**INTERPROCESS COMMUNICATION COMPARISON**

**CS-220 OPERATING SYSTEMS**

**GROUP MEMBERS:**

19K-0206 Muhammad Taha Tasleem.

19k-0125 Affan Siddiqui.

19k-1287 Sharjeel Khan.

**Section:**

4J

# **INTRODUCTION:**

Inter-process Communication (IPC) is the mechanism provided by the Operating System, that allows process to communicate with each other.

This communication could involve a process letting another process know that some event has occurred or the transferring of data from one process to another.

There are various ways through which Inter-process communication can be implemented. Some of its mechanisms are Pipes, Shared Memory, Message Queues and etc.

# **DESCRIPTION:**

In this project we have implemented 3 problems using 2 methods of Interprocess Communication. The 3 problems are Fibonacci Series, Bubble Sorting and Matrix Multiplication. The 2 methods that we have used are Pipes and Shared Memory.

# **OBJECTIVE:**

The objective of the project is to compare the two different Inter-process communication with each other and determine the better inter-process communication in times of efficiency, when applied to different problems.

# **REQUIREMENTS AND SETTINGS:**

***Development Environment:*** Linux.

***Language Used:*** “C”.

***Libraries Used:*** time.h, shm.h, stdio.h, ipc.h, sys/wait.h, sys.types.h, and etc.

# **PROCESS EXPLANATION:**

As we have discussed earlier, three problems are implemented. Eight different files are made in terms of 2 files for each problem using shared memory problem and one file for each problem using pipe. In shared memory, one file is made for client, while the other is made for the server.

Each problem is implemented with pipes in which one pipe is writing so other can only read and vice versa. Fork function is created which divides the work in two different process, one for parent and other for children.

After that, the problem is implemented with shared memory in which the memory is attached from client side to server side and process is performed.

# **COMPARISON:**

## **FIBONACCI SERIES:**

## USING PIPES:

Time for Parent Process: 0.000130

Time for Child Process: 0.000194

Total Time: 0.000328

## USING SHARED MEMORY:

Time for Parent Process: 0.000119

Time for Child Process: 0.000138

Total Time: 0.000257

## **SORTING:**

## USING PIPES:

Time for Parent Process: 0.000731

Time for Child Process: 0.0001028

Total Time: 0.0008338

## USING SHARED MEMORY:

Time for Parent Process: 0.00099

Time for Child Process: 0.000153

Total Time: 0.001143

## **MATRIX MULTIPLICATION:**

## USING PIPES:

Time for Parent Process: 0.000282

Time for Child Process: 0.001035

Total Time: 0.001317

## USING SHARED MEMORY:

Time for Parent Process: 0.000111

Time for Child Process: 0.000135

Total Time: 0.000246

# **RESULT:**

The result shows that the any problem done with pipes was slower more than 5 times as compared with shared memory. So, we can say that shared memory is better and faster than pipes.

# **GRAPH:**

As we can see from graph as well, shared memory is much faster and efficient as compared to Pipes.

# **SCREENSHOTS:**

