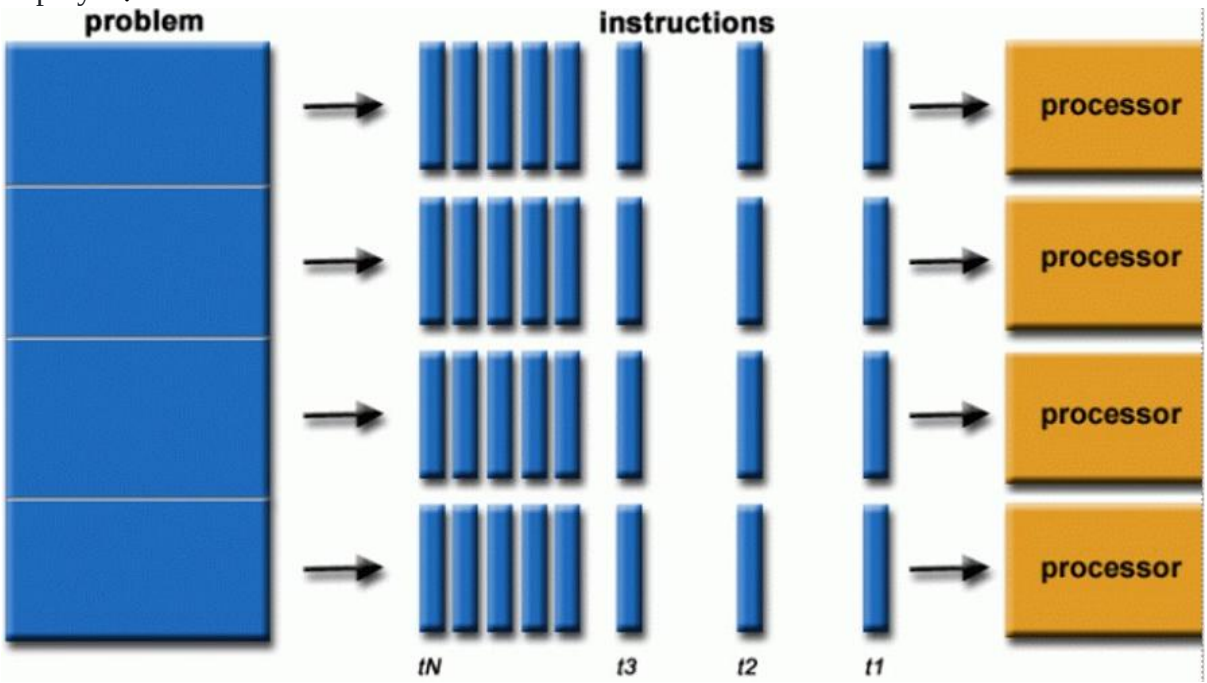
Experiment No 9

Aim: Implement Programs using parallel programming paradigm

Theory:

Parallel Programming

Parallel Programming is the process of decomposing a problem into smaller tasks that can be executed at the same time using multiple compute resources.It is used interchangeable with parallel processing or in conjunction with parallel computing, which refers to the systems that enable the high efficiency of parallel programming.In parallel programming tasks are parallelized so that they can be run at the same time by using multiple computers or multiple cores within a CPU. In the simplest sense, parallel computing is the simultaneous use of multiple compute resources to solve a computational problem.A problem is broken into discrete parts that can be solved concurrently.Each part is further broken down to a series of instructions.Instructions from each part execute simultaneously on different processors.An overall control/coordination mechanism is employed.



Inter Process Communication (IPC)

A process can be of two types:

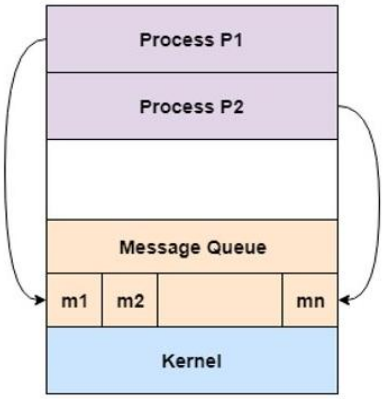
- Independent process.
- Co-operating process.

An independent process is not affected by the execution of other processes while a co-operating process can be affected by other executing processes. Though one can think that those processes, which are running independently, will execute very efficiently, in reality, there are many situations when co-operative nature can be utilized for increasing computational speed, convenience, and modularity. Inter-process communication (IPC) is a mechanism that allows processes to communicate with each other and synchronize their actions. The communication between these processes can be seen as a method of co-operation between them. Processes can communicate with each other through both:

1. Shared Memory
2. Message passing
3.

Message Passing:-

Message passing model and shared memory model are models of inter-process communication. Message passing model allows multiple processes to read and write data to the message queue without being connected to each other.Messages are stored on the queue until their recipient retrieves them. Message queues are quite useful for inter-process communication and are used by most operating systems.

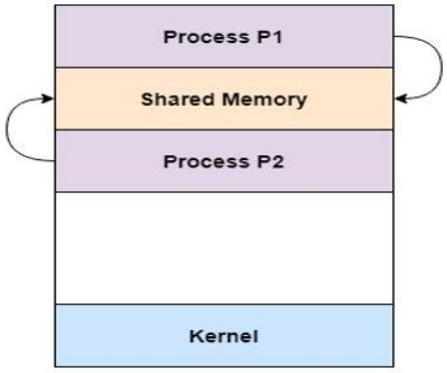


Message Passing Model

In the above diagram, both the processes P1 and P2 can access the message queue and store and retrieve data.An advantage of message passing model is that it is easier to build parallel hardware.This is because message passing model is quite tolerant of higher communication latencies. It is also much easier to implement than the shared memory model.However, the message passing model has slower communication than the shared memory model because the connection setup takes time.

Shared Memory Process Communication Model

The shared memory in the shared memory model is the memory that can be simultaneously accessed by multiple processes. This is done so that the processes can communicate with each other.



Shared Memory Model

In the above diagram, the shared memory can be accessed by Process 1 and Process 2.An advantage of shared memory model is that memory communication is faster as compared to the message passing model on the same machine.However, shared memory model may create problems such as synchronization and memory protection that need to be addressed.

Example: Producer Consumer Problem

Step 1 − A **producer process produces information** that is **consumed by the consumer process.** For example, a compiler that produces assembly code which is consumed by an assembler. The assembler can produce object modules which are consumed by the loader.
Step 2 − It uses the **shared memory** concept.
Step 3 − If we want to allow producer and consumer processes to run concurrently, we have to provide a buffer of items which can be filled by producer and empty the buffer by consumer.
Step 4 − this buffer will reside in a region of memory which is shared by producer and consumer process.
Step 5 − A producer can produce one item while the consumer is consuming another item.
Step 6 − the producer and consumer must be synchronised, so that the consumer does not try to consume an item which is not yet produced.

Socket Programming
It helps us to connect a client to a server. Client is message sender and receiver and server is just a listener that works on data sent by client.
What is a Thread?
A thread is a light-weight process that does not require much memory overhead, they are cheaper than processes.
What is Multi-threading Socket Programming?
Multithreading is a process of executing multiple threads simultaneously in a single process.

Lab Assignments to complete in this session

1) Implement Solution to the producer consumer problem using semaphore

2)Implement client server communication using socket programming