Job Startup at Exascale:
Challenges and Solutions

Sourav Chakraborty

Advisor: Dhabaleswar K (DK) Panda

The Ohio State University

Current Trends in HPC

- Tremendous increase in system and job sizes
- Interconnects like InfiniBand and OmniPath is dominant
- Dense many-core systems like KNL are more common
- Hybrid MPI+PGAS models becoming popular

Fast and scalable job-startup is essential!

Why is Job Startup Important?

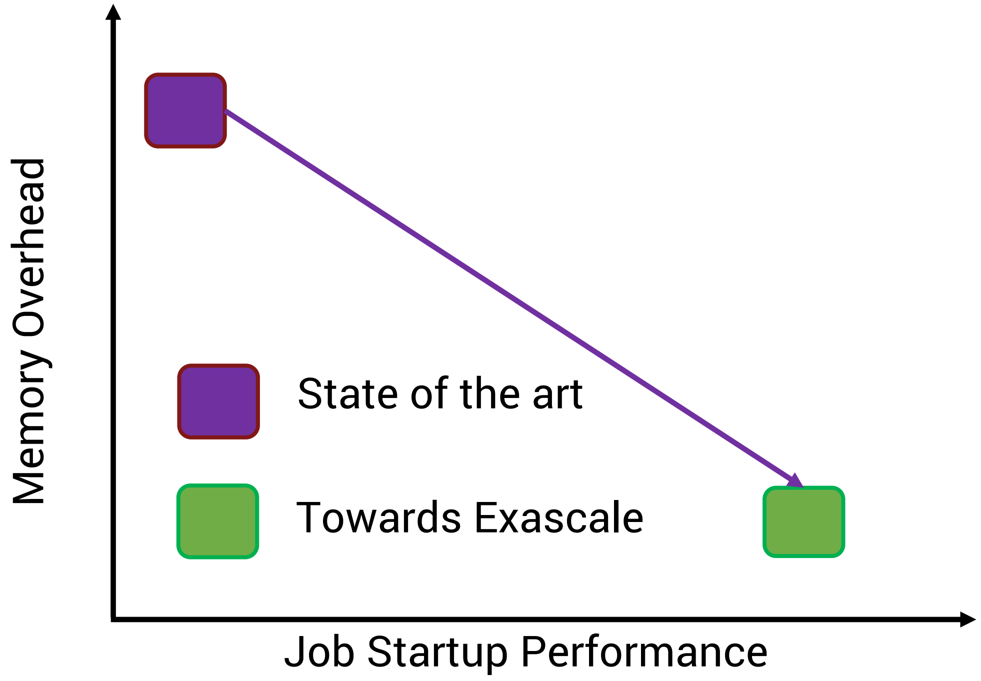
Development and debugging

Regression / Acceptance testing

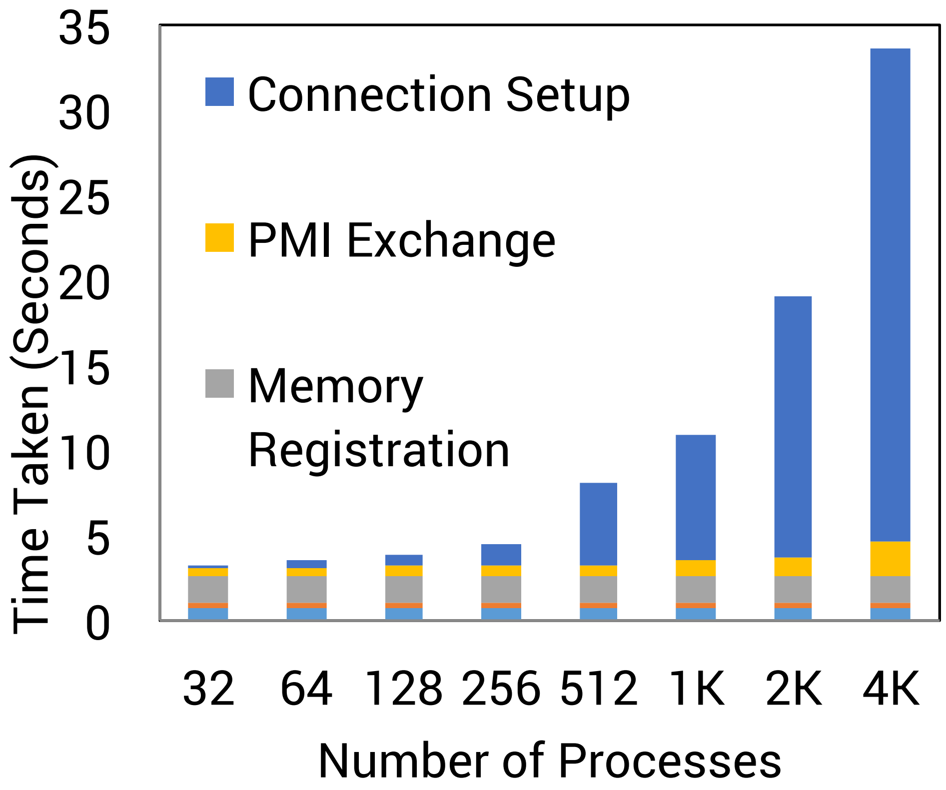
Checkpoint - Restart

Towards Exascale: Challenges to Address

- Dynamic allocation of resources
- Leveraging high-performance interconnects
- Exploiting opportunities for overlap
- Minimizing memory usage



Challenge: Avoid All-to-all Connectivity

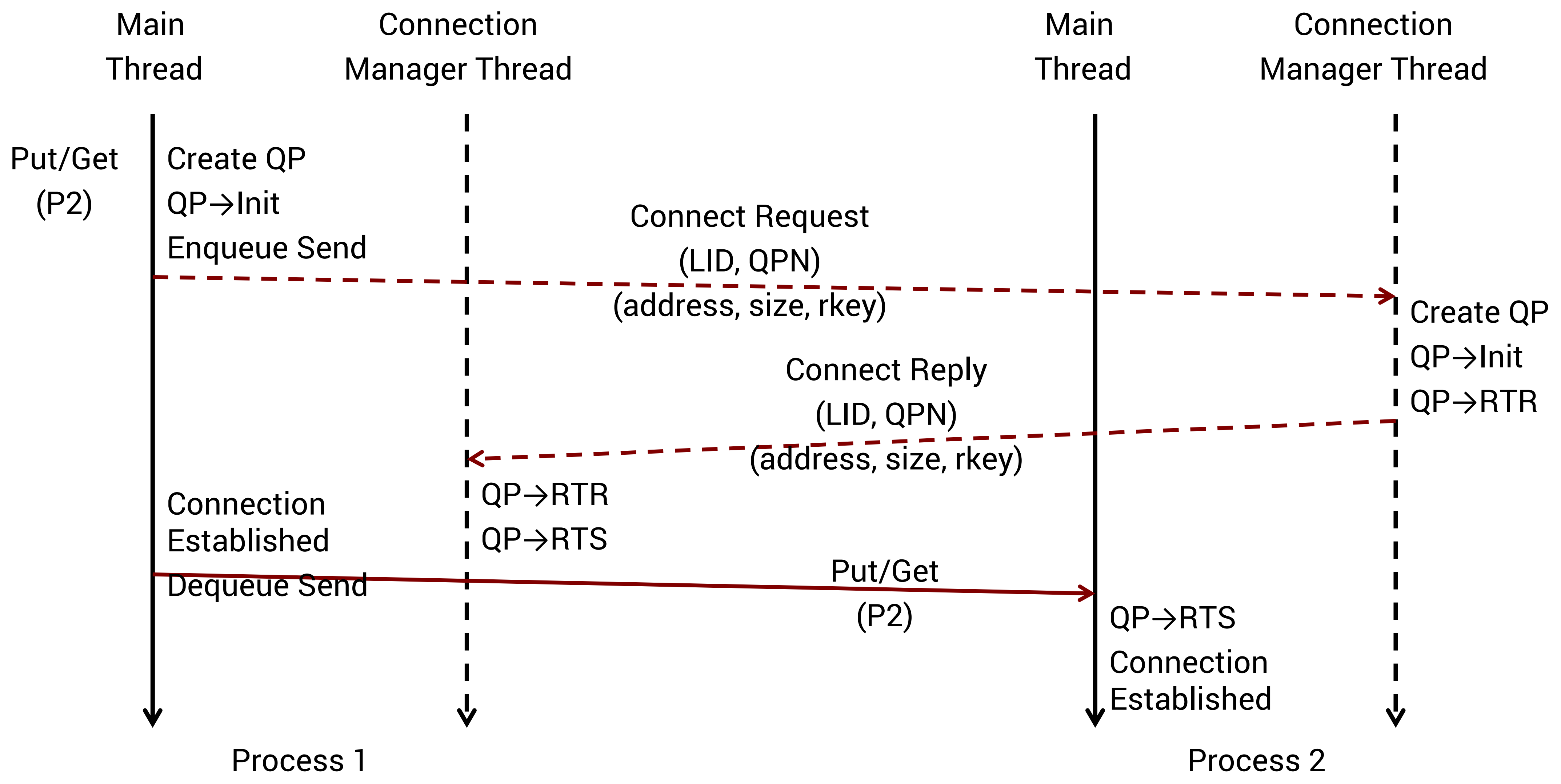


Application	Processes	Average Peers
BT	64	8.7
	1024	10.6
EP	64	3.0
	1024	5.0
MG	64	9.5
	1024	11.9
SP	64	8.8
	1024	10.7
2D Heat	64	5.3
	1024	5.4

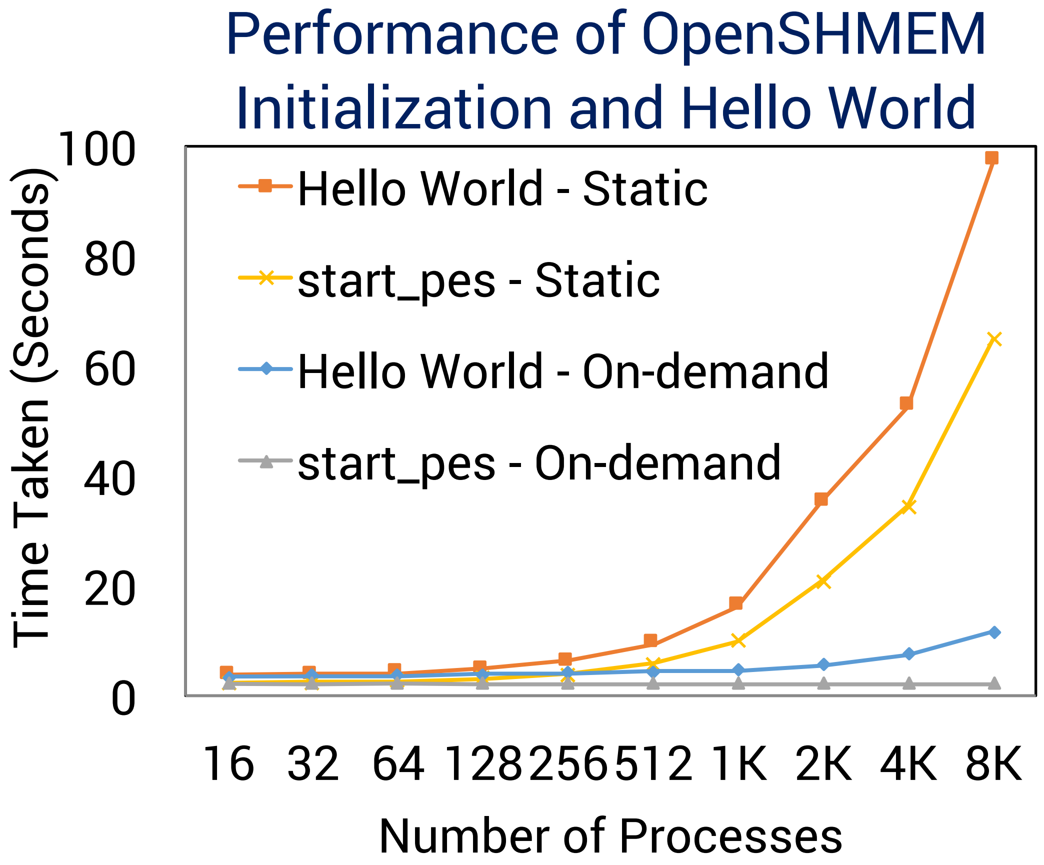
Connection setup phase takes 85% of initialization time with 4K processes

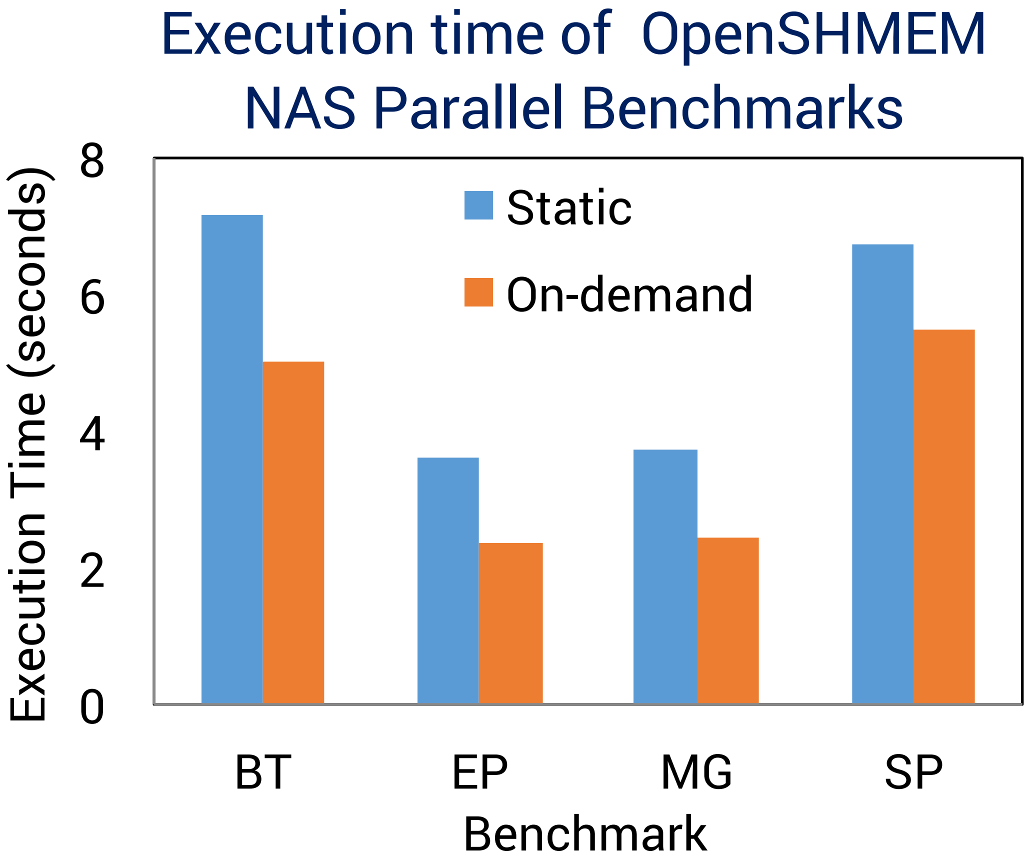
Applications rarely require full all-to-all connectivity

On-demand Connection Management



Results - On-demand Connections

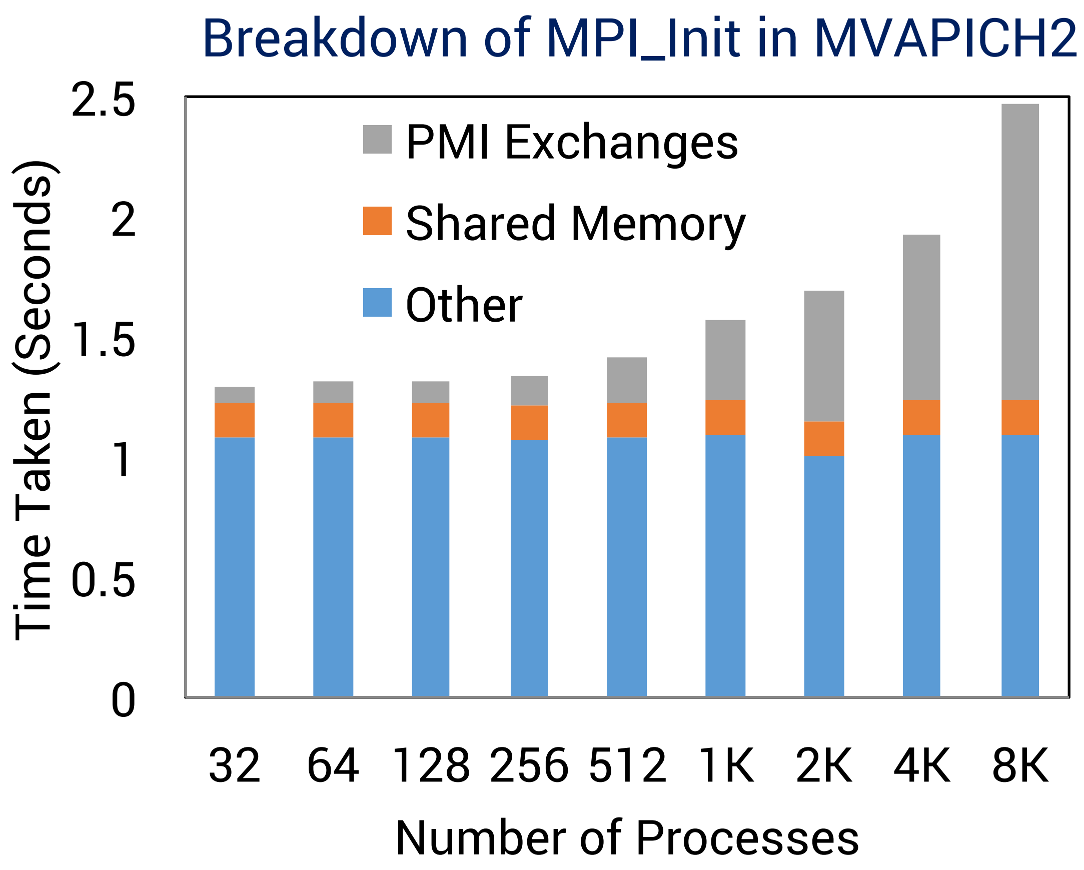




Initialization – 29.6 times faster

Total execution time – 35% better

Challenge: Exploit High-performance Interconnects in PMI



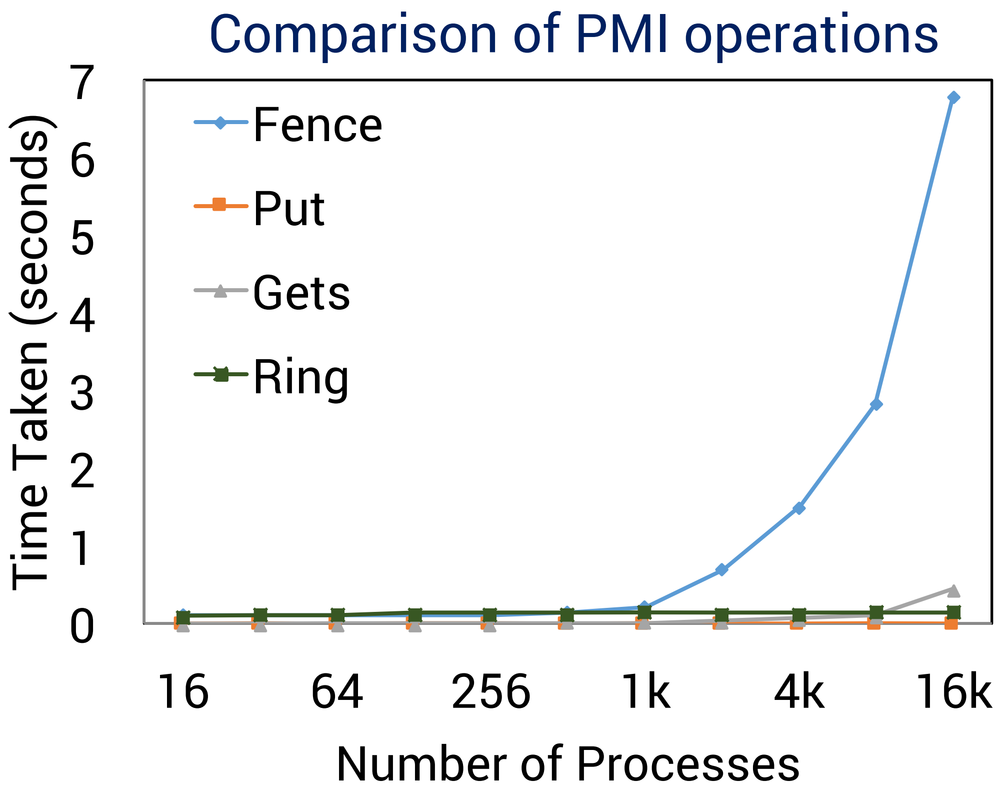
- Used for network address exchange, heterogeneity detection, etc.
  - Used by major parallel programming frameworks
- Uses TCP/IP for transport
  - Not efficient for moving large amount of data
  - Required to bootstrap high-performance interconnects

PMI = Process Management Interface

PMIX_Ring: A Scalable Alternative

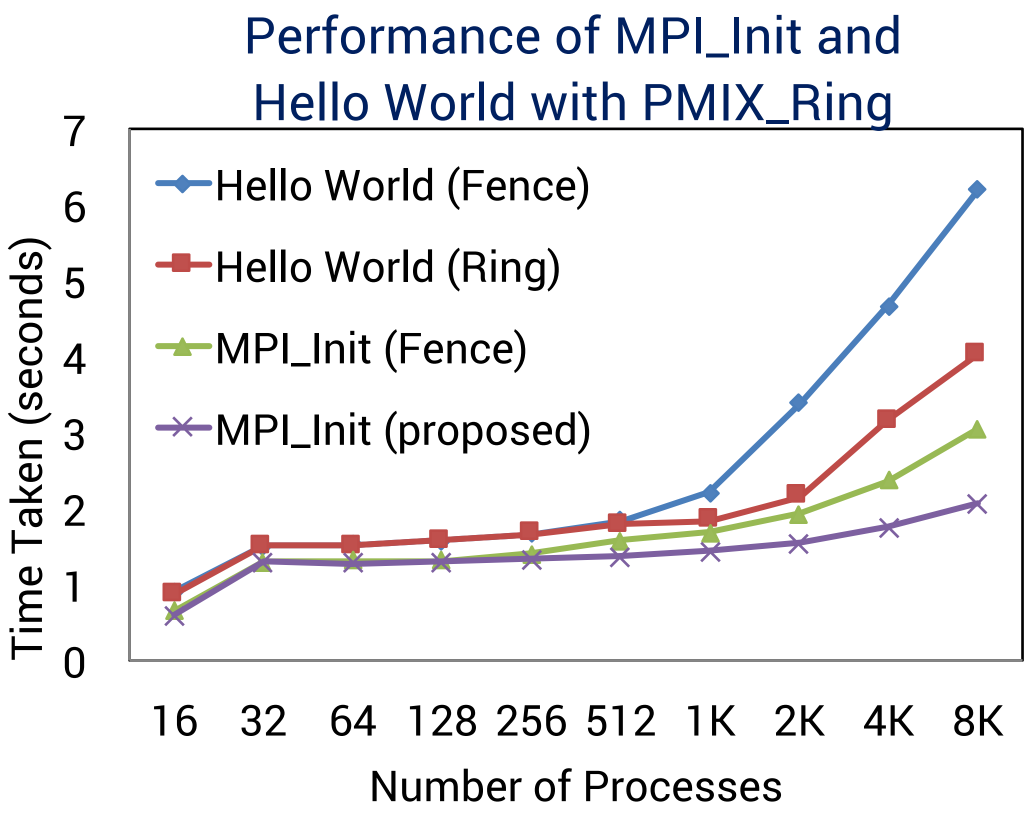
- Exchange data with only the left and right neighbors over TCP
- Exchange bulk of the data over High-speed interconnect (e.g. InfiniBand, OmniPath)

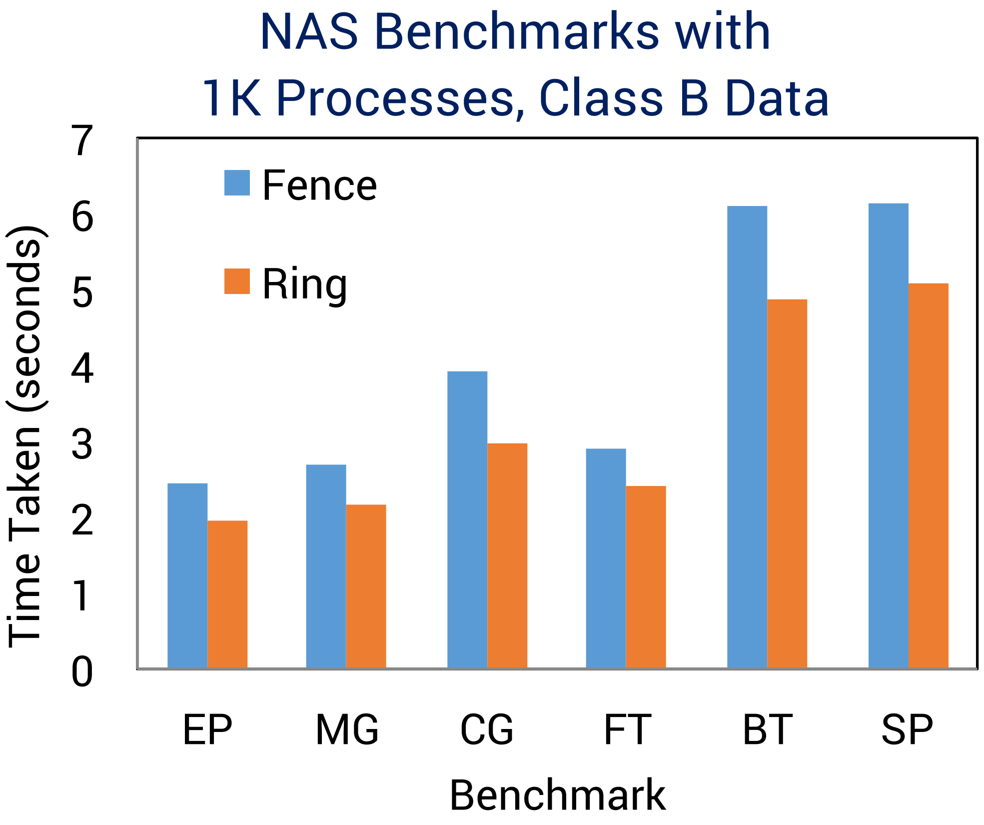
```
int PMIX_Ring(
      char value[],
      char left[],
      char right[],
…)
```



PMIX_Ring is more scalable

Results - PMIX_Ring





33% improvement in MPI_Init

Total execution time – 20% better

Challenge: Exploit Overlap in Application Initialization

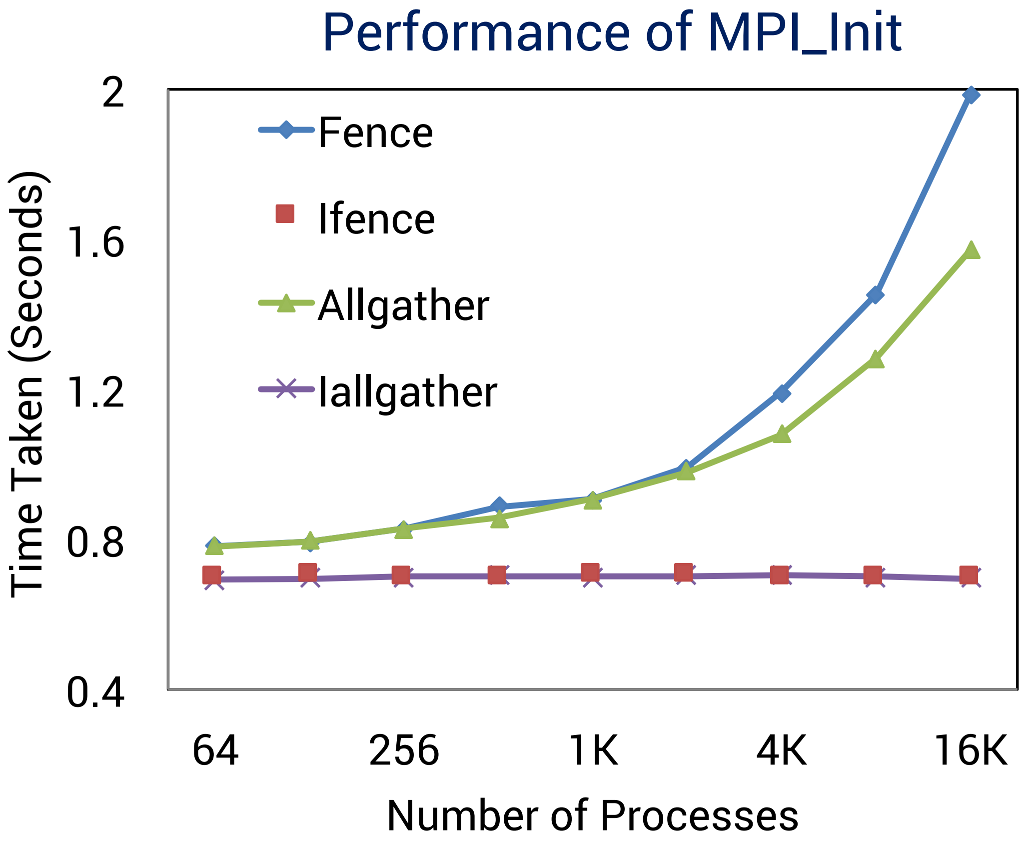
- PMI operations are progressed by the process manager
- MPI/PGAS library is not involved
- Can be overlapped with other initialization tasks / application computation
- Put+Fence+Get combined into a single function - Allgather

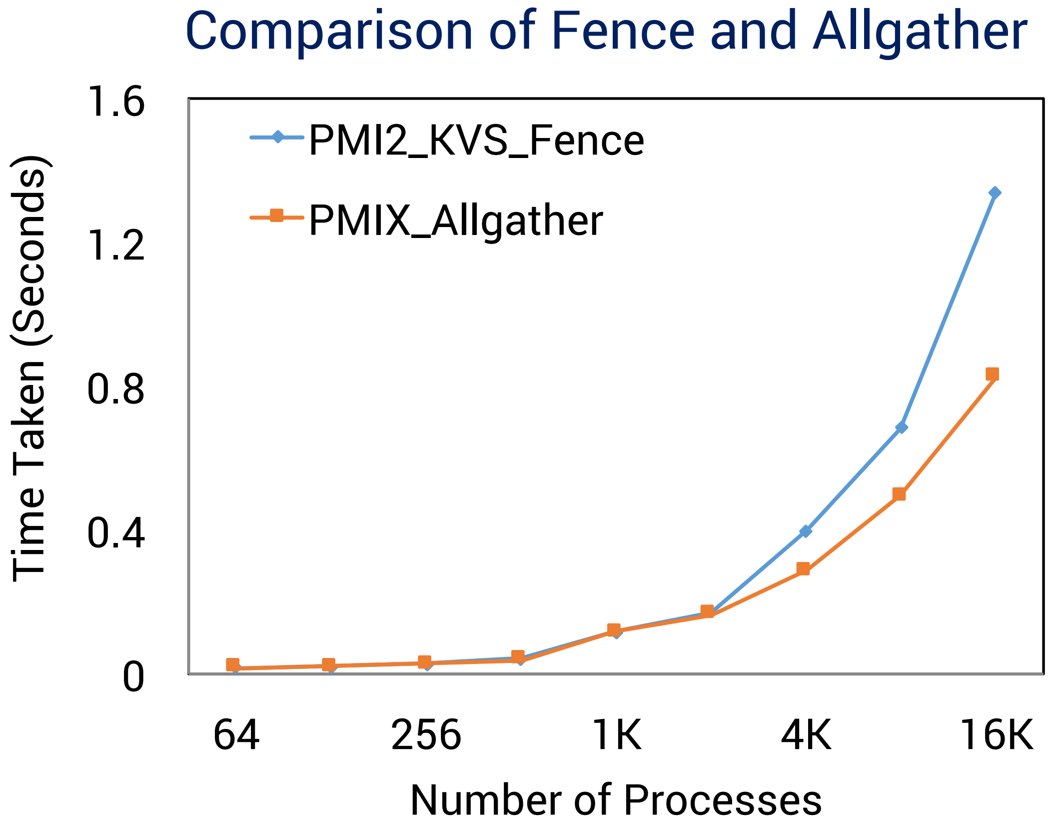
```
int PMIX_KVS_Ifence(
    PMIX_Request *request)

int PMIX_Iallgather(
    const char value[],
    char buffer[],
    PMIX_Request *request)

int PMIX_Wait(
    PMIX_Request request)
```

Results - Non-blocking PMI Collectives





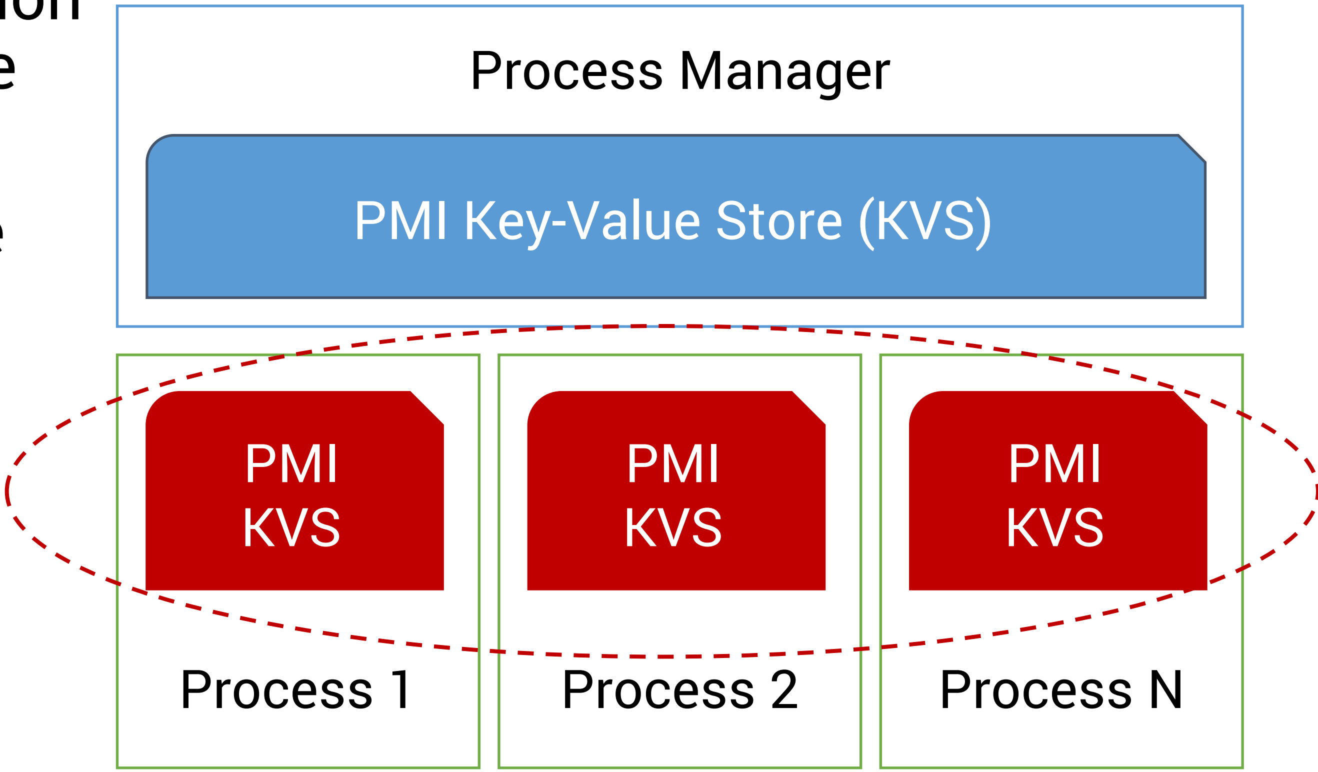
Near-constant MPI_Init at any scale

Allgather is 38% faster than Fence

Challenge: Minimize Memory Footprint

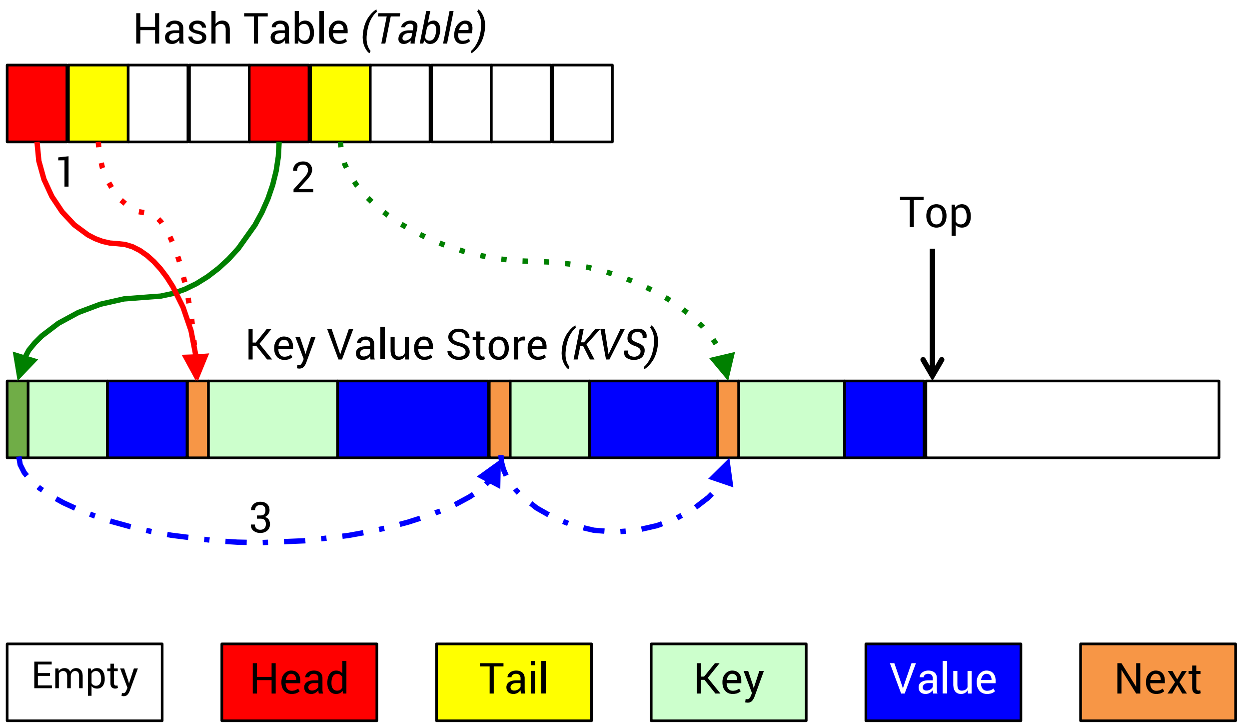
- Address table and similar information is stored in the PMI Key-value store (KVS)
- All processes in the node duplicate the KVS

- PPN redundant copies per node



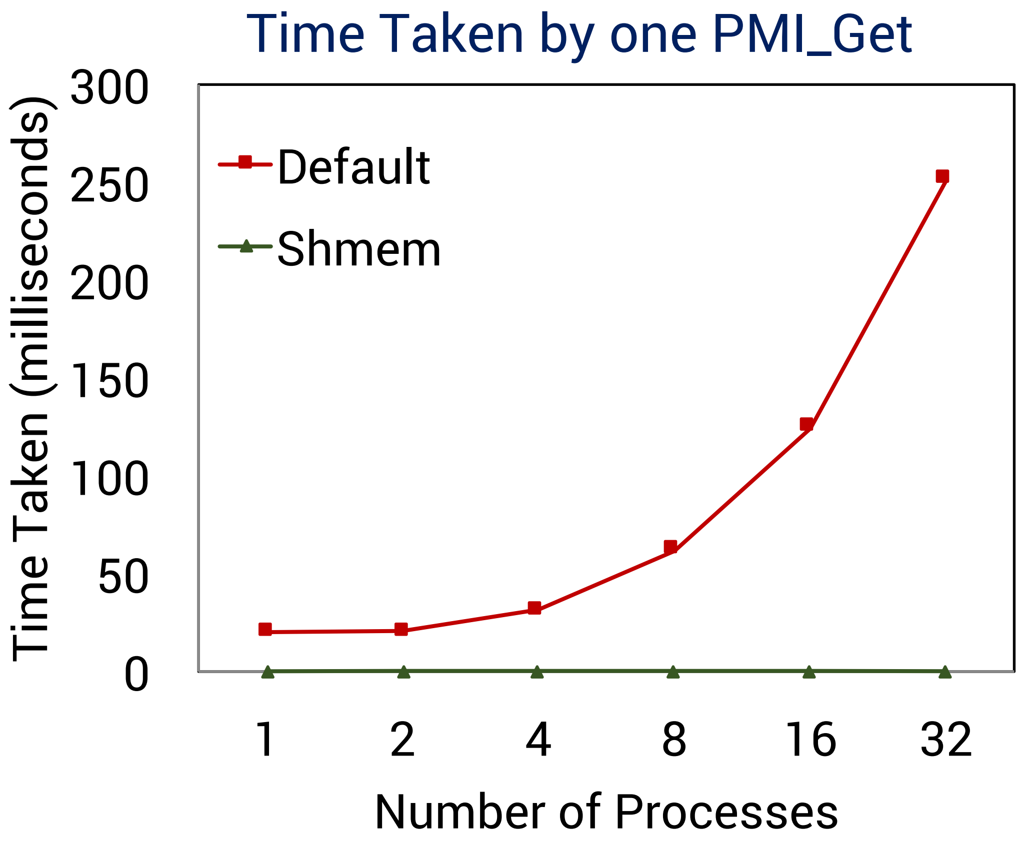
PPN = Number of Processes per Node

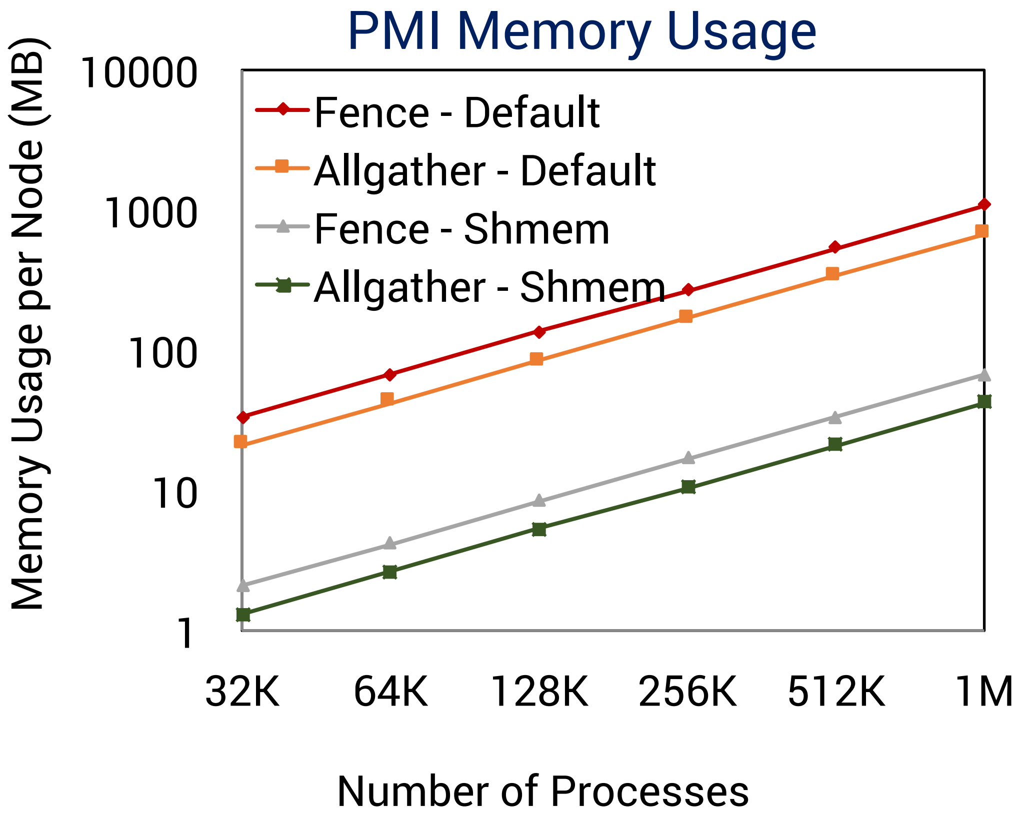
Shared Memory based PMI



- Process manager creates and populates shared memory region
- MPI processes directly read from shared memory
- Dual shared memory region based hash-table design for performance and memory efficiency

Shared Memory based PMI

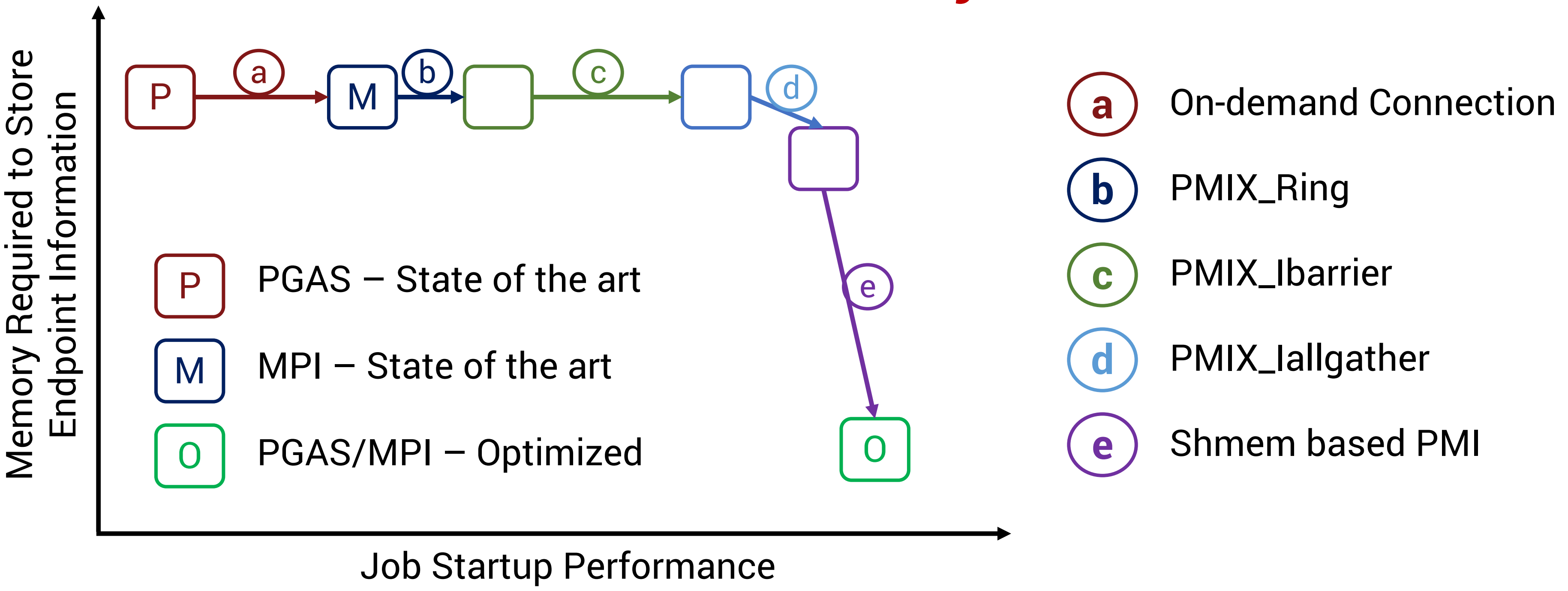




PMI Gets are 1000x faster

Memory footprint reduced by O(PPN)

Summary



- Near constant MPI/OpenSHMEM initialization at any process count
- 10x and 30x improvement in startup time of MPI and OpenSHMEM with 16,384 processes (1,024 nodes)
- O(PPN) reduction in PMI memory footprint

Availability and Impact

- Tested at large-scale on Stampede and LLNL clusters
- All designs available as part of MVAPICH2 / MVAPICH2-X
  - MVAPICH powers Sunway TaihuLight - the #1 SuperComputer in the world!
  - 13th, 241,108-core (Pleiades) at NASA
  - 17th, 462,462-core (Stampede) at TACC
- Can be easily adopted by other MPI libraries and Resource Managers
  - Design of PMIX_Ring contributed to SLURM 15
- Other enhancements available as patches from MVAPICH2 website
  - Ongoing discussion to include them in future SLURM releases

Thank You!

http://go.osu.edu/mvapich-startup

http://mvapich.cse.ohio-state.edu/

chakrabs@cse.ohio-state.edu

panda@cse.ohio-state.edu