RV College of Engineering®

Experiential Learning Phase - I :

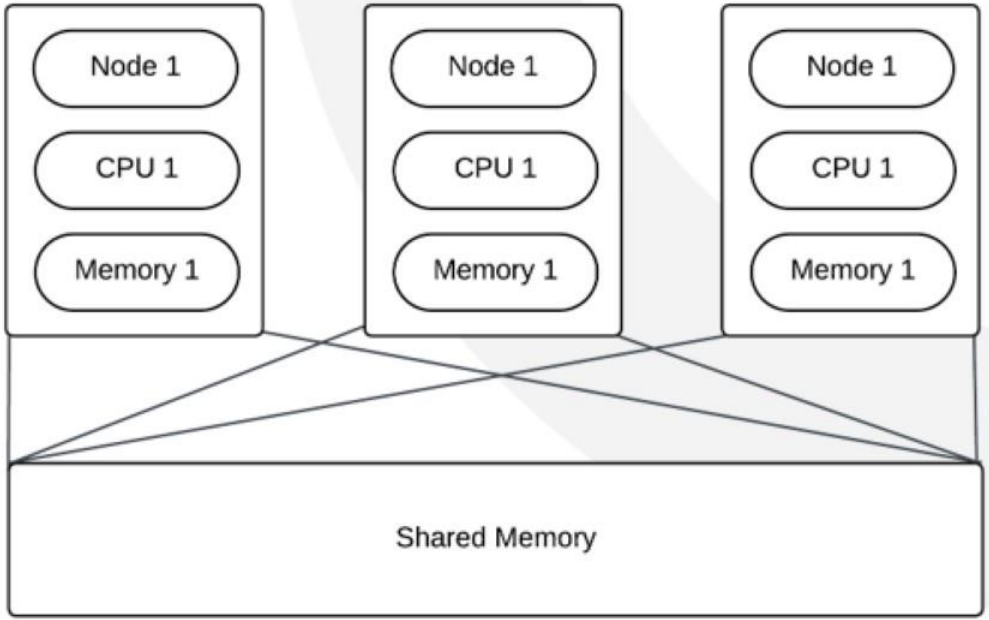
Operating Systems
CS235AI

Distributed Shared Memory

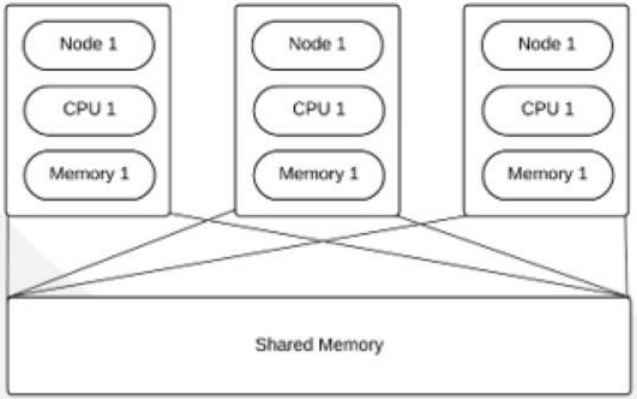
ARYA HARIHARAN (1RV22CS029)
PROFESSOR – Dr. PAVITHRA H

Go, change the world®

INTRODUCTION



- A **cluster of computers** are present, each with its **own physical memory**.
- Through DSM, each computer/node can also access a **shared memory space**.
- It creates the **illusion of a single, coherent memory space** across the network, allowing access to shared data as if it were local.



Distributed shared memory is a service that manages memory across multiple nodes so that applications will have the illusion that they are running on a single shared-memory machine.

PROBLEM STATEMENT

To simulate Distributed Shared Memory (DSM) architecture by implementing the central server algorithm using sockets in Python

OBJECTIVES

- Simulate the working of a distributed shared memory system using socket programming in Python.
- Implement the central server DSM algorithm.
- Clearly define read and write accesses for different clients.

RELEVANCE TO COURSE

This project involves several concepts that are core OS concepts or are deeply interconnected to the OS course –

- It simulates a memory management technique, generally applied among various nodes accessing the same data.
- It employs sockets to simulate the client and server processes.
- It uses threads to process the requests of each client.
- It uses mutex locks to control writing to resources.

RV College of Engineering®

METHODOLOGY

Go, change the world®

1

Create Sockets

Create two different .py files, one for defining server behaviour and the other, for client behaviour. Create a server socket and client socket in the programs respectively using the socket library. Perform the necessary connection and binding.

2

Set Up the Communication channel

Using the in-built functions in Python, set up communication between the client sockets and server sockets. Define exit conditions and entry channels.

3

Define Different Accesses

Clearly define the different accesses and allowances for different modes such as read and write. Create threads for each client to process the requests

4

Initialize Page Table and Mutex Locks for Each File for Writing

Maintain a page table using a two dimensional list to maintain a list of clients, files it has access to and what operation it can perform. Mutex locks have to be implemented to ensure only one client can write at a time.

METHODOLOGY

5

Create the Files for Sharing

Implement a number of files present in the shared memory space. These files can be written by only one client at a time, but can be read by multiple clients at a time.

6

Run the Simulation

Run the simulation. Ensure edge case conflicts are rectified to ensure the integrity of data, such as multiple write, read-write at the same time, etc.

RV College of Engineering®

TOOLS/API

Some of the tools and APIs that will be used in this project are -

- Python
- socket API in Python
- thread API in Python

RV College of Engineering®

WHY DSM?

1

Simplified Parallel Programming

A DSM system allows extending these familiar models (like pthreads and OpenMP) to a distributed environment, simplifying the transition to distributed computing.

2

Utilization of Cluster Resources

Clusters of machines offer a cost-effective way to increase computing power and resources.

3

Resource Efficiency

DSM systems can help in efficient resource utilization.

4

Load Balancing and Scalability

A DSM system can intelligently distribute memory pages across machines, achieving load balancing and improving scalability.

RV College of Engineering®

APPLICATIONS

Computationally Intensive Tasks

- A scientific research project requires running **computationally intensive simulations or numerical computations**.
- These simulations often **involve complex mathematical models** and require **significant computational resources** to execute.
- Computational workload can be **distributed across multiple nodes** or clusters.
- To **coordinate data access** and **sharing of intermediate results** between distributed nodes, DSM can be employed.

RV College of Engineering®

APPLICATIONS

Computationally Intensive Tasks

- Lighting and illumination effects in virtual environments
- Simulating **the propagation of light rays through a scene**, accounting for factors such as reflection, refraction, and scattering
- Typically involves **ray tracing or radiosity methods**
- DSM can aid in **distributing the computational workload** across multiple nodes
- Distribution and sharing of scene data using DSM

REFERENCES

1. Nelson J, Holt B, Myers B, Briggs P, Ceze L, Kahan S, Oskin M. {Latency-Tolerant} software distributed shared memory. In2015 USENIX Annual Technical Conference (USENIX ATC 15) 2015 (pp. 291-305).
2. El-Ghazawi, Tarek, William Carlson, Thomas Sterling, and Katherine Yelick. *UPC: distributed shared memory programming*. John Wiley & Sons, 2005.
3. Wu J. Distributed system design. CRC press; 2017 Dec 14.
4. B. Nitzberg and V. Lo, "Distributed shared memory: a survey of issues and algorithms," in Computer, vol. 24, no. 8, pp. 52-60, Aug. 1991, doi: 10.1109/2.84877.
5. M. Stumm and S. Zhou, "Algorithms Implementing Distributed Shared Memory", *Computer*, vol. 23, no. 5, pp. 54-64, May 1990.