MSA Workshop

MExMeMo / IFCES2
June 19th, 2023

Carsten Clauss, Sonja Happ, Simon Pickartz
ParTec AG

Enabling HPC

- ParTec is a strong HPC specialist for more than two decades
  - ParaStation research project: 1995 (Univ. of Karlsruhe, Germany)
  - ParTec founded as a spin-off in 1999
  - HPC full service provider since 2004
  - HPC full systems provider since 2021
- Pioneering the Modular Supercomputing Architecture (MSA) for >10 years
- ParaStation Modulo is extensively used in production environments
  - Serves as the basis for co-design and co-development
  - Also enables ParTec Support services: on-site/remote system operations
- ParaStation Modulo serves as a platform for research activities
  - Used and further developed in Exascale-related projects like DEEP, {DEEP, RED, IO}-SEA, EUPEX
  - Also serves as a platform for MSA in Quantum- and AI-related projects like HPCQS, QSolid and CoE RAISE

PARASTATION MODULO SOFTWARE SUITE

ParTec MODULAR SUPERCOMPUTING

ParaStation CLUSTERTOOLS | ParaStation HEALTHCHECKER | ParaStation TICKETSUITE | ParaStation MPI

Tools for Provisioning and Management

- System management CLI
- Image management
- Rolling updates
- Stateless & stateful booting
- Post-install configuration
- Slurm integration
- Distributed database for system configuration
- HealthChecker integration

Integrity of the Computing Environment

- Automated error detection & error handling
- Various hook-in points
- No interference with jobs
- TicketSuite integration
- Highly configurable

- 100+ tests (HW/SW):
- Node/System/Fabric level

Issue Tracking on System Level

- Manual and automatic ticket creation
- Prioritization
- Routing/Triage
- Documentation and central information hub
- Maintenance planning
- Interfaces with external ticketing systems

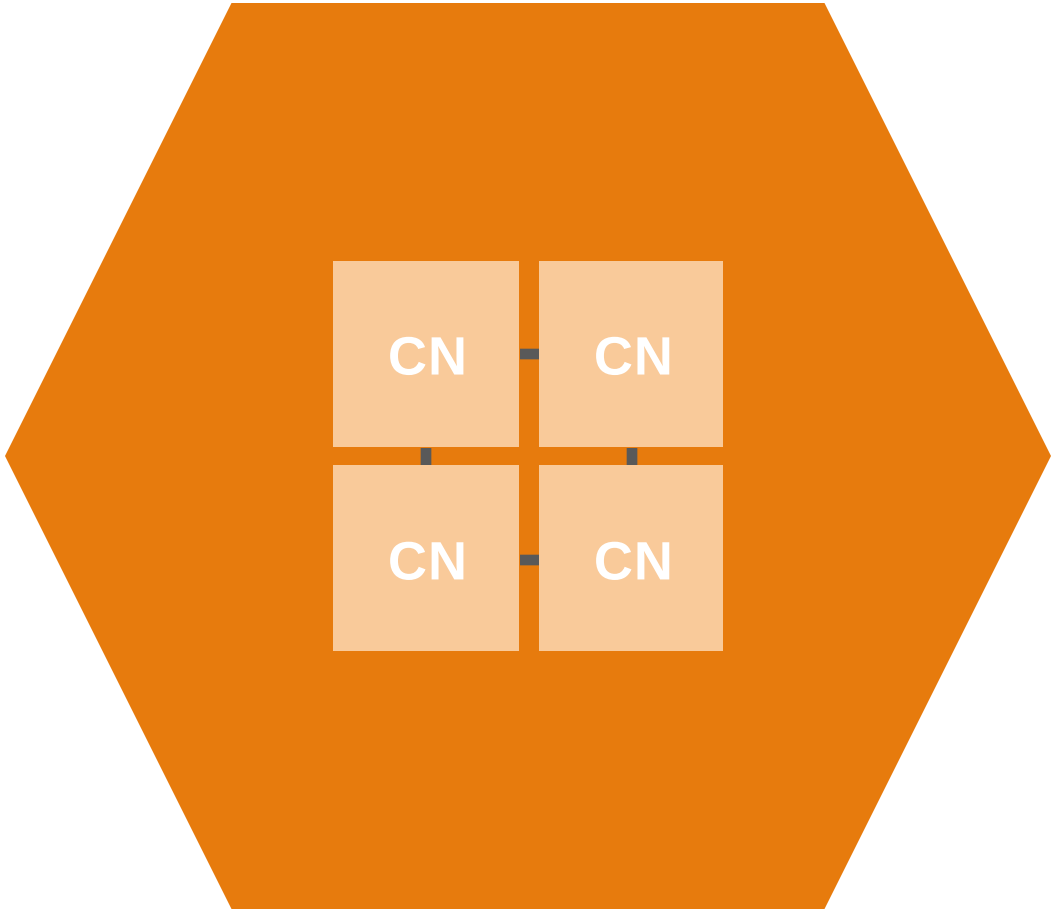
Execution Environment and MPI Library

- MPI-4.0-compliant
- MPICH ABI-compatible
- Supports multiple interconnects in parallel
- Modularity support
- Network bridging
- PMIx support
- Full Slurm integration

FROM TRADITIONAL HPC TO MODULAR SUPERCOMPUTING

HOMOGENEOUS CLUSTERS

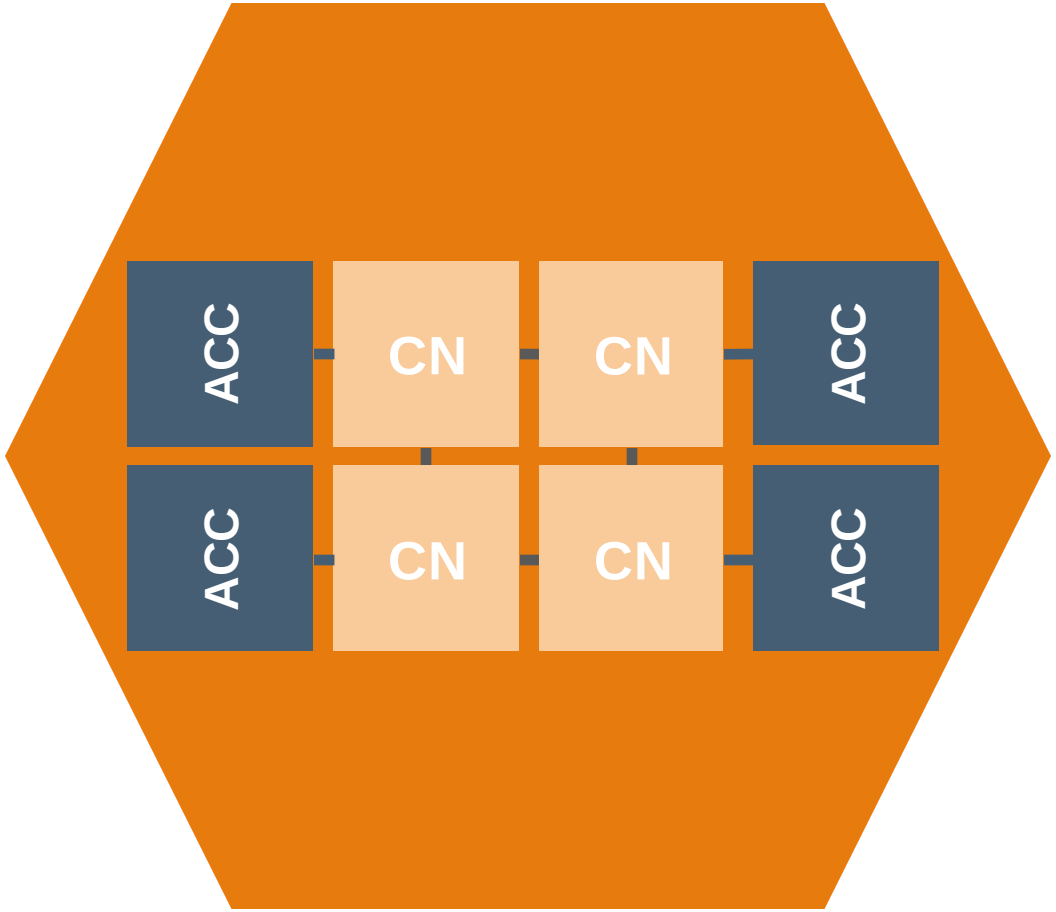
- Cluster Nodes: general purpose (multi-core) processor technology
- Same processor characteristics in all nodes
- Single high-speed network connecting them all
- Good concept but limited efficiency for selected HPC applications (general purpose ballast)



FROM TRADITIONAL HPC TO MODULAR SUPERCOMPUTING

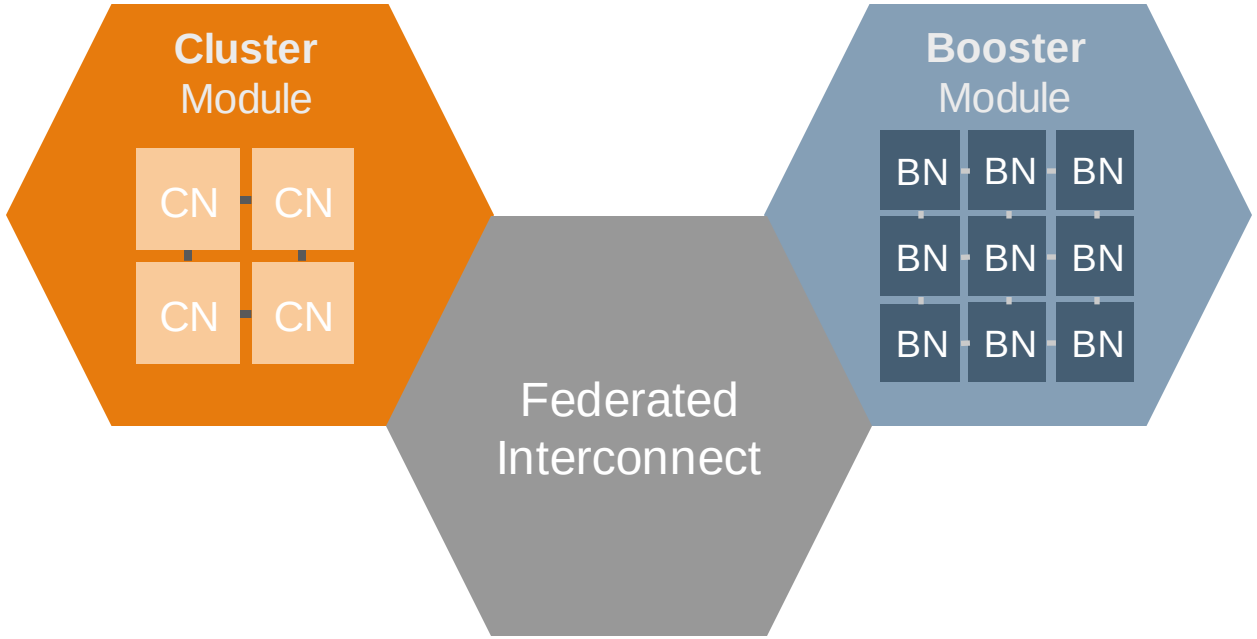
ACCELERATED CLUSTERS

- Fixed ratio and assignment of accelerators to CPUs
- Commonly realized by means of “Fat”-Nodes
- Static management of resources
- Accelerators do not act autonomously
- General-purpose cluster interconnect
- Programming via local offload interfaces



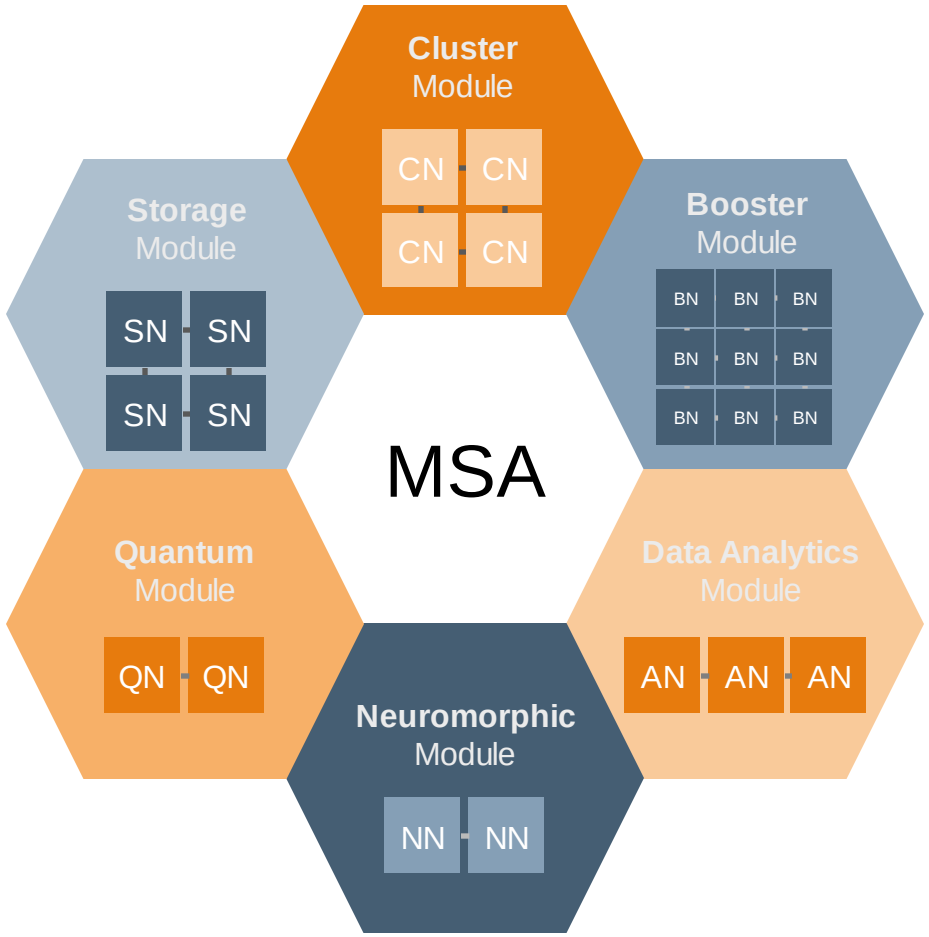
CLUSTER-BOOSTER ARCHITECTURE

- No fixed ratio or assignment between resources (multicore & manycore nodes)
- Dynamic management and association of resources
- High-throughput network in the Booster
- Per-module SPMD
- System-wide, transparent communication and execution environment provided by MPI



MODULAR SUPERCOMPUTING ARCHITECTURE

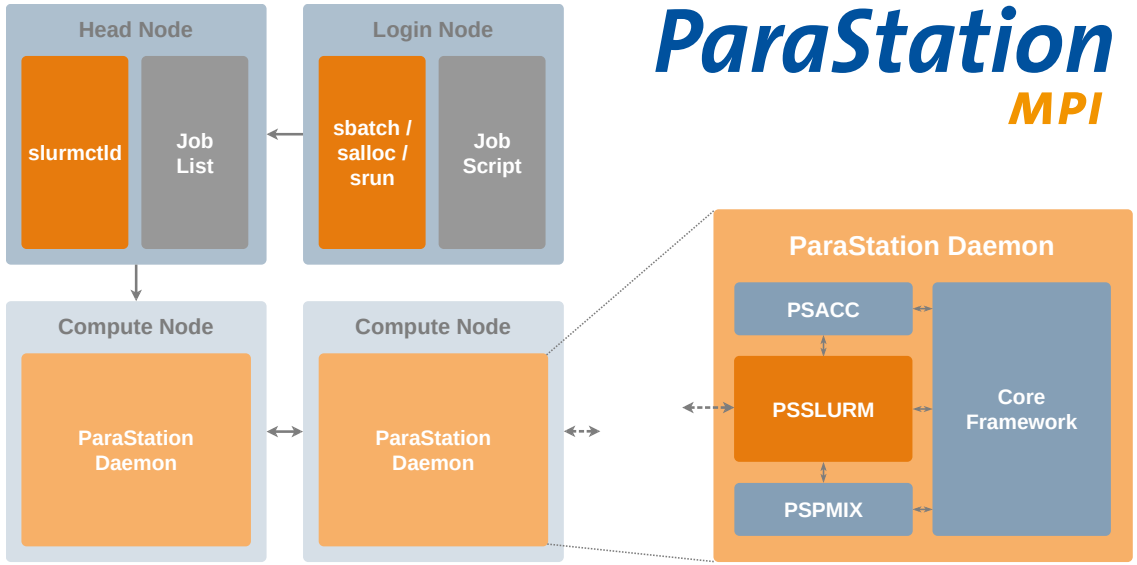
- Generalization of the Cluster-Booster Concept
  - Heterogeneity on the system level
  - Effective resource sharing
- Any number of (specialized) modules possible
  - Cost-effective scaling
  - Extensibility of existing modular systems by adding modules
- Fit application diversity
  - Large-scale simulations
  - Data analytics
  - Machine/Deep Learning, AI
  - Hybrid-quantum Workloads
- Achieve leading scalability and energy efficiency
  - Exascale-ready!
- Unified software environment for running across all modules
  - Enabled by the ParaStation Modulo software suite



PROCESS MANAGEMENT

- Scalable network of MPI process management daemons
  - One instance running on each of the computational nodes
  - Responsible for process startup and control
  - Responsible for intra-job resource assignment
  - Provides precise resource monitoring
  - Provides a PMIx server to the application
  - Guarantees proper cleanup after jobs

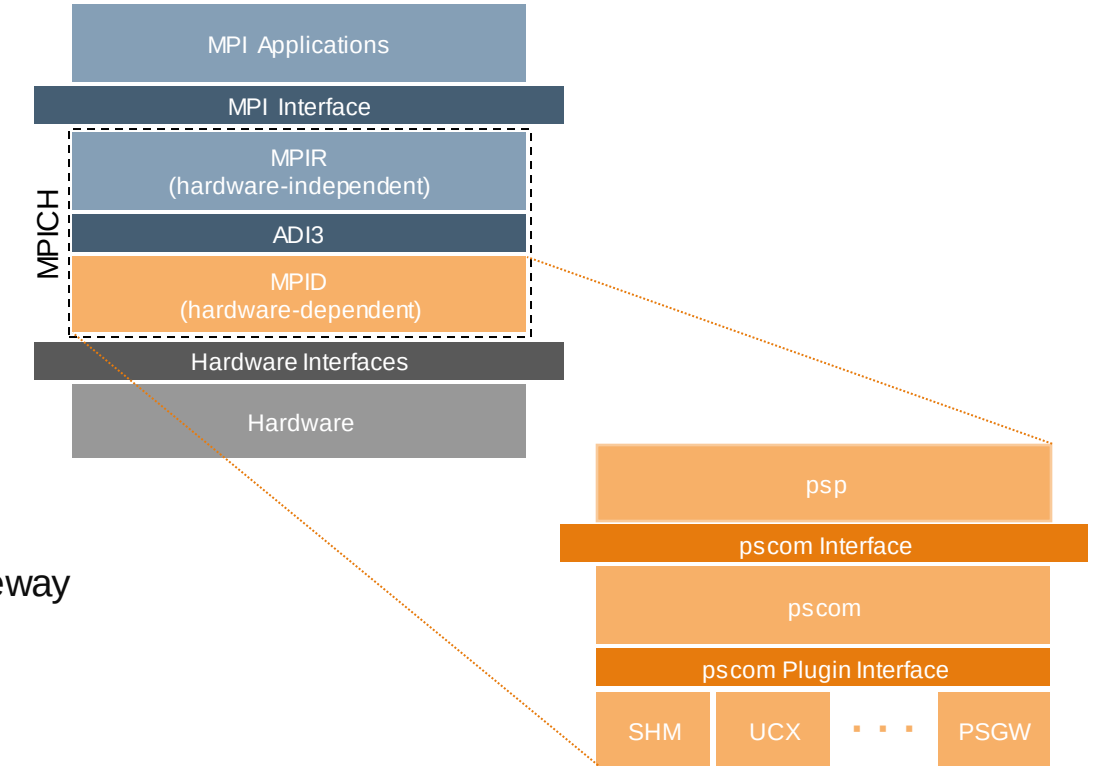
- psslurm: Full integration for Slurm
  - Implemented as plugin (i.e., loadable shared library) to the ParaStation Management daemon
  - Replaces node-local Slurm daemons
  - Enforces resource limits
  - Collects misc. information, e.g., accounting, energy, file system usage, ... and forwards it to the slurmctld



ParaStation MPI

PARASTATION MPI

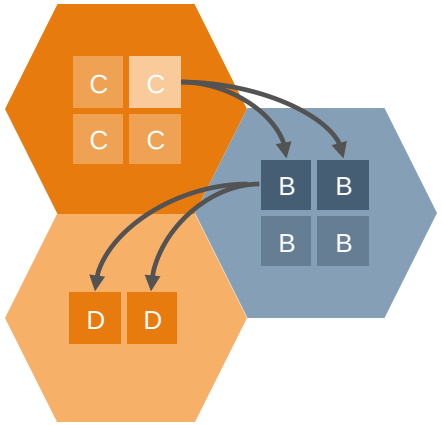
- Based on MPICH 4.1.1
  - Support MPICH tools for tracing, debugging, etc.
  - Integrates into MPICH on the MPID layer by implementing an ADI3 device
  - The PSP Device is powered by pscom – a low-level point-to-point communication library
  - Support the MPICH ABI Compatibility Initiative
- Support for various transports / protocols via pscom plugins
  - Support for InfiniBand, Omni-Path, BXI, SHM, etc.
  - Concurrent usage of different transports
  - Transparent bridging between any pair of networks enabled by gateway capabilities
- Proven scalability to more than 140,000 processes



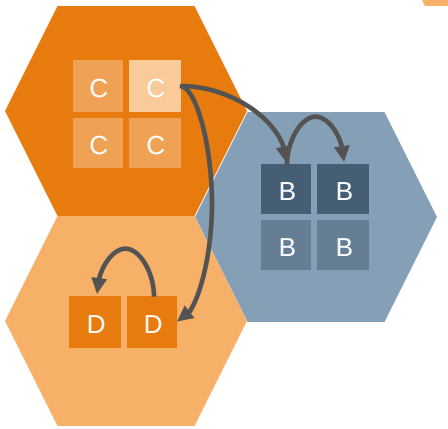
ParaStation MPI

MSA AWARENESS

- Support for multi-level hierarchy-aware collectives
  - Optimize communication patterns to the topology of the MSA
  - Assumption: Inter-module communication is the bottleneck
  - Dynamically update the communication patterns (experimental)
- API extensions for accessing modularity information
  - New MPI split type for communicators (`MPIX_COMM_TYPE_MODULE`)
  - Provide the module id via the `MPI_INFO_ENV` object
- MPI Network Bridging
  - Connect any pair of interconnect and protocol
  - Transparent to the application layer



Binary Tree
(default)



Hierarchical
(MSA-aware)

MODULAR MPI JOBS

```
#
## Example for a modular job on JUWELS
#

# Use of srun with colon notation
$ srun … : …

# Modules JUWELS Cluster and JUWELS Booster
$ srun --partition=batch … : --partition=booster …

# 8 Nodes / 8 Procs. per node on the Cluster and 16 Nodes / 4 Procs. per node on the booster
$ srun --partition=batch –N 8 --ntasks-per-node=8 … : --partition=booster -N 16 --ntasks-per-node=4 …
```

HIERARCHICAL COLLECTIVES

- General rules used to optimize collectives
  1. First do all module-internal gathering and/or reduction operations – if required
  2. Then perform the inter-module operation with only one process per module
  3. Finally, distribute the data within each module in a strictly module-local manner

- Multi-level hierarchy awareness
  - Apply this set of rules *recursively* (i.e., first on module level, second on node level, etc.)

- Usage: Set environment variables
  - `PSP_MSA_AWARENESS=1`
  - `PSP_SMP_AWARE_COLLOPS=1`
  - `PSP_MSA_AWARE_COLLOPS=1|2`

These features are not always beneficial and/or are still experimental, they are *disabled* by default

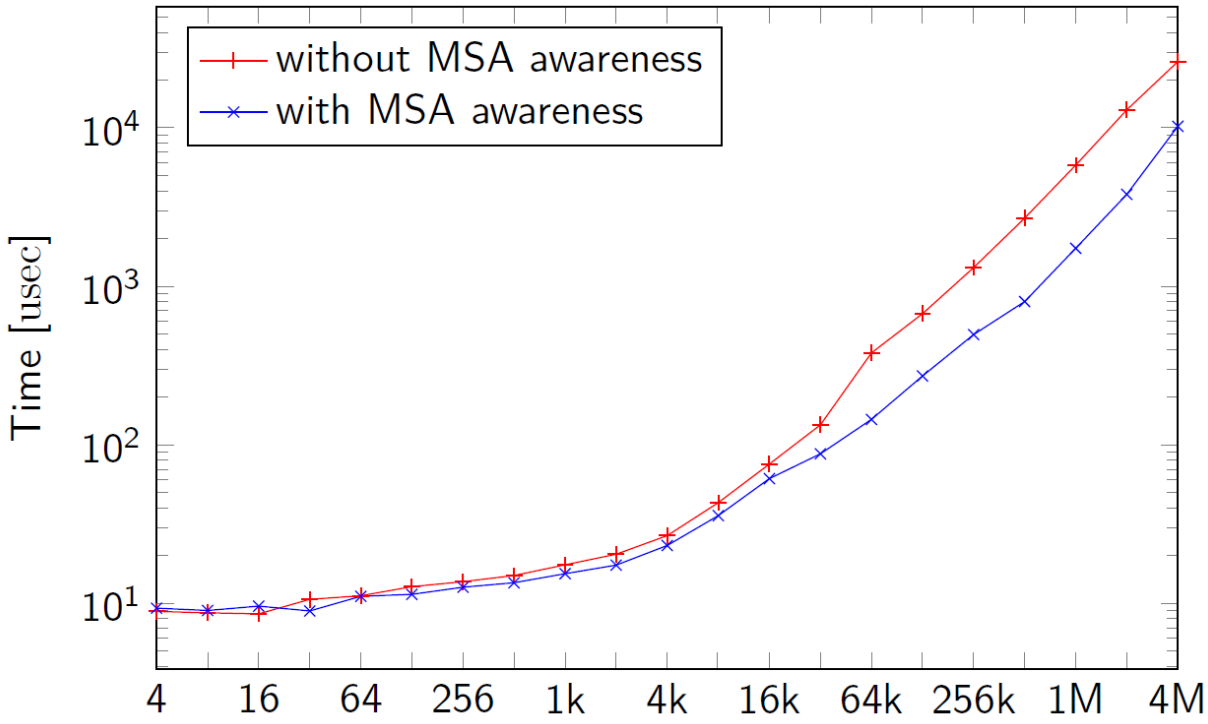
1: apply either SMP or MSA awareness
2: apply both recursively

HIERARCHICAL COLLECTIVES – PERFORMANCE IMPROVEMENT

- Improvement heavily depends on the setting, e.g.
  - Number of Processes / gateway nodes involved
  - Rank distribution in the communicator
  - Message sizes of the communication pattern
  - … and the pattern itself

- Usage: Set environment variables
  - `MPI_Bcast / MPI_Ibcast`
  - `MPI_Reduce / MPI_Ireduce`
  - `MPI_Allreduce / MPI_Iallreduce`
  - `MPI_Scan / MPI_Iscan`
  - `MPI_Barrier`

IMB MPI Benchmarks: Allreduce with 8 (CN) + 8 (DAM-EXT) nodes, 8 procs per node and 1 Gateway (GW) node on DEEP



API EXTENSIONS

- Support for MSA awareness on the application level

- Retrieve explicit or implicit topology information by
  - Querying the module ID via the `MPI_INFO_ENV` object

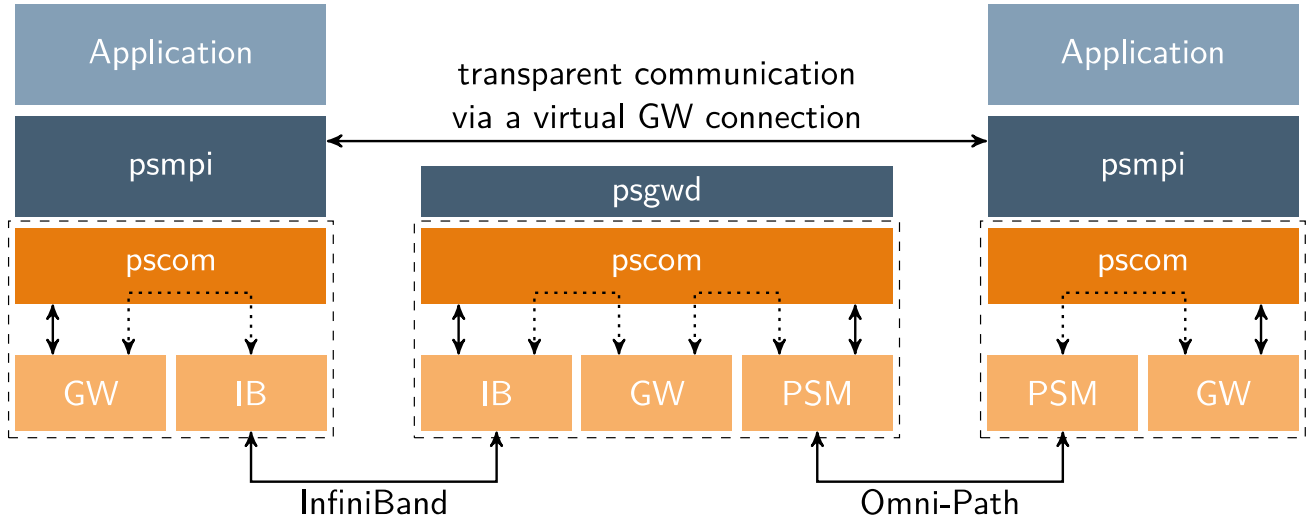
```
MPI_Info_get(MPI_INFO_ENV, “msa_module_id”, …, value, …);
```

  - Splitting communicators according to the topology by using the `MPIX_COMM_TYPE_MODULE` split type

```
MPI_Comm_split_type(oldcomm, MPIX_COMM_TYPE_MODULE, …, &newcomm);
```

MPI NETWORK BRIDGING

- Transparent communication across networks
  - Use a gateway when two processes are not directly connected through the same network
  - Bridging between any pair of interconnects supported by pscom (e.g., InfiniBand, Omni-Path, BXI, etc.)
- Static routing
  - Use the same gateway for different destinations
  - Virtual GW connections provide full transparency to the application layer
- Successfully deployed in production environments
  - Implemented first for the JURECA Cluster-Booster System
  - Bridging between Mellanox EDR and Intel Omni-Path

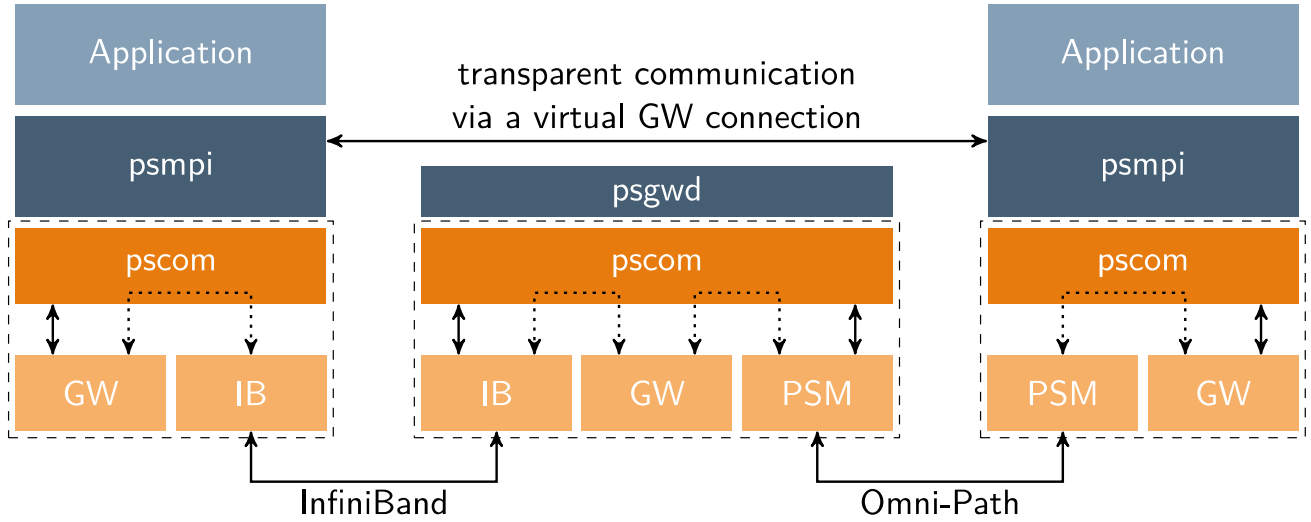


MPI NETWORK BRIDGING

- Compensate for Slurm's inability to handle global resources
- `psgw` plugin to ParaStation Management Daemon + spank plugin
- Extends `salloc`, `srun`, and `sbatch` with these options …

```
--gw_num=number    Number of gateway nodes (required)
--gw_file=path     Path to the gateway routing file
--gw_plugin=string Name of the route plugin
--gw_env=string    Additional gateway environment variables
--gw_cleanup       Automatically cleanup the route file
```

… to allocate gateway resources, automatically start gateway daemons, and create a routing file



MODULAR WORKFLOWS

- An MPI job started with colon notation via `srun` will run in a single `MPI_COMM_WORLD`
- Workflows may demand for multiple `MPI_COMM_WORLDs` that may connect (and later disconnect with each other during runtime)
- Simple job script example for such a case

```
#!/bin/bash
#SBATCH --partition=batch
#SBATCH --nodes=8
#SBATCH --ntasks-per-node=8

#SBATCH hetjob

#SBATCH --partition=booster
#SBATCH --nodes=16
#SBATCH --ntasks-per-node=4

srun --het-group=0 ./mpi_hello_accept &
srun --het-group=1 ./mpi_hello_connect &

wait
```

Start 2 *separate* `MPI_COMM_WORLDs`

THANK YOU FOR YOUR ATTENTION

QUESTIONS

ParTec AG, Possartstr. 20, D-81679 München – www.par-tec.com
{clauss, sonja.happ, pickartz}@par-tec.com