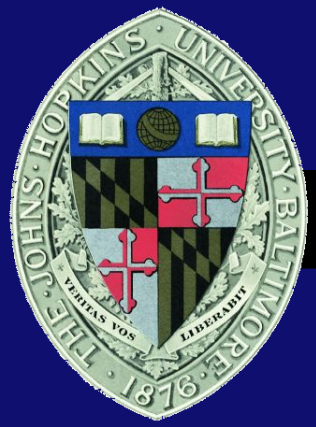
Lecture 3.1
Factors Against Parallelism

EN 600.320/420/620

Instructor: Randal Burns

7 February 2018



Department of Computer Science, *Johns Hopkins University*

3 Factors Against Parallelism

- **Startup costs** associated with initiating processes
  - May often overwhelm actual processing time (rendering ||ism useless)
  - Involve thread/process creation, data movement

- **Interference**: slowdown resulting from multiple processors accessing shared resources
  - Resources: memory, I/O, system bus, sub-processors
  - Software synchronization: locks, latches, mutex, barriers
  - Hardware synchronization: cache faults, HW mutexes and locks

- **Skew:** when breaking a single task into many smaller tasks, not all tasks may be the same size
  - Not all tasks finish at the same time

Example: Startup Costs

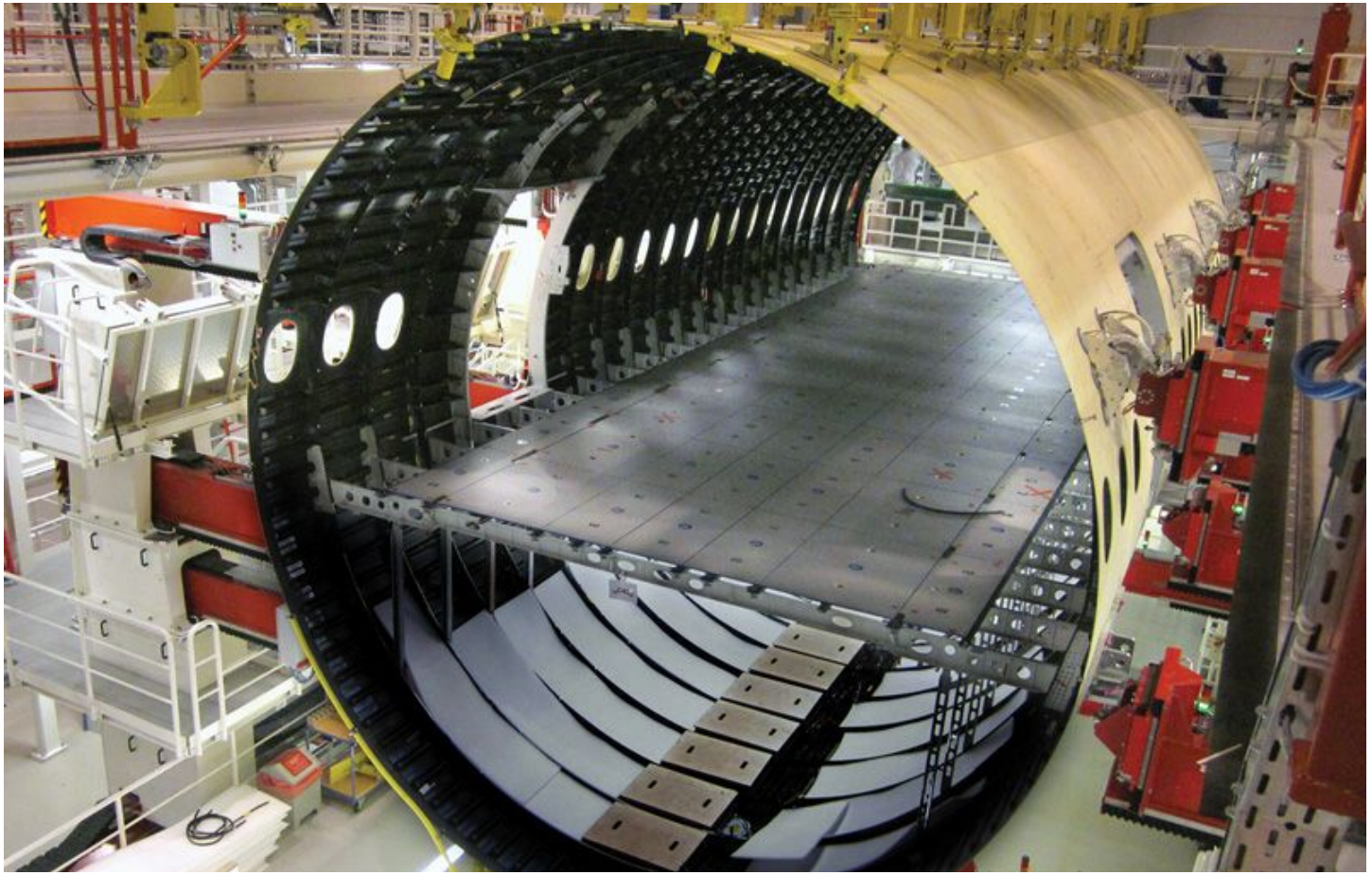
- Szkieletor in Krakow Poland
  - *Too expensive to complete or demolish*

Image from http://en.wikipedia.org/wiki/File:Szkieletor2.jpg

Example: Interference

Image from
http://crowdcentric.net/2011/05/can-you-help-us-solve-the-worlds-traffic-congestion/

Example: Skew



Airbus A350 waiting for the incomplete nose section

http://www.ainonline.com/?q=aviation-news/dubai-air-show/2011-11-12/

Factors and Communication

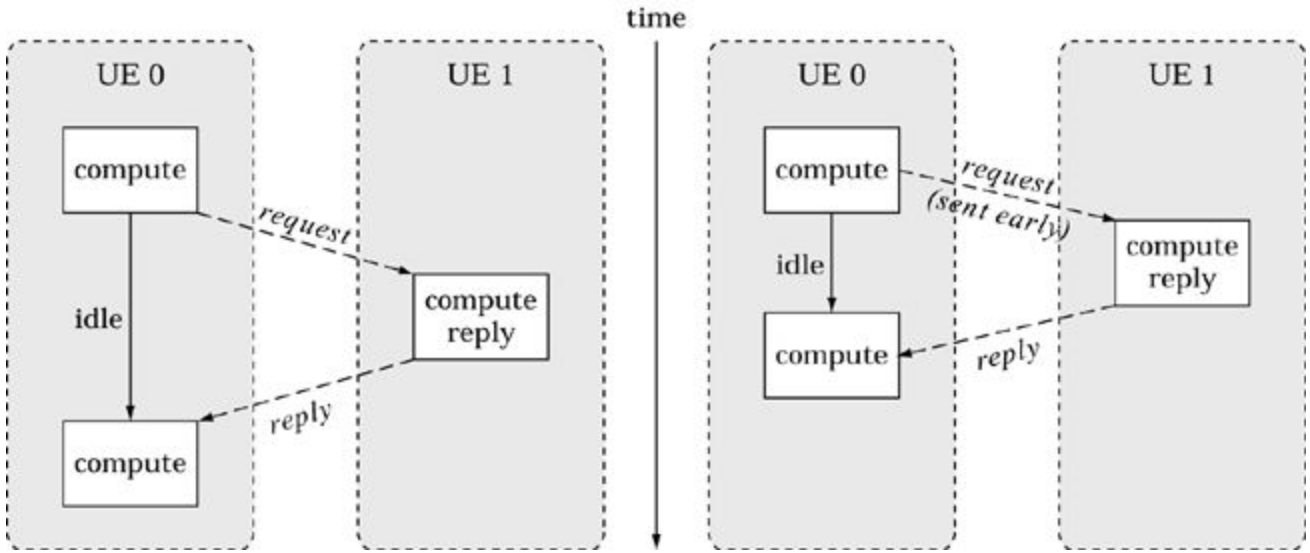
- Real things that degrade parallelism
  - I/O (memory and storage)
  - Network communication
  - Failures—particularly slow/failed processes
- But, HPC community focuses on communication (among processes) as the major source
  - This can be via memory or networking
  - I/O may be as important now: can be modeled as communication. One way? Often bundled as startup costs.

Communication

- Parallel computation proceeds in phases
  - Compute (evaluate data that you have locally)
  - Communicate (exchange data among compute)
- Communication is governed by:
  - Latency: fixed cost to send a message
    - Round trip time (speed of light and switching costs)
  - Bandwidth: marginal cost to send a message
    - Link capacity
- Latency dominates small messages and bandwidth dominates large
  - Almost always better to increase message size for performance, but difficult to achieve in practice

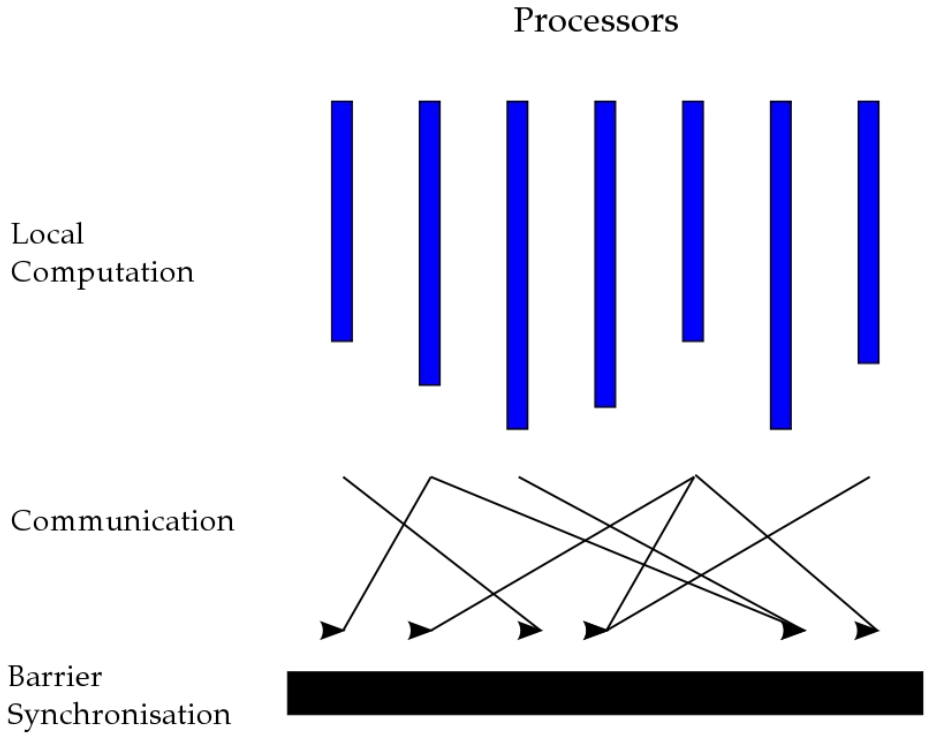
Overlapped Communication

- Messaging and computation that occur in parallel are *overlapped*
  - Reduces wait time between computing phases
  - Best codes overlap



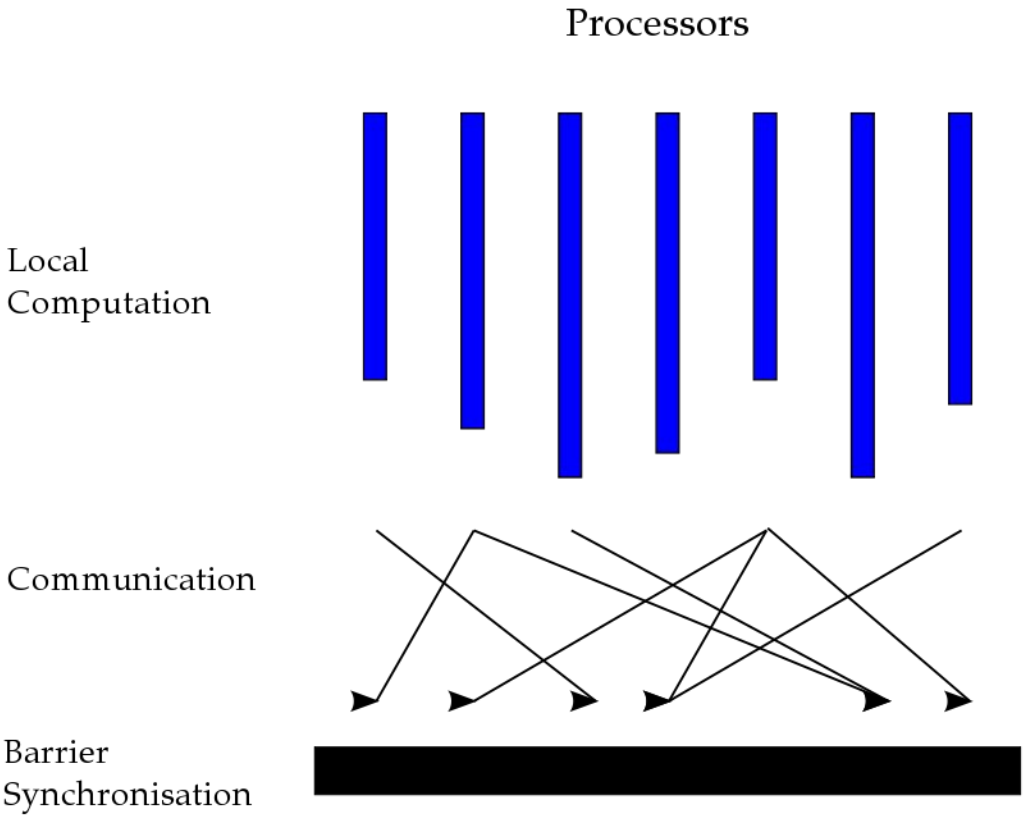
Bulk Synchronous Parallel (BSP)

- A natural abstraction for parallel computation
  - Used in most MPI and map/reduce programs
- Three phases of processing per “superstep”
  - Compute
  - Communication
  - Barrier
- Allows for no overlap
- Cloud framework
  - Hama



Amdahl's Law and BSP

- Any area that is not blue contributes to the 'sequential' part of the code, i.e. unoptimized
  - Communication
  - Barrier
  - Skew



Conclusions

- Factors against parallelism are the most important design consideration.
  - This is the non-parallel part in Amdahl's law
- Typical experience
  - Design a parallel code
  - Test on n=2 or 4 nodes (works great)
  - Deploy on >16 nodes (sucks eggs)
  - Measure factors against parallelism
  - Redesign/reimplement