CSE 6230 – HPC Tools and Applications

Ramakrishnan Kannan

Shruti Shivakumar

Day II

Recap

- Terminologies
  - Parallel Computing, HPC, Cluster Computing, Many/Multiple Cores, Embarassingly parallel, Pipelining
- Speedup, Work, Efficiency
- Amdahls Law
  - Model, Challenges, Hope
- Scaling
  - Weak and Strong Scaling

ARCHITECTURAL TAXONOMIES

- These classifications provide ways to think about problems and their solution.
- The classifications were originally in terms of hardware, but there are natural software analogues.
- Many systems blend approaches, and do not exactly correspond to the classifications.

$$\left\{ \begin{array}{c} \mathrm{S} \\ \mathrm{M} \end{array} \right\} \mathrm{I} \left\{ \begin{array}{c} \mathrm{S} \\ \mathrm{M} \end{array} \right\} \mathrm{D}$$

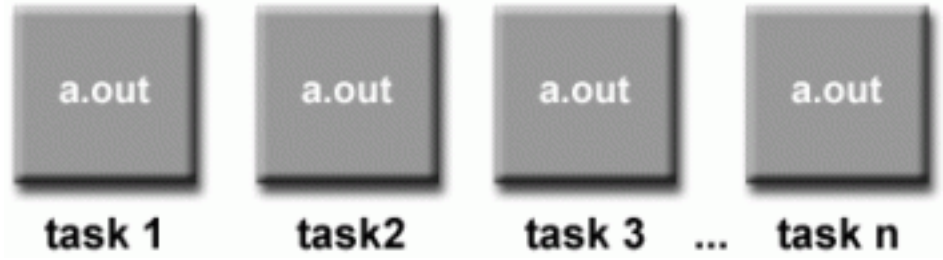
SI : Single **I**nstruction: All processors execute the same instruction.

MI : Multiple **I**nstruction: Different processor may be executing different instructions.

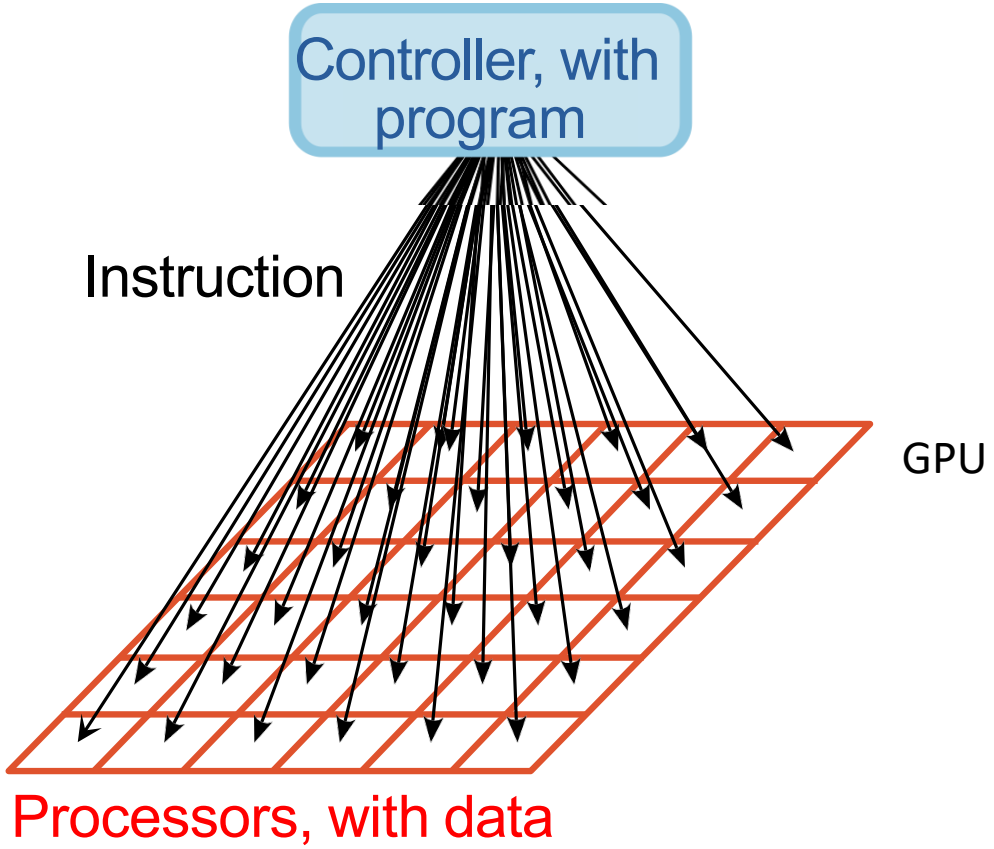
SD : Single **D**ata: All processors are operating on the same data.

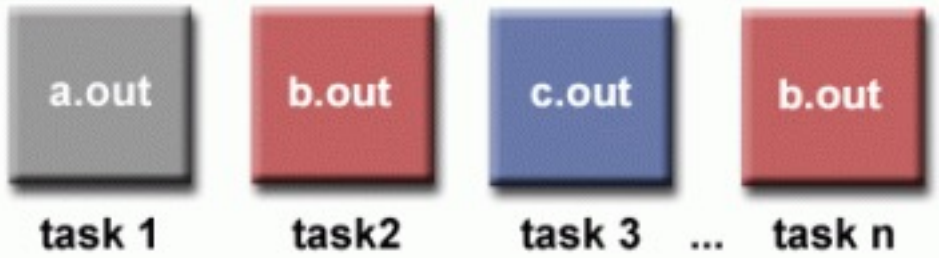
MD: Multiple **D**ata: Different processors may be operating on different data.

SIMD & MIMD



SIMD – Parallel for

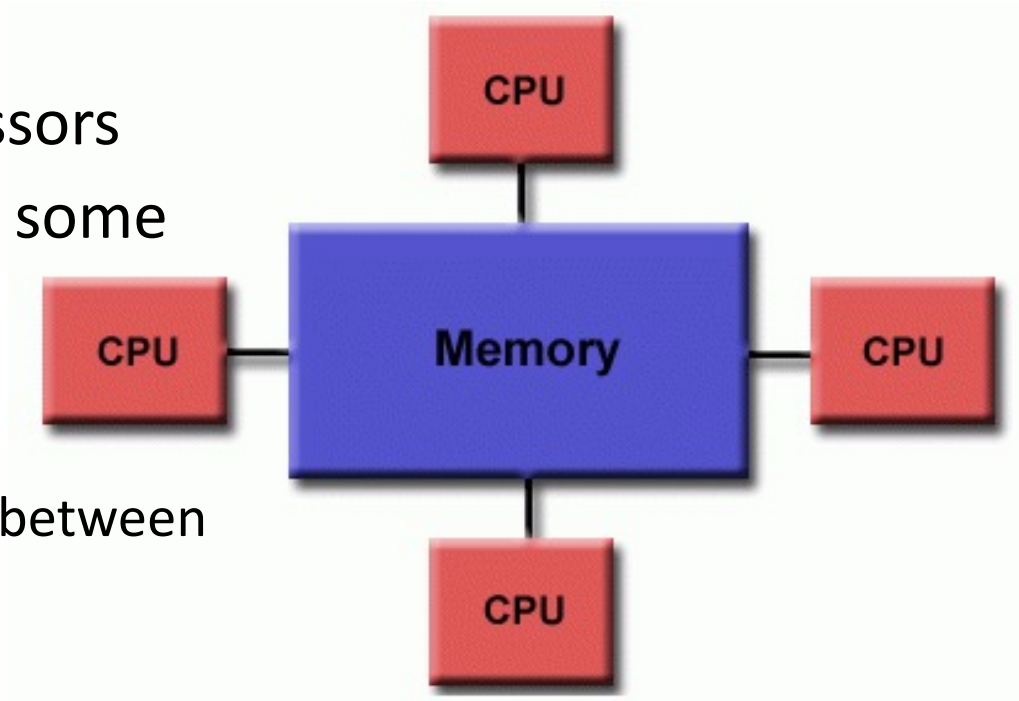




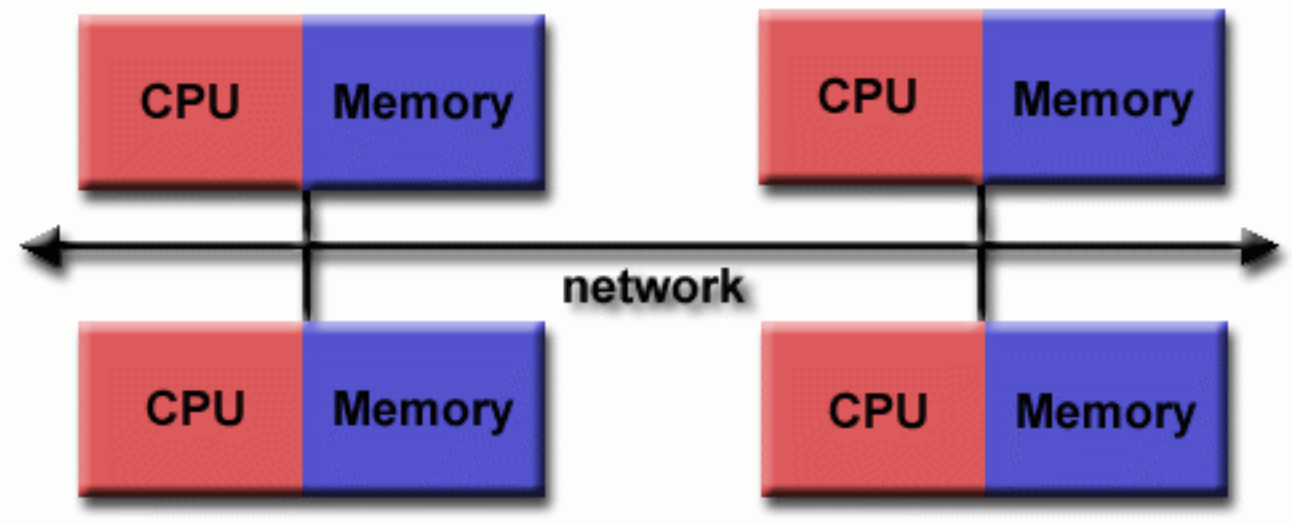
MIMD

Shared Memory – A node or a computer

- Global memory space, accessible by all processors
- Processors may have local copies (in cache) of some global memory, consistency of copies usually maintained by hardware (cache coherency)
- Advantages:
  - Global address space is user-friendly Data sharing between tasks is fast
- Disadvantages:
  - Shared memory - to - CPU path may be a bottleneck (is bandwidth of the network sufficient?)
  - Often: Non-Uniform Memory Access (NUMA)
    - ⇒ access time varies, depends on physical distance
  - Programmer responsible for correct synchronization
- Programming Models
  - OpenMP, Cilk

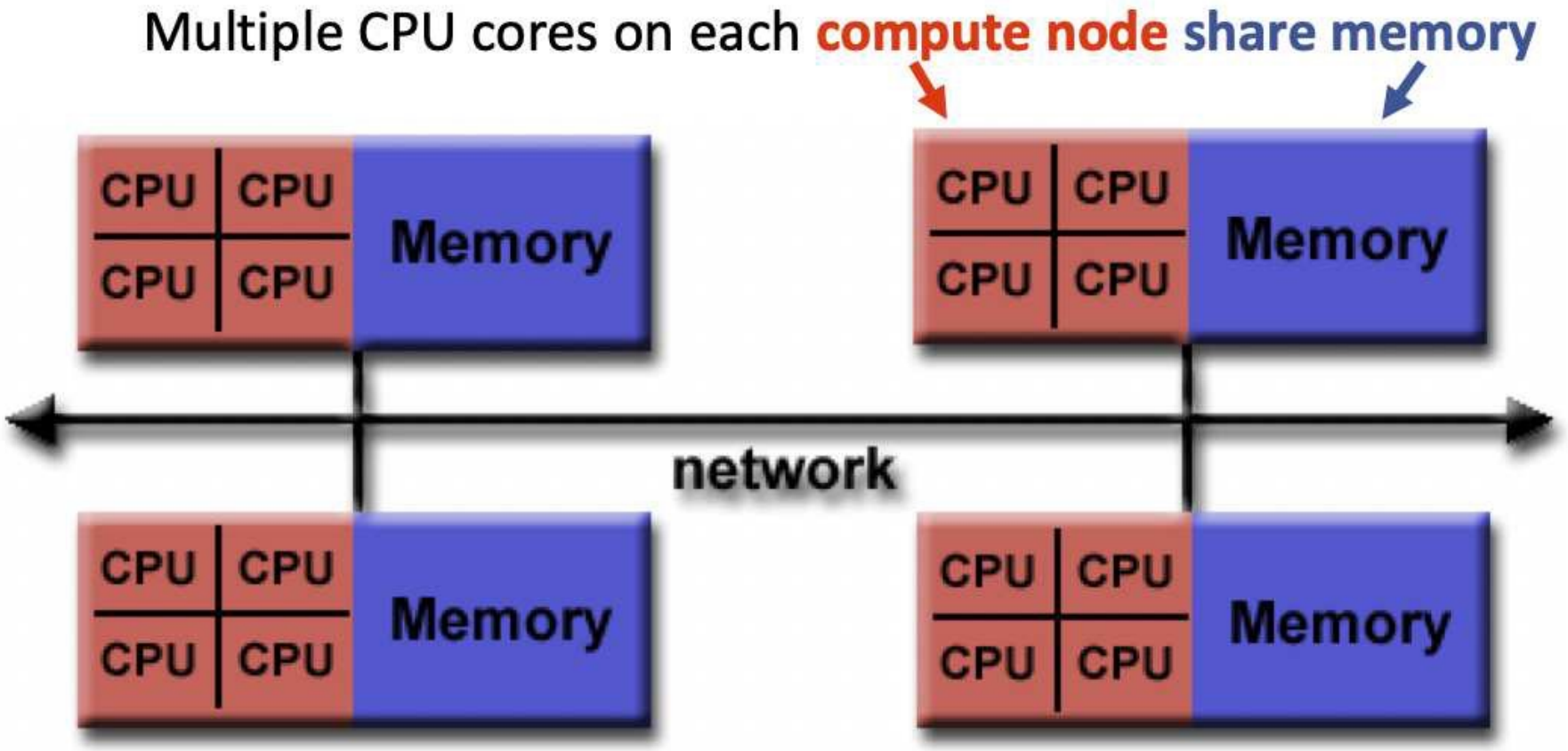


Distributed Memory



- If processor A needs data in processor B, then B must send a message to A containing the data. Thus DM systems also known as message passing systems.
- Programming Models - MPI
- Advantages:
  - Memory is scalable with number of processors
  - Each processor has rapid access to its own memory
  - Cost effective:can use commodity parts
- Disadvantages:
  - Programmer is responsible for many of the details of the communication, easy to make mistakes.
  - May be difficult to distribute the data structures

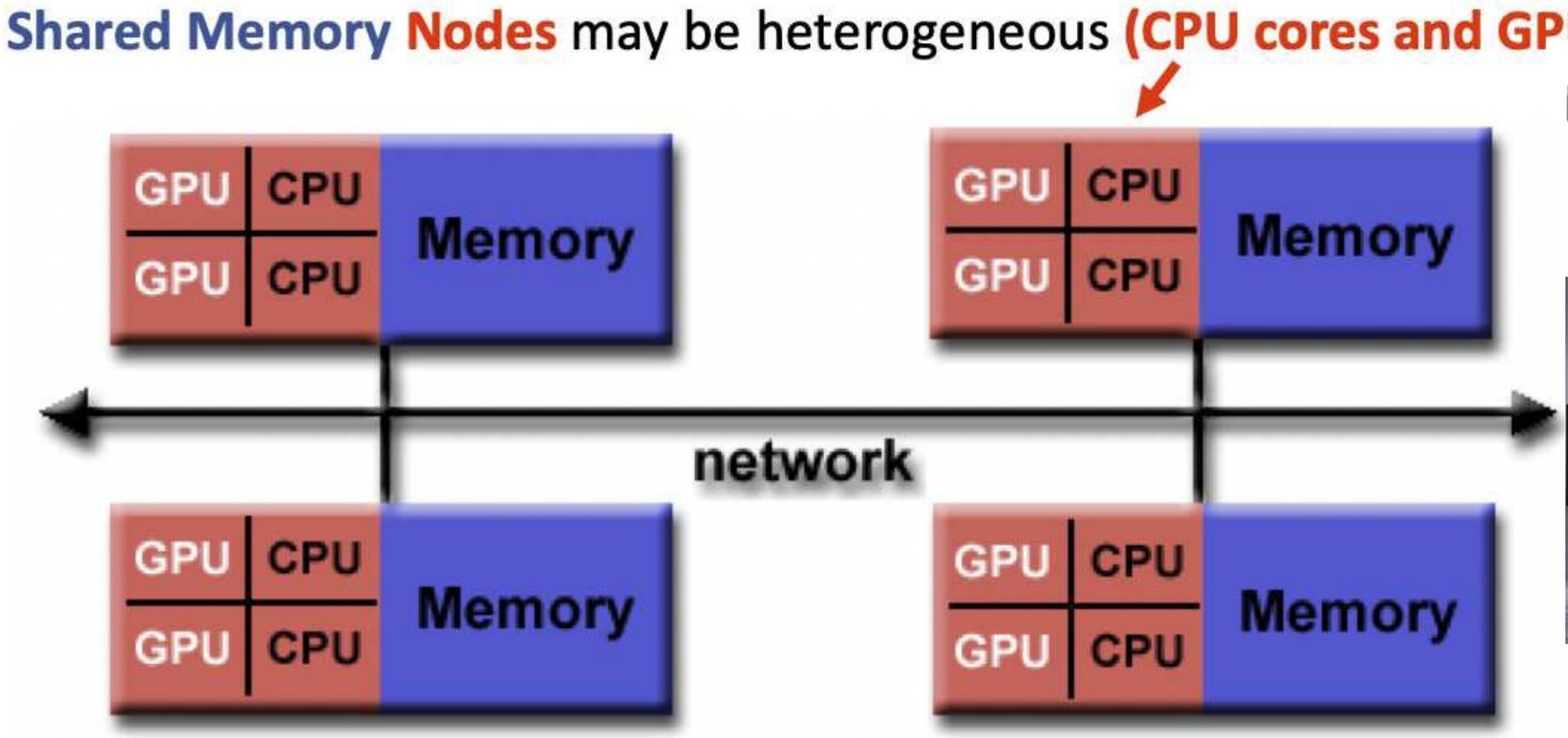
A General HPC Architecture

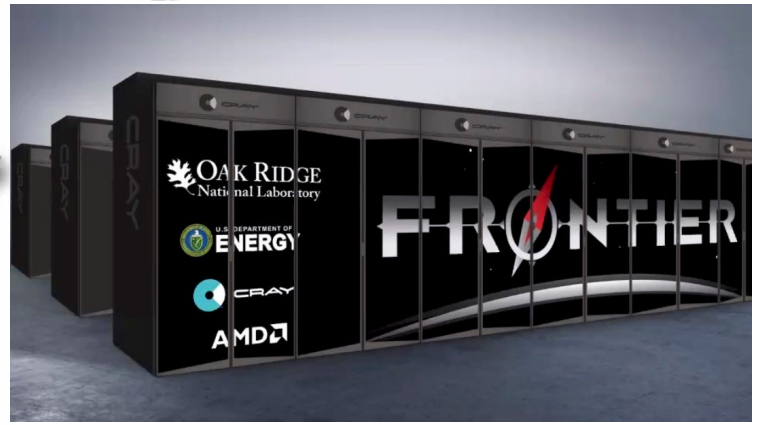


Shared Memory concept within a node

plus *Distributed Memory* concept: Non-local data can be sent across the network to other CPUs

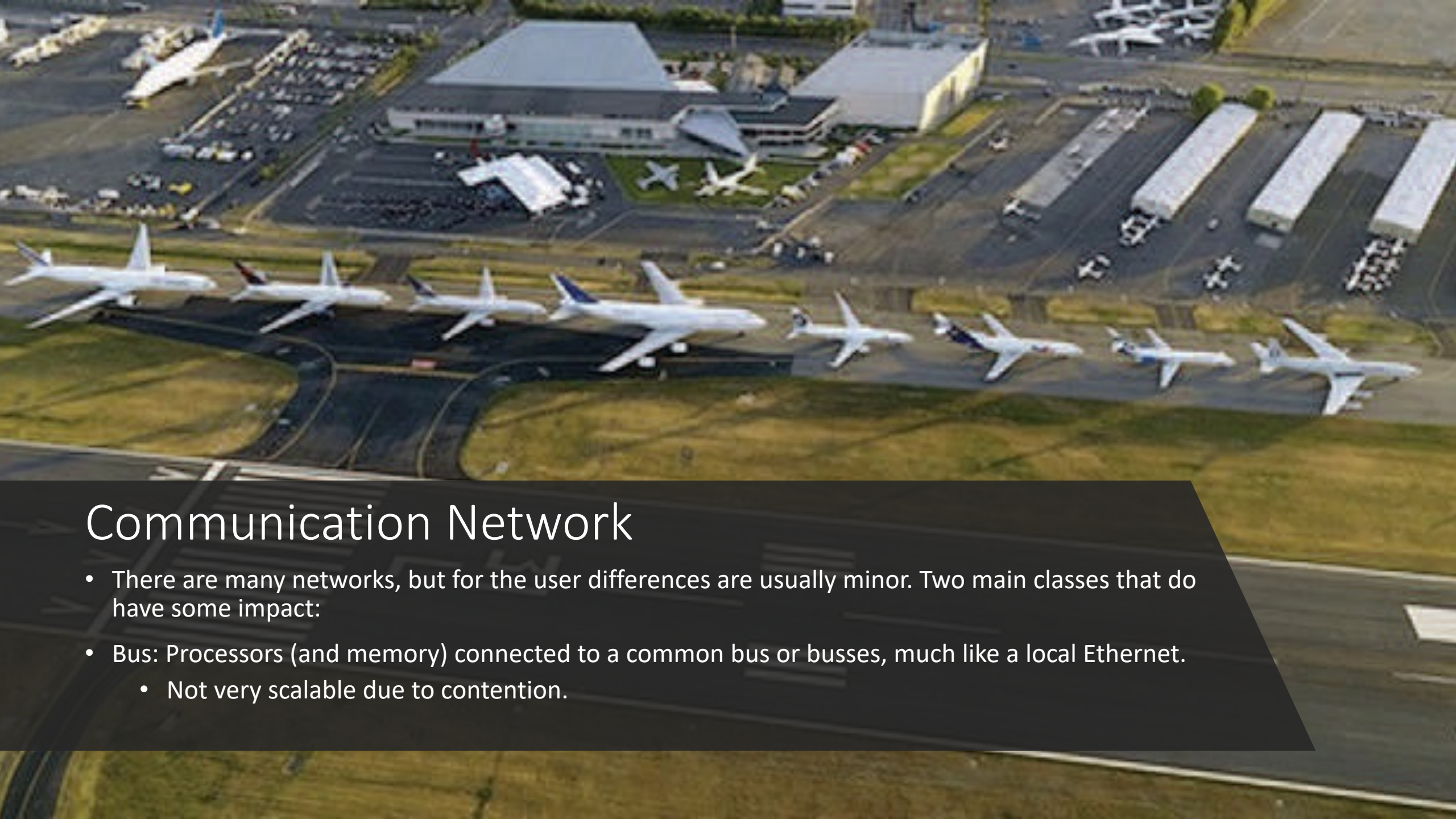
HPC Architectures with Accelerators





Shared Memory within a node with CPUs and GPUs

plus *Distributed Memory* concept: Non-local data can be sent across the network to other CPUs

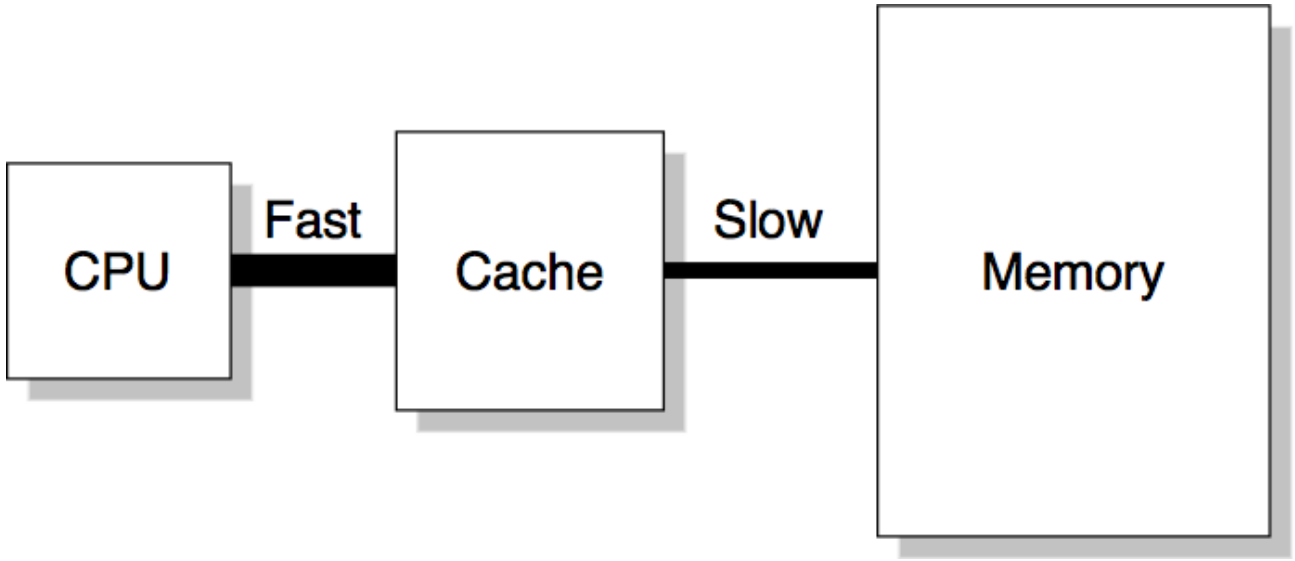


Communication Network

- There are many networks, but for the user differences are usually minor. Two main classes that do have some impact:
- Bus: Processors (and memory) connected to a common bus or busses, much like a local Ethernet.
  - Not very scalable due to contention.

Memory Hierarchy

- *von Neumann bottleneck*: processor much faster than memory, sits idle waiting for data. Unfortunately, faster memory higher $/byte, physics imposes size constraints.
- To ameliorate latency, data moved between levels in blocks (cache lines, pages). For efficiency:
  - use entire block while resident in the faster memory



Domain and Functional Decomposition

- Domain decomposition: Partition a (perhaps conceptual) space. Different processors do similar work on different pieces (quilting bee, teaching assistants for discussion sections, etc.)
- Functional decomposition: Different processors work on different types of tasks (workers on an assembly line, sub-contractors on a project, etc.)
  - Functional decomposition rarely scales to many processors, so we'll concentrate on domain decomposition.

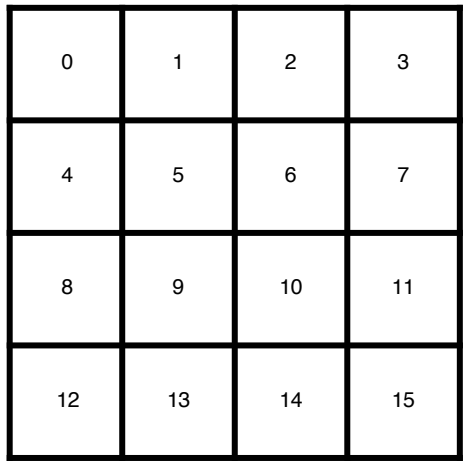
Static Decompositions

- Often just evenly dividing space among the processors yields acceptable load balance, with acceptable performance if communication minimized.
- This approach works even if the objects have varying computational requirements, as long as there are enough objects so that the worst processor is likely to be close to the average (law of large numbers).
- Will be discussed in terms of distributed memory, but basic ideas also apply to shared memory.

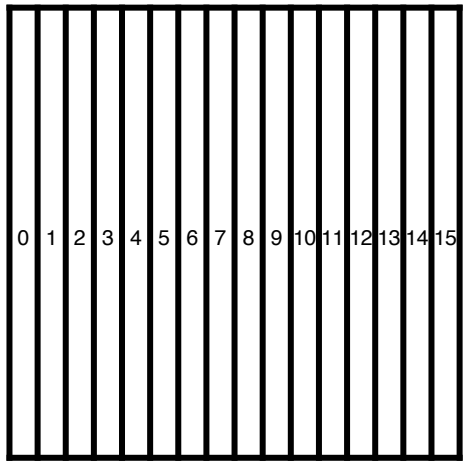
Matrix Decompositions

- Suppose work at each position only depends on value there and nearby ones, equivalent work at each position.
- Dependencies force communication along boundary

Minimizes Bandwidth

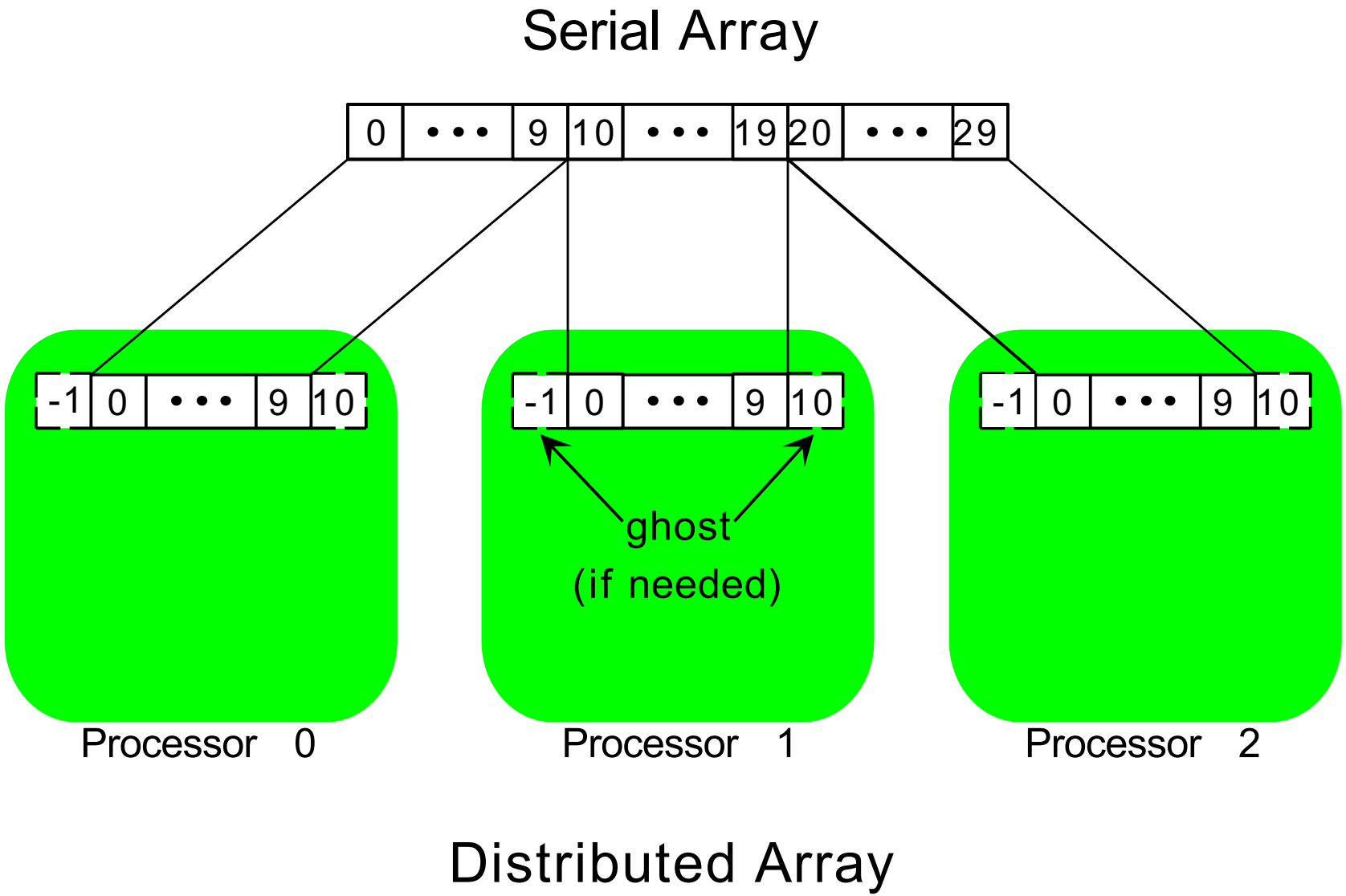


Minimizes Latency



Minimize boundary (2D) Minimize # neighbors (1D)

Local vs Global Arrays



MPI Ranks -- Linear vs 2D Grid

–

MPI ranks 0. . . 15

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

logical rows and columns 0 . . . 3

For MPI_COMM_RANK = i and MPI_COMM_SIZE = p

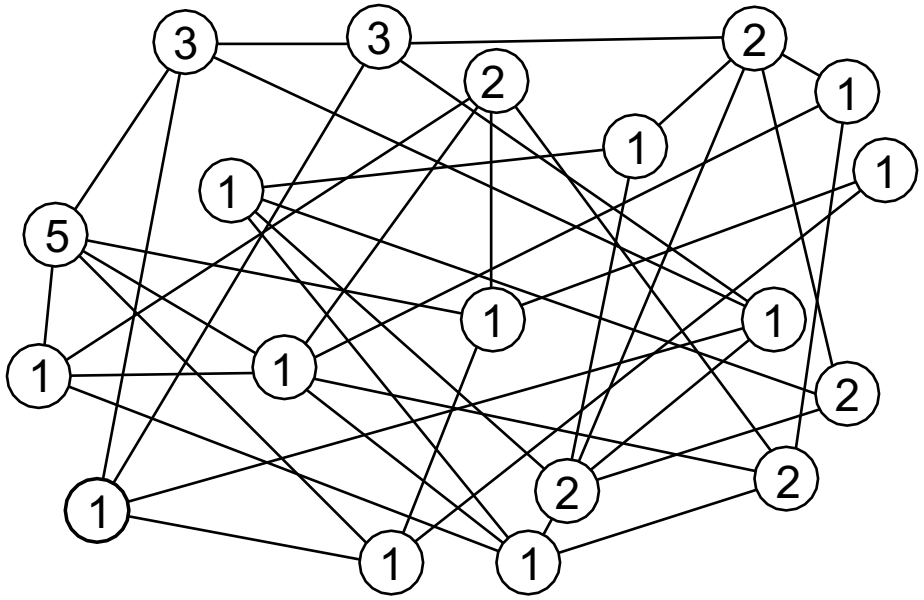
$\text{my\_row} = \lfloor \text{i}/\sqrt{\text{p}} \rfloor$ and $\text{my\_col} = \text{i} - \text{my\_row} * \sqrt{\text{p}}$

to send to logical		send to MPI_COMM_RANK
Right	(my_row, my_col +1)	$\text{i} + 1$
Left	(my_row, my_col -1)	$\text{i} - 1$
Up	(my_row -1, my_col)	$\text{i} - \sqrt{\text{p}}$
Down	(my_row +1, my_col)	$\text{i} + \sqrt{\text{p}}$

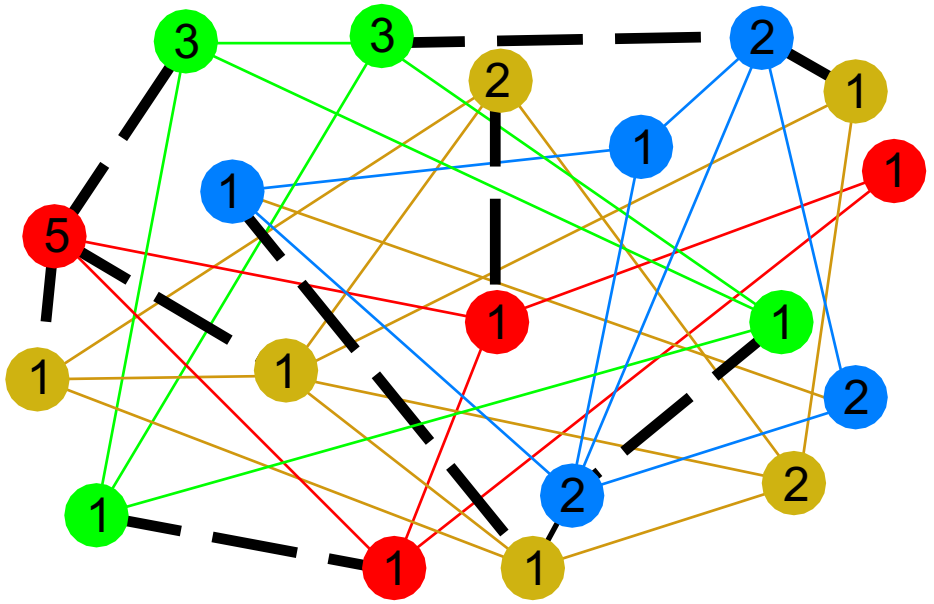
Graph Decompositions

- Graph decomposition techniques can be used when dependencies are less regular. Once again, dependencies determine communication.
  - Objects (calculations) represented as vertices (with weights if calculation requirements uneven)
  - Communication represented as edges (with weights if communication requirements uneven).
- Goals:
  - assign vertices to processors to evenly distribute the number/weight of vertices: balance computation
  - minimize and balance the number/weight of edges between processors: minimize communication

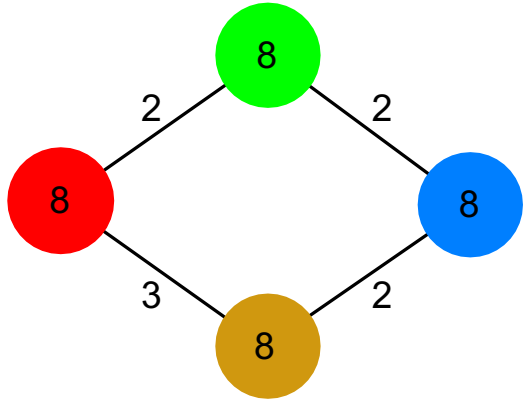
Illustration of Graph Decomposition



Numbers indicate work,
want to use 4 processors.



Processors:

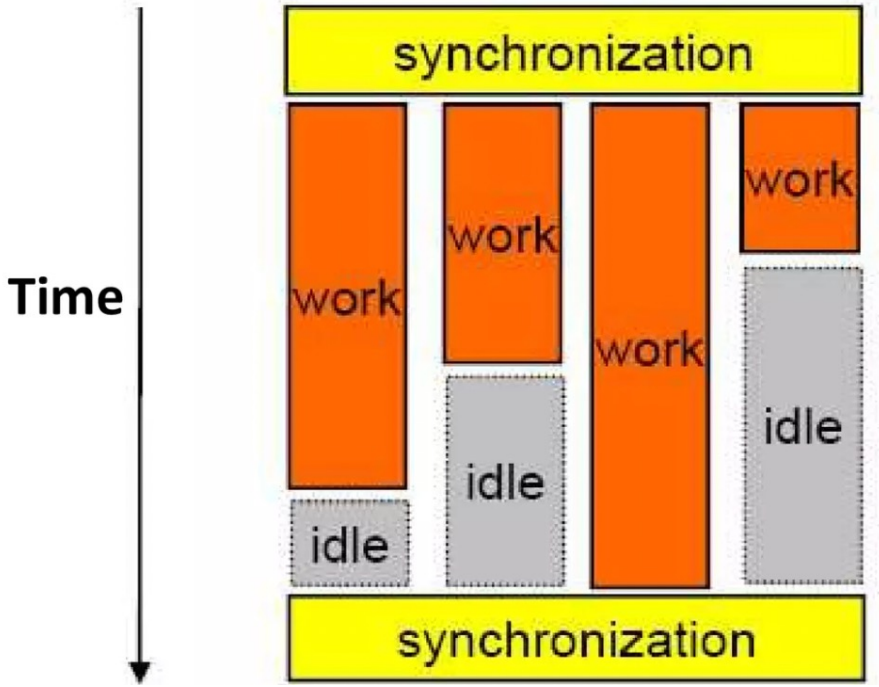


Geometric Decompositions

- When the objects have an underlying geometrical basis: finite elements in crash simulation, polygons representing census blocks in a geographical information system, stars in a galaxy, etc., the geometry can often be exploited
  - if dependencies predominately involve nearby objects.
- Geometric decompositions can be based on recursive bisectioning, quad- or oct-trees, space-filling curves, etc., and can incorporate weights.
- Warning: geometric approaches not nearly as useful on high-dimensional data.

Summary : Important Bottlenecks

Understand the serial portions of the code



Load Imbalance

Tame Communication

Engineering Challenges – Take more time to develop over serial code and debugging is hard

Administrivia

- TA office hours
  - Tuesday 3:45pm to 4:45pm
  - Zoom link
- Piazza
- PACE ICE (Instructional Cluster Environment)
- Github classroom

PACE ICE

- Running assignments
- Login nodes : coding, running small scripts
  - Please do not use the login nodes for any resource-intensive activities, as it prevents other students from using the cluster.
- Compute nodes : running assignments by submitting jobs to queue
  - Batch or interactive jobs

- For more information
  - https://pace.gatech.edu/pace-ice-instructional-cluster-environment-education

Github classroom

- Check Canvas/Piazza for the link to the Assignment 0
- Modify the contents of README.md to the following 5 lines:
  - First name
  - Last name
  - Preferred name
  - Github username
  - GT username
- Commit and push to your repo.
- Graded for 1 point. Due Jan 17, 5pm EST.