Objective of this lecture:

- Distributed programming paradigms
- Models of distributed computation: synchronous vs asynchronous systems, failure models

Reading list: Mullender Chapter 5.1-5.2, pp 97-102,
### Parameters in Computation Models

- Synchronous vs. asynchronous
- Failure semantics (for processes)
- Failure semantics (for communication)
- Network topology
- Deterministic vs. randomized processes
Synchronous vs. Asynchronous Systems

- A system is **asynchronous** if there is no bound on message delay, clock drift, or the time to execute a step.

- A system is **synchronous** if it satisfies the following properties:
  1. Known upper bound $\delta$ on message delay
  2. Every process $p$ has a local clock $C_p$ with a known bounded rate of drift $\rho \geq 0$ with respect to real-time. i.e., $\forall p$ and $\forall t > t'$,
     \[
     (1 + \rho)^{-1} \leq \frac{C_p(t) - C_p(t')}{t - t'} \leq (1 + \rho)
     \]
  3. Known upper and lower bounds on the time required by a process to execute a step.

- Observation: For a synchronous system, it is possible to implement approximately synchronized clocks even in the presence of failures!

- Observation: variable or unexpected workloads are sources of asynchrony $\rightarrow$ synchronous assumptions are probabilistic!

---

Process Failure Models

A model of failure specifies in what way a faulty process can deviate from its prescribed specification.

- **Failstop**: A process fails by halting and remains in that state; the failure is detectable.

- **Crash**: A process fails by halting and remains in that state; the failure may not be detectable.

- **Send omission**: A faulty process fails by halting or by transmitting only a subset of the messages that it attempts to send.

- **Receive omission**: A process fails by halting or by receiving only a subset of the messages that has been sent to it.

- **General omission**: send and/or receive omission.
Byzantine failures: A process fails by exhibiting arbitrary behavior, e.g., change its state arbitrarily.

Timing failures: A process fails in one/more of the following ways:
- Commits general omission failures.
- Local clock drift exceeds the specified bound (clock failure).
- Violates the bound on the time to execute a step (performance failure).

Communication Failures

- Crash: A faulty link stops transporting messages.
- Omission: A faulty link intermittently omits to transport.
- Byzantine failures: A faulty link exhibit arbitrary behavior. e.g., generate extra messages.
- Timing failures: A faulty link transports messages faster or slower than its specification.