INTRODUCTION TO PARALLEL PROGRAMMING

Yi Nutanong [1]

[1] Department of Computer Science & Software Engineering,
The University of Melbourne

EXTENDED TUTORIAL ABSTRACT

As the demand for computational power has grown beyond what the present uniprocessor technology can offer, parallel computing seems to be the most logical way to meet the demand. In parallel computing, the computational speed is increased by using multiple processing elements (PE's) to cooperatively solve a single problem. Writing a program that performs such computation is called *parallel programming*.

The main objective of this tutorial is to familiarise the participants with parallel programming. It focuses on techniques and tools for parallel programming.

This tutorial session is intended for new parallel programmers and undergraduate students who are potentially considering *parallel and distributed computing* as a research area for their further studies.

There is no explicit prerequisite for this tutorial. However, some background knowledge of operating systems and computer networks is helpful.

This tutorial covers the following topics:

- Parallel Architectures
- Parallel Complexity and Performance Issues
- Parallel Programming Methodologies
- Parallel Programming Techniques and Tools

The “Parallel Architectures” topic briefly reviews parallel computer architectures: SISD, SIMD, MISD, and MIMD. This topic emphasises on the two classes of MIMD machines: shared-memory, and distributed-memory. It also covers the cluster machine architecture.

The “Parallel Complexity and Performance Issues” topic concerns what type of work is available to be done in parallel, and what we need to take care of, in order to build an efficient parallel program. The discussion includes issues like: levels of parallelism, granularity, and speed-up factor.

The “Parallel Programming Methodologies” topic discusses the two main methodologies for parallel programming: data decomposition and algorithm decomposition. In data decomposition, the data is divided into several parts, where they are processed by duplicates of the main operation in parallel. In functional decomposition, on the other hand, the parallelism is obtained by decomposing the main algorithms into several sub-algorithms and running them simultaneously on the same data.

The topic on “Parallel Programming Techniques and Tools” demonstrates two different techniques: multi-threaded programming, and message-passing programming. For the multi-threaded programming, the discussion covers the basic concept, then Java Threads programs and POSIX Threads (PThread) programs are demonstrated. For message passing programming, two message passing tools: Message Passing Interface (MPI), and Parallel Virtual Machine (PVM) are demonstrated.