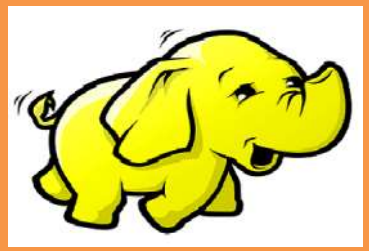
SNS COLLEGE OF TECHNOLOGY

UNIT IV

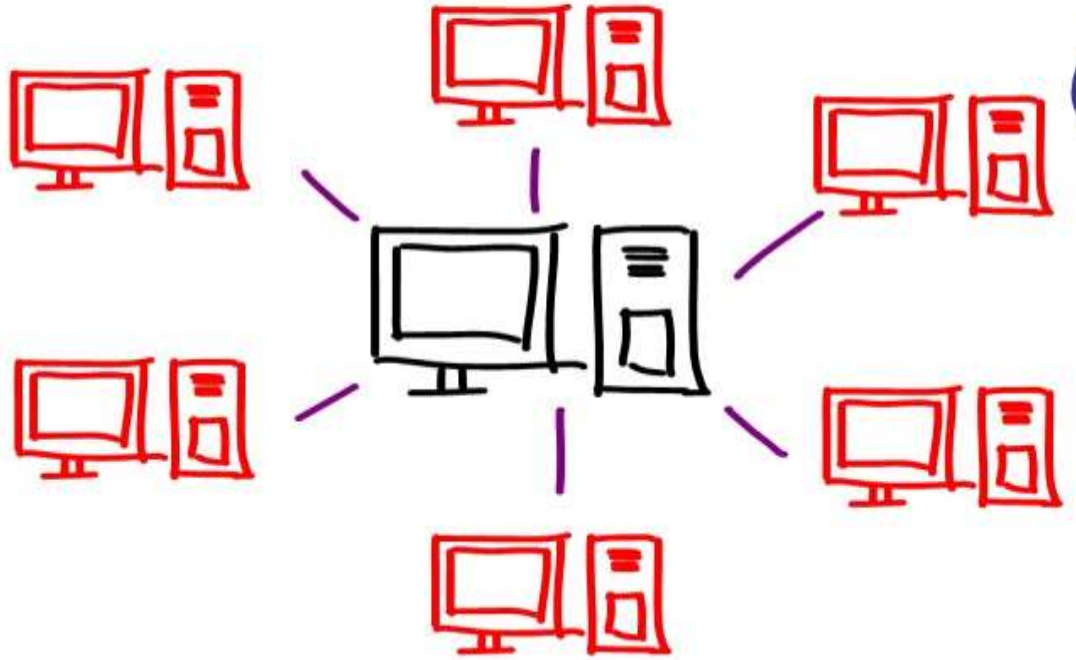
DEPARTMENT OF COMPUTER APPLICATIONS
SNS COLLEGE OF TECHNOLOGY
COIMBATORE – 64035



PROGRAMMING MODEL

- Parallel and Distributed Programming Paradigms
- MapReduce , Twister and Iterative MapReduce
- Hadoop Library from Apache
- Mapping Applications
- Programming Support - Google App Engine, Amazon AWS
- Cloud Software Environments -Eucalyptus, Open Nebula, OpenStack, Aneka, CloudSim

Distributed Computing



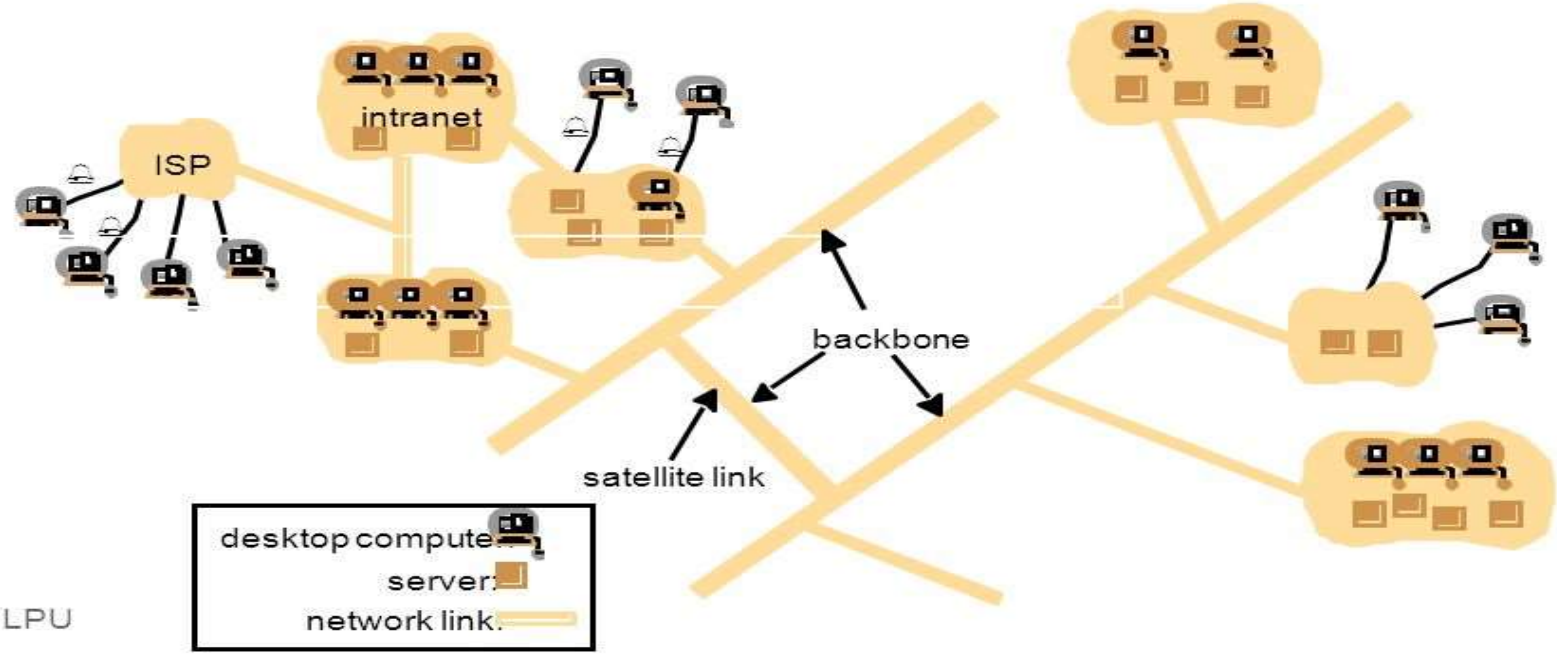
A set of computational engines connected by a network to achieve a common goal of running a job or an application

Distributed System: Example

Examples of Distributed Systems

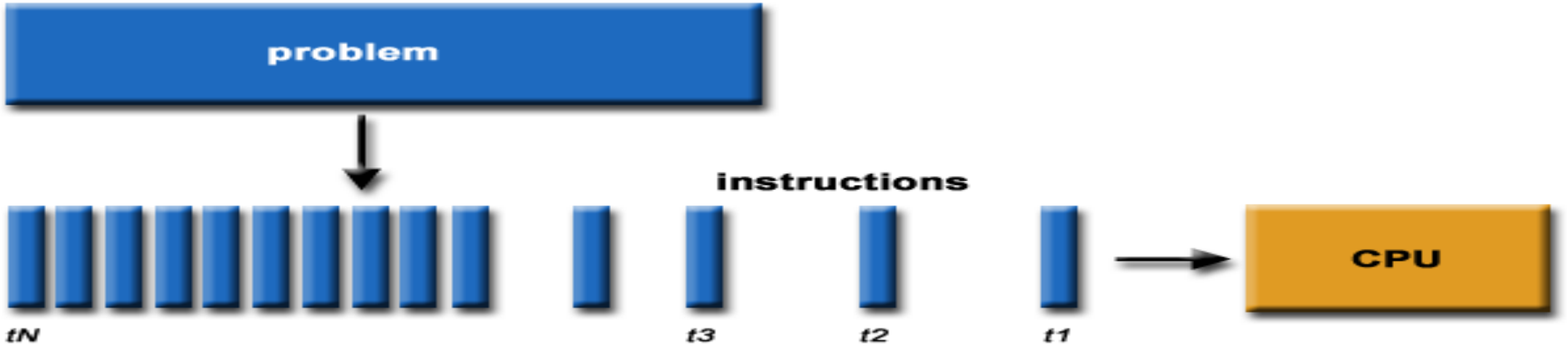
1. The internet

- Heterogeneous network of computers and applications
- Implemented through Internet protocol



Parallel Computing

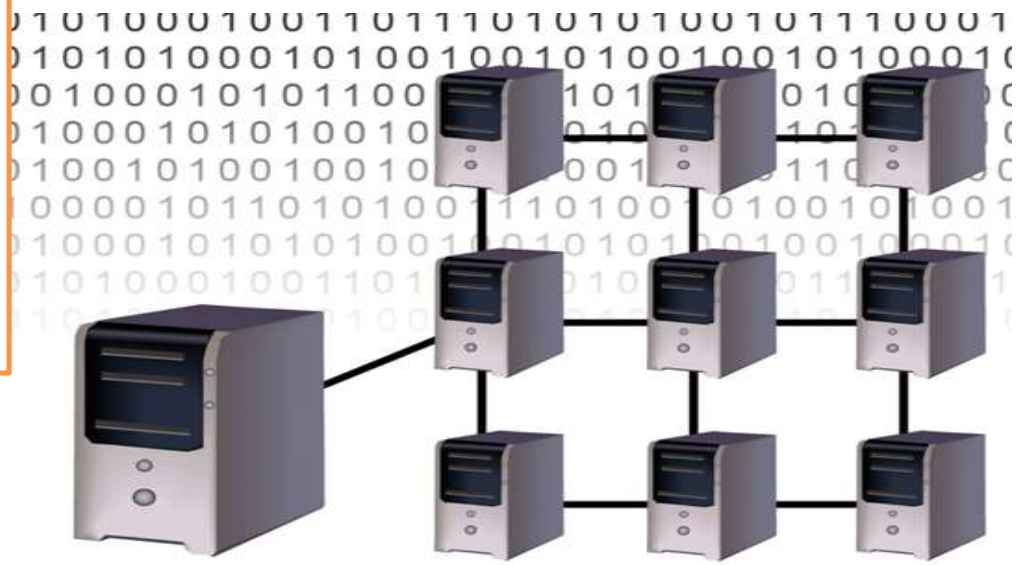
- Traditionally, software has been written for serial computation
- A problem is broken into a discrete series of instructions. Instructions are executed one after another



Parallel Computing

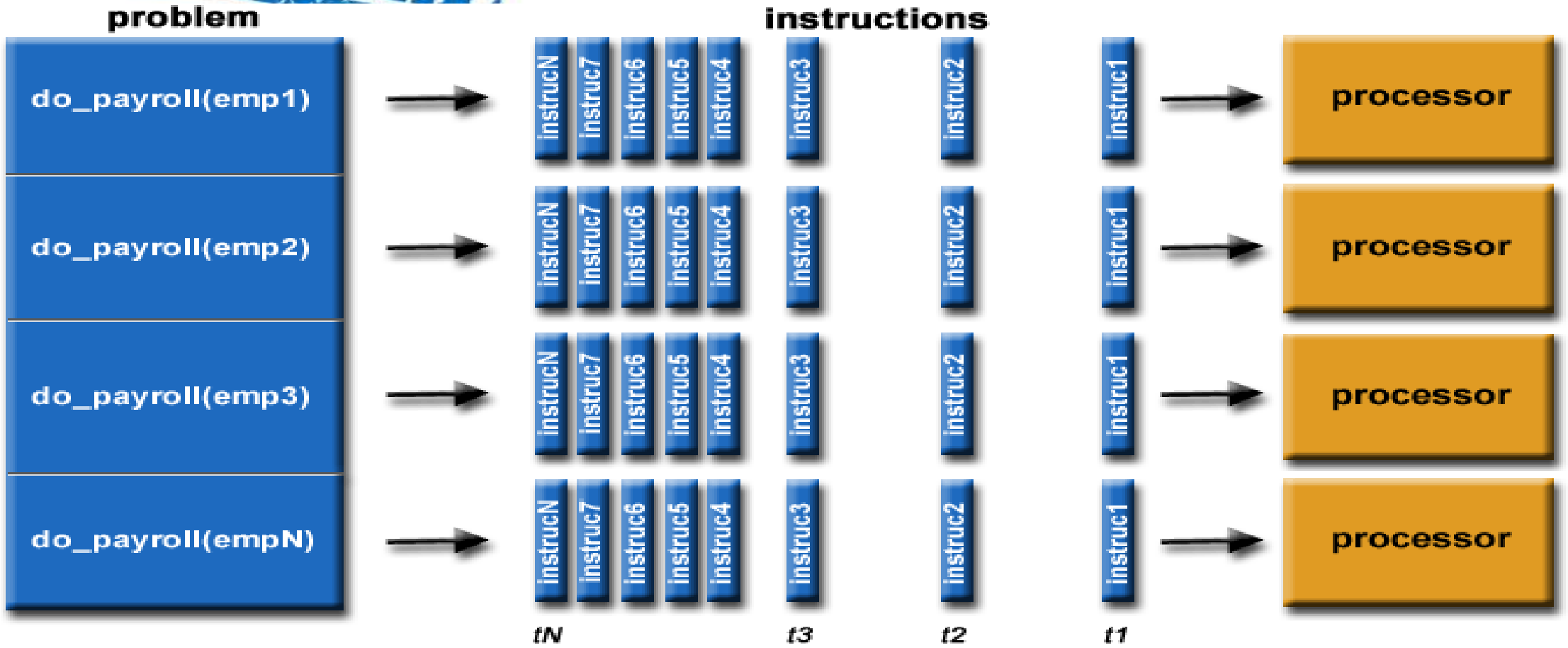
It is the simultaneous use of more than one computational engine (not necessarily connected via a network) to run a job or an application

- It may use either a distributed or a non-distributed computing system
- Advantage: decreases application response time; it increases throughput and resource utilization



Parallel Computing

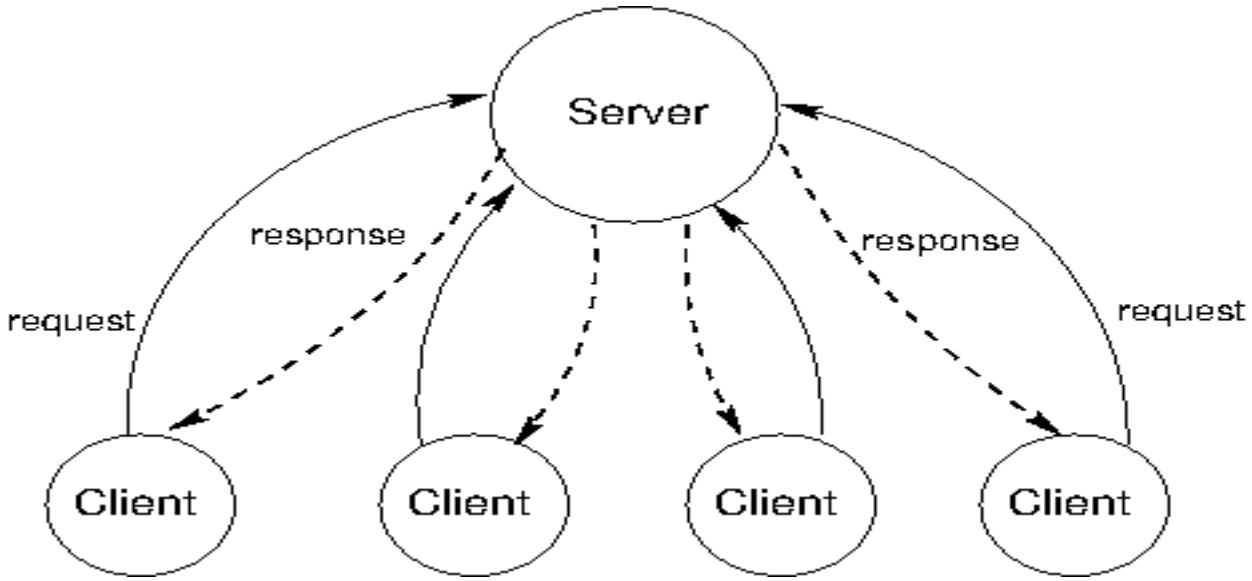
EXAMPLE



Parallel Vs Distributed Computing

Parallel System	Distributed system
Tightly coupled system Shared memory	Weakly coupled system Distributed memory
Global clock control SIMD, MIMD	No Global clock control Synchronization algorithm used
Order of Tbps (Processor interconnection	Order of Gbps
Performance Scientific computing	Performance Reliability / Availability Information/Resource sharing

Distributed System Parallel computing



PROCESS

- **Partitioning:** applicable to both computation and data
- **Computation partitioning:** Splits a given job/program into smaller task based on identifying portions of the job run concurrently
- **Data partitioning:** splits data into smaller pieces
- **Mapping:** assign smaller parts of a program/pieces of data to underlying resources

PROCESS

- **Synchronization:** Synchronization and coordination among workers is necessary so that race conditions are prevented and data dependency among different workers is properly managed
- **Communication:** It is always triggered when the intermediate data is sent to workers
- **Scheduling:** When the number of computation parts (tasks) or data pieces is more than the number of available workers, a scheduler selects a sequence of tasks or data pieces to be assigned to the workers

MOTIVATION

- requires specialized knowledge of programming
- implicity of writing parallel programs is an important metric for parallel / distributed programming paradigms
- to improve productivity of programmers
- to decrease programs' time to market
- to leverage underlying resources more efficiently

MOTIVATION

- to increase system throughput
- to support higher levels of abstraction
- loose coupling of components in these paradigms makes them suitable for VM implementation and leads to much better fault tolerance and scalability

Solution ?

Thank You