totalNumber];
in.open("merge_arr.xls");
for(i=0;i<5*totalNumber;i++)
{
in>>arr[i];
cout<<i<<"."<<arr[i]<<endl;
}
in.close();

for(i=0;i<5*totalNumber;i++)
{
for(int j=i+1;j<5*totalNumber;j++)
{
if(arr[i]>arr[j]
{
double temp=arr[i];
arr[i]=arr[j];
arr[j]=temp;
}
}
}

FILE *fp;
fp=fopen("merge_arr.xls","w");
for(i=0;i<5*totalNumber;i++)
{
fprint(fp,"1.9f\n",arr[i]);
}
fclose(fp);
in.open("merge_dep.xls");
for(i=0;i<5*totalNumber;i++)
{
in>>arr[i];
cout<<i<<"."<<arr[i]<<endl;
}
in.close();
for(i=0;i<5*totalNumber;i++)
{
for(int j=i+1;j<5*totalNumber;j++)
{
if(arr[i]>arr[j]
{
double temp=arr[i];
arr[i]=arr[j];
arr[j]=temp;
}
}
}
fp=fopen("merge_dep.xls","w");
for(i=0;i<5*totalNumber;i++)
{
fprintf(fp,"%1.9lf"\n",arr[i]);
}
fclose(fp);
}

void queue()
{
ifstream in("mrg_dep.xls");
ifstream newin("mrg_arr.xls");
double data_d;
double *arr;
arr=new double[5*totalNumber];
double gsum=0;
int i;
for(i=0;i<5*totalNumber;i++)
{
newin>>arr[i];
}
for(i=0;i<5*totalNumber;i++)
{
double sum=0;
```

```
in>>data_d;
for(int j=0;data_d>arr[j]&&j<5*totalNumber;j++)
{
sum++;
}
if(i==0)
{
cout<<sum<<endl;
}
gsum+=sum;
}
cout<<"sum="<<gsum<<endl;
double qu=double(gsum)/double(4.9*totalNumber);
cout<<"qu="<<qu<<endl;
in.close();
newin.close();
}
void merge_arr()
{
int i;
double x,arr=0.0;
FILE *fp, *fp1,*fp2,*fp3,*fp4,*fp5;
fp1=fopen("bar_arr.xls","r");
fp2=fopen("eqp_arr.xls","r");
fp3=fopen("exp_arr.xls","r");
fp4=fopen("Gauss_arr.xls","r");
fp5=fopen("Geo_arr.xls","r");
fp=fopen("mrg_arr.xls","w");

for(i=0;i<totalNumber;i++)
{
fscanf(fp1,"%lf",&x);
arr=arr+x;
fprintf(fp,"%1.9lf",arr);
}
for(i=0;i<totalNumber;i++)
{
fscanf(fp2,"%lf",&x);
arr=arr+x;
fprintf(fp,"%1.9lf",arr);
}
for(i=0;i<totalNumber;i++)
{
fscanf(fp3,"%lf",&x);
arr=arr+x;
fprintf(fp,"%1.9lf",arr);
}
for(i=0;i<totalNumber;i++)
{
fscanf(fp4,"%lf",&x);
arr=arr+x;
fprintf(fp,"%1.9lf",arr);
}
for(i=0;i<totalNumber;i++)
{
fscanf(fp5,"%lf",&x);
arr=arr+x;
fprintf(fp,"%1.9lf",arr);
}
fclose(fp1);
fclose(fp2);
fclose(fp3);
fclose(fp4);
fclose(fp5);
fclose(fp);
}
void merge_dep()
{
int i;
double x,arr=1.0;
FILE *fp, *fp1,*fp2,*fp3,*fp4,*fp5;
fp3=fopen("exp_dep.xls","r");
fp=fopen("mrg_dep.xls","w");
for(i=0;i<totalNumber;i++)
{
fscanf(fp3,"%lf",&x);
arr=arr+x;
fprintf(fp,%1.9lf",arr);
}
fclose(fp3);
fclose(fp);
}
```

REFERENCES

[1] Riktesh Srivastava ,*"Buffer estimation of Internet Server using Queuing Theory"*Ph.D. Thesis, Dr. R.M.L. Avadh University, Faizabad, India, 2006-07 (Submitted).

[2] UDDI, *"Universal description, discovery, and integration of business for the web,"* 2004.

[3] Mazen Zari, Hossein Saiedain, Muhammed Naeem, *"Understanding and Reducing Web delays"*, pp. 30-37, IEEE Journal for Electronics and Computer Science, Dec. 2001, Vol. 34, No.12.

[4] J. C. Mogul, "Operating systems support for busy internet servers," in *Fifth Workshop on Hot Topics in Operating Systems(HotOS-V)*, Orcas Island, WA, 1995.

[5] P. Barford and M. Crovella, *"Generating representative web workloads for network and server performance evaluation,"* in *Measurement and Modeling of Computer Systems*, 1998, pp. 151–160.

[6] L. Kleinrock, *Queueing Systems, Vol. 2, Applications*. John Wiley, 1976.

[7] D. Menasce and V. Almeida, *Capacity Planning for Web Services: Metrics, Models, and Methods*. Prentice Hall PTR, 2001.

[8] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik, Eds., *Quantitative system performance: computer system analysis using queueing network models*. Prentice-Hall, Inc., 1984.

[9] T. F. Abdelzaher, C. Lu, *"Modeling and Performance Control of Internet Servers, Invited Paper", 39th IEEEConference on Decision and Control*, Sydney, Australia,December 2000.

[10] Yeon Seung Ryu, Kern Koh, *"A Dynamic Buffer Management Technique for Minimizing the Necessary Buffer Space n a Continuous Media Server",* p. 0181, IEEE International Conference on Multimedia Computing and System (ICMCS), 1996.

[11] Darrell Anderson, Ken Yocum, Jeff Chase, *"A Case for Buffer Servers",* p. 82, IEEE Seventh Workshop on Hot Topics in Operating Systems, 1999.

[12] Jim W. Roberts, "Traffic Theory and the Internet", IEEE Communications, January 2001.

[13] Steven H. Low, R. Srikant, *"A Mathematical Framework for Designing a Low-Loss, Low-Delay Internet",* IEEE transaction on Communications, November 29, 2002.

[14] Xiangping Chen, Prasant Mohapatra , *"Performance Evaluation of Service Differentiating Internet Servers",* pp. 1368-1375, Vol. 51, No. 11, 2002.

[15] Lei Ying, G. E. Dullerud and R. Srikant, *"Global Stability of Internet Congestion Controllers with Heterogeneous Delays",* IEEE Transactions on Communications, 2003.

[16] M. Arlitt and C. Williamson, *"Internet Web Servers: Workload Characterization And Performance Implications,"* Networking, IEEE/ACM Transactions on, vol. 5, no. 5, pp. 631-645, Oct. 1997.
[17] D. Dias, et al., *"A Scalable And Highly Available Web Server,"* in *COMPCON '96. Technologies for the Information Superhighway Digest of Papers*., San Jose, CA., 1996, pp. 85-92.
[18] A. Iyengar, et al., *"High-Performance Web Site Design Techniques," IEEE Internet Computing*, vol. 4, no. 2, pp. 17-26, 2000.
[19] L. P. Slothouber, *"A Model of Web Server Performance",*http://www.geocities.com/webserverperformance/modelpaper.html, June, 1995.
[20] P. R. Kumar and S. P. Meyn, *"Stability of Queueing Networks and Scheduling Policies"*, IEEE Transactions on Automatic Control, pp. 251-260, vol. 40, no. 2, February 1995.