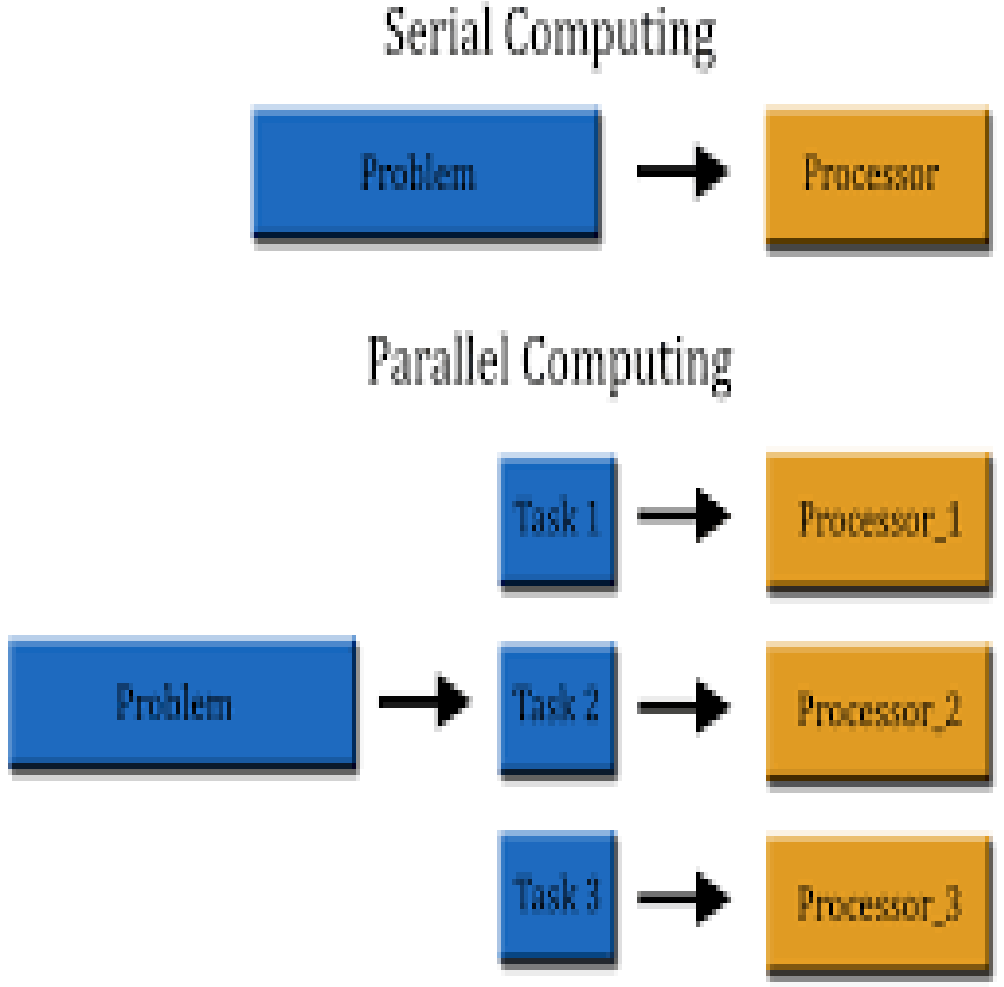
Cloud Programming, Resource Management and Scheduling:

By: Neelam Singh

Parallel & Distributed Programming Paradigms

- Parallel Computing: In parallel computing multiple processors performs multiple tasks assigned to them simultaneously. Memory in parallel systems can either be shared or distributed. Parallel computing provides concurrency and saves time and money.
- Parallel computing on a single computer uses multiple *processors* to process tasks in parallel.
- 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



- Parallel computers can be roughly classified according to the level at which the hardware supports parallelism, with multi-core and multi-processor computers having multiple processing elements within a single machine, while clusters, MPPs, and grids use multiple computers to work on the same task. Specialized parallel computer architectures are sometimes used alongside traditional processors, for accelerating specific tasks.

Why Use Parallel Computing?

- Compared to serial computing, parallel computing is much better suited for modeling, simulating and understanding complex, real world phenomena.
- SAVE TIME AND/OR MONEY
  - In theory, throwing more resources at a task will shorten its time to completion, with potential cost savings.
  - Parallel computers can be built from cheap, commodity components.
- SOLVE LARGER / MORE COMPLEX PROBLEMS
  - Many problems are so large and/or complex that it is impractical or impossible to solve them using a serial program, especially given limited computer memory.
  - Example: "Grand Challenge Problems" (en.wikipedia.org/wiki/Grand_Challenge) requiring petaflops and petabytes of computing resources.
  - Example: Web search engines/databases processing millions of transactions every second
- PROVIDE CONCURRENCY
  - A single compute resource can only do one thing at a time. Multiple compute resources can do many things simultaneously.
  - Example: Collaborative Networks provide a global venue where people from around the world can meet and conduct work "virtually".

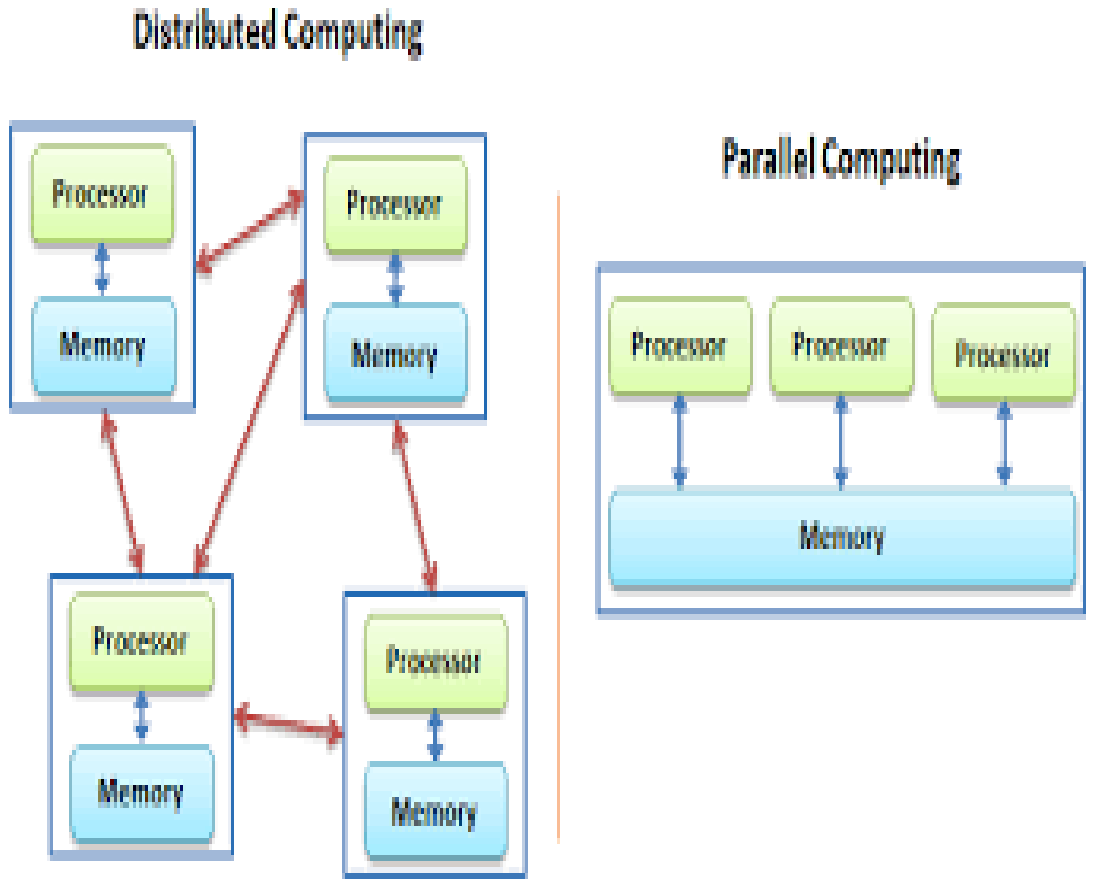
- TAKE ADVANTAGE OF NON-LOCAL RESOURCES
  - Using compute resources on a wide area network, or even the Internet when local compute resources are scarce or insufficient.
- MAKE BETTER USE OF UNDERLYING PARALLEL HARDWARE
  - Modern computers, even laptops, are parallel in architecture with multiple processors/cores.
  - Parallel software is specifically intended for parallel hardware with multiple cores, threads, etc.
  - In most cases, serial programs run on modern computers "waste" potential computing power.

Who is Using Parallel Computing?

- **Science and Engineering**
  - Historically, parallel computing has been considered to be "the high end of computing", and has been used to model difficult problems in many areas of science and engineering:
  - Atmosphere, Earth, Environment
  - Physics - applied, nuclear, particle, condensed matter, high pressure, fusion, photonics
  - Bioscience, Biotechnology, Genetics
  - Chemistry, Molecular Sciences
  - Geology, Seismology
  - Mechanical Engineering - from prosthetics to spacecraft
  - Electrical Engineering, Circuit Design, Microelectronics
  - Computer Science, Mathematics
  - Defense, Weapons
- **Industrial and Commercial**
  - Today, commercial applications provide an equal or greater driving force in the development of faster computers. These applications require the processing of large amounts of data in sophisticated ways. For example:
  - "Big Data", databases, data mining
  - Artificial Intelligence (AI)
  - Oil exploration
  - Web search engines, web based business services
  - Medical imaging and diagnosis
  - Pharmaceutical design
  - Financial and economic modeling
  - Management of national and multi-national corporations
  - Advanced graphics and virtual reality, particularly in the entertainment industry
  - Networked video and multi-media technologies

Distributed Computing:

- In distributed computing we have multiple autonomous computers which seems to the user as single system. In distributed systems there is no shared memory and computers communicate with each other through message passing. In distributed computing a single task is divided among different computers.



- A computer program that runs within a distributed system is called a **distributed program** (and distributed programming is the process of writing such programs).There are many different types of implementations for the message passing mechanism, including pure HTTP, RPC-like connectors and message queues.
- *Distributed computing* also refers to the use of distributed systems to solve computational problems. In *distributed computing*, a problem is divided into many tasks, each of which is solved by one or more computers, which communicate with each other via message passing.

Architectures

- Distributed programming typically falls into one of several basic architectures: client–server, three-tier, *n*-tier, or peer-to-peer; or categories: loose coupling, or tight coupling.
  - **Client–server**: architectures where smart clients contact the server for data then format and display it to the users. Input at the client is committed back to the server when it represents a permanent change.
  - **Three-tier**: architectures that move the client intelligence to a middle tier so that stateless clients can be used. This simplifies application deployment. Most web applications are three-tier.
  - ***n*-tier**: architectures that refer typically to web applications which further forward their requests to other enterprise services. This type of application is the one most responsible for the success of application servers.
  - **Peer-to-peer**: architectures where there are no special machines that provide a service or manage the network resources. Instead all responsibilities are uniformly divided among all machines, known as peers. Peers can serve both as clients and as servers. Examples of this architecture include BitTorrent and the bitcoin network.