Chapter 1

Introduction

1.1 Concurrency and Parallelism

A concurrent program, unlike a sequential program, has multiple threads of control which may be executed as parallel processes.
A *thread* and *process* in Computer Science have multiple, overlapping, and inconsitent meanings. Generally, a *"process"* is associated with an instance of a running software program, which contains its own address space, whereas a *"thread"* refers to a *"thread of control"* within a given process and therefore does not contain its own address space but shares the space of the process with multiple other threads.

In order to execute a concurrent program different approaches can be chosen:

- MULTIPROGRAMMING:
  The processes are sharing one or more processors.
- MULTIPROCESSING:
  Each process runs on its own processor with access to a shared memory.
- DISTRIBUTED PROCESSING:
  Each process runs on its own processor while being connected to others via a network.

1.2 CHALLENGES: Safety and Liveness

1.3 Expressing Concurrency

1.3.1 Process Creation

1.3.2 Communication and Synchronization