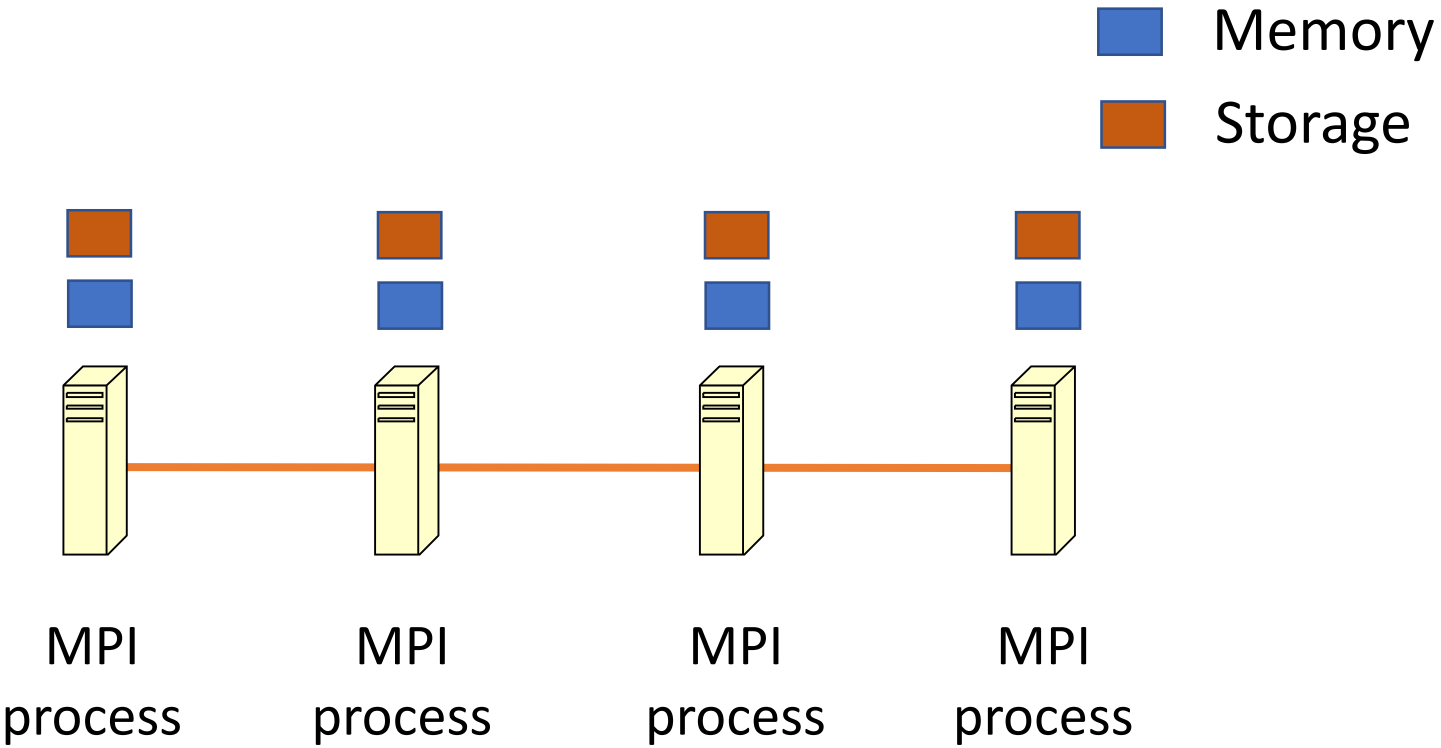
MPI Processes

Jan 8, 2019

Hardware Model



🟦 Memory

🟧 Storage

MPI process

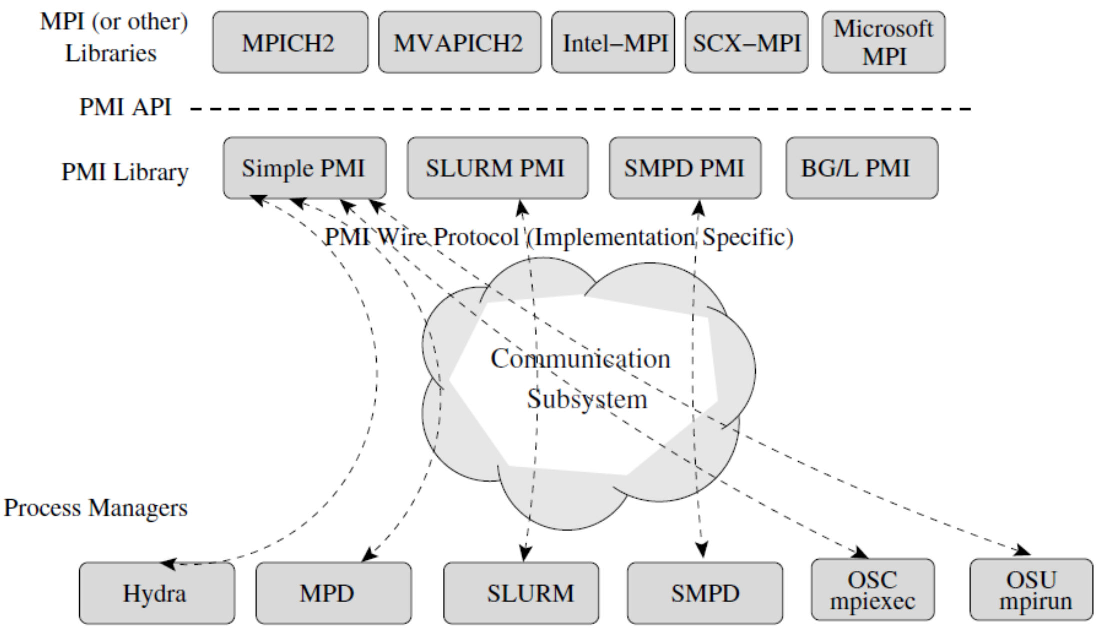
MPI process

MPI process

MPI process

NO centralized server/master

Process Management Setup



Parallel program library (e.g. MPI)

Process management interface (PMI)

Resource manager/ Job scheduler

Reference

[PMI: A Scalable Parallel Process-Management Interface for Extreme-Scale Systems]

Internals

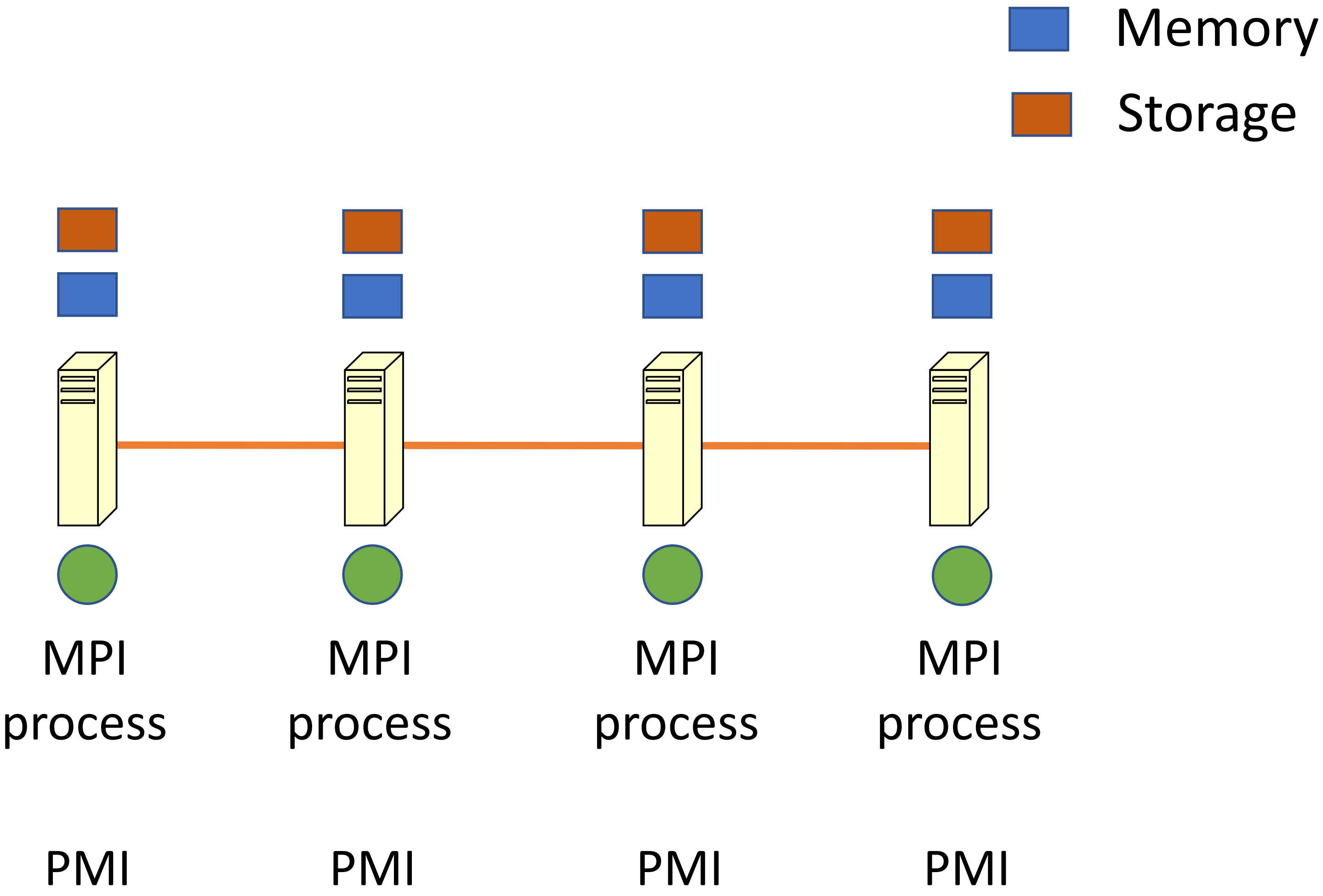
Process Manager

- Start and stop processes in a scalable way
- Setup communication channels for parallel processes
- Provide system-specific information to processes

Process Management Interface

- Processes can exchange information about peers by querying PMI
- Provides a logically centralized service for all processes in an MPI job
- Uses key-value store for process-related data

Process Launch



🟦 Memory

🟧 Storage

MPI process

MPI process

MPI process

MPI process

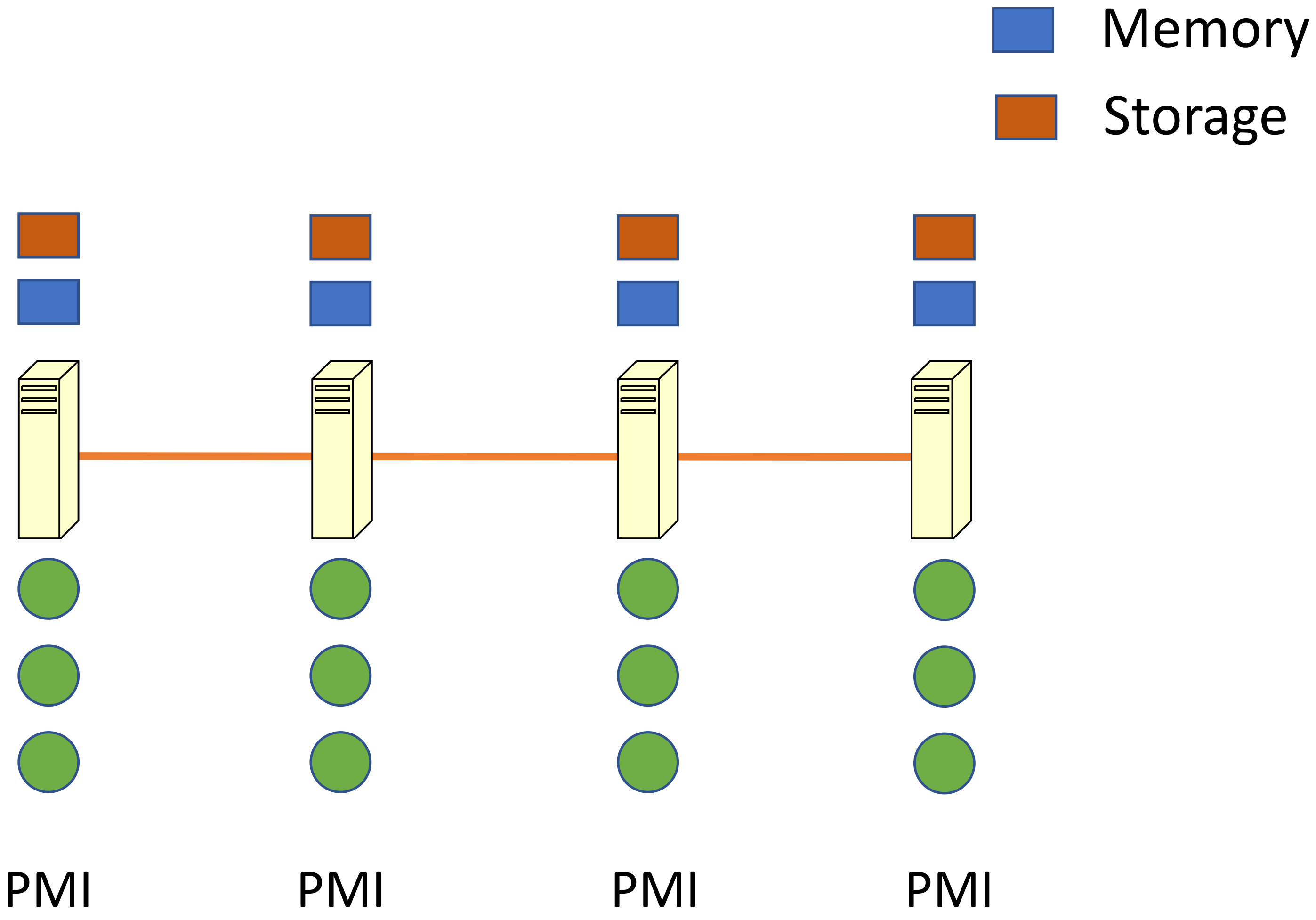
PMI

PMI

PMI

PMI

Process Launch



🟦 Memory

🟧 Storage

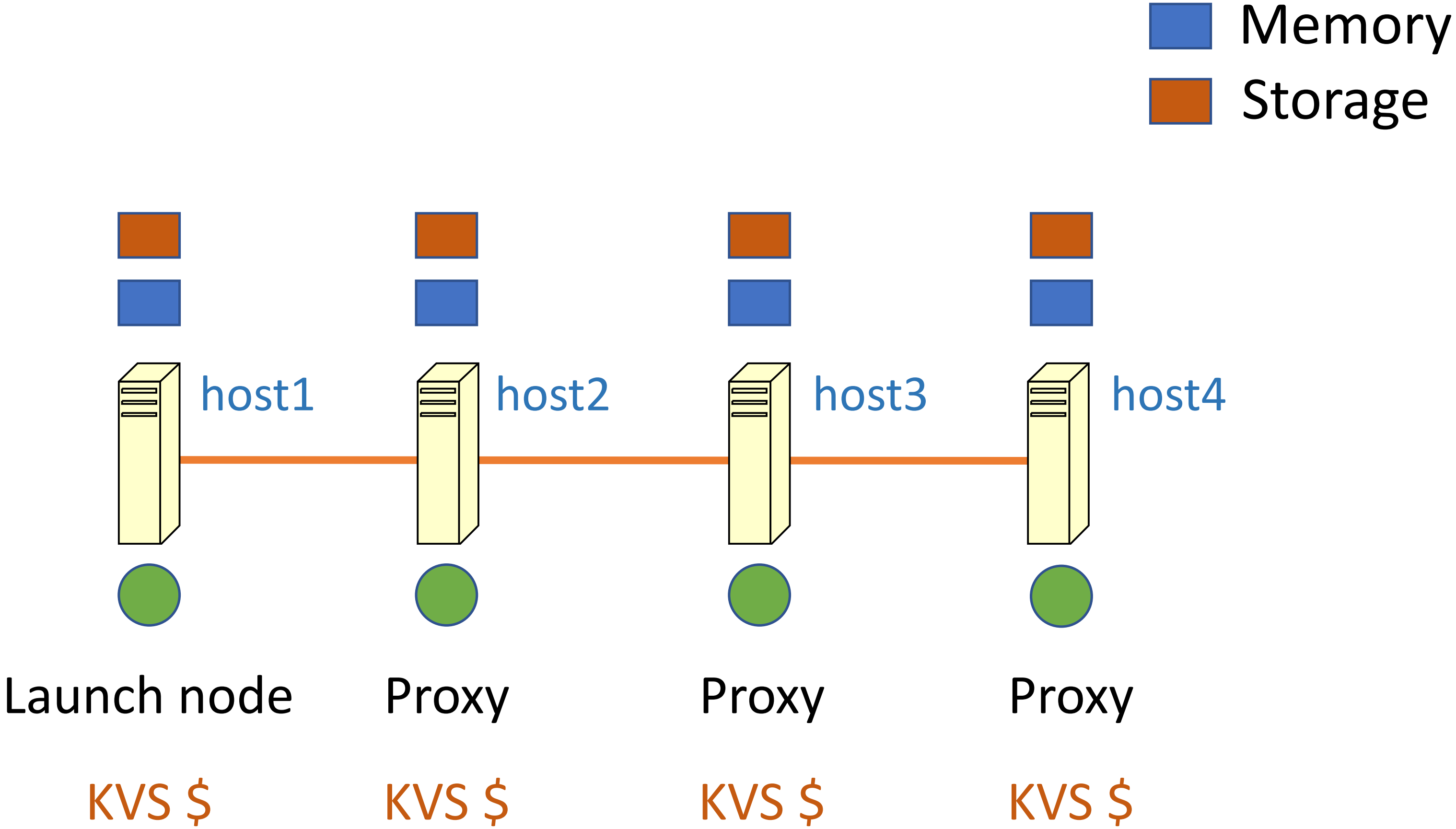
PMI PMI PMI PMI

Hydra Process Manager

- A process management system for starting parallel jobs
- Uses existing daemons (viz. ssh) to start MPI processes
- Automatically detects resource managers and interacts with them
- $ mpiexec ./app
  - Hydra gets information about allocated resources and launches processes
- Passes environment variables from the shell on which mpiexec is launched to the launched processes

There are others – mpd, gforker, slurm, etc.

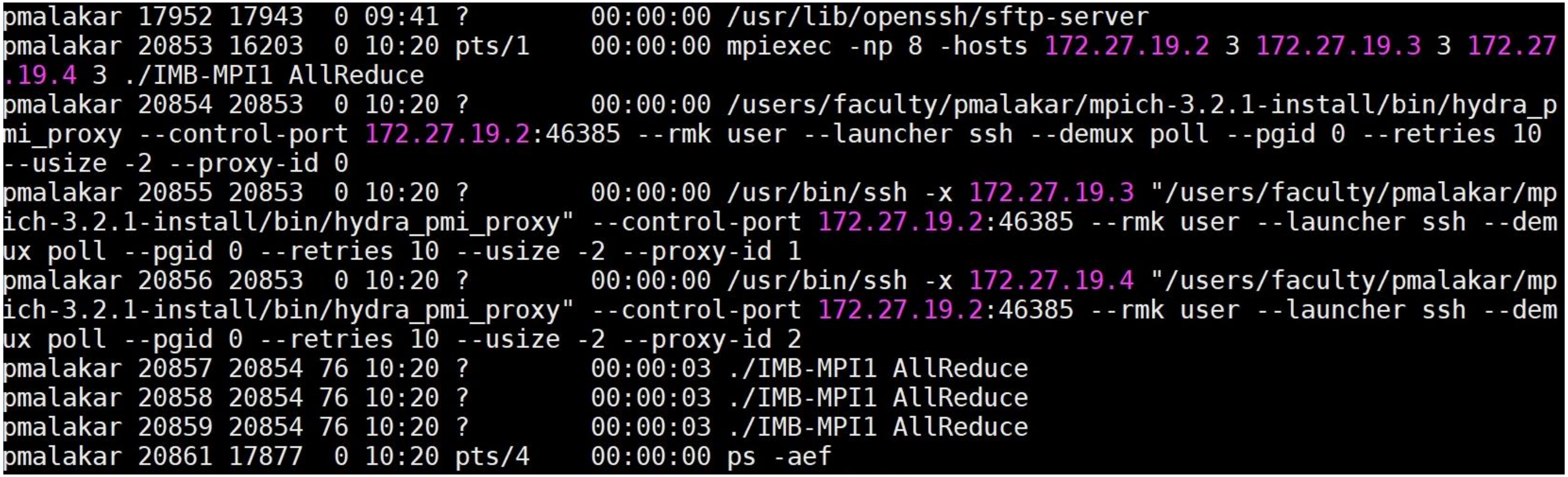
mpiexec



mpiexec –n 4 –hosts host1,host2,host3,host4 ./exe

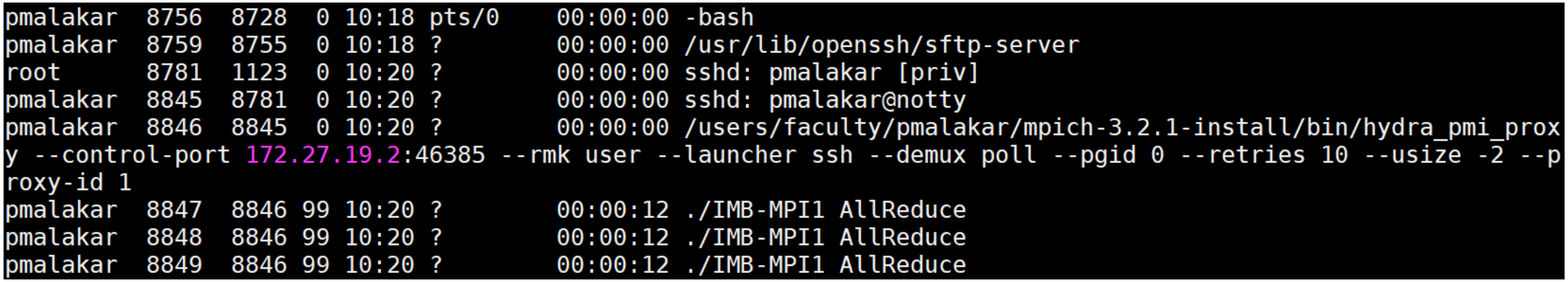
Launch Node

mpiexec -np 8 -hosts host1:3,host2:3,host3:3 ./exe

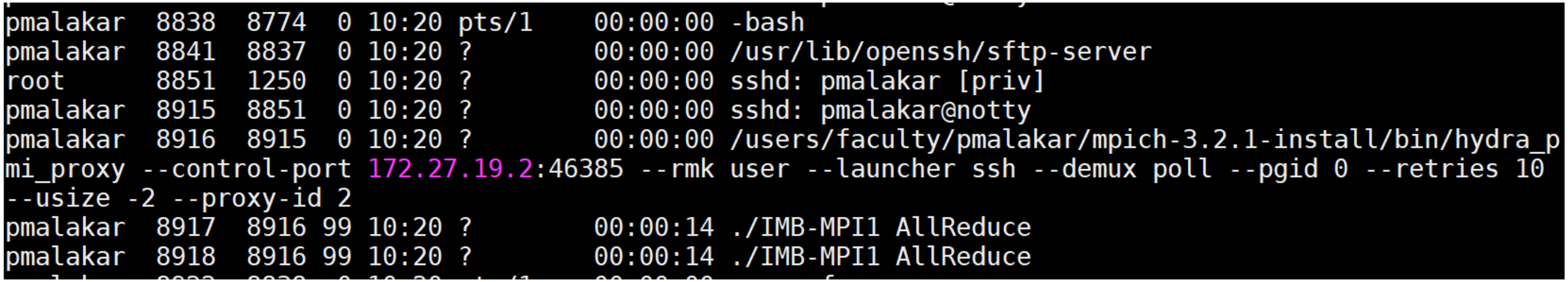


```
pmalakar 17952 17943  0 09:41 ?        00:00:00 /usr/lib/openssh/sftp-server
pmalakar 20853 16203  0 10:20 pts/1    00:00:00 mpiexec -np 8 -hosts 172.27.19.2 3 172.27.19.3 3 172.27
.19.4 3 ./IMB-MPI1 AllReduce
pmalakar 20854 20853  0 10:20 ?        00:00:00 /users/faculty/pmalakar/mpich-3.2.1-install/bin/hydra_p
mi_proxy --control-port 172.27.19.2:46385 --rmk user --launcher ssh --demux poll --pgid 0 --retries 10
--usize -2 --proxy-id 0
pmalakar 20855 20853  0 10:20 ?        00:00:00 /usr/bin/ssh -x 172.27.19.3 "/users/faculty/pmalakar/mp
ich-3.2.1-install/bin/hydra_pmi_proxy" --control-port 172.27.19.2:46385 --rmk user --launcher ssh --dem
ux poll --pgid 0 --retries 10 --usize -2 --proxy-id 1
pmalakar 20856 20853  0 10:20 ?        00:00:00 /usr/bin/ssh -x 172.27.19.4 "/users/faculty/pmalakar/mp
ich-3.2.1-install/bin/hydra_pmi_proxy" --control-port 172.27.19.2:46385 --rmk user --launcher ssh --dem
ux poll --pgid 0 --retries 10 --usize -2 --proxy-id 2
pmalakar 20857 20854 76 10:20 ?        00:00:03 ./IMB-MPI1 AllReduce
pmalakar 20858 20854 76 10:20 ?        00:00:03 ./IMB-MPI1 AllReduce
pmalakar 20859 20854 76 10:20 ?        00:00:03 ./IMB-MPI1 AllReduce
pmalakar 20861 17877  0 10:20 pts/4    00:00:00 ps -aef
```

Compute Node Processes

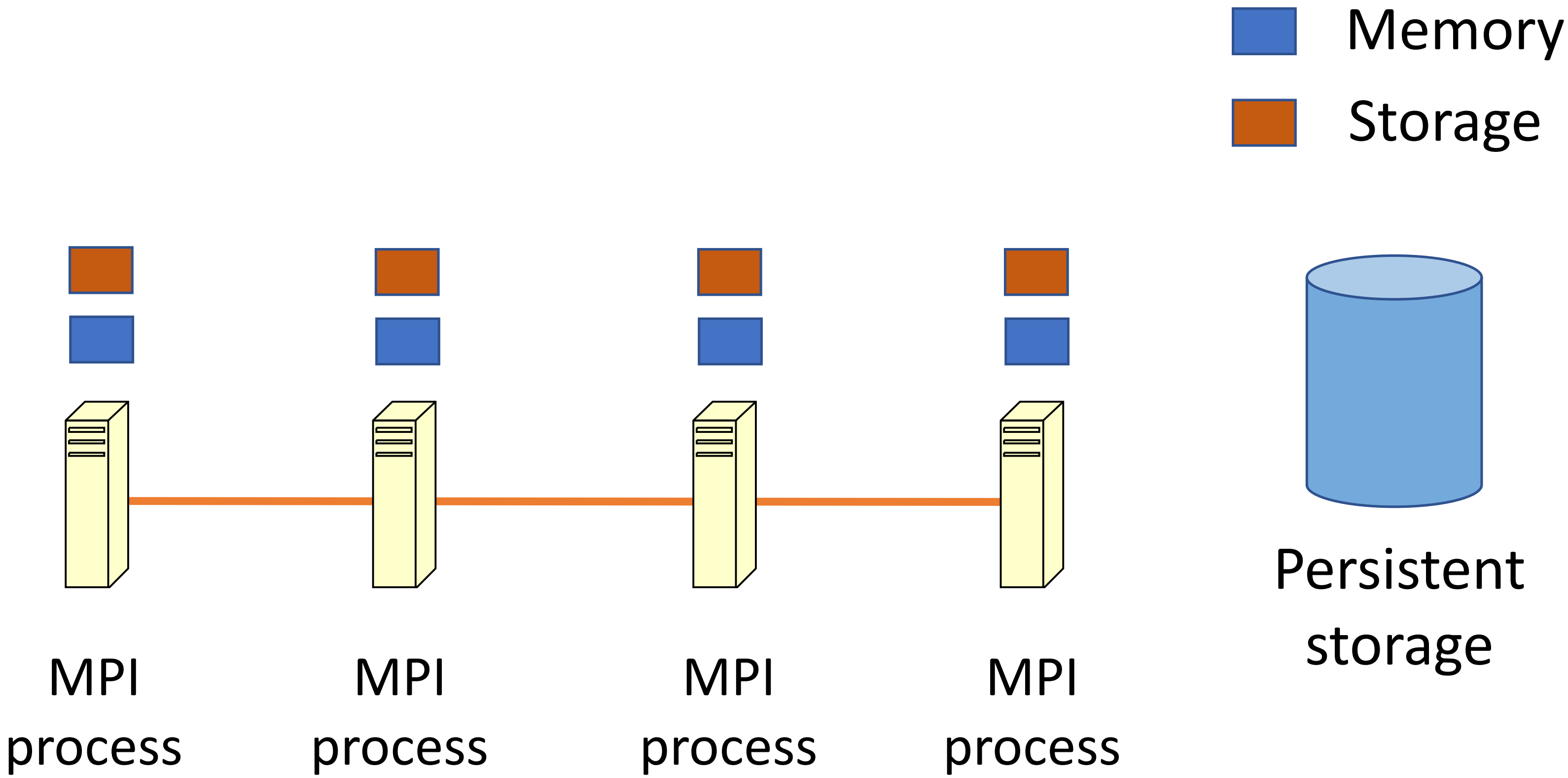


```
pmalakar  8756  8728  0 10:18 pts/0    00:00:00 -bash
pmalakar  8759  8755  0 10:18 ?        00:00:00 /usr/lib/openssh/sftp-server
root      8781  1123  0 10:20 ?        00:00:00 sshd: pmalakar [priv]
pmalakar  8845  8781  0 10:20 ?        00:00:00 sshd: pmalakar@notty
pmalakar  8846  8845  0 10:20 ?        00:00:00 /users/faculty/pmalakar/mpich-3.2.1-install/bin/hydra_pmi_prox
y --control-port 172.27.19.2:46385 --rmk user --launcher ssh --demux poll --pgid 0 --retries 10 --usize -2 --p
roxy-id 1
pmalakar  8847  8846 99 10:20 ?        00:00:12 ./IMB-MPI1 AllReduce
pmalakar  8848  8846 99 10:20 ?        00:00:12 ./IMB-MPI1 AllReduce
pmalakar  8849  8846 99 10:20 ?        00:00:12 ./IMB-MPI1 AllReduce
```



```
pmalakar  8838  8774  0 10:20 pts/1    00:00:00 -bash
pmalakar  8841  8837  0 10:20 ?        00:00:00 /usr/lib/openssh/sftp-server
root      8851  1250  0 10:20 ?        00:00:00 sshd: pmalakar [priv]
pmalakar  8915  8851  0 10:20 ?        00:00:00 sshd: pmalakar@notty
pmalakar  8916  8915  0 10:20 ?        00:00:00 /users/faculty/pmalakar/mpich-3.2.1-install/bin/hydra_p
mi_proxy --control-port 172.27.19.2:46385 --rmk user --launcher ssh --demux poll --pgid 0 --retries 10
--usize -2 --proxy-id 2
pmalakar  8917  8916 99 10:20 ?        00:00:14 ./IMB-MPI1 AllReduce
pmalakar  8918  8916 99 10:20 ?        00:00:14 ./IMB-MPI1 AllReduce
```

Hardware Model



🟦 Memory

🟧 Storage

MPI process

MPI process

MPI process

MPI process

Persistent storage

What are different ways in which you can share data among distributed processes?

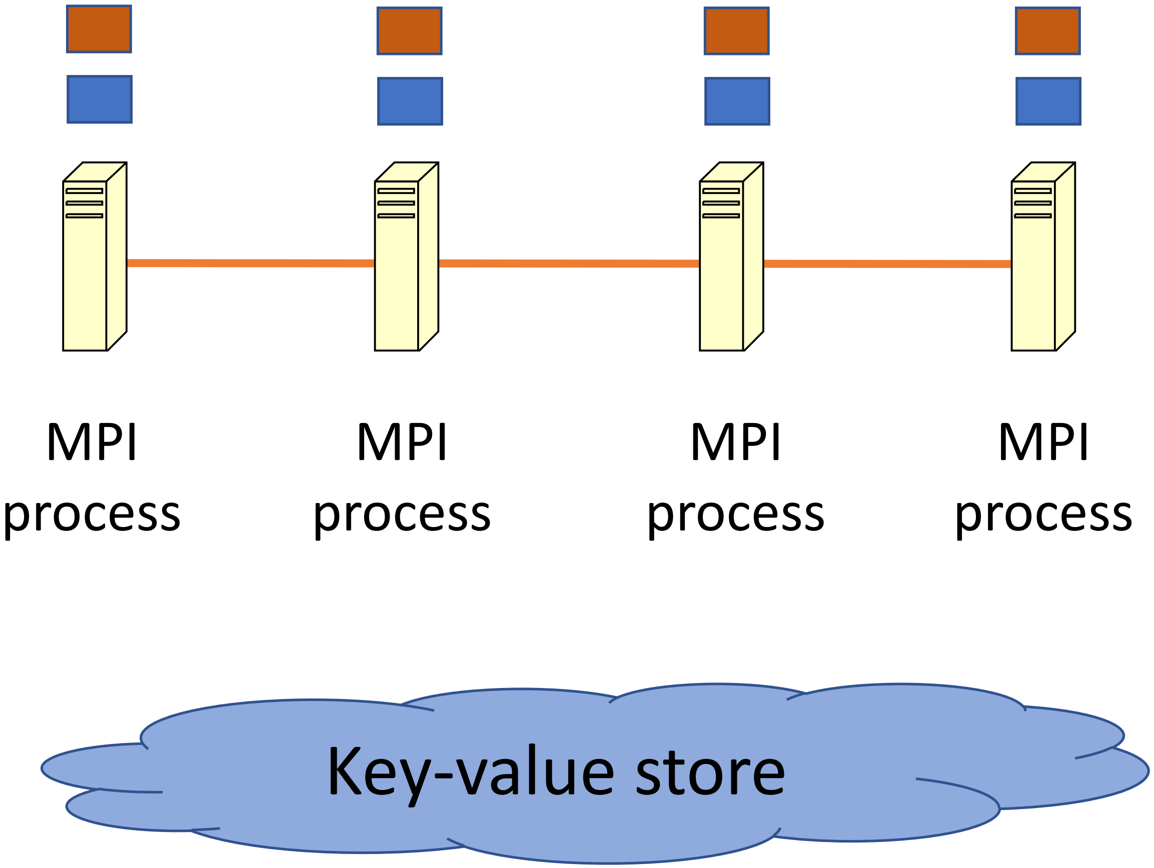
Message Passing Paradigm

- Message sends and receives
- Explicit communication

Communication types

- Blocking
- Non-blocking

Hardware Model



MPI process

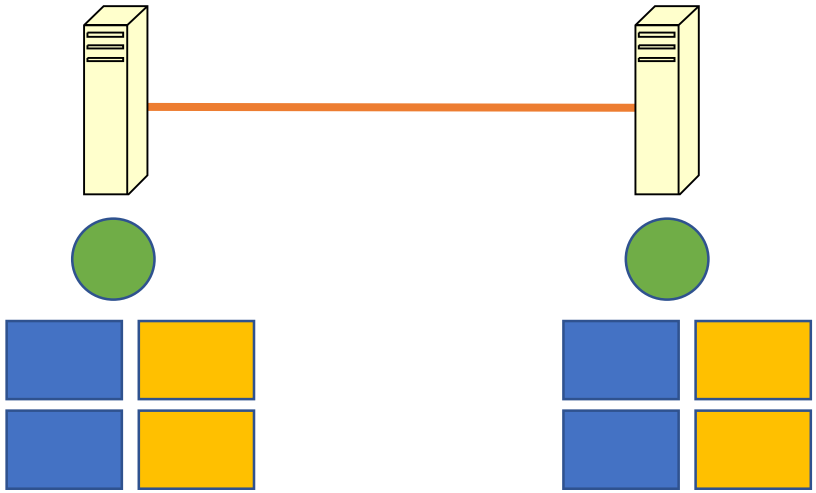
MPI process

MPI process

MPI process

Key-value store

Communication Channels

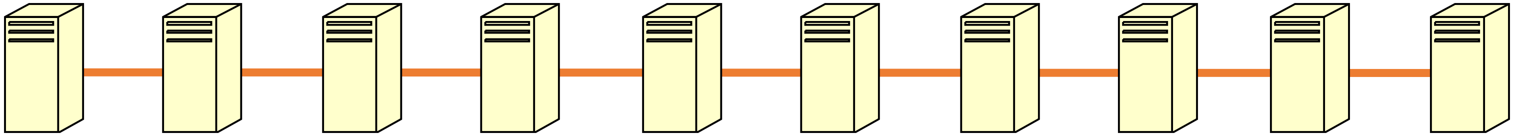


- Sockets for network I/O (wire protocol in PMI)
- PMI is responsible for creating/initializing/cleanup
- MPI handles communications, progress etc.

Reading: Design and Evaluation of Nemesis, a Scalable, Low-Latency, Message-Passing Communication Subsystem by Buntinas et al.

Interconnects

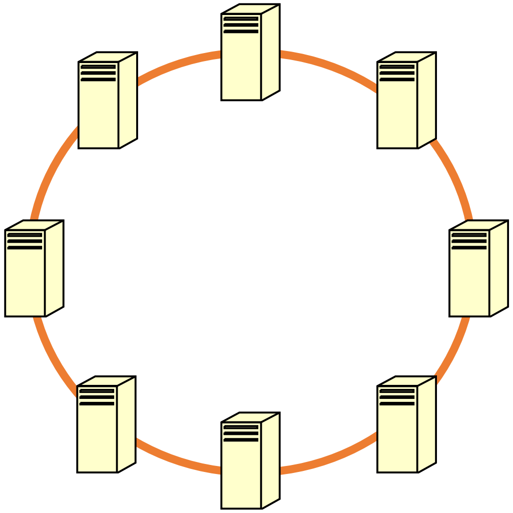
Linear Array



Attributes / Parameters

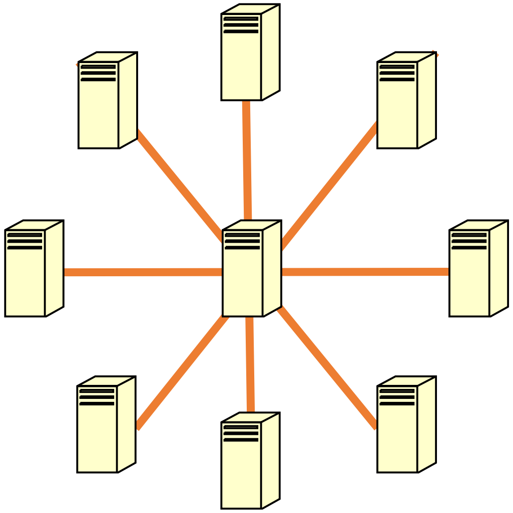
- Topology
- Diameter p-1
- Bisection width 1
- Cost p-1

Ring Interconnect



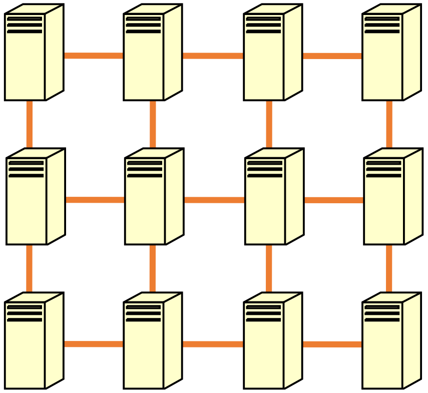
- Diameter p/2
- Bisection width 2
- Cost p

Star Interconnect



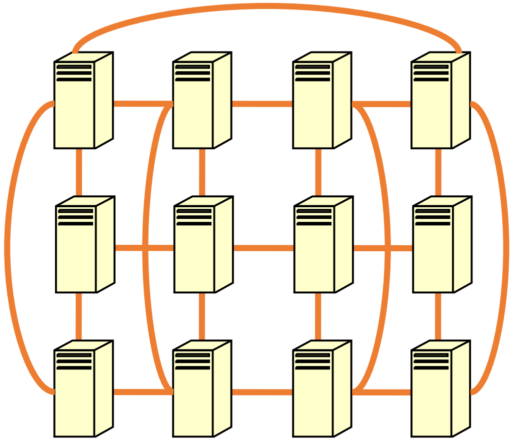
- Diameter 2
- Bisection width 1
- Cost p-1

Mesh Interconnect



- Diameter $2(\sqrt{p} - 1)$
- Bisection width $\sqrt{p}$
- Cost $2(p - \sqrt{p})$

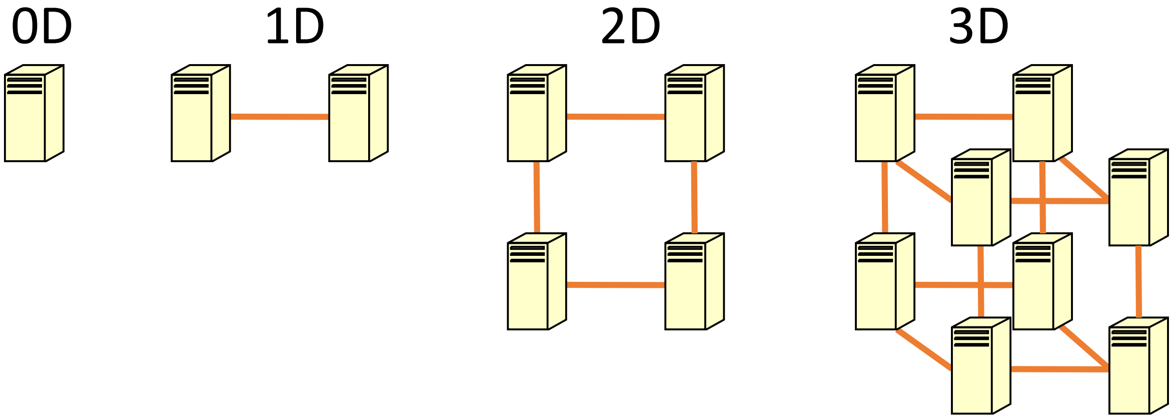
Torus Interconnect



What is the advantage of torus over mesh?

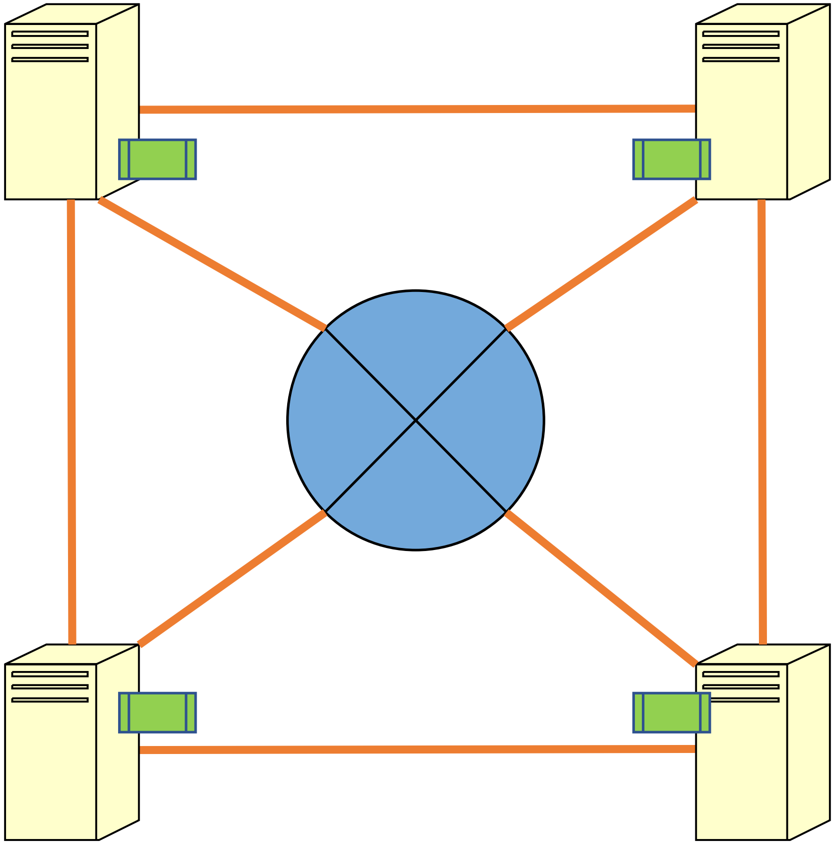
- Diameter 2 (√p/2)
- Bisection width √p
- Cost 2p

Hypercube Interconnect

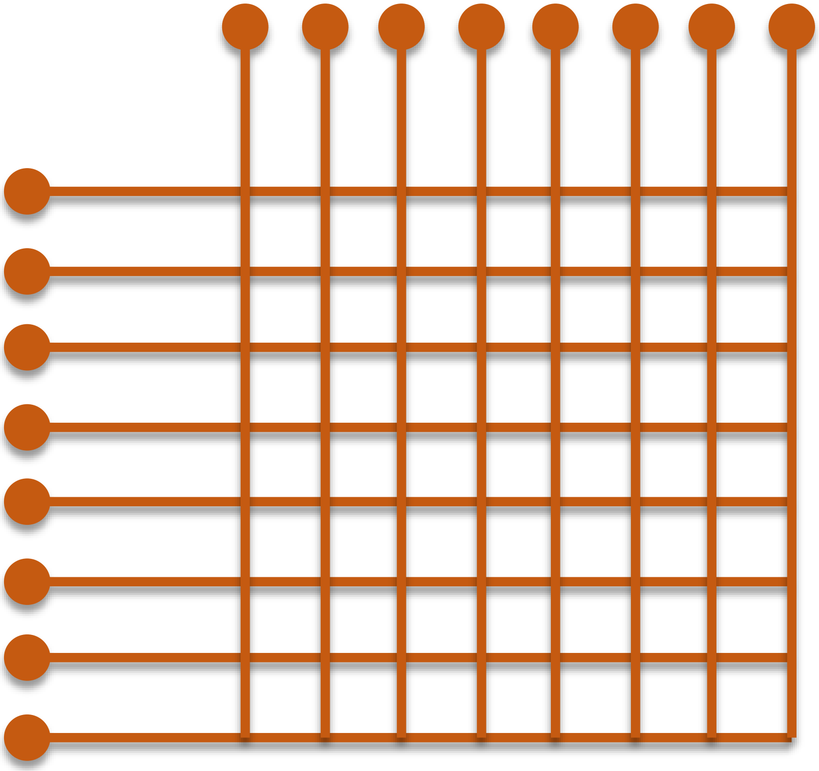


- Diameter log p
- Bisection width p/2
- Cost (p log p)/2

Switched Network



Crossbar Switch



Connectivity ?

Latency ?

Communication Cost

- Startup time (t_s)
- Latency (t_h)
- Bandwidth (t_w)

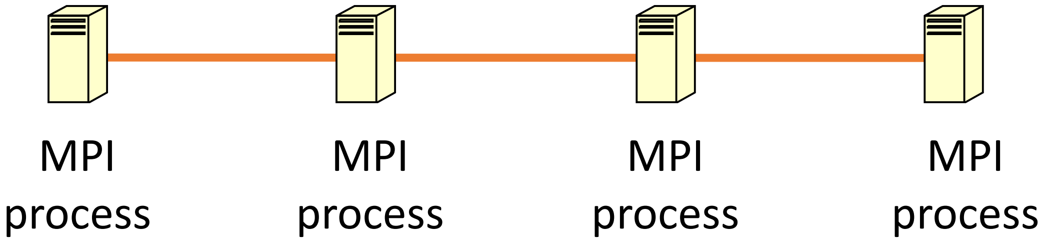
Transfer time = $t_s + t_h + n/t_w$

Communication time = Transfer time + Overhead

MPI Programming

Parallel Program Execution

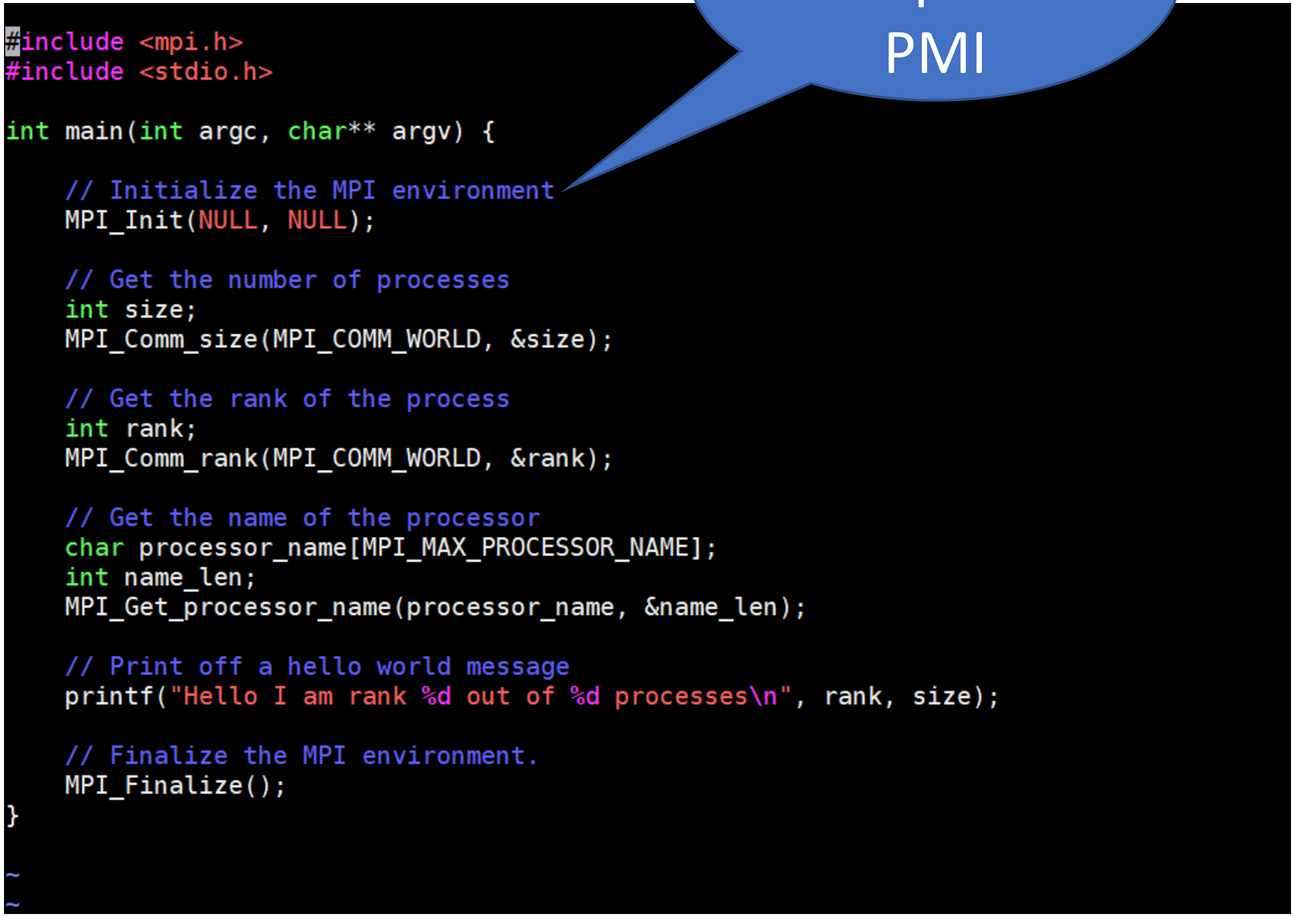
- Launch MPI processes on cluster nodes
- Communication setup



MPI

- Standard for message passing
- Explicit communications
- High programming complexity
- Requires communication scope

Getting Started



```c
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {

    // Initialize the MPI environment
    MPI_Init(NULL, NULL);

    // Get the number of processes
    int size;
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    // Get the rank of the process
    int rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    // Get the name of the processor
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);

    // Print off a hello world message
    printf("Hello I am rank %d out of %d processes\n", rank, size);

    // Finalize the MPI environment.
    MPI_Finalize();
}

~
~
```

Initializes and queries PMI

MPI_Init

- gather information about the parallel job
- set up internal library state
- prepare for communication

Communication Scope

Process

- belongs to a group
- identified by a rank within a group

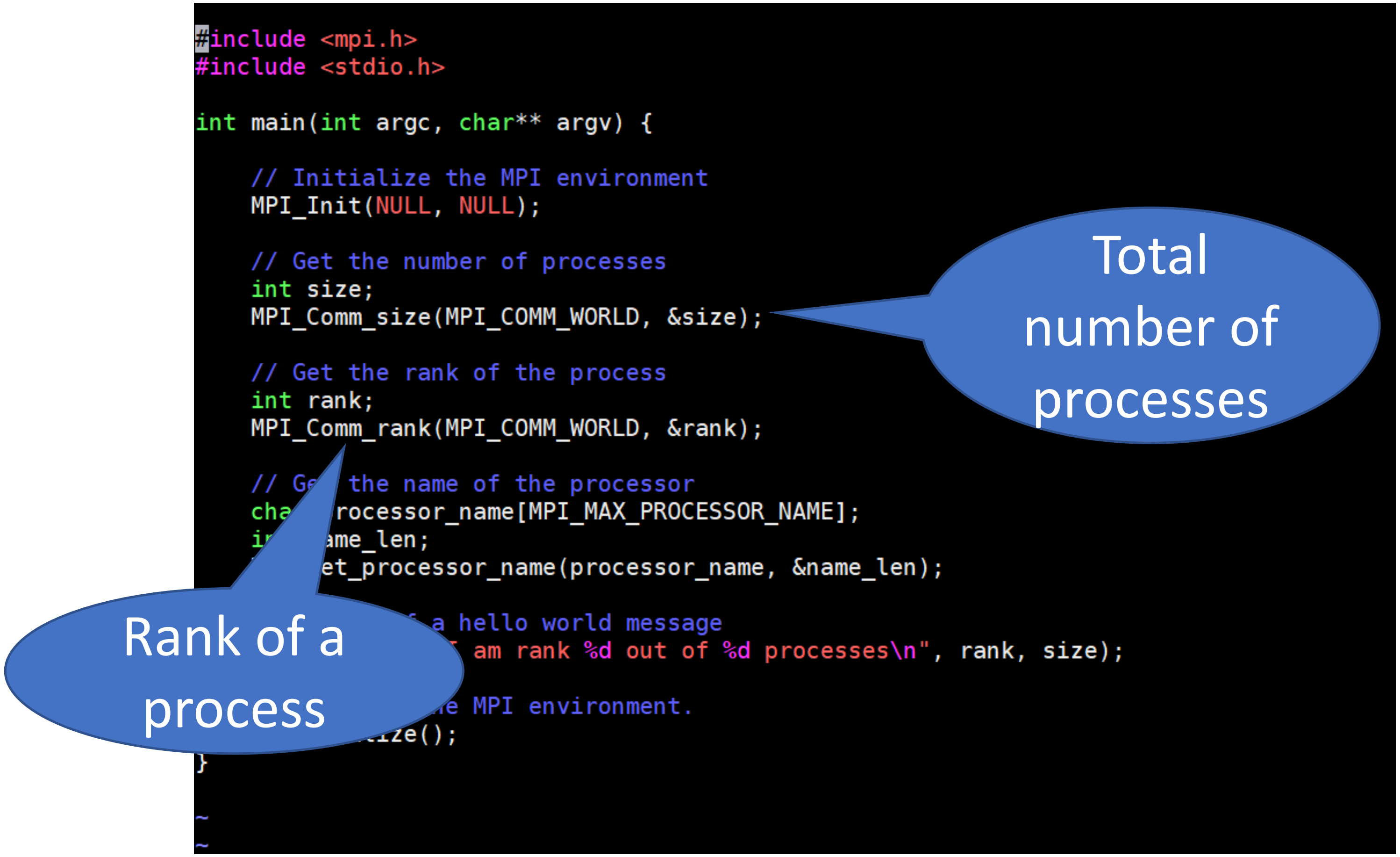
Message

- context
- tag

A communication handle Communicator defines the scope

Adapted from Sathish Vadhiyar's slides

Getting Started



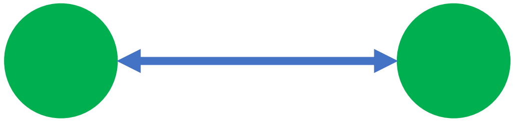
```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {

    // Initialize the MPI environment
    MPI_Init(NULL, NULL);

    // Get the number of processes
    int size;
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    // Get the rank of the process
    int rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    // Ge  the name of the processor
    cha  rocessor_name[MPI_MAX_PROCESSOR_NAME];
    in  ame_len;
          et_processor_name(processor_name, &name_len);

                f a hello world message
                 I am rank %d out of %d processes\n", rank, size);

                  e MPI environment.
              ize();
}

~
~
```

Total number of processes

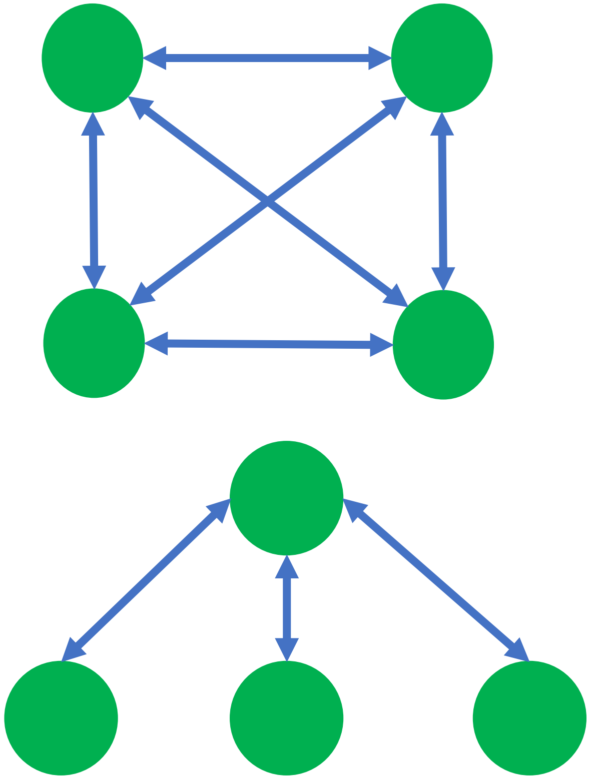
Rank of a process

MPI Communication Types

Point-to-point

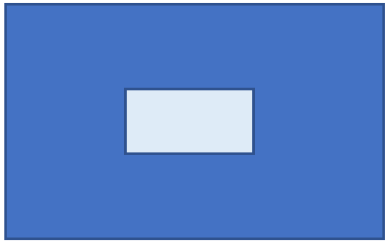


Collective



Basic MPI Communication

- MPI_Send

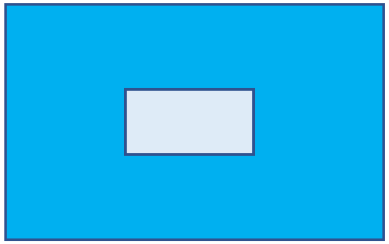


Blocking send and receive

int MPI_Send (const void *buf, int count,
MPI_Datatype datatype, int dest, int tag,
MPI_Comm comm)

Tags should match

- MPI_Recv



int MPI_Recv (void *buf, int count,
MPI_Datatype datatype, int source, int tag,
MPI_Comm comm, MPI_Status *status)

Teaching Assistants

- Dixit Kumar (dixit)
- Kawal Preet (kawal)
- Nirjhar Roy (nirjhar)
- Soumya Banerjee (soumyab)

- Add them to your Git repo