CLOUD PROGRAMMING ENVIRONMENT

DR. MANJUNATH V HEGDE AND DR. VIDYA RAO

L-26-29

PRINCIPLES OF PARALLEL AND DISTRIBUTED COMPUTING (CONTD..)

- **The term parallel computing and distributed computing are often used interchangeably, even though they mean slightly different things.**
- **The term parallel implies a tightly coupled system, where as distributed systems refers to a wider class of system, including those that are tightly coupled.**
- **More precisely, the term parallel computing refers to a model in which the computation is divided among several processors sharing the same memory.**
- **The architecture of parallel computing system is often characterized by the homogeneity of components: each processor is of the same type and it has the same capability as the others.**

PRINCIPLES OF PARALLEL AND DISTRIBUTED COMPUTING (CONTD..)

- The shared memory has a single address space, which is accessible to all the processors.
- Parallel programs are then broken down into several units of execution that can be allocated to different processors and can communicate with each other by means of shared memory.
- Originally parallel systems are considered as those architectures that featured multiple processors sharing the same physical memory and that were considered a single computer.
  - Over time, these restrictions have been relaxed, and parallel systems now include all architectures that are based on the concept of shared memory, whether this is physically present or created with the support of libraries, specific hardware, and a highly efficient networking infrastructure.

PRINCIPLES OF PARALLEL AND DISTRIBUTED COMPUTING (CONTD..)

- The term distributed computing encompasses any architecture or system that allows the computation to be broken down into units and executed concurrently on different computing elements, whether these are processors on different nodes, processors on the same computer, or cores within the same processor.
- Distributed computing includes a wider range of systems and applications than parallel computing and is often considered a more general term.
- Even though it is not a rule, the term distributed often implies that the locations of the computing elements are not the same and such elements might be heterogeneous in terms of hardware and software features.
- Classic examples of distributed computing systems are
    - Computing Grids
    - Internet Computing Systems

WHAT IS PARALLEL PROCESSING?

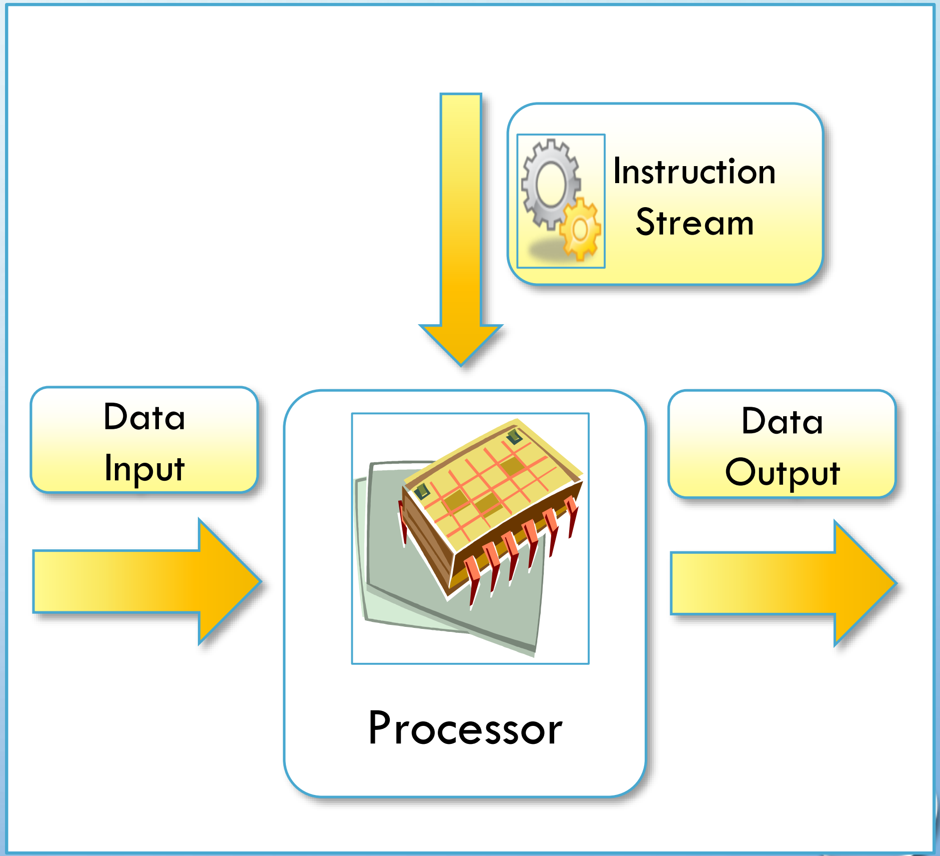
- Processing of multiple tasks simultaneously on multiple processors is called parallel processing.
- The parallel program consists of multiple active processes (tasks) simultaneously solving a given problem.
- A given task is divided into multiple subtasks using a divide-and-conquer technique, and each subtask is processed on a different central processing unit (CPU).
- Programming on multi processor system using the divide-and-conquer technique is called parallel programming.
- Many applications today require more computing power than a traditional sequential computer can offer.
- Parallel Processing provides a cost effective solution to this problem by increasing the number of CPUs in a computer and by adding an efficient communication system between them.
- The workload can then be shared between different processors. This setup results in higher computing power and performance than a single processor a system offers.

HARDWARE ARCHITECTURES FOR PARALLEL PROCESSING

- **The core elements of parallel processing are CPUs. Based on the number of instructions and data streams, that can be processed simultaneously, computing systems are classified into the following four categories:**
  - **Single-instruction, Single-data (SISD) systems**
  - **Single-instruction, Multiple-data (SIMD) systems**
  - **Multiple-instruction, Single-data (MISD) systems**
  - **Multiple-instruction, Multiple-data (MIMD) systems**

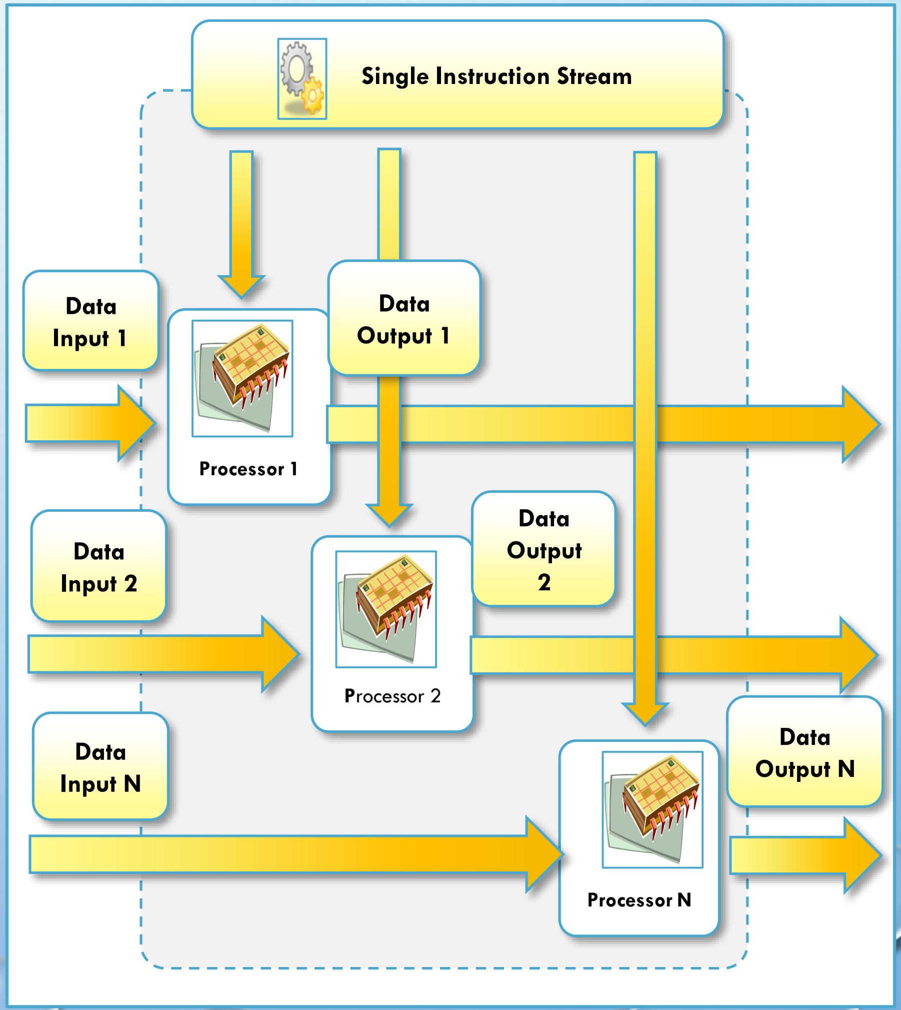
SINGLE – INSTRUCTION , SINGLE DATA (SISD) SYSTEMS

- SISD computing system is a uni-processor machine capable of executing a single instruction, which operates on a single data stream.
- Machine instructions are processed sequentially, hence computers adopting this model are popularly called sequential computers.
- Most conventional computers are built using SISD model.
- All the instructions and data to be processed have to be stored in primary memory.
- The speed of processing element in the SISD model is limited by the rate at which the computer can transfer information internally.
- Dominant representative SISD systems are IBM PC, Macintosh, and workstations.



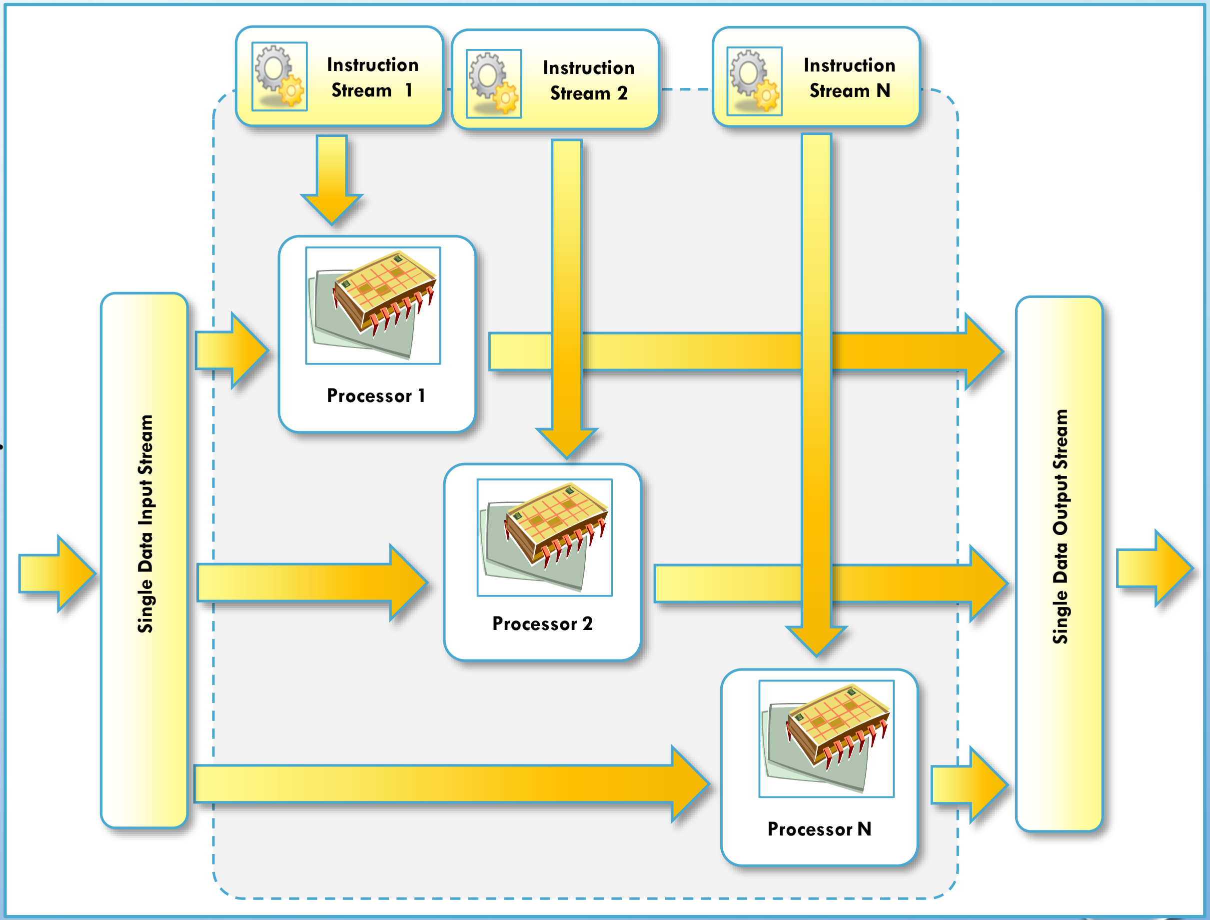
SINGLE – INSTRUCTION , MULTIPLE DATA (SIMD) SYSTEMS

- SIMD computing system is a multiprocessor machine capable of executing the same instruction on all the CPUS but operating on different data streams.
- Machines based on this model are well suited for scientific computing since they involve lots of vector and matrix operations.
- For instance statement Ci = Ai * Bi, can be passed to all the processing elements (PEs), organized data elements of vectors A and B can be divided into multiple sets (N-sets for N PE systems), and each pe can process one data set.
- Dominant representative SIMD systems are cray's vector processing machine.



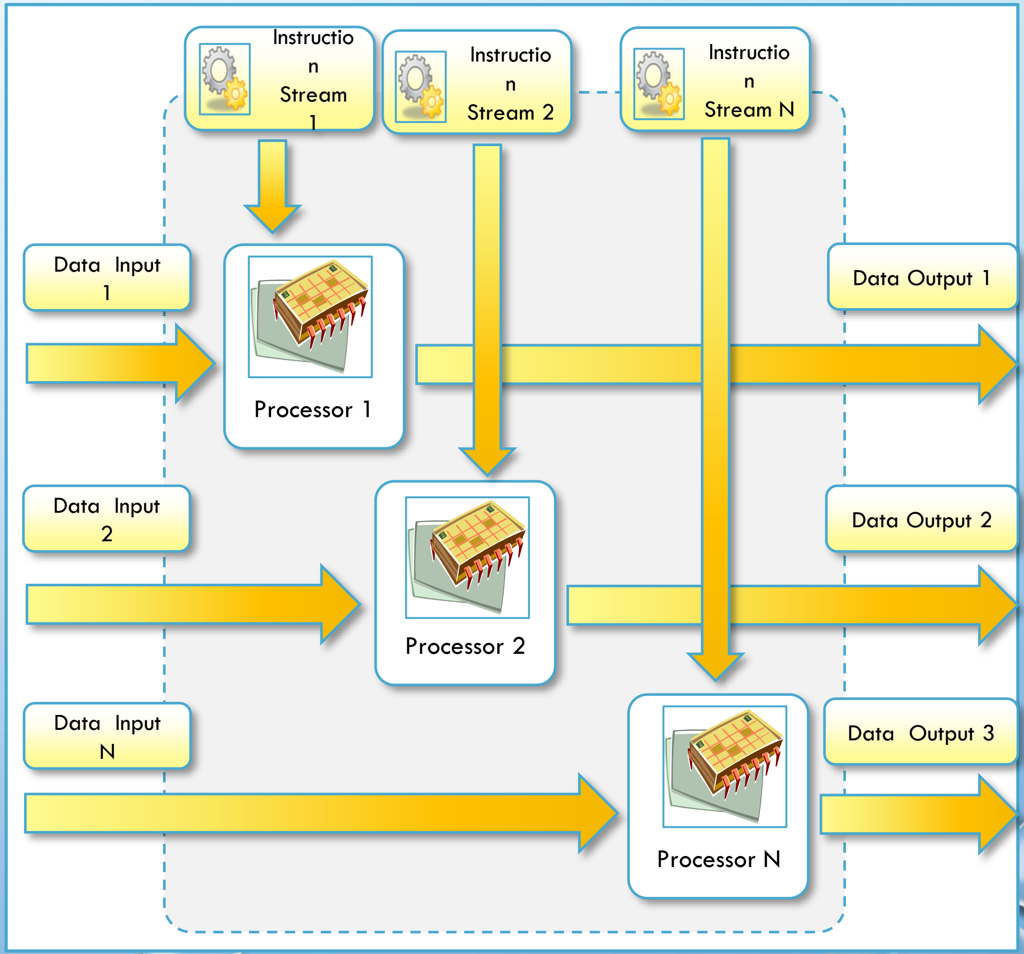
MULTIPLE – INSTRUCTION , SINGLE DATA (MISD) SYSTEMS

- MISD computing system is a multi processor machine capable of executing different instructions on different pes all of them operating on the same data set.
- For example
  - Y = sin(x) + cos(x) + tan(x)
- Machines built using MISD model are not useful in most of the applications.
- Few machines are built but none of them available commercially.
- This type of systems are more of an intellectual exercise than a practical configuration.



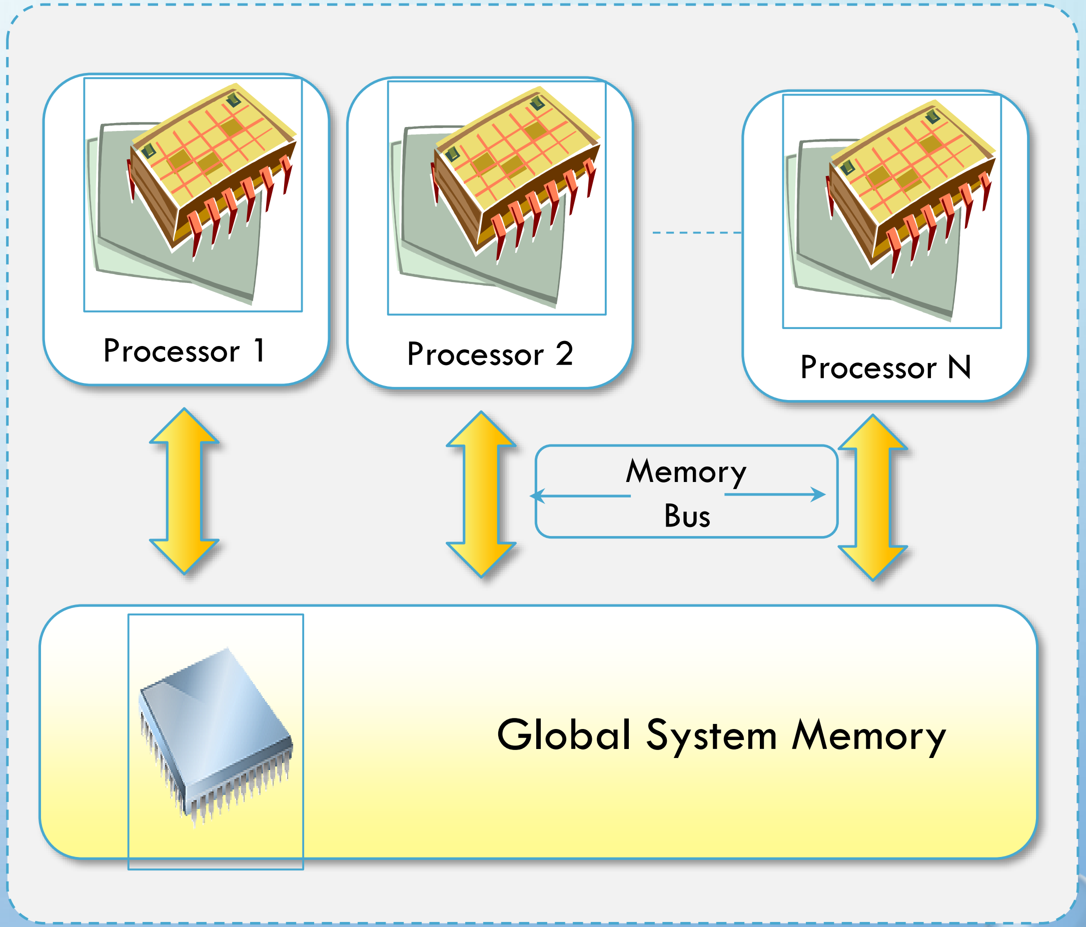
MULTIPLE – INSTRUCTION , MULTIPLE DATA (MIMD) SYSTEMS

- MIMD computing system is a multi processor machine capable of executing multiple instructions on multiple data sets.
- Each pe in the MIMD model has separate instruction and data streams, hence machines built using this model are well suited to any kind of application.
- Unlike SIMD, MISD machine, PEs in MIMD machines work asynchronously,
- MIMD machines are broadly categorized into shared-memory MIMD and distributed memory MIMD based on the way pes are coupled to the main memory.



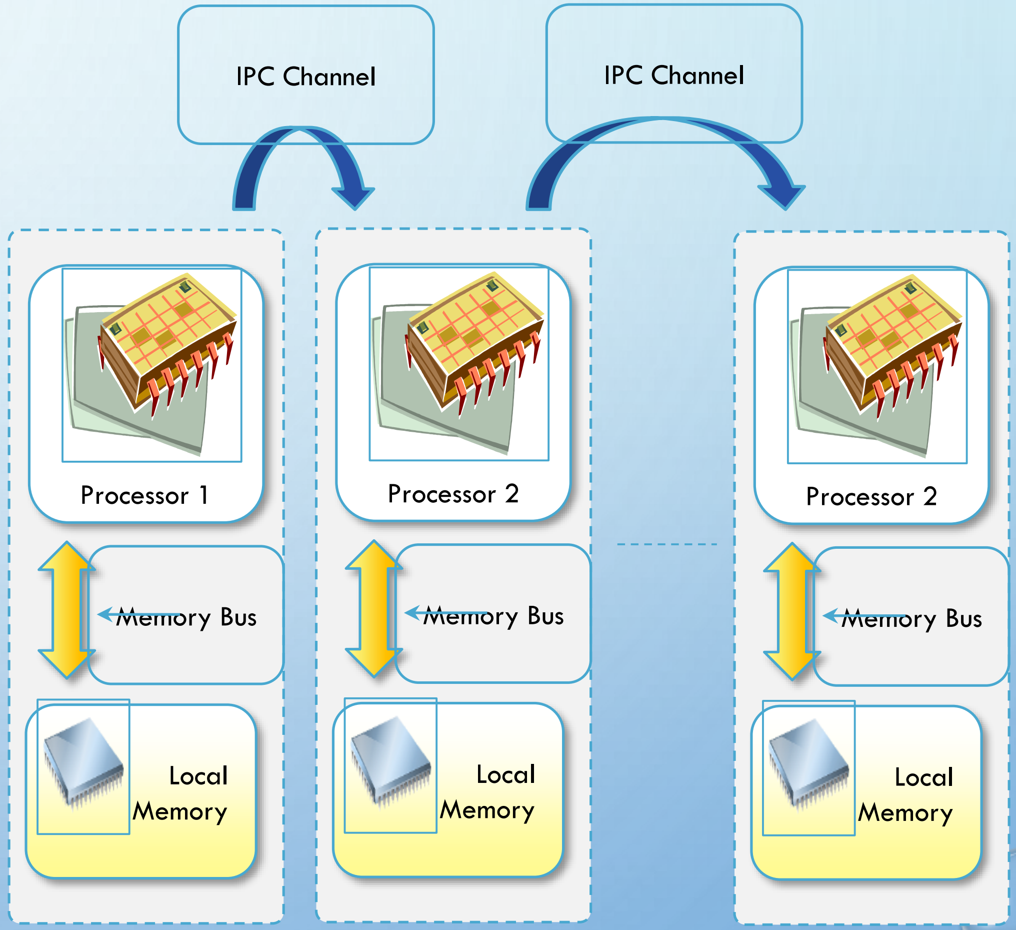
SHARED MEMORY MIMD MACHINES

- All the pes are connected to a single global memory and they all have access to it.
- Systems based on this model are also called tightly coupled multi processor systems.
- The communication between PEs in this model takes place through the shared memory.
- Modification of the data stored in the global memory by one PE is visible to all other PEs.
- Dominant representative shared memory MIMD systems are silicon graphics machines and Sun/IBM SMP (symmetric multi-processing).



DISTRIBUTED MEMORY MIMD MACHINES

- All pes have a local memory. Systems based on this model are also called loosely coupled multi processor systems.
- The communication between PEs in this model takes place through the interconnection network, the inter process communication channel, or IPC.
- The network connecting pes can be configured to tree, mesh, cube, and so on.
- Each PE operates asynchronously, and if communication/synchronization among tasks is necessary, they can do so by exchanging messages between them.



SHARED VS DISTRIBUTED MIMD MODEL

- The shared memory MIMD architecture is easier to program but is less tolerant to failures and harder to extend with respect to the distributed memory MIMD model.
- Failures, in a shared memory MIMD affect the entire system, whereas this is not the case of the distributed model, in which each of the PEs can be easily isolated.
- Moreover, shared memory MIMD architectures are less likely to scale because the addition of more PEs leads to memory contention.
- This is a situation that does not happen in the case of distributed memory, in which each PE has its own memory.
- As a result, distributed memory MIMD architectures are most popular today.

APPROACHES TO PARALLEL PROGRAMMING

- A sequential program is one that runs on a single processor and has a single line of control.
- To make many processors collectively work on a single program, the program must be divided into smaller independent chunks so that each processor can work on separate chunks of the problem.
- The program decomposed in this way is a parallel program.
- A wide variety of parallel programming approaches are available.

APPROACHES TO PARALLEL PROGRAMMING

- The most prominent among them are the following.
  - Data Parallelism
  - Process Parallelism
  - Farmer-and-worker model
- The above said three models are suitable for task-level parallelism. In the case of data level parallelism, the divide-and-conquer technique is used to split data into multiple sets, and each data set is processed on different PEs using the same instruction.
- This approach is highly suitable to processing on machines based on the SIMD model.

APPROACHES TO PARALLEL PROGRAMMING

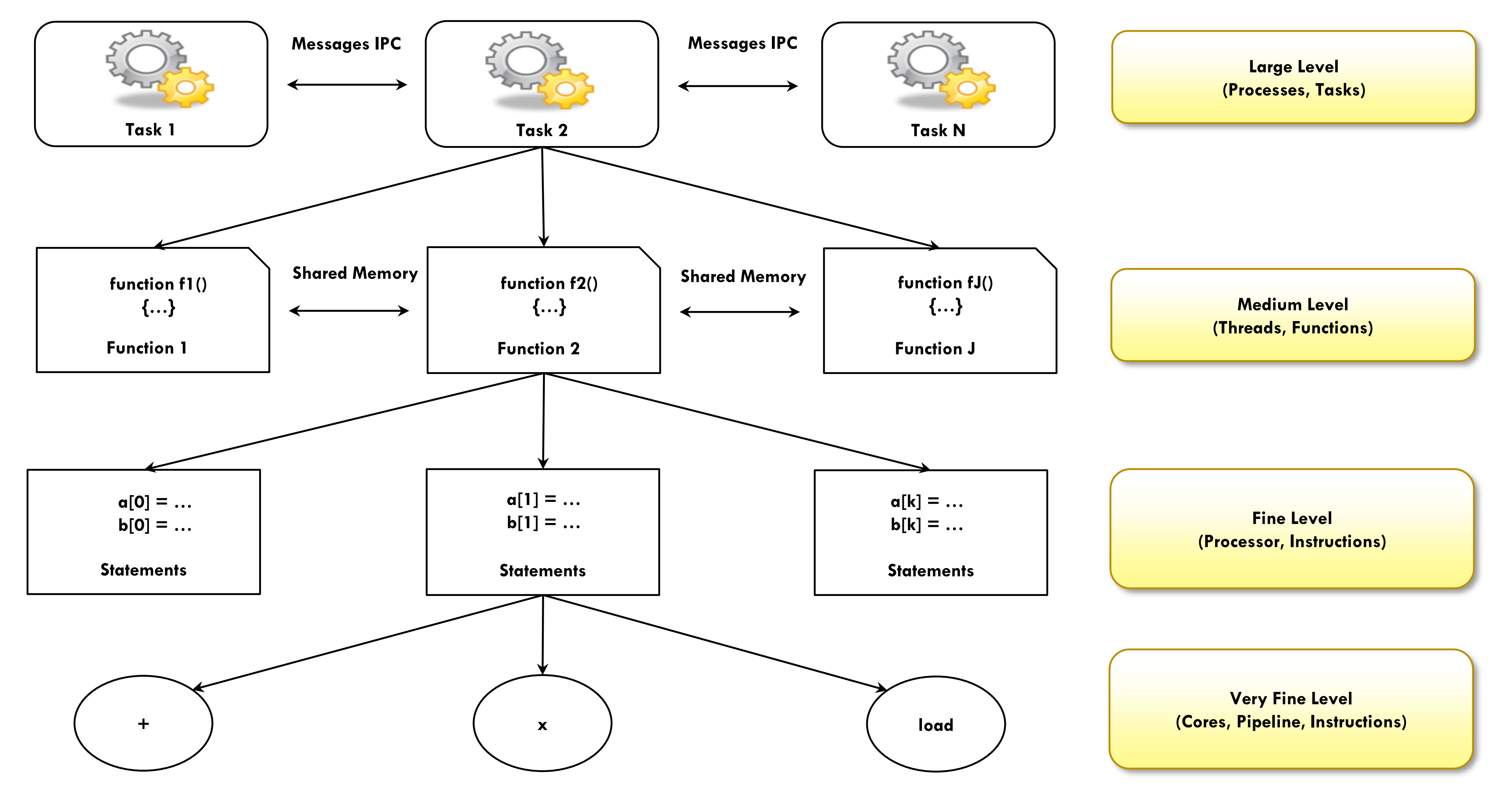
- In the case of Process Parallelism, a given operation has multiple (but distinct) activities that can be processed on multiple processors.
- In the case of Farmer-and-Worker model, a job distribution approach is used, one processor is configured as master and all other remaining PEs are designated as slaves, the master assigns the jobs to slave PEs and, on completion, they inform the master, which in turn collects results.
- These approaches can be utilized in different levels of parallelism.

LEVELS OF PARALLELISM

- Levels of Parallelism are decided on the size of code (grain size) that can be a potential candidate of parallelism.
- The table shows the levels of parallelism.
- All these approaches have a common goal
  - To boost processor efficiency by hiding latency.
  - To conceal latency, there must be another thread ready to run whenever a lengthy operation occurs.
- The idea is to execute concurrently two or more single-threaded applications. Such as compiling, text formatting, database searching, and device simulation.

Grain Size	Code Item	Parallelized By
Large	Separate and heavy weight process	Programmer
Medium	Function or procedure	Programmer
Fine	Loop or instruction block	Parallelizing compiler
Very Fine	Instruction	Processor

LEVELS OF PARALLELISM (CONTD..)



ELEMENTS OF DISTRIBUTED COMPUTING

- In the previous section we discussed techniques and architectures that allow introduction of parallelism within a single machine or system and how parallelism operates at different levels of the computing stack.
- Here extend these concepts and explore how multiple activities can be performed by leveraging systems composed of multiple heterogeneous machines and systems.
- We discuss what is generally referred to as distributed computing and more precisely introduce the most common guidelines and patterns for implementing distributed computing systems from the perspective of the software designer.

GENERAL CONCEPTS AND DEFINITIONS

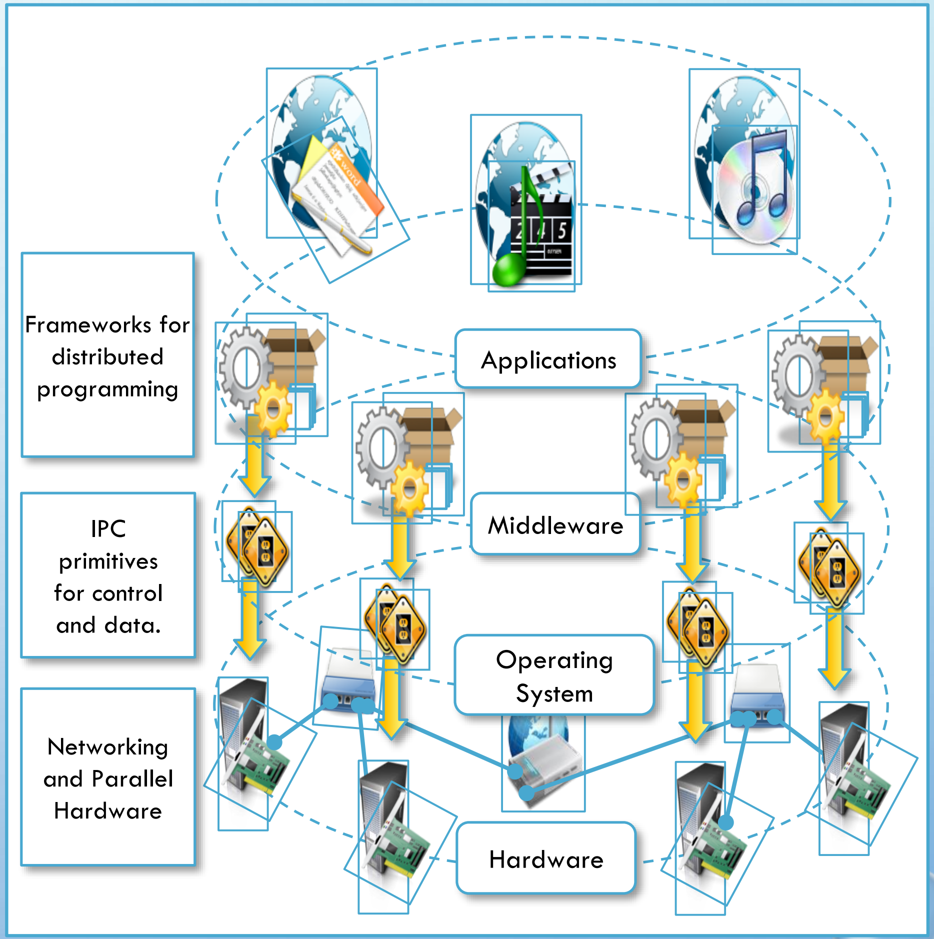
- Distributed computing studies the models, architectures, and algorithms used for building and managing distributed systems.
- As general definition of the term distributed system, we use the one proposed by Tanenbaum
  - A distributed system is a collection of independent computers that appears to its users as a single coherent system.
- This definition is general enough to include various types of distributed computing systems that are especially focused on unified usage and aggregation of distributed resources.

GENERAL CONCEPTS AND DEFINITIONS
(CONTD..)

- Here, we focus on the architectural models, that are used to harness independent computers and present them as a whole coherent system.
- Communications is another fundamental aspect of distributed computing. Since distributed systems are composed of more than one computer that collaborate together, it is necessary to provide some sort of data and information exchange between them, which generally occurs through the network.
  - A distributed system is one in which components located at networked computers communicate and coordinate their action only by passing messages.
- As specified in this definition, the components of a distributed system communicate with some sort of message passing. This is a term the encompasses several communication models.

COMPONENTS OF DISTRIBUTED SYSTEM

- A distributed system is the result of the interaction of several components that traverse the entire computing stack from hardware to software.
- It emerges from the collaboration of several elements that- by working together- give users the illusion of a single coherent system.
- The figure provides an overview of the different layers that are involved in providing the services of a distributed system.
- At the very bottom layer, computer and network hardware constitute the physical infrastructure; these components are directly managed by the operating system, which provides the basic services for inter process communication (IPC), process scheduling and management, and resource management in terms of file system and local devices.
- Taken together these two layers become the platform on top of which specialized software is deployed to turn a set of networked computers into a distributed system.



TECHNOLOGIES FOR DISTRIBUTED COMPUTING

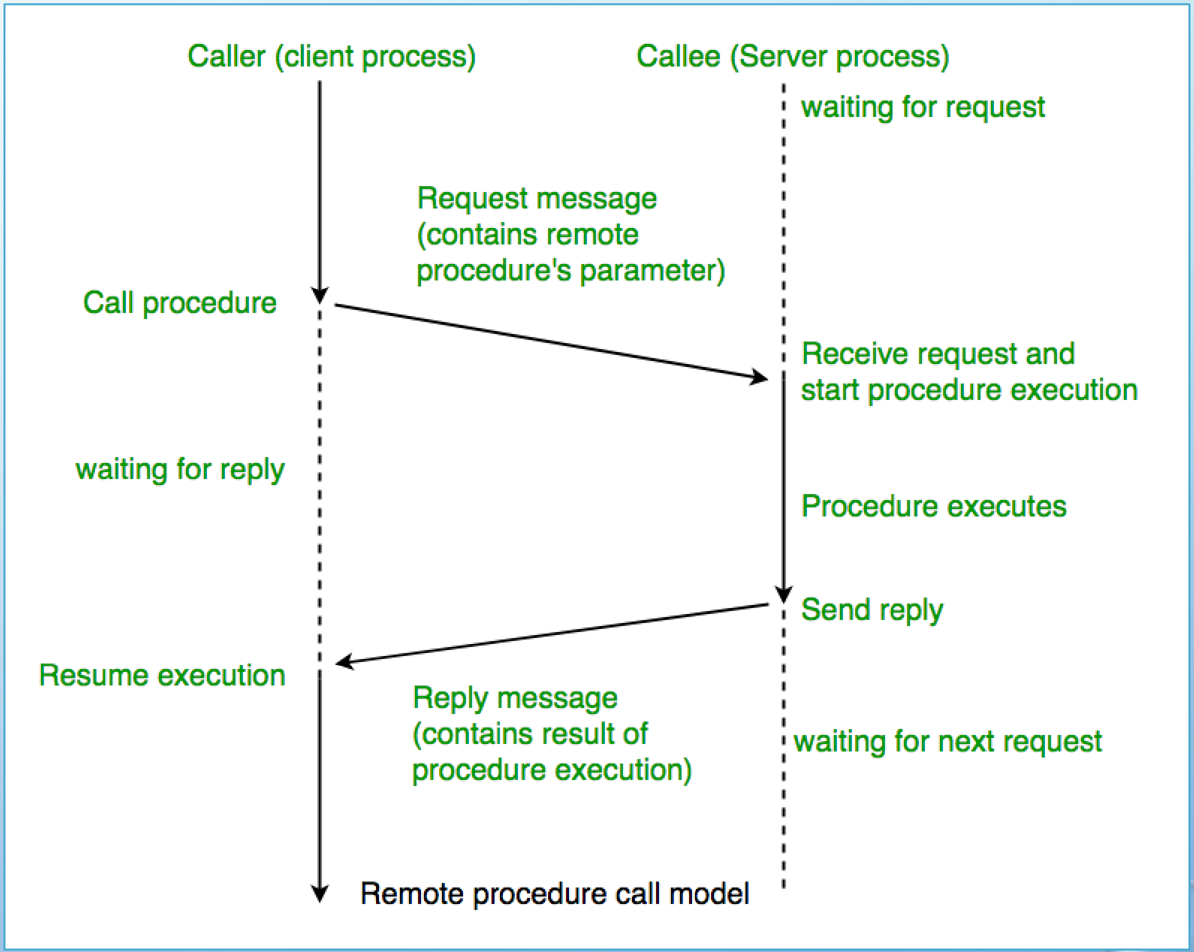
- Remote Procedure Call (RPC)
  - RPC is the fundamental abstraction enabling the execution procedures on clients' request.
  - RPC allows extending the concept of a procedure call beyond the boundaries of a process and a single memory address space.
  - The called procedure and calling procedure may be on the same system or they may be on different systems.
  - The important aspect of RPC is marshalling and unmarshalling.
- Distributed Object Frameworks
  - Extend object-oriented programming systems by allowing objects to be distributed across a heterogeneous network and provide facilities so that they can be coherently act as though they were in the same address space.

REMOTE PROCEDURE CALL (RPC)

- Remote Procedure Call (RPC) is a technique for constructing distributed, client-server based applications. It is based on extending the conventional local procedure calling so that the called procedure need not exist in the same address space as the calling procedure.
- The two processes may be on the same system, or they may be on different systems with a network connecting them.

REMOTE PROCEDURE CALL (RPC)

- The calling environment is suspended, procedure parameters are transferred across the network to the environment where the procedure is to execute, and the procedure is executed there.
- When the procedure finishes and produces its results, its results are transferred back to the calling environment, where execution resumes as if returning from a regular procedure call.

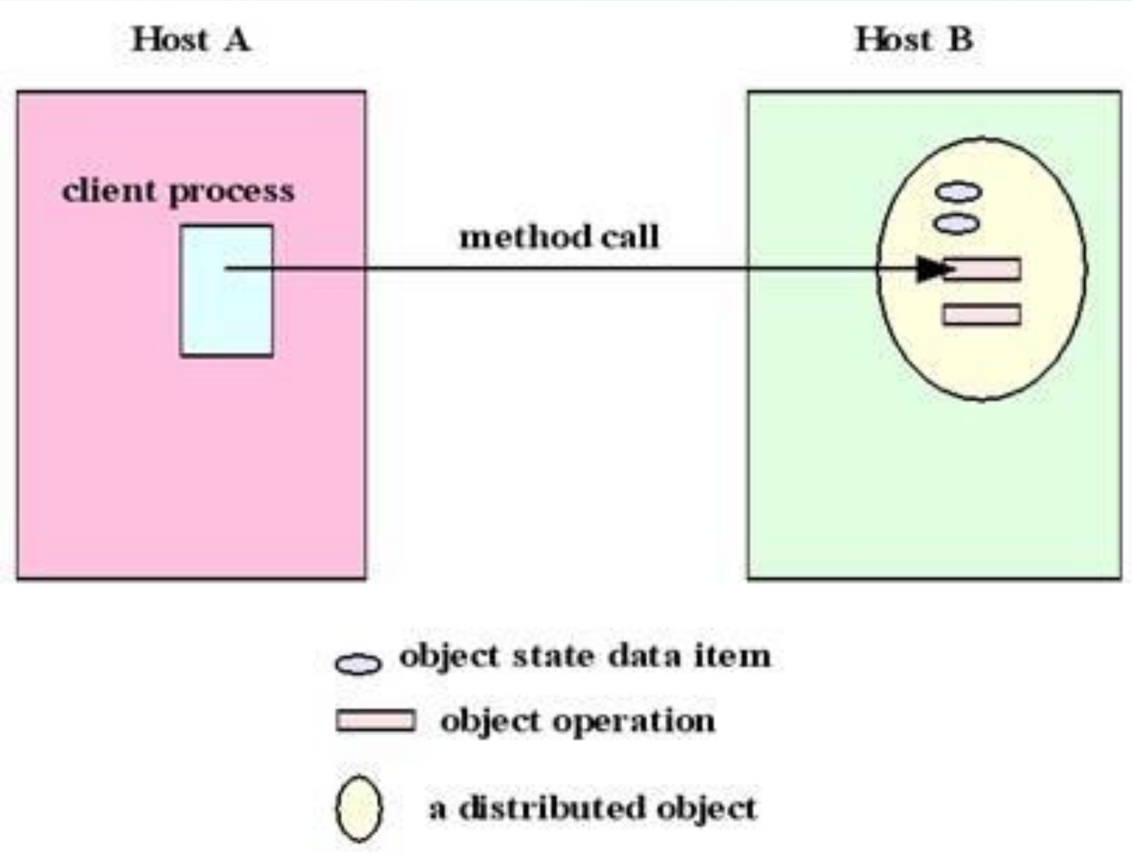


DISTRIBUTED OBJECT

- The distributed object paradigm
  - provides abstractions beyond those of the message-passing model.
  - In object-oriented programming, objects are used to represent an entity significant to an application.
- Each object encapsulates:
  - the state or data of the entity: in Java, such data is contained in the instance variables of each object;
  - the operations of the entity, through which the state of the entity can be accessed or updated.
- Local Objects vs. Distributed Objects
  - Local objects are those whose methods can only be invoked by a local process, a process that runs on the same computer on which the object exists.
  - A distributed object is one whose methods can be invoked by a remote process, a process running on a computer connected via a network to the computer on which the object exists.

THE DISTRIBUTED OBJECT PARADIGM

- In a distributed object paradigm, network resources are represented by distributed objects.
- To request service from a network resource, a process invokes one of its operations or methods, passing data as parameters to the method.
- The method is executed on the remote host, and the response is sent back to the requesting process as a return value.q



NEXT CLASS….