> No: 1
22.7.22

Test the Performance of the given web application using Jmeter

Aim:

To test the performance of web application using Jmeter.

Tool Description:

Apache Jmeter is a testing tool used for analysis and measuring the performance of different software servic and products.

Its an open source software for testing web applicatim or FTP application. Used to execute performance testing, load testing to check an app's strength.

Steps for Installation:

1. Download Jmeter from google under libraries dowinload the zip file from google and extract it.
2. Open the file having the bin folder.
3. In the bin folder seled and check Jmeter bat to start
4. Test the applicatim.

Steps for Testing:

1. Intialy choose the thread accordingly between any range and thread group.
2. In the thread group add the http request which request do the browser
3. Place the website link to the server box.
4. we can add listner http request and view the result in different forms such as table, graph, tree, etc..
5. Run the tool and view the result.

No.2
29.7.22

Performance Acceptance testing for a web application using selenium

Aim:

To perform acceptance testing using the tool selenium

Tool Description:

Selenium is a powerful tool for controlling web browser through programs and performing browser communication. It functions for all web browser. works on all major operating system and its scripts are written in python, java etc..

Selenium components are IDE and webbrowser.

Steps for installation:

1. Open google browser and navigate to chrome web store.
2. A window will open up and click "Add to chrome".
3. wait for the download to complete then on "add".
4. Installation is completed. After that click on "OK".
5. Click on selenium icon then selenium will open.

Steps for acceptance testing:

1. Every statement is converted to url.
2. Browser drives uses an Http server to receive requests.

Once accepted, the requests are passed to actual browser.

3. If it post, browsers does an action if get then response is generated and over http.

No: 3
5.8.22

Validate the prediction performance of multi dimensional data using ML on WEKA.

Aim:

To validate the prediction performance of multi dimensional data using machine learning models on WEKA.

Tool Description:

WEKA contains a collection of visualisation tools and algorithms for data analysis and predictive modelling together with graphical user interface for easy access to those functions. It supports several standard like data mining tasks and also specifically data preprocessing, clustering, feature selection and classification.

Steps for Installation:

1. Go to chrome browser and go to weka download page.
2. Click on stable or developer version accordingly to your system configuration.

Steps for validating:

1. Make ready for your own dataset to validate and open WEKA tool.
2. Click the explorer button to launch the explorer
3. Click the select attributes to access feature selection and search methods, which will enable us to choose the correct attributes.
4. Click the preprocess tab to select the attributes.
5. Click the classify tab to open up the classifiers.
6. Under test options, click percentage split and up a percentage.
7. Under filter choose your required model.
8. Now a model's relationship is displayed.

Exp No 4
12.8.22

Analyse TCP UDP packets using wireshark

Aim:

To generate TCP, UDP packet and analyse them using wireshark tool.

Tool description:

Its a free open source tool which is widely used for analyzing. It captures network traffic from ethernet, bluetooth etc.. It captures and stores the data for offline analysis. Also captures network traffic from local networks.

Steps for Installation:

1. Go to chrome and download the wireshark tool according to your systems configuration.

Steps for analysing:

1. Open the wireshark tool
2. The interfaces present in the system will be listed. Select one or more interfaces and press the start capturing the packets.
3. Open and in windows and execute command nping .. tcp-test .. ip 102.168.10.1 ... p-200 - privileged - c5000000 - data length 1 - delay 0 -e -N.
4. similarly we can be generating UDP and ICMP packets using their particular commands.

Analyze packets using wireshark tool.

Exp No: 5

19.8.22

2D Animation

Design an advertisement using any open source 2D animation tool

Aim:

To create an advertisement using any open source 2D animatic tool.

Tool Description:

Blender is a free and opensource 2D creation suite. It supports the features of the 3D pipeline modelling, rigging, animation etc.. It can also be used for 3D.

Installation:

Install from official website or stream.

Satisfy the system requirements

And download the tool according to system OS.

Tool Execution:

1. Press shift+a and add torus from mesh
2. Adjust the minor and major radius
3. Switch form object mode to edit mode
4. Create deformities by using proportional tool.
5. Switch back to object mode and smooth out surface.
6. Switch to edit mode and use the X-ray tool
7. Press shift to create duplicate.
8. Press p to make something a different object.

Exp No: 6
6.8.22

3D Animation

Create a short film using an open source 3D animation tool

Aim:

To create a short film using any open source 3D animation tool.

Tool Description:

Blender is a free open source 3D creation suite. It supports the entirety of 3D pipeline modelling, rigging, animation, simulation, rendering, compositing, motion tracking and video editing.

Installation:

Install from official website.

Download according to system os and Install.

Tool Execution:

1. Create a new file in general mode
2. Select a customed shape and press s to reshape it.
3. Using shift+a add a plane, choose plane from mesh
4. Make sure the cuboid is above the newly added plane
5. Add multiple cubes by pressing shift+a and select.
6. To move, select cuboid and click i, set the location, timing, and more.
7. Make changes in cubes to animate the surface.

Exp No: 7
2/9/22

Create and manage users, group and their access rights by using DCL commands.

Aim:

To create and manage users, groups and their access rights by using DCL commands. (ie Data control language). Also, to test the data integrity and consistency with on Delete / null Options.

Tool Description:

To execute DCL commands as well as to test the data integrity and use mySQL. mySQL is a relational database management system developed by oracle corporation. Its licenses are GNU free / libre.

Tool Installation:

1. Download mySQL installer and execute it.
2. Install the server instance and configure it.
3. After installation it gets opened automatically.

Tool Execution:

(a) DCL commands: GRANT and REVOKE

1. Create user example @ 'localhost' identified by 'mypass'
2. Grant all on my-Table to example @ localhost.
3. Grant select on users to example @ localhost.
4. syntax: Grant select, update on my_table to user1, user2;
5. Revoke select, update on my-table from user1, user2;

(b) Data Integrity and Consistency:

on Delete cascade and on Delete set null.

1. Syntax for on Delete cascade: Constraint on Delete cascade.

2. Syntax for on Delete set null:

Constraint

on Delete set null.

The on delete cascade is used to automatically remove and matching records from the child table when we delete the rows from parent table. on delete set null is used to set the foreign key columns to null.

Open mp

Develop parallel program to perform matrix multiplication using open mp constructs.

Aim:

To develop a parallel program for matrix multiplication using ope mp constructs.

Tool Description:

MXM_OpenMP, a code which sets up a dense matrix multiplication problem C = A*B, using ope mp for parallel execution. The matrices A and B are chosen so that C = (N+1) * I, where N is the order of I is the identity.

Algorithm:

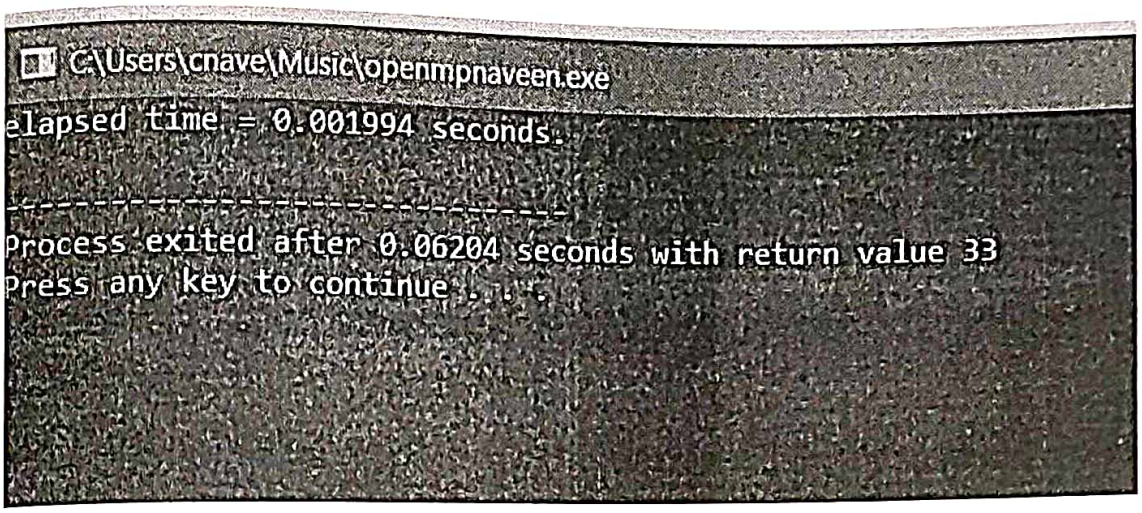
1. Define the number of rows of A, B, C matrix
2. Specify the number of chunks.
3. Compare thread id to identify master thread.
4. Input matrix A, B and store.
5. For iterative computation of each matrix, indicate a new thread.
6. Print the matrix as result.

PROGRAM:

```
#include
<pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#include
<sys/time.h>
#define N 100
int A[N][N];
int B[N][N];
int C[N][N];

int main()
{
  int i,j,k;
  struct timeval tv1,
  tv2; struct timezone
  tz;
        double elapsed;
 omp_set_num_threads(omp_get_num_proc
 s()); for (i= 0; i< N; i++)
    for (j= 0; j< N; j++)
       {
       A[i][j] = 2;
       B[i][j] = 5;
       }
  gettimeofday(&tv1, &tz);
#pragma omp parallel for private(i,j,k)
 shared(A,B,C) for (i = 0; i < N; ++i) {
    for (j = 0; j < N; ++j) {
       for (k = 0; k < N; ++k) {
```

```
            C[i][j] += A[i][k] * B[k][j];
        }
      }
    }
    gettimeofday(&tv2, &tz);
    elapsed = (double) (tv2.tv_sec-tv1.tv_sec) + (double) (tv2.tv_usec-tv1.tv_usec) *
    1.e-6; printf("elapsed time = %f seconds.\n", elapsed);
}
```

OUTPUT:



RESULT:

Thus the parallel program to perform matrix multiplication using openmp constructs has been designed.

Exp No: 9
16.9.22

Analyse the performance of connectim, connectimless transport protocols in a simulatim environment usig NS3

Aim:

To analyze the performance of connectim, connectim-less transport protocols in a simulatim environment using NS3.

Tool Description:

Setting up a network to do some real experiments is the best way for studying about communicatim in internet. However setting up a network is expensive and not easy. For this reason, a virtual network provided by network stimulator is used for experimenting NS3 is free and easy to use and popular all over the world.

Tool Installation:

1. Install Ubuntu OS, network animatim
2. Instal NAM - sudo apt-get install -y nam

Install NS2 - sudo-apt get install-y ns2

3. Declare stimulatim and setting output file.
4. setting node and link
5. setting agent
6. For UDP, we use UDP agent for sender and receiver set to null agent.
7. For TCP, we set sender agent as TCP agent and receiver as TCP using agent.

Setting UP:

1. UDP uses CBR applicatim while TCP uses FTP applicatim.
2. Setting time schedule for stimulatim.
3. Declare Finish.

Execution:

1. To run the test file, "file_name.tcl" is listed.
2. After simulatim, NAM gets started. If it does not run., 'non out num' is entered.

NS2 – UDP

```
set ns [new Simulator]

set tr [ open "out.tr" w]
$ns trace-all $tr

set ftr [open "out.nam" w]
$ns namtrace-all $ftr

set n0 [$ns node]
set n1 [$ns node]

$ns duplex-link $n0 $n1 2Mb 4ms DropTail

set udp1 [new Agent/UDP]

set sink [new Agent/UDPSink]

$ns attach-agent $n0 $udp1
$ns attach-agent $n1 $sink

$ns connect $udp1 $sink

set ftp [new Application/FTP]

$ftp attach-agent $udp1

proc finish { } {
        global ns tr ftr
        $ns flush-trace
        close $tr
        close $ftr
        exec nam out.nam &
        exit
        }

$ns at .1 "$ftp start"
$ns at 2.0 "$ftp stop"

$ns at 2.1 "finish"
```

NS2-TCP

```
set ns [new Simulator]

set tr [ open "out.tr" w]
$ns trace-all $tr

set ftr [open "out.nam" w]
$ns namtrace-all $ftr

set n0 [$ns node]
set n1 [$ns node]

$ns duplex-link $n0 $n1 2Mb 4ms DropTail

set tcp1 [new Agent/TCP]

set sink [new Agent/TCPSink]

$ns attach-agent $n0 $tcp1
$ns attach-agent $n1 $sink

$ns connect $tcp1 $sink

set ftp [new Application/FTP]

$ftp attach-agent $tcp1

proc finish { } {
        global ns tr ftr
        $ns flush-trace
        close $tr
        close $ftr
        exec nam out.nam &
        exit
        }

$ns at .1 "$ftp start"
$ns at 2.0 "$ftp stop"

$ns at 2.1 "finish"
```

Develop parallel program for all pairs, shortest path problem using MPI library

Aim:

To develop a parallel program for all pairs of shortest path using MPI library.

Tool Description:

This is a parallel implementation of Dijkstra's / Floyd shortest path algorithm for a weighted directed graph given as an adjacency matrix. It finds the shortest path from node to every other vertex.

Algorithm:

1. Read the matrices which gives the adjacency of vertices
2. Scatter the process
3. Block row distribution among the process
4. Convert a row of the matrix to a string and then print the string.
5. Implement distribution of Floyd / Dijkstra algorithm.
6. Find the shortest path between all pairs of vertices.
7. Display the result.

PROGRAM:

```
#include
<stdio.h>
#include
<stdlib.h>
#include <string.h> /* for debugging
*/ #include <mpi.h>

const int INFINITY = 1000000;

void Read_matrix(int local_mat[], int n, int my_rank, int p, MPI_Comm comm);

void Print_matrix(int local_mat[], int n, int my_rank, int p, MPI_Comm comm);

void Floyd(int local_mat[], int n, int my_rank, int p, MPI_Comm comm);

int Owner(int k, int p, int n);

void Copy_row(int local_mat[], int n, int p, int row_k[], int k);

void Print_row(int local_mat[], int n, int my_rank, int i);

int main(int argc, char* argv[])
{
int n;
int* local_mat;
MPI_Comm
comm; int p,
my_rank;

MPI_Init(&argc,  &argv);
comm =
MPI_COMM_WORLD;
```

```
MPI_Comm_size(comm, &p);
MPI_Comm_rank(comm,
&my_rank); if (my_rank == 0)
{
printf("How many
vertices?\n"); scanf("%d",
&n);
}
MPI_Bcast(&n, 1, MPI_INT, 0,
comm);          local_mat          =
malloc(n*n/p*sizeof(int));          if
(my_rank == 0)
  printf("Enter the local_matrix\n");
Read_matrix(local_mat, n, my_rank, p,
comm); if (my_rank == 0)
  printf("We got\n");
Print_matrix(local_mat, n, my_rank, p,
comm); if (my_rank == 0)
  printf("\n");
Floyd(local_mat, n, my_rank, p,
comm); if (my_rank == 0)
  printf("The solution is:\n");

Print_matrix(local_mat, n, my_rank, p,
comm); free(local_mat);
MPI_Finalize(
); return 0;

}
 /* main */
```

```
void Read_matrix(int local_mat[], int n, int my_rank, int p, MPI_Comm comm)
{
int i, j;
int* temp_mat =
NULL; if (my_rank ==
0)
{
temp_mat = malloc(n*n*sizeof(int));

for (i = 0; i < n;
 i++) for (j = 0; j <
 n; j++)
    scanf("%d", &temp_mat[i*n+j]);

MPI_Scatter(temp_mat, n*n/p, MPI_INT, local_mat, n*n/p, MPI_INT, 0,
comm); free(temp_mat);
}
else
{
MPI_Scatter(temp_mat, n*n/p,
MPI_INT, local_mat, n*n/p, MPI_INT,
0, comm);
}
}
/* Read_matrix */
void Print_row(int local_mat[], int n, int my_rank, int i)
{
 char char_int[100];
 char
 char_row[1000]; int
 j, offset = 0;

 for (j = 0; j < n; j++)
```

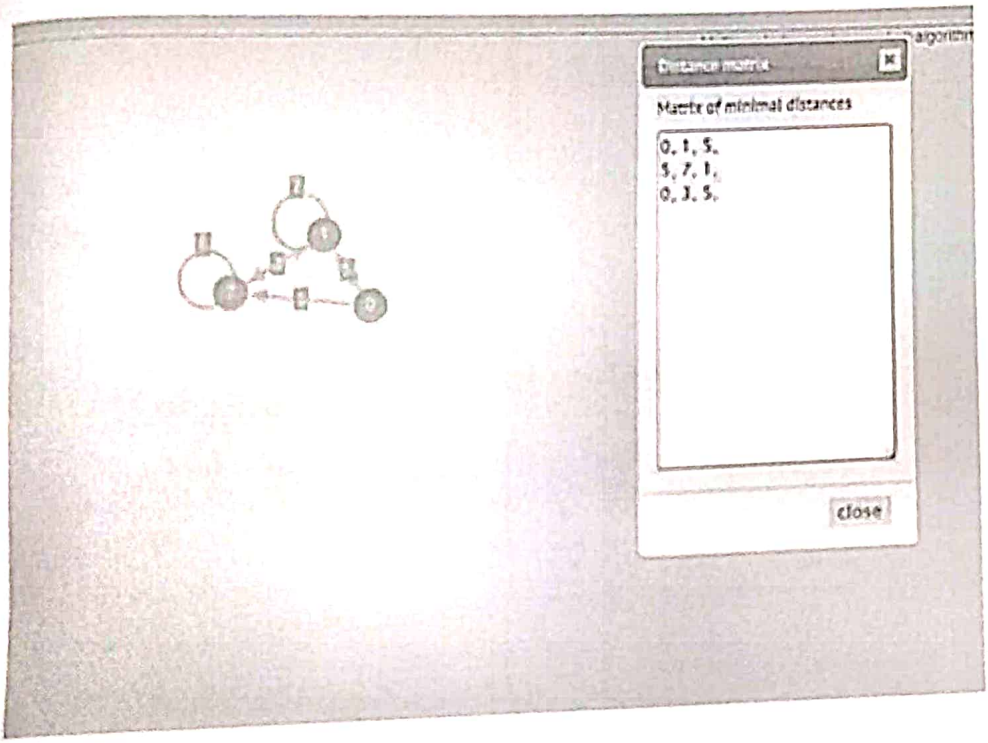
```
{
if (local_mat[i*n + j] ==
  INFINITY) sprintf(char_int, "i
  ");
else
  sprintf(char_int, "%d ", local_mat[i*n +
  j]);  sprintf(char_row  +  offset,  "%s",
  char_int); offset += strlen(char_int);
}

printf("Proc %d > row %d = %s\n", my_rank, i, char_row);


}
/* Print_row */

void Print_matrix(int local_mat[], int n, int my_rank, int p, MPI_Comm comm)
{
 int i, j;
 int* temp_mat = NULL;

 if (my_rank == 0)
 {
 temp_mat = malloc(n*n*sizeof(int));
 MPI_Gather(local_mat, n*n/p,
 MPI_INT, temp_mat, n*n/p,
 MPI_INT, 0, comm); for (i = 0; i < n;
 i++)
 {
 for (j = 0; j < n; j++)
 if (temp_mat[i*n+j] ==
 INFINITY) printf("i ");
```

```
else
printf("%d ",
temp_mat[i*n+j]); printf("\n");
}
free(temp_mat);
}
else
{
MPI_Gather(local_mat, n*n/p,
MPI_INT, temp_mat, n*n/p,
MPI_INT, 0, comm);
}
}

/* Print_matrix */
void Floyd(int local_mat[], int n, int my_rank, int p, MPI_Comm comm)
{
int global_k, local_i, global_j,
temp; int root;
int* row_k = malloc(n*sizeof(int));
for (global_k = 0; global_k < n; global_k++)
{
root = Owner(global_k, p,
n); if (my_rank == root)
Copy_row(local_mat, n, p, row_k,
global_k); MPI_Bcast(row_k, n,
MPI_INT, root, comm); for (local_i = 0;
local_i < n/p; local_i++)
for (global_j = 0; global_j < n; global_j++)
{
temp = local_mat[local_i*n + global_k] + row_k[global_j];
```

```
if (temp <
local_mat[local_i*n+global_j])
local_mat[local_i*n + global_j] =
temp;
}
}
free(row_k);
}

/* Floyd */
int Owner(int k, int p, int n)
{
 return k/(n/p);
} /* Owner */
void Copy_row(int local_mat[], int n, int p, int row_k[], int k)
{
 int j;
 int local_k = k %
 (n/p); for (j = 0; j < n;
 j++)
 row_k[j] = local_mat[local_k*n + j];
} /* Copy_row */
```

OUTPUT:

```
How many vertices?
3
Enter the local_matrix
0 1 5
5 7 1
0 3 5
We got
0 1 5
5 7 1
0 3 5
```



RESULT:

Hence the program for all pairs shortest path using MPI has been designed.

Minix OS

Aim:

To demonstrate Minix OS.

Tool Description:

Minix is a free open source OS designed to be highly reliable and secure. It is based on a tiny microkernel running in kernel mode with the rest of OS running as a number of isolated, protected process as in user mode.

Tool Installation:

1. Download the CD-Rom image (ie iso file)
2. Install and mount the ISO file to a CD-Rom and then boot the computer from it. (If installing in PC)
3. Configure virtual machine to use the ISO file as its CD-Rom and then then boot it (if installing in Vm)
4. Login as root and type "select".

Minix commands:

1. which - find the absolute path of each of commands.
2. cd - change directory
3. cp - copy files from one location to another.
4. cat - concatenate.
5. du - display the number of blocks used for files.
6. df - display the amount of disk space available.
7. Find - walking a file hierachy
8. ls - list files
9. more - read files and display the text one screen at a time.
10. ps - check the status of active processes
11. rm - remove files you no longer need.
12. shutdown - shutsdown the system.

Shutdow -r : reboot

Shoutdown -h : halt

shutdow -d : powerbtf.

Result:

Thus, Minix OS is successfully implemented.