14. what is the major revolution introduced by web 2.0. How it shifted from a static to a dynamic version of the Internet

15. Main characteristics of a service orientation.

* It supports loose coupling everywhere in the project.
* SOA supports interoperability,
* It increases the quality of service.
* It supports vendor diversity,
* It promotes discovery & federation.
* It is location-transparent.
* It is still maturing and achievable idea.

16. what is utility computing?

A service provisioning model that offers computing resources to clients as and when they require them on an-demand basis.

17. what are the major advantage of cloud computing.

* Cost savings * Security * Flexibility * Mobility
* Insight * Increased collaboration * Quality control
* Disaster Recovery. * Loss prevention * Sustainability
* Automatic s/w updates * competitive Edge

18. How cloud development different from traditional software development.

The purpose of cloud based s/w vs on-premise s/w is to expand its scale not being limited by physical storage. This allows us to expand the capabilities of your s/w indefinitely add new features & stores large amount of data while maintaining stable and continuous performance.

19. What are the different parameters that has to be considered while building a cloud computing environment?

* Distributed design * operational excellence
* security & compliance * Reliability * performance.

Distributed design:- It is inherent when architecting a cloud platform.

* It is a complex challenge that must be addressed from the beginning.
* The strategic aspects are:
  * Global networking * User location
  * Regulatory requirements
  * Resource management
  * Identity & access management
  * cloud services
  * vendor lock
  * product best practices.

Operational Excellence:- * An ecosystem's ability to maintain optimal operational performance standards, under either normal or extreme circumstances requires addressing aspects such as automation of processes, observability and inclusion of contingency plans.

* The deploy solutions are:
  * logging & monitoring
  * Disaster recovery planning and execution.

Security & compliance:- * It is for mission-critical systems.

* It must be made to ensure potential vulnerabilities are addressed.
* compliance must be addressed and constantly compiled and evaluated.
* compliances are:- * Authentication & Authorization
  * data security
  * Audit

Reliability :- * A company's cloud reliability can contribute significantly to its operational excellence and ability to serve its customers.

* when analyzing production errors & associated improvements.

* reliability topics to consider:-
  * Site reliability engineering
  * Service metrics.
  * observability
  * High availability.
  * Deployment strategy.

performance :- * A soln's elasticity when dealing with n/w & connectivity changes.

* performance aspects on target:
  * Autoscaling
  * Tracing
  * debugging
  * profiling
  * Cost optimization
  * forecast calculation.

80. what are the key elements of computing. explain.

* architecture * compilers * appln * prblm solving environment.

architecture :- * It refers to various cloud technology components, such as hardware, s/w, virtual resources capabilities and virtual n/w systems interact and connect to create cloud computing environments.

* Compilers :- Compilers are portable and reduces the storage space.

* which provide most convenient tool to compile code and remove the errors.

Applns :- s/w that runs the processing logic and data storage s/w in different systems.

* Some processing takes place on an end user's.
* Backup and recovery
* Big data analytics
* E commerce appln.

Problm solving environment :- A system that provides all the computational facilities necessary to solve a target class of problems.

* It uses language of the target class & specialized knowledge of the h/w or s/w.

21. what are the 3 phases of computing.

* research & development
* commercialization
* commoditization.

22. what is parallel & distributed computing.

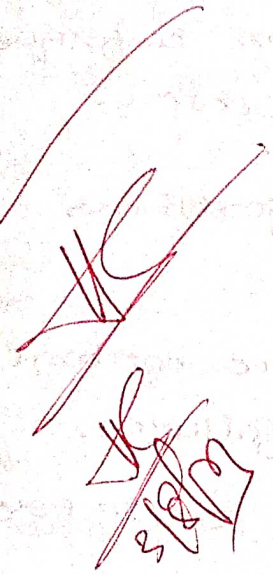
Parallel C :- the process of breaking down larger probms into smaller, independent & combining of the results.

Distributed C :- It is a system whose components are located on different networked computers, which commu-nicate & coordinate their actions by passing msgs to one another

23. Differentiate b/w parallel and distributed computing.

Parallel	Distributed.
* Tightly coupled * we need to divide & solve the problem * All process run at same time.	* Tightly coupled used for wider class. * Manage larger problem. * It will wait for other process to complete then it will run

* It initiate once the result is ready	* It will wait until the process complete that it depends on.
* memory will remain same.	* data will be given to each & every process.
* Backup is not there.	* It follow particular mechanism.
* It is interdependent.	* we can fetch previous data if there are mish-anding of data happens.
* It is difficult	



84. what is parallel processing or parallel programing. It is a method in computing of running two or more processors to handle separate parts of an overall task.

Parallel processing :- processing of multiple tasks simultaneously on multiple processors.

The task is divided into multiple subtasks using a divide & conquer technique & each subtasks is processed on different CPU.

parallel programming :- programming on a multiprocessor system using the divide & conquer technique.

* parallel processing provides a cost-effective solution to problems.

25. Explain

The Development of parallel processing is being influenced by many factors :-

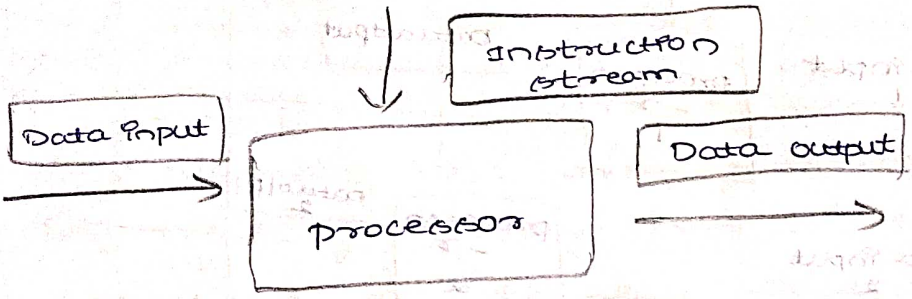
* computational requirements increasing in the area of scientific & business computing.
* Sequential architecture to get high computational speed.
* Hardware improvements in pipelining, superscalar & sophisticated compiler technology.
* Vector processing works for certain kind of problems.
* the technology of parallel processing is mature & exploited commercially.
* Development in networking technology
* It is suitable for heterogeneous computing.

Hardware architectures for parallel processing :-

26. Based on the no. of instruction & data streams that can be processed simultaneously, computing systems are classified into 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 (SSID) systems.

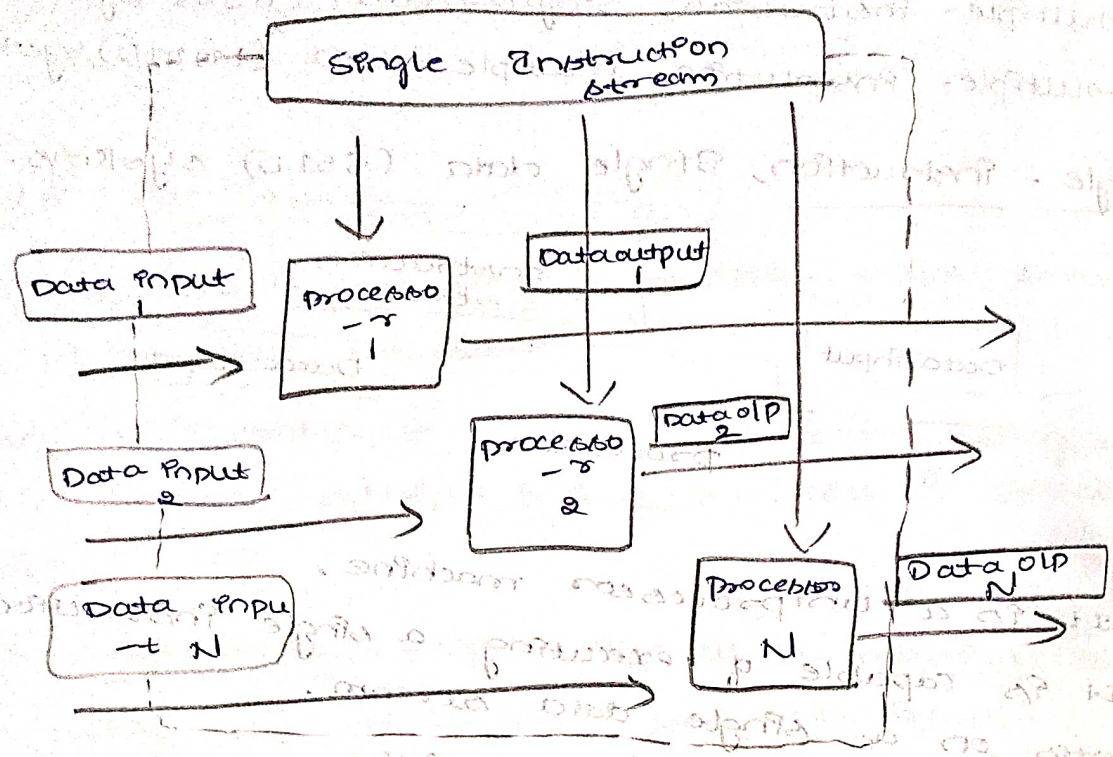


* It is a uniprocessor machine.
* It is capable of executing a single instruction, which operates on a single data stream.
* these are processed sequentially.
* Hence, computer adopt this model called sequential computers.
* All the instructions & data to be processed stored in primary memory.
* Speed of processing element is limited.
* Dominant representative SSID systems are IBM PC, Macintosh & workstations.

Single-instruction, multiple-data (SIMD) systems.

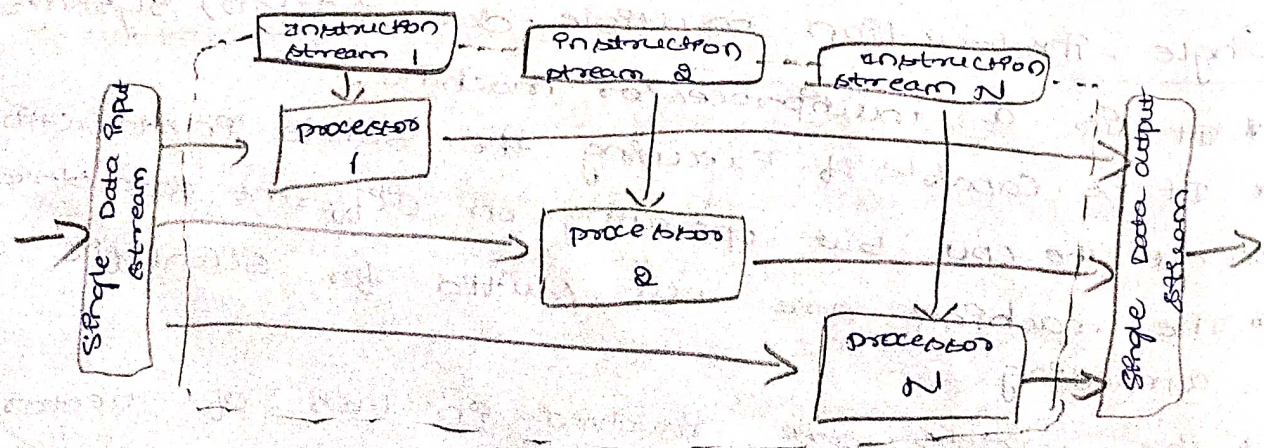
* It is a multiprocessor machine.
* It is capable of executing the same instruction on all the cpu but operating on different data streams.
* The machines are well suited for scientific computing.
* The models are involved in lots of vector operations.

- $C_i = A_i * B_i$;
- This can be passed to all the processing element -s (PEs).
- A & B are the vectors divided into multiple sets.
- The Dominant representative SIMD systems are Cray's vector processing machine and Thinking Machines' cm*.



Multiple-Instruction, Single-data (MISD) systems.

- It is a multiprocessor machine.
- It is capable of executing different instructions on the different processing elements but all of them operating on the same data set.
- For instance, $y = \sin(x) + \cos(x) + \tan(x)$.
- It is more intellectual exercise than practical configuration.

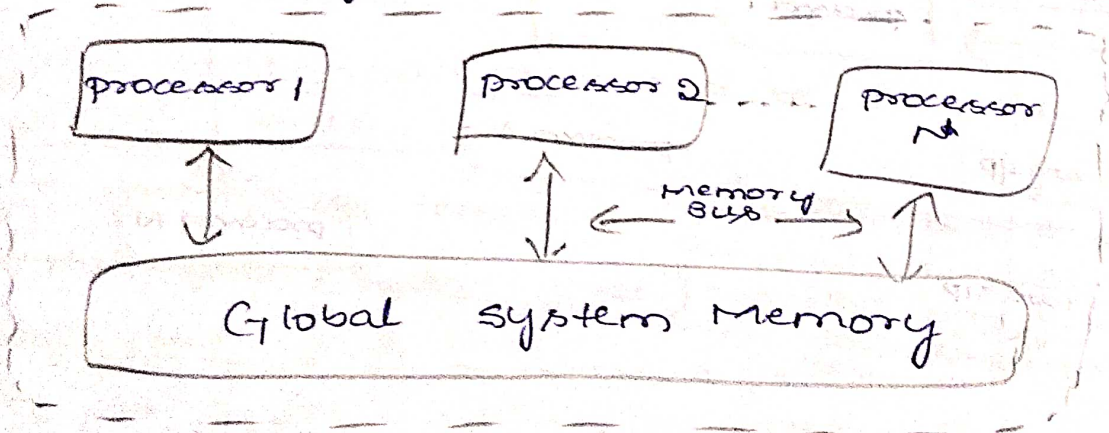


Multiple-instruction, multiple-data (MIMD) systems

* It is a multiprocessor machine.
* It is capable of executing multiple instructions on multiple data sets.
* Each PE in the MIMD model has separate instruction & data streams.
* It is well suited to any kind of appln.
* PEs in MIMD machines work asynchronously.
* MIMD machines are broadly categorized into shared-memory MIMD & distributed-memory MIMD.
* Dominant representative shared memory are SPI-con Graphics machines & Sun / IBM's SMP (Symmetric multi-processing)

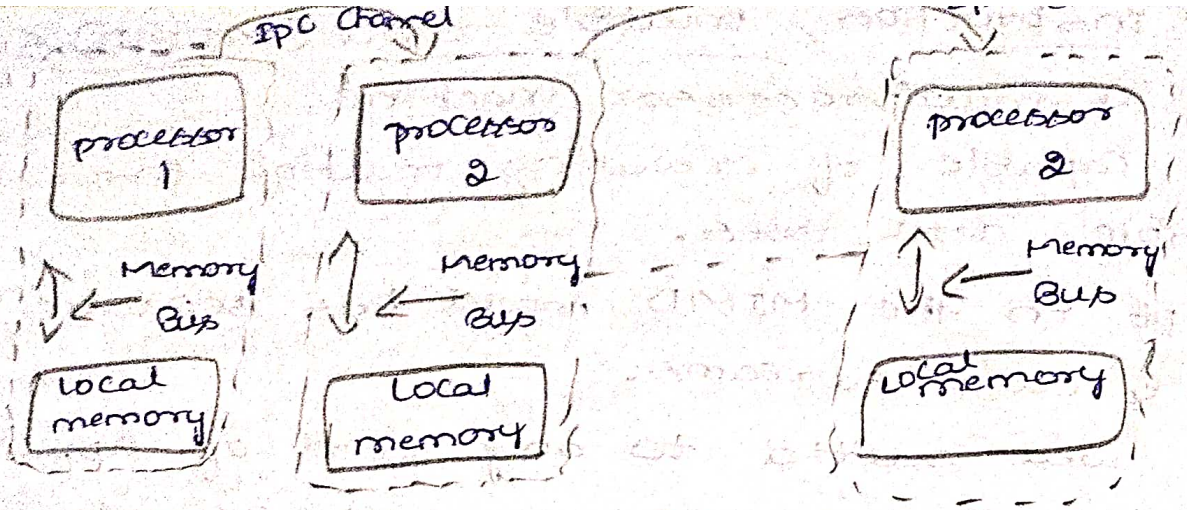
Shared memory MIMD machines:-

* In this all the PEs are connected to a single global memory & they have access to it.
* It is based on tightly coupled multiprocessor systems.
* The communication blw PEs takes place through the shared memory.
* Modification of the data stored in the global memory is visible to all.

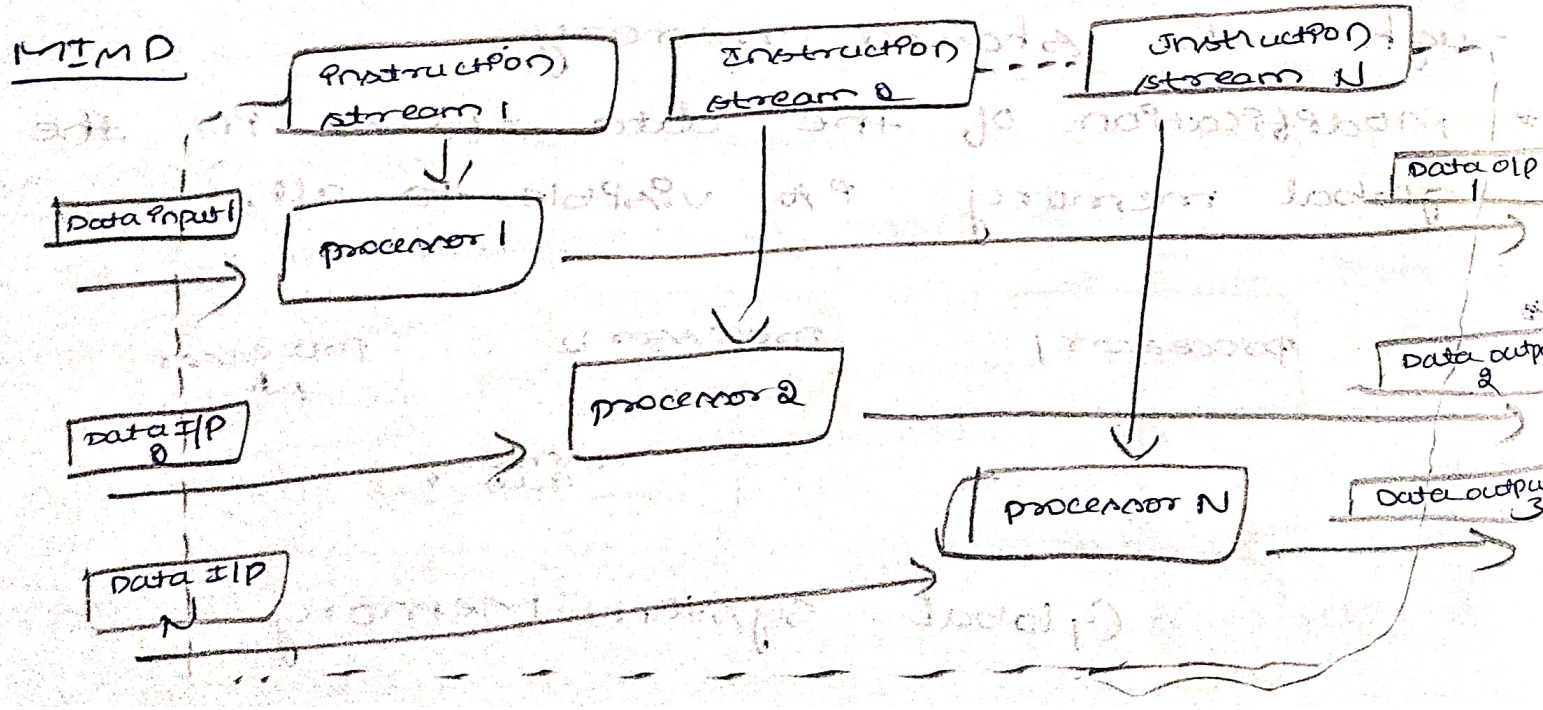


Distributed memory MIMD machines

* All PEs have a local memory.
* It is based on loosely coupled multiprocessor systems.
* The comm^n blw PE's in this model takes place through the interconnection network.
* The nlw is configured to tree, mesh, cube



* Shared memory MIMD architecture is easier to program but is less tolerant to failures.
* Harder to extend w.r.to distributed memory MIM
* Failures of shared memory affect the entire syst em.
* whereas distributed memory will not.
* Shared memory MIMD is less scalable than distributed memory MIMD.
* Distributed memory MIMD is most popular today.



9 7 Oct

Approaches to parallel programming:-

* Data parallelism
* process parallelism
* Farmer-and-worker model.

} these 3 models are all suitable for task level parallelism.

28. Explain U ... form the master to collect results.

Levels of parallelism:-

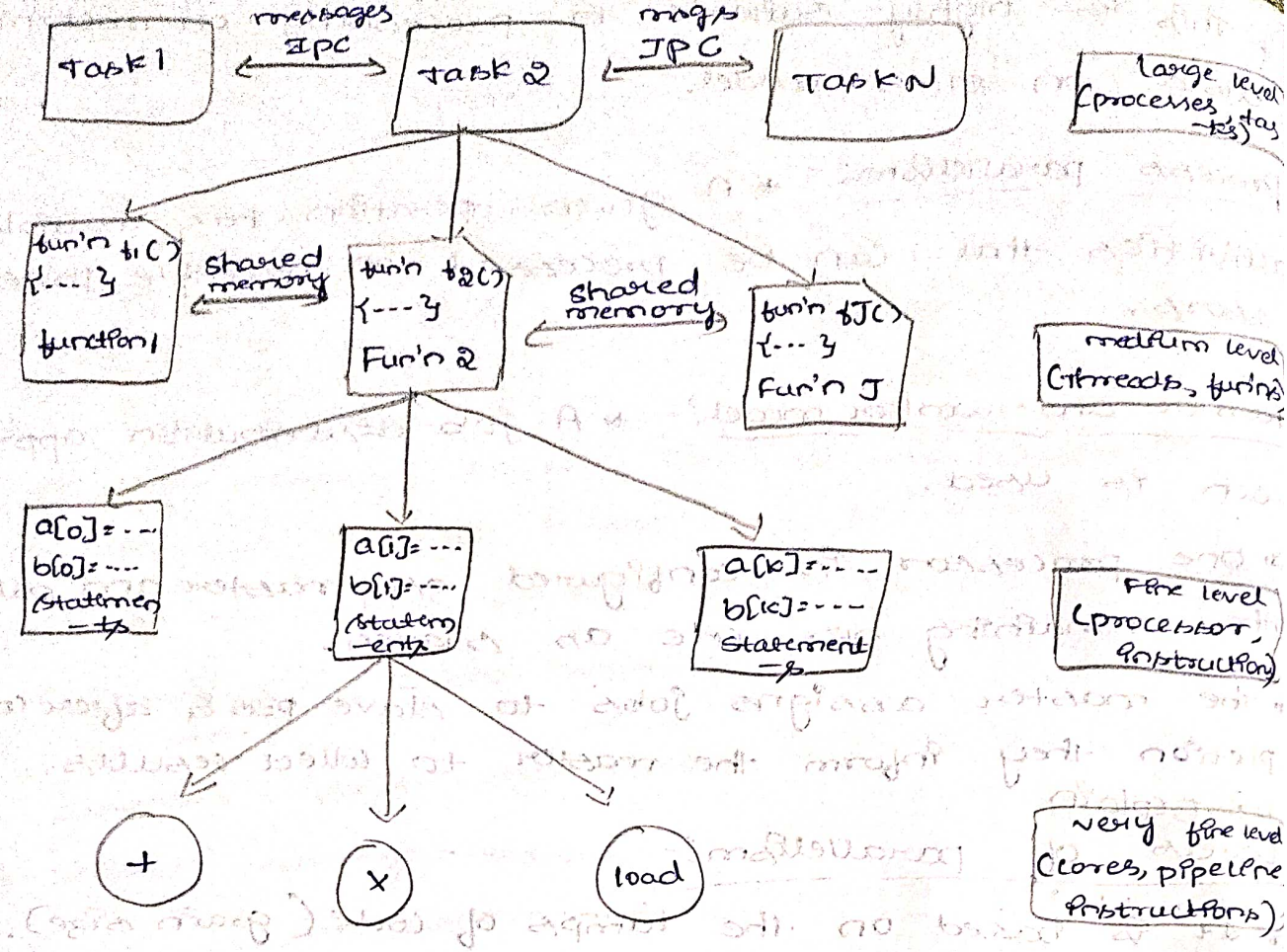
* It is based on the lumps of code (grain size).
* Lists categories of code granularity for parallelism
* TO boost processor efficiency by hiding latency.
* To execute concurrently two or more single-th-readed applns, such as compiling, text formating database searching & device simulation.

parallelism within an appln can be detected at several levels.

* Large grain (task level)
* medium grain / control level } This involves the distribution of the computation among multiple threads or processes.
* Fine grain (data level)
* very fine grain (multiple-instruction issue)

Levels of parallelism

Grain size	Code item	parallelized by
Large	separate & heavy weight processes	programmer
Medium	Fun'n or procedure	Programmer
Fine	loop or instruction block	Parallelizing compiler
Very fine	Instruction	processor



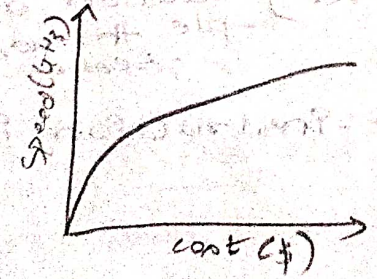
Levels of parallelism in an appln.

2a. Best

Laws of caution:-

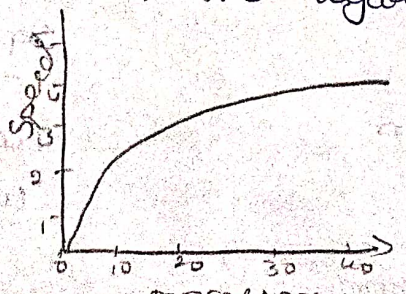
Parallelism:- It is used to perform multiple activities together to increase its throughput or speed of the system.

Speed of computation is proportional to the square root of system cost; they never increase linearly. The more before the faster a system becomes, the more expensive it is to increase its speed.



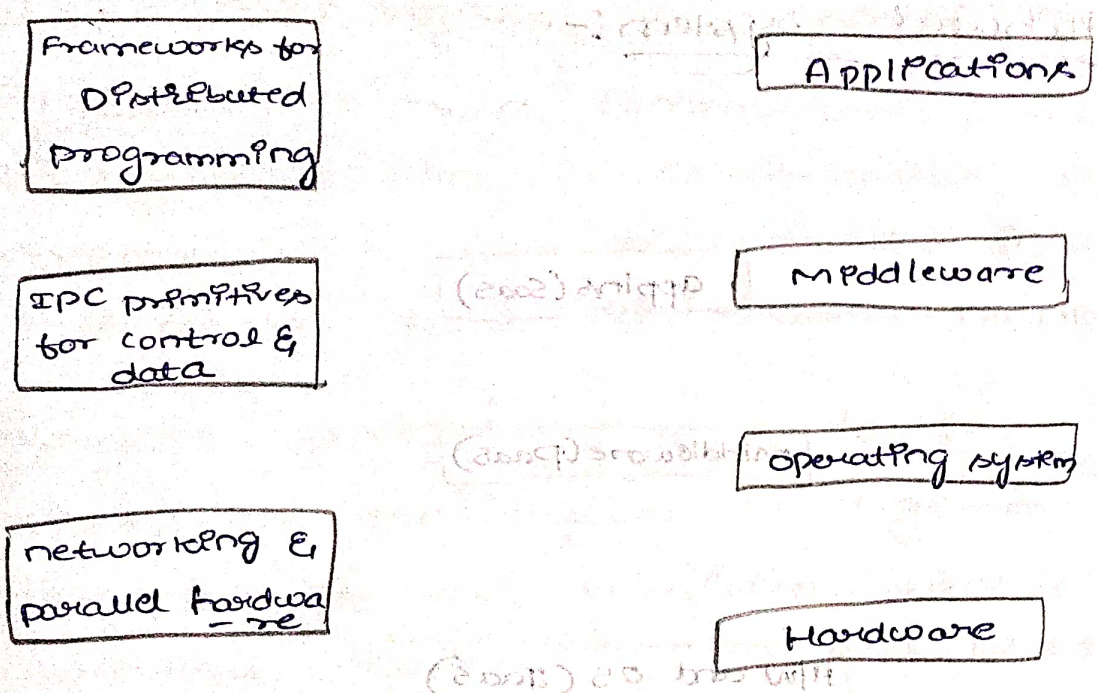
Speed by a parallel computer increases as the logarithm of the no. of processors.

$$y = k * \log(N)$$



30. explain

Components of a distributed system:-



Layered view of a distributed system.

* At bottom, hardware constitute the physical infrastructure.
* These are directly managed by operating system, that provides the basic service for IPC, process scheduling & management, and resource management in terms of file system & local devices.
* OS, network connectivity & different devices is controlled by standards.
* IPC services are implemented on top of standardized communication protocols are TCP/IP or UDP.

Middleware:- * It is for development & deployment of distributed applns.

programming paradigms for distributed systems

31. define

Component :- It represents a unit of s/w that encapsulates a fun'n or a feature of the system.

eg :- prgms, objects, processes, pipes & filter.

Connector :- It is a communication mechanism that allows cooperation & coordination among components.

Connectors are not encapsulated in a single entity.

* Implemented in a distributed manner.

32. explain

Software architectural styles :-

* Based on the logical arrangement of s/w components
* they provide an intuitive view of the whole system.
* Identify main abstractions used to shape the components of the system & expected interaction patterns b/w them.

The models constitute the foundations on top of which distributed systems are designed from a logical point of view :-

Data Centered architectures :-

This identify the data as the fundamental element of the s/w system.

Category	Most common architectural styles.
Data-Centered	Repository, Blackboard
Data-flow	pipe & filter, Batch sequential
Virtual machine	Rule-based system, Interpreter
Call & return	main prgm & subroutine call/ top-down systems, object oriented systems, Layered systems.

Independent components.

Event systems.

Two main components

Repository :- * central data structure :- which represents the current state of the system & collects the independent components.

* Repository based architecture :- controls the shared data structure.

Blackboard :- three main components :-

* knowledge sources :- these are the entities that upd-ates the knowledge base maintained in blackboard.

* Blackboard :- Represents the data structure that shar-d among the knowledge sources & stores it.

* control :- control is the collection of triggers & procedures that react with blackboard & update the state of the knowledge appln.

Blackboard maintains knowledge in the form of ass-ertions & rules.

Dataflow architecture :-

* it is availability of data that controls the computation.

Batch sequential style :- Sequence of separate prgms executing one after the other.

Pipe-and-filter style :- Style for expressing the activity of a sw system, as sequence of data transformations.

Virtual machine architectures :-

presence of an abstract execution environment that emulates features are not available in sw | hw.

* Indirection layer b/w applns & hosting environment.

Rule-Based style :- Representing the abstract execution environment as an inference engine.

* used in network intrusion detection systems (NIDS)

Interpreter style :- It is presence of an engine used to interpret a pseudo-program expressed in a format. (Java, C#)

* useful in high-level programming & scripting languages (Awk, PERL)

Call & return architectures :- It identifies all systems are organised into components, connected by method calls.

* It is possible to identify three major subcategories
  ↦ top-down style, object-oriented style & layered style.

* Top down style :- Leads to divide & conquer approach to problem resolution.

* accomplishing main prgm into sub prgms.

* method calls used for remote procedure call (RPC)

Object-oriented style :- Encompasses a wide range of systems are designed & implemented as objects.

Layered style :- Allows the design and implementation of as layers, which provide a different level of abstraction of the system.

Independent Components:- * It have their own life cycle, which interact with each other to perform their activities.

* Communicating processes:- Represented by independent processes that leverage IPC facilities for co-ordination management.

* Event systems:- these are loosely coupled and connected. for manipulation & performing operations
* call back fn happen to handle the event.

33. explain

System architectural styles:-

* physical organization of components & processes over a distributed infrastructure.

* they provide a set of reference models for deployment

Two fundamental reference styles:- Client/server and peer-to-peer.

Client/Server:- * It has two major components server & Client.

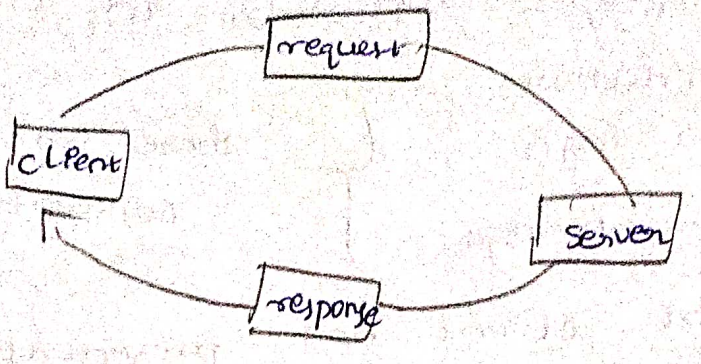
* these two components interact with each other through a network connection.

* comm^n is unidirectional.

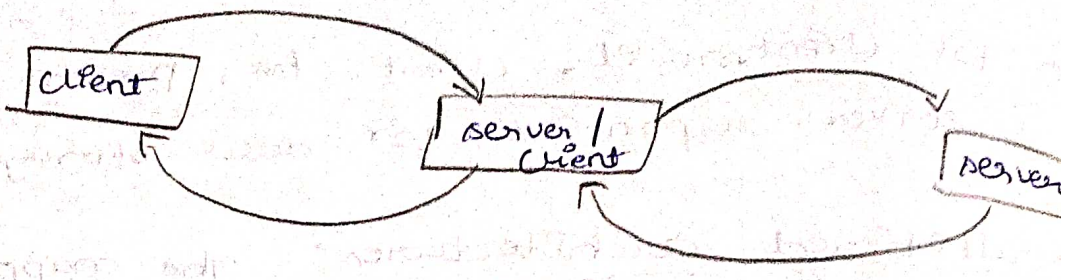
* Important operations in the client-server are request, accept (client side) and listen, response (server side)

* Services are centralized & accessed through a single access point.

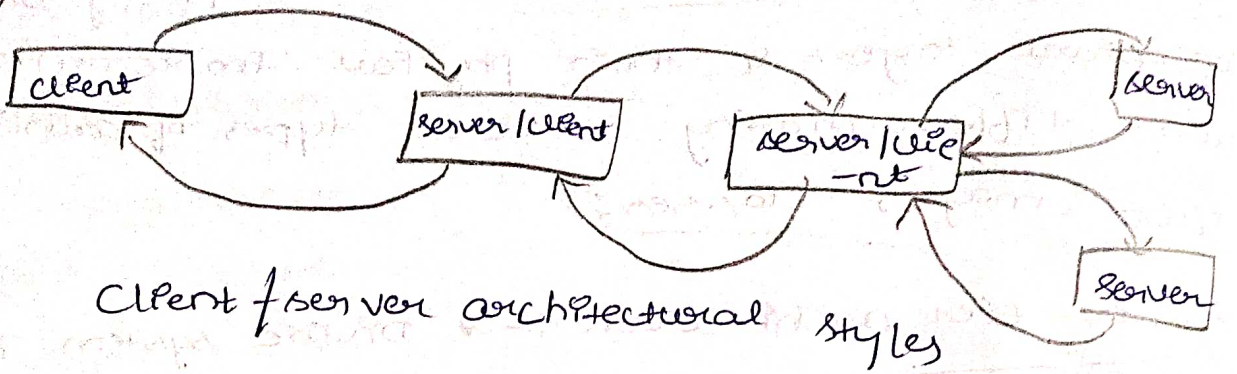
Two tier (Classic model)



Three tier



N tier

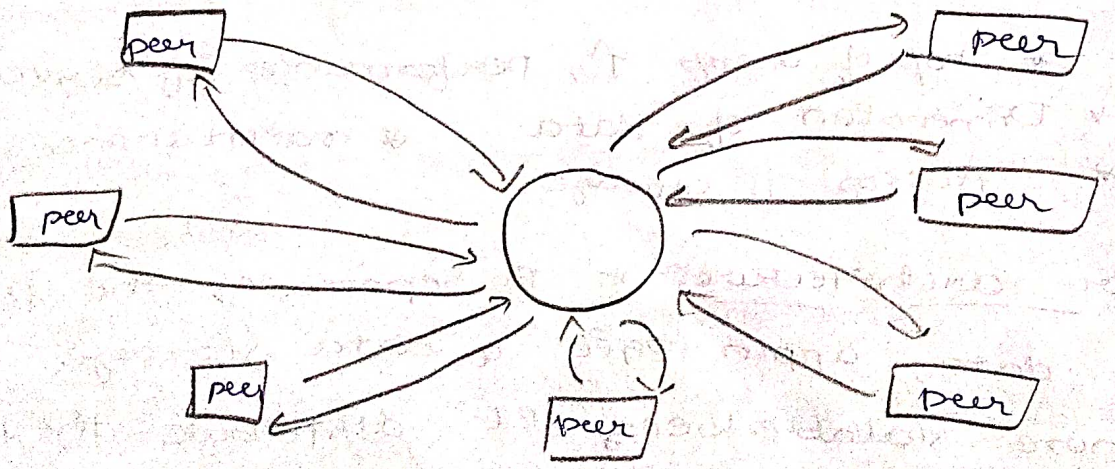


Client / server architectural styles

Peer-to-peer :- ⇒ plays the same role.

- Both client & server can acts as server & client.
- Same responsibility to each component.
- eg :- File-sharing applns : Gnutella, BitTorrent & kazza.

Disadvantage :- management of implementation of algorithms.



peer-to-peer architectural style.

3H. explain

Mesg - based Comm^{n} :- Abstractions that are presented to developers for progming the interaction of distributed components.

* It encompasses any form of data representation that is limited in size & time.

→ Mesg passing :- Exchanging information in the form of message.

eg :- Msg - passing Interface (MPI) & openMp.

→ Remote procedure call (RPC) :- triggers the execution of code in remote processes.

* Here client / server process take place.
* The use of msg within this context is referred as marshaling of parameters.

→ Distributed objects :-

* this is for implementation of RPC for object-oriented paradigm.
* the Comm^{n} b/w caller & remote process made through msgs.
* It is independent.

eg :- CORBA (common object request broker architecture) com, DCom, com+ (component object model), RMI (Java Remote method invocation) & .NET.

→ Distributed agents & active objects :- these are based on agents & presence of instances, where the agents of objects & the existance of requests.

* triggers execution of methods

* more complex semantic attached to msg.

→ Web Service:- * provides implementation of RPC over HTTP.

* web service as remote object hosted on web server.

eg:- SOAP (Simple object access protocol), REST (Representational state transfer).

35. explain Models for msg-based comm^n :-

* Point-to-point msg model:-

* comm^n among single components.

* Each msg is sent from one component to another.

* It must know how to address another component.

* comm^n initiated by sender.

* useful for implementing systems based on one-to-one or many-to-one comm^n.

Publish-and-subscribe msg model:-

* There are two major roles publisher & subscriber.

* publisher create messages attached to specific event.

* subscriber register to the event.

there are two major strategies for dispatching the event to the subscribers:-

(a) Push strategy:- It is responsibility of publisher to notify all subscribers.

(b) Pull strategy:- It is responsibility of subscribers to check msg on the event is registered or not.

* this is suitable for one-to many comm^n model

Request-reply msg model:-

* It checks for each msg sent by a process, there is a reply.

* point-to-point model based on this.