



Department of Computer Science

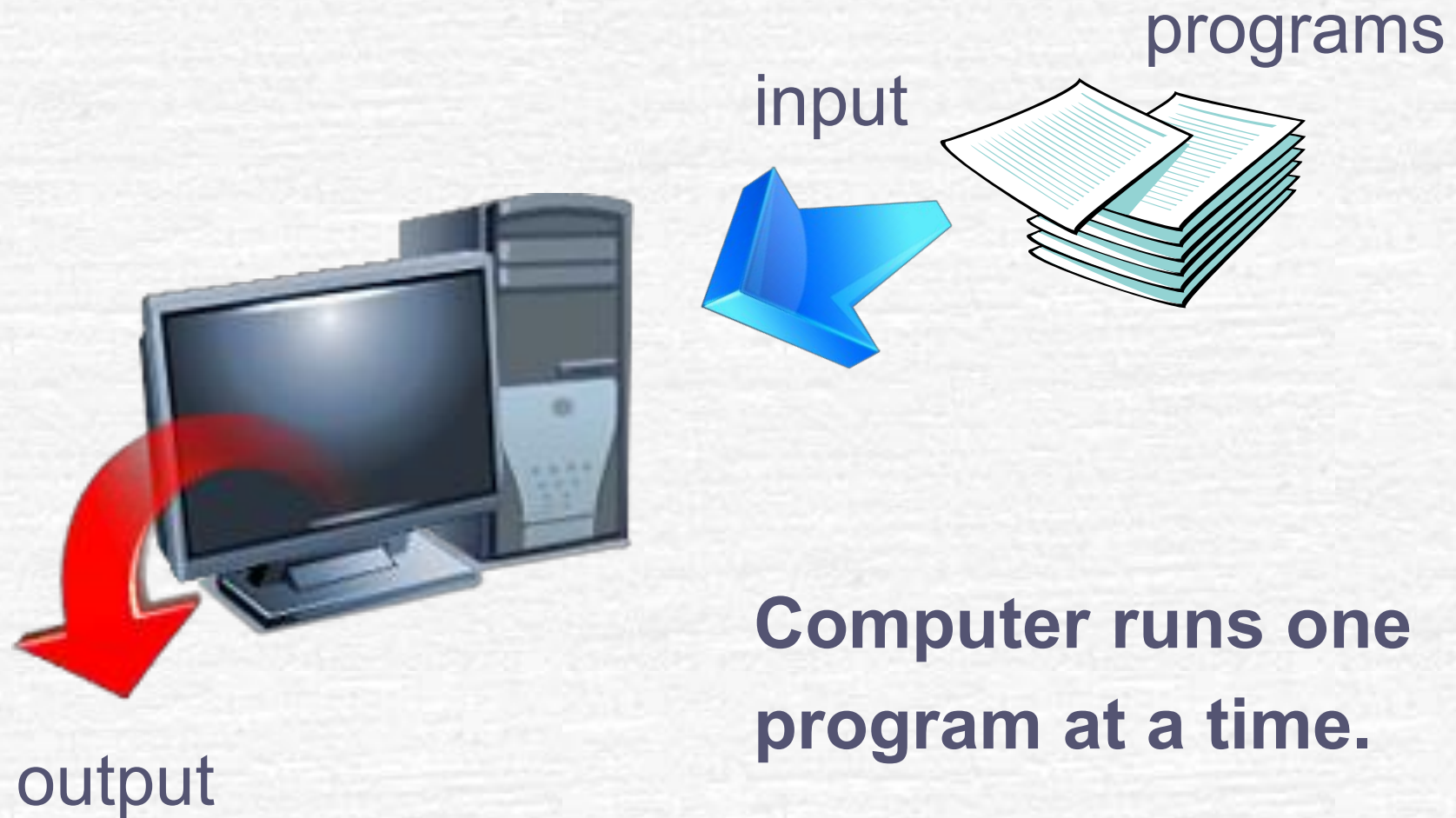
# Introduction to Parallel Computing

Sofien GANNOUNI

Computer Science

E-mail: [gnnosf@ksu.edu.sa](mailto:gnnosf@ksu.edu.sa) ; gansof@yahoo.com

# Serial hardware and software



# Parallelism

## **Parallel Computing:**

- is a fundamental technique by which computations can be accelerated.
- is a form of computation in which many calculations are carried out simultaneously.

## **Parallel Programming:**

- Decomposing a programming problem into tasks
- Deploy the tasks on multiple processors and run them simultaneously
- Coordinating work and communications of those processors



# Parallel Computers

classified according to the level at which the hardware supports parallelism:

- **Multi-core** and **multi-processor** (Symmetric Multiprocessing) computers

- **Multi-core** computers have multiple processing elements (cores) within a single machine.
- **Symmetric Multiprocessing** (SMP) - multiple processors sharing a single address space, OS instance, storage, etc. All processors are treated equally by the OS

- **Clusters** and **grids** use multiple computers to work on the same task.

**Static** ● **Cluster**: A parallel system consisting of a combination of commodity units (processors or SMPs).

- **Grid**: is a group of networked computers that work together, only when idle, as a virtual supercomputer to perform large tasks.

**fixed**

}

# Why do we need Parallel Processing?

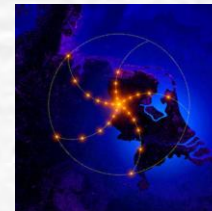
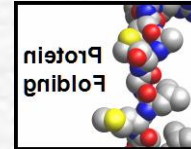
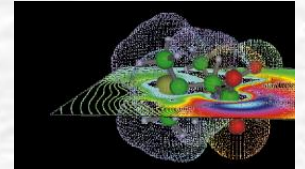
web

OX:

- Many applications need much faster machines
- Sequential machines are reaching their speed limits.
- Use multiple processors to solve large problems faster.
- Microprocessors are getting cheaper and cheaper.
- Cheap multiprocessors and multi-core CPUs bring parallel processing to the desktop.

# Challenging Applications

- Modeling ozone layer, climate, ocean
- Quantum chemistry
- Protein folding
- General: *computational science*
- Aircraft modeling
- Using volumes of data from scientific instruments
  - astronomy
  - high-energy physics
- Computer chess
- Analyzing multimedia content





# History

---

- ✎ 1950s: first ideas (see Gill's quote)
- ✎ 1967 first parallel computer (ILLIAC IV)
- ✎ 1970s programming methods, experimental machines
- ✎ 1980s: parallel languages (SR, Linda, Orca), commercial supercomputers
- ✎ 1990s: software standardization (MPI), clusters, large-scale machines (Blue Gene)
- ✎ 2000s: grid computing: combining resources world-wide (Globus)

# Opportunities to use parallelism

## **Instruction Level Parallelism**


- Hidden Parallelism in computer programs

## **Single computer level**

- Multi-core computers: Chip multi-processors
  - Dual-core, Quad-core
- Multi-processor computers: Symmetric multi-processors
  - Super-computers

## **Multiple computers level**

- Clusters, Servers, Grid computing

 **Need for programs that have multiple instruction streams**



# Parallelism in Computer Programs

- ☛ The main motivation for executing program instructions in parallel is to complete a computation faster.
- ☛ But programs are written assuming that:
  - Instructions are executed in **sequential**.
  - Most programming languages embed sequential execution.

# Instruction Level Parallelism (IPL)

☞  $(a+b) * (c+d)$

- could be computed simultaneously.

☞ **Separation of instructions and data.**

- Instructions and memory references execute in parallel without interfering.

☞ **Instruction Execution is pipelined**

☞ **Processors initiate more than one instruction at a time.**

# How do we write parallel programs?

## Task parallelism

- Partition various tasks carried out solving the problem among the cores.

## Data parallelism

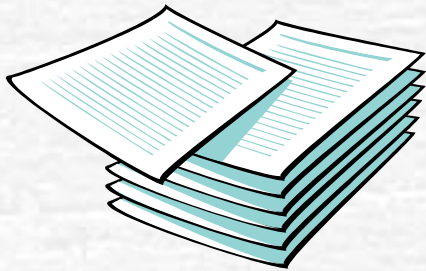
- Partition the data used in solving the problem among the cores.
- Each core carries out similar operations on it's part of the data.



# Professor P

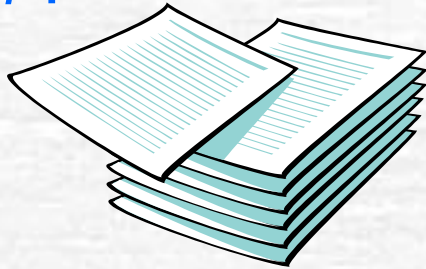
15 questions

300 exam copies

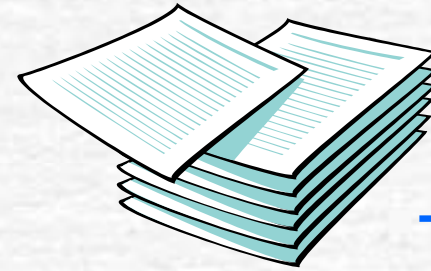


# Division of work – data parallelism

TA#1

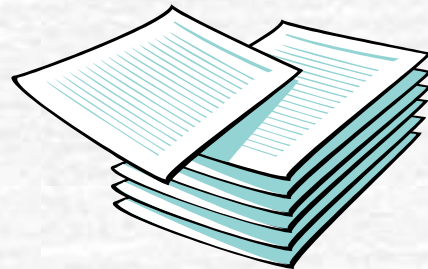


100 exams



TA#3

100 exams



TA#2

100 exams

# Division of work – task parallelism

TA#1



Questions 1 - 5

or Questions 1 - 7



TA#3

Questions 11 - 15

or Questions 12 - 15



TA#2

Questions 6 - 10

or Questions 8 - 11

Partitioning strategy:  
- either by number  
- Or by workload



# Parallel Computing vs. Distributed Computing

## Parallel computing

- Provides performance that a single processor can not give.
- Interaction among processors is frequent
  - Fine grained with low overhead
  - Assumed to be reliable.

## Distributed Computing

- Provides convenience:
  - Availability, Reliability
  - Physical distribution, Heterogeneity
- Interaction is infrequent,
  - Coarse grained - heavier weight
  - Assumed to be unreliable.

# Distributed Systems

## What is a Distributed System?

✍ A distributed system is one that looks to its users like an ordinary, centralized, system but runs on multiple independent CPUs

## ✍ Symptoms? Shroeder:

- Multiple, independent processing units
- Processors communicate via a hardware interconnect
- Processing unit failures are independent: No single point of failure
- No global clock
- State is shared among processors

# Aspects of Parallel Computing

## Parallel Computers Architecture

## Algorithms and applications

- Reasoning about performance
- Designing parallel algorithms.

## Parallel Programming

- Paradigms
- Programming Models
- Programming languages
- Frameworks
- Dedicated environments