# COL380

## Introduction to
## Parallel & Distributed Programming

# Formal Models of Computation

- Simplify specifying, reasoning, analyzing algorithms
- Must abstract away many details
  - ➡ Should predict computability
- Should track performance
- General classes
  - ➡ Shared Memory vs Distributed Memory
  - ➡ Synchronous vs Asynchronous

# Concurrent/Parallel Computation Models

- Communicating Sequential Processes
- Actors
- Parallel Random Access Machine
- Bulk Synchronous Parallel computation

# CSP

- Compose sequential processes passing messages
  - ➡ Synchronous: send completes when message received (and vice versa)
  - ➡ Processes names known to senders (used for send and receive)
- Sender?gotvalue || Recipient!somevalue
- Guard; sender?P -> {post arrival code}
  - ➡ Wait for predicate to become true (or fail if it becomes false)

```
[ Sender1? msg -> process(msg, Q) ||
  Sender2? msg -> process(msg, R)) ||
  Sender3? msg -> process(msg, S)    ]
```

(Selection)

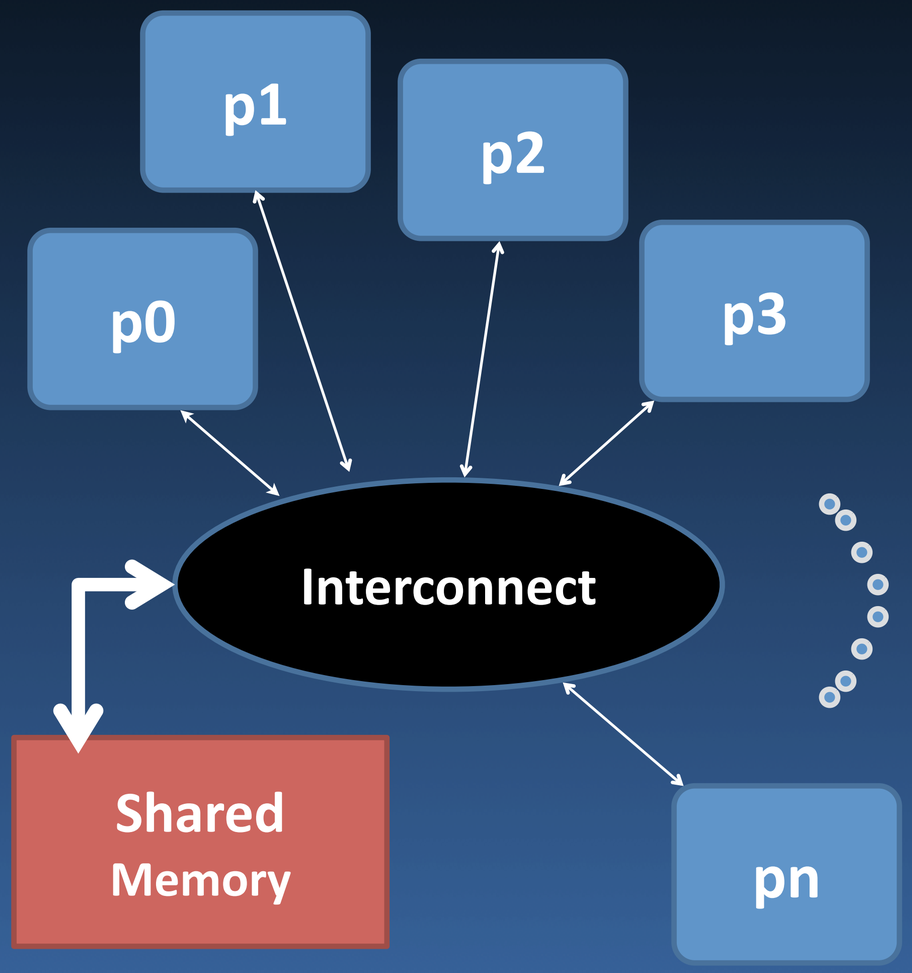
[Hoare,"Communicating sequential processes,". CACM 21 (8), 1978]

# Actors Model

- Actors: autonomous computing agents
  - ➡ No shared state; interact with each other using messages
    - ▸ Asynchronous, Lossless, Unordered
    - ▸ Have a (addressed) mailbox for communication (allows buffering)
- Actors process messages in their mailbox, in response:
  - ➡ execute a specified method
  - ➡ sends zero or more messages
  - ➡ creates zero or more new actors
  - ➡ changes its own local state (impacts the next message)

[See: Hewitt, Bishop, Steiger,"A Universal Modular Actor Formalism for Artificial Intelligence," IJCAI 1973]

# PRAM Model

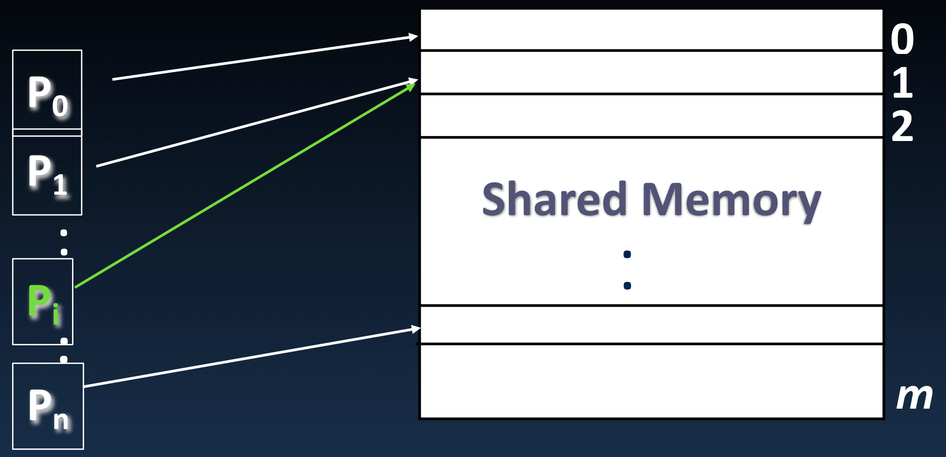



- Synchronous, Shared Mem
  - ➡ *Arbitrary* number of cells
- Arbitrary number of processors, each:
  - ➡ has local memory (*Arbitrary* number of cells)
  - ➡ knows its ID
  - ➡ can access a shared memory location in *constant* time

Unrealistic?

Can be often simulated

# PRAM Model Steps




- At each time-step each $P_i$ can:
  1. read some memory cell
  2. perform a local computation step
  3. write a memory cell (Read and write are in two phases)

     ➡ Co-access may be restricted
- Thus, a pair of processors $P_i$ $P_j$ can communicate in two steps

  ➡ constant time

# More about PRAM

- Inputs/Outputs are placed at designated addresses
  - ➡ Technically also a ‘start’ protocol to activate processors
- Each instruction takes O(1) time
- Processors are synchronous
  - ➡ Asynchronous PRAM models exist as well
- Cost analysis:
  - ➡ Cost, Work, Time (taken by the longest running processor)
  - ➡ Maximum number of active processors and memory cells