UNIVERSITY OF ENGINEERING AND TECHNOLOGY, TAXILA

FACULTY OF TELECOMMUNICATION AND INFORMATION ENGINEERING

COMPUTER/SOFTWARE ENGINEERING DEPARTMENT

PARALLEL PROCESSING

2K5-CP,SE

INSTRUCTOR: ENGR.WAJAHAT ABBAS

Lecture-2

Parallel Processing Concepts and Definitions

Concepts and Terminology
Some General Parallel Terminology

§ Task

- A logically discrete section of computational work. A task is typically a program or program-like set of instructions that is executed by a processor.

§ Parallel Task

- A task that can be executed by multiple processors safely (yields correct results)

§ Serial Execution

- Execution of a program sequentially, one statement at a time. In the simplest sense, this is what happens on a one processor machine. However, virtually all parallel tasks will have sections of a parallel program that must be executed serially.

§ Parallel Execution

- Execution of a program by more than one task, with each task being able to execute the same or different statement at the same moment in time.

§ Shared Memory

- From a strictly hardware point of view, describes a computer architecture where all processors have direct (usually bus based) access to common physical memory. In a programming sense, it describes a model where parallel tasks all have the same "picture" of memory and can directly address and access the same logical memory locations regardless of where the physical memory actually exists.

§ Distributed Memory

- In hardware, refers to network based memory access for physical memory that is not common. As a programming model, tasks can only logically "see" local machine memory and must use communications to access memory on other machines where other tasks are executing.

§ Communications

- Parallel tasks typically need to exchange data. There are several ways this can be accomplished, such as through a shared memory bus or over a network, however the actual event of data exchange is commonly referred to as communications regardless of the method employed.

§ Synchronization

- The coordination of parallel tasks in real time, very often associated with communications. Often implemented by establishing a synchronization point within an application where a task may not proceed further until another task(s) reaches the same or logically equivalent point. Synchronization usually involves waiting by at least one task, and can therefore cause a parallel application's wall clock execution time to increase.

§ Granularity

- In parallel computing, granularity is a qualitative measure of the ratio of computation to communication.
- Coarse: relatively large amounts of computational work are done between communication events
- Fine: relatively small amounts of computational work are done between communication events

§ Observed Speedup

- Observed speedup of a code which has been parallelized, defined as:

$$\frac{\text{Wall-clock time of serial execution}}{\text{Wall-clock time of parallel execution}}$$

- One of the simplest and most widely used indicators for a parallel program's performance.

§ Parallel Overhead

- The amount of time required to coordinate parallel tasks, as opposed to doing useful work. Parallel overhead can include factors such as:
    - Task start-up time
    - Synchronizations
    - Data communications
    - Software overhead imposed by parallel compilers, libraries, tools, operating system, etc.
    - Task termination time

§ Massively Parallel

- Refers to the hardware that comprises a given parallel system - having "many" processors. The meaning of "many" keeps increasing, but currently means more than "1000".

§ Scalability

- Refers to a parallel system's (hardware and/or software) ability to demonstrate a proportionate increase in parallel speedup with the addition of more processors.
- Factors that contribute to scalability include:
    - Hardware - particularly CPU-Memory bandwidth and network communications
    - Application algorithm
    - Parallel overhead related
    - Characteristics specific to application and coding