6CS005 Learning Journal - Semester 1 2019/20

Name: Pranish Acharya

UV\_ID: 2038527

# Table of Contents

[Table of Contents 1](#_Toc20306348)

[1 Parallel and Distributed Systems 2](#_Toc20306349)

[1.1 Answer of First Question 2](#_Toc20306350)

[1.2 Answer of Second Question 2](#_Toc20306351)

[1.3 Answer of Third Question 2](#_Toc20306352)

[1.4 Answer of Fourth Question 2](#_Toc20306353)

[1.5 Answer of Fifth Question 2](#_Toc20306354)

[1.6 Answer of Sixth Question 2](#_Toc20306355)

[2 Applications of Matrix Multiplication and Password Cracking using HPC-based CPU system 2](#_Toc20306356)

[2.1 Single Thread Matrix Multiplication 2](#_Toc20306357)

[2.2 Multithreaded Matrix Multiplication 3](#_Toc20306358)

[2.3 Password cracking using POSIX Threads 3](#_Toc20306359)

[3 Applications of Password Cracking and Image Blurring using HPC-based CUDA System 3](#_Toc20306360)

[3.1 Password Cracking using CUDA 3](#_Toc20306361)

[3.2 Image blur using multi dimension Gaussian matrices 3](#_Toc20306362)

# Parallel and Distributed Systems

## Answer of First Question

Threads are the independent stream of instructions that can be scheduled to run by OS. Threads allow application to perform multiple tasks concurrently. Threads allow to perform multiple computations parallelly.

## Answer of Second Question

The two process scheduling policies are :

1. Pre-emptive: The schedular is in charge of how long a process runs for. If a process exceeds its time slice, it is stopped by the schedular.
2. Co-operative: Each process is in-charge of how long it runs for. When a process feels like co-operating, it will surrender execution.

Pre-emptive is preferable as every process gets their equal execution time. The choice of policies may alter the execution time period of the Java threads.

## Answer of Third Question

1. Centralized Systems
2. Distributed Systems

1.

Centralized Systems uses client-server architecture where one or more clients nodes are directly connected to the central server.

1.

Distributed Systems uses peer-to-peer architecture where every nodes make its own decision. The final behavior of the system is the aggregate of the decision of every nodes.

2.

This system consists of one component with non-autonomous parts.

2.

This system consists of multiple autonomous components.

3.

In this system, all the resources are accessible.

3.

In this system, resources may not be accessible.

4.

Components are shared by users all the time.

4.

Components are not shared by all the users.

## Answer of Fourth Question

Transparency in Distributed System is defined to hide something. Transparency is an important issue to realize the single system image which makes systems as easy to use as a single processor system.

Classification of the Transparency:

1. Access Transparency: Data and resources can be used in a consistent way.
2. Location Transparency: A user cannot tell where resources are located.
3. Migration Transparency: Resources can move at will without changing their names.
4. Concurrency Transparency: Multiple users can share resource automatically.
5. Failure Transparency: A user does not notice resource failure.
6. Performance Transparency: Systems are reconfigured to improve performance as loads vary.
7. Scaling Transparency: Systems can expand in size without changing the system structure and the application programs.

## Answer of Fifth Question

Include your code using a text file in the submitted zipped file under name Task1.5

## Answer of Sixth Question

# Applications of Matrix Multiplication and Password Cracking using HPC-based CPU system

## Single Thread Matrix Multiplication

* The analysis of the algorithm’s complexity. (1 mark)
* Suggest at least three different ways to speed up the matrix multiplication algorithm given here. (Pay special attention to the utilisation of cache memory to achieve the intended speed up). (1 marks)
* Write your improved algorithms as pseudo-codes using any editor. Also, provide reasoning as to why you think the suggested algorithm is an improvement over the given algorithm. (1 marks)

Paste your algorithm’s pseudo code here

* Write a C program that implements matrix multiplication using both the loop as given above and the improved versions that you have written. (1marks)

Include your code using a text file in the submitted zipped file under name Task2.1

* Measure the timing performance of these implemented algorithms. Record your observations. (Remember to use large values of N, M and P – the matrix dimensions when doing this task). (1 marks)

Insert a paragraph that hypothesises how long it would take to run the original and improved algorithms. Include your calculations.

Explain your results of running time.

## Multithreaded Matrix Multiplication

* Include your code using a text file in the submitted zipped file under name Task2.2
* Insert a table that has columns containing running times for the original program and your multithread version. Mean running times should be included at the bottom of the columns.
* Insert an explanation of the results presented in the above table.

## Password cracking using POSIX Threads

* Include your code using a text file in the submitted zipped file under name Task2.3.1, Task2.3.3, Task2.3.5
* Insert a table of 10 running times and the mean running time.
* Insert a paragraph that hypothesises how long it would take to run if the number of initials were to be increased to 3. Include your calculations.
* Explain your results of running your 3 initial password cracker with relation to your earlier hypothesis.
* Write a paragraph that compares the original results with those of your multithread password cracker.

# Applications of Password Cracking and Image Blurring using HPC-based CUDA System

## Password Cracking using CUDA

* Include your code using a text file in the submitted zipped file under name Task3.1
* Insert a table that shows running times for the original and CUDA versions.
* Write a short analysis of the results

## Image blur using multi dimension Gaussian matrices

* Include your code using a text file in the submitted zipped file under name Task3.2
* Insert a table that shows running times for the original and CUDA versions.
* Write a short analysis of the results