Computer Architecture: Assignment 2
Name: Shayeri Das
CSE(A) 1551009

1. Write short notes on Amdahl's law and Gustafson's Law.

Ans: Amdahl's Law

In Amdahl's law, computational workload W is fixed while the number of processors that can work on W can be increased.
Denote the execution rate of i processors as R_i, then in a relative comparison they can be simplified as $R_1 = 1$ and $R_n = n$. The workload is also simplified. We assume that the workload consists of sequential work αW and n parallel work $(1-\alpha)W$ where α is between 0 and 1. More specifically, this workload can be written in a vector form as, $W = (\alpha, 0, \dots 0, \alpha - 1) W$, or $W_1 = \alpha W$, $W_n = (1-\alpha)W$, and $W_i = 0$ for all $i \neq 1, n$.
The execution time of the given work by n processors is then computed as,

$$T_n = \frac{W_1}{R_1} + \frac{W_n}{R_n}$$

Speedup of n processor system is defined using a ratio of execution time, i.e.,

$$S_n = \frac{T_1}{T_n}$$

Substituting the execution time in relation W gives,

$$S_n = \frac{W/1}{\frac{\alpha W}{1} + \frac{(1-\alpha)W}{n}} = \frac{n}{1+(n-1)\alpha} \quad \dots (1)$$

Eq.(1) is called the Amdahl's Law. If the number of processors is increased to infinity, the speedup becomes,

$$S_\infty = \frac{1}{\alpha} \quad \dots (2)$$

Notice that the speedup can NOT be increased to infinity even if the number of processors is increased to infinity. Therefore, Eq.(2) is referred to as a sequential bottleneck of multiprocessor systems.

Gustafson's Law

This law says that increase of problem size for large machines can retain scalability with respect to the number of processors.
Assume that the workload is scaled up on an n-node machine as,

$$W' = \alpha W + (1-\alpha) n W$$

Speedup for the scaled up workload is then,

$$S'_n = \frac{\text{Single Processor Execution Time}}{n\text{- Processor Execution Time}}$$

$$S'_n = \frac{(\alpha W + (1-\alpha) n W)/1}{\frac{\alpha W}{1} + \frac{(1-\alpha) n W}{n}} \quad \dots\dots (3)$$

Simplifying Eq.(3) produces the Gustafson

$$S_n' = \alpha + (1-\alpha) n \quad \dots (4)$$

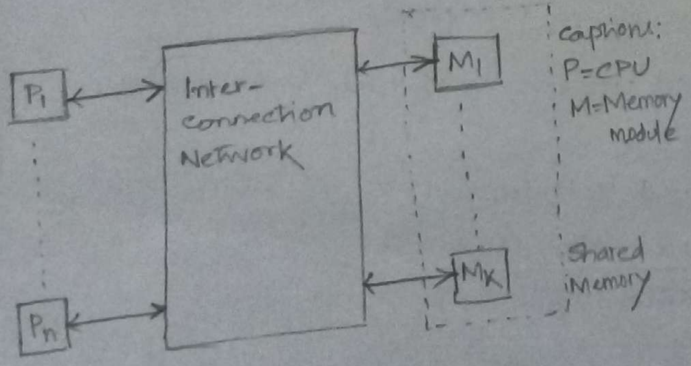
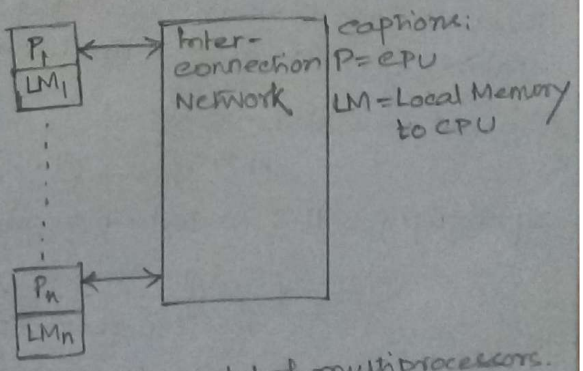
Notice that if the workload is scaled up to maintain a fixed execution time as the number of processors increases, the speedup increases linearly. What Gustafson's law says is that the true parallel power of a large multiprocessor system is only achievable when a large parallel problem is applied.

2. What is the difference between UMA and NUMA architecture? Are these SIMD or SISD or MIMD or MISD architecture?

Ans: UMA (Uniform-Memory-Access) Model and NUMA (Nonuniform-Memory-Access) Model are two different sets of architectural models available for a multiprocessor.

These models being multiprocessor systems belong to MIMD class of computer as classified by M.J. Flynn.

Differences between UMA and NUMA architecture:

UMA Model	NUMA Model
• shared memory system as all the processors access the physical main memory uniformly	• distributed-memory system in which a local memory is attached with each processor. All local memories distributed throughout the system form a global shared memory accessible by all processors
• all processors have equal access time to all memory word; uniform memory access	• a memory word access time varies with the location of the memory word in the shared memory.
• also termed as tightly coupled systems (TCS) due to high degree of resource sharing	• also known as loosely coupled system (LCS) or distributed system.
• slower to access main memory due to added delay through interconnection networks.	• faster to access local memory with a local processor.
 P1 Pn Inter-connection Network M1 Mx captions: P=CPU M=Memory module Shared Memory Fig: UMA model of multiprocessor.	 P1 LM1 Pn LMn Inter-connection Network captions: P=CPU LM=Local Memory to CPU Fig: NUMA model of multiprocessors.

3. How would you classify Systolic Array using Flynn's classification?

Ans: In parallel computer architectures, a systolic array is a homogeneous network of tightly coupled data processing units (DPUs) called cells or nodes. Each node or DPU independently computes a partial result as a function of the data received from its upstream neighbours, stores the result within itself and passes it downstream. They are sometimes classified as multiple-instruction single-data (MISD) architecture under Flynn's taxonomy, but this classification is questionable because a strong argument can be made to distinguish systolic arrays from any of Flynn's four categories: SISD, SIMD, MISD, MIMD.

4. What is the difference between loosely coupled and tightly coupled architecture? Add block diagrams of each along with your answers.

Ans:

1) Shared Memory Multiprocessor called TIGHTLY COUPLED system, they communicate through shared main memory.
   - Performance degration when 2 or more processors try to access same memory locations
   - can perform with high degree of interaction between tasks, at a higher performance level.

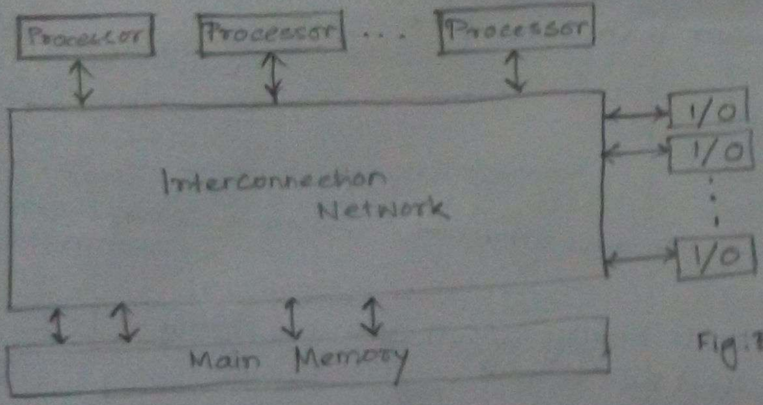


Fig: BLOCK DIAGRAM: Tightly Coupled Architecture

2) Distributed Memory Multiprocessors called LOOSELY COUPLED system, they communicate through message transfer system (MTS)
   - Multiple inexpensive computers are connected
   - Efficient when the interaction between task is low.

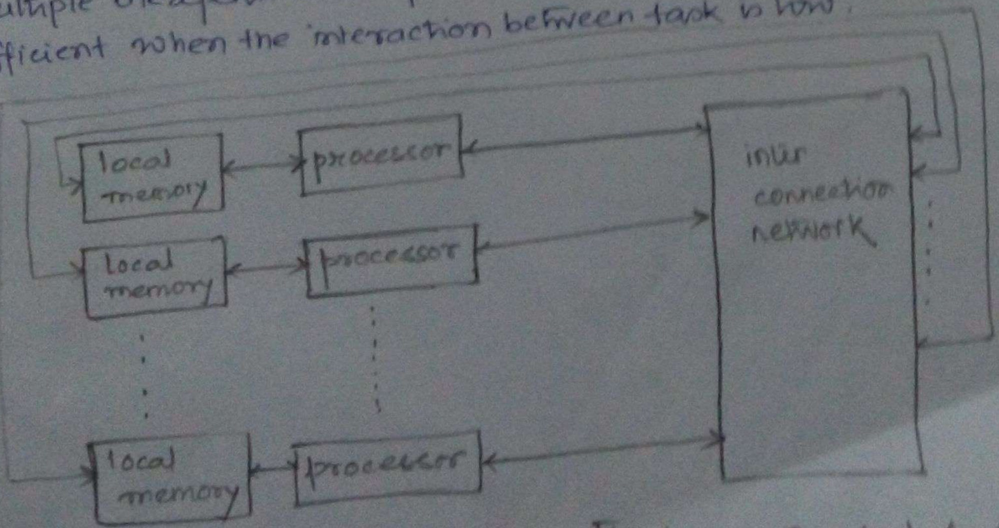


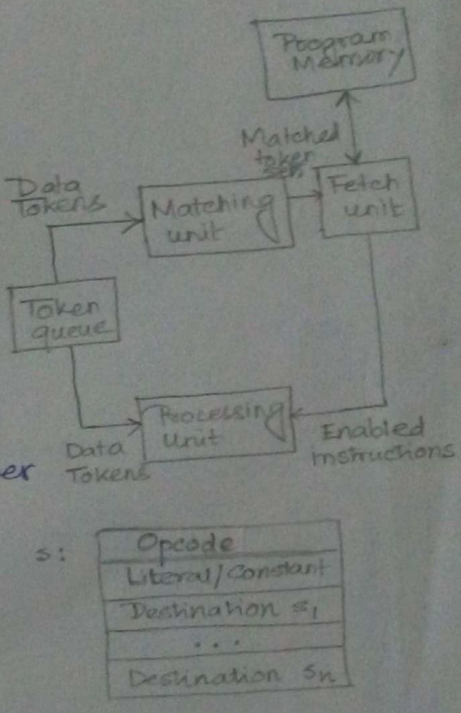
Fig: Loosely Coupled Architecture BLOCK DIAGRAM

Scanned by CamScanner

Loosely coupled architectures means changes in one module/section/ component hardly affect other components and each module is somewhat independent of each other. This architecture is robust, easy to maintain and scale. On the other hand, tightly coupled architecture promotes inter dependent applications and code. Tightly coupled architecture is fragile as minor issue in one segment can bring the whole architecture down.

5. Explain with a simple diagram the functioning of a dynamic data flow architecture machine.

Ans: The functioning of a dynamic dataflow architecture machine are as follows:-



DYNAMIC DATAFLOW ARCHITECTURE

- Separate data tokens and control
  - Token: labeled packet of information
- Allows multiple iterations to be simultaneously active
  - shared control (instruction)
  - Separate data tokens
  - a data token can carry a loop iteration number
- Match tokens' tags in matching store via assoc. search
  - if match not found, make entry, wait for partner
- When there is a match, fetch corr. instruction from program memory.
- Requires large associative search
  - to match tags.
- Adds "structure storage"
  - access via select function — index and structure descriptor as inputs.

6. What is miss penalty? State briefly what are the ways of reducing miss rate, miss penalty and hit time?

Ans: The difference between lower level access time and cache access time is called miss penalty.

- Reducing Miss Rate

1. Reduce Misses via Larger Block Size
2. Reduce Conflict Misses Via Higher Associativity
3. Reducing conflict Misses via Victim Cache
4. Reducing conflict Misses via Pseudo-Associativity
5. Reducing Misses by HW Prefetching Instruction, Data
6. Reducing Misses by SW Prefetching Data
7. Reducing Capacity/Conflict Misses by Compiler Optimizations

• Reducing Miss Penalty

1. Read Priority over Write on Miss
2. Subblock Placement to Reduce Miss Penalty
3. Early Restart and Critical Word First
4. Non-blocking Caches to reduce stalls on misses
5. Second Level Cache.

• Reducing Hit Time

1. Fast Hit times via Small and Simple Caches
2. By Avoiding Address Translation
3. Via Pipelined Writes
4. Fast Writes on Misses via Small Subblocks.

7. What are the cache coherence protocols?

Ans: The multi-cache inconsistency in multiprocessor systems known as the 'cache coherence problem' can be avoided by implementing the cache coherence protocols. They have been divided into software and hardware protocols.

Software Protocol: Relies on the compiler to deal with the problem. The compiler analyzes the data items shared by different processors. It tags the writable shared items as non-cacheable. Therefore, a reference by any processor to this shared items is made directly to the main memory. Conversely, a read only segment of data items, which is shared by several processors, need not be non-cacheable. This approach is simple and less expensive as no hardware circuitry is required and the solution is achieved at the compiler time.

Hardware Protocols: 1. Snoopy Protocol
2. Directory Protocol

→ Snoopy Protocol: In this protocol, a snoopy cache controller is attached with every processor that constantly monitors the operations on the bus by other processors. Every processor keeps track of the other processor's memory writes. Two common approaches are adopted based on snooping:

• Write update Protocol: When a processor updates ~~[illegible]~~ a shared item in its cache, it sends message to all other processors and supplies the new updated value of the item so that other processors can update their local caches immediately. If the shared items can be identified, then the extent of broadcasting can be reduced.

• Write invalidate Protocol: Multiple copies of an item are allowed. However, whenever a processor modifies location x in its cache, it must check the other caches to invalidate possible copies. This operation is called as cross-interrogate (XI).

→ Directory Protocol : In this protocol, a centralized controller is maintained that is a part of the main memory controller and a directory is kept in main memory. The directory contains global state information about the contents of the various local caches. When a processor modifies the information in its cache the central memory controller checks the directory and finds which processors are affected. Only the affected processors are informed about the change by the central controller.

MESI Protocol : To provide cache consistency, the data cache often supports a protocol known as Modified Exclusive Shared Invalid (MESI).

—— x ——