To Share or Not to Share: a case for MPI in Shared-Memory

Julien Adam[1], Jean-Baptiste Besnard[1], Adrien Roussel[2,3], Julien Jaeger[2,3], Patrick Carribault[2,3], Marc Pérache[2,3]

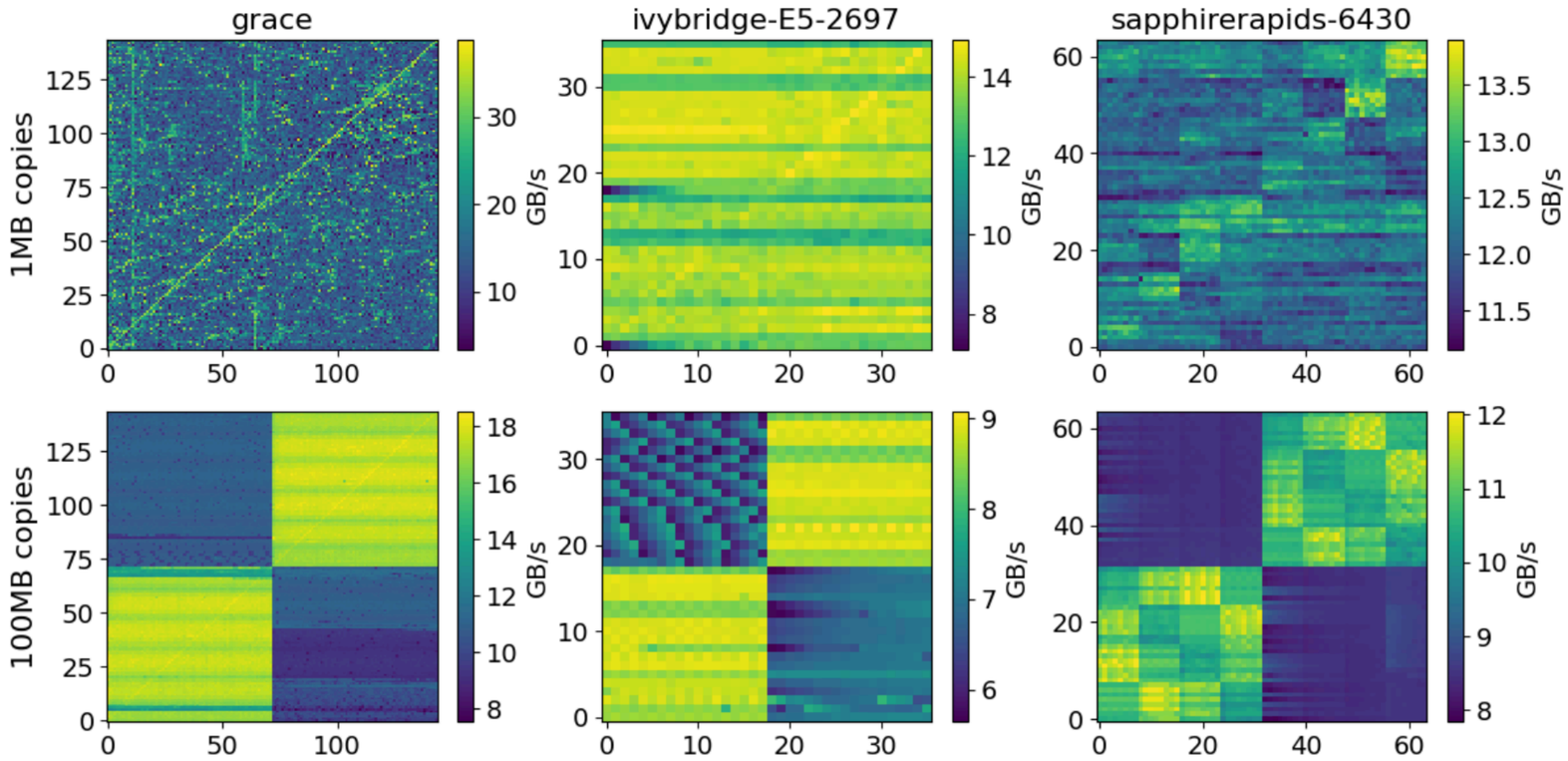
1. ParaTools SAS, Bruyères-le-Châtel, France
2. CEA, DAM, DIF, F91297 Arpajon, France
3. Université Paris-Saclay, CEA, Laboratoire en Informatique Haute Performance pour le Calcul et la simulation
91680 Bruyères-le-Châtel, France

Discussing how Shared-Mem can Benefit to MPI

- The Obvious Starting Point
- MPI and Compute Topology
- Horizontalization of Compute
- Current Support of MPI in SHMEM
- Sessions at the Rescue

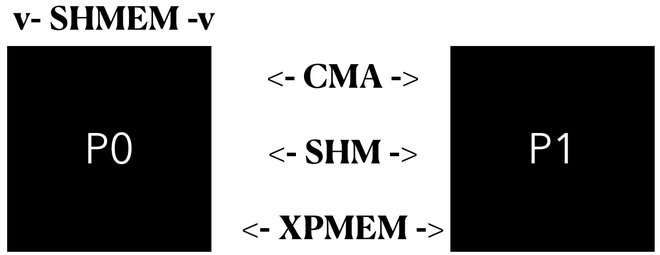
The Obvious Starting Point

Nodes Have a Topology

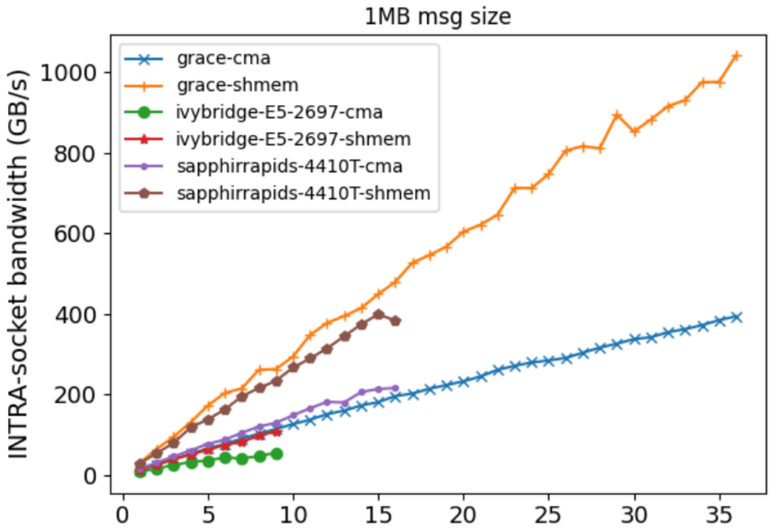


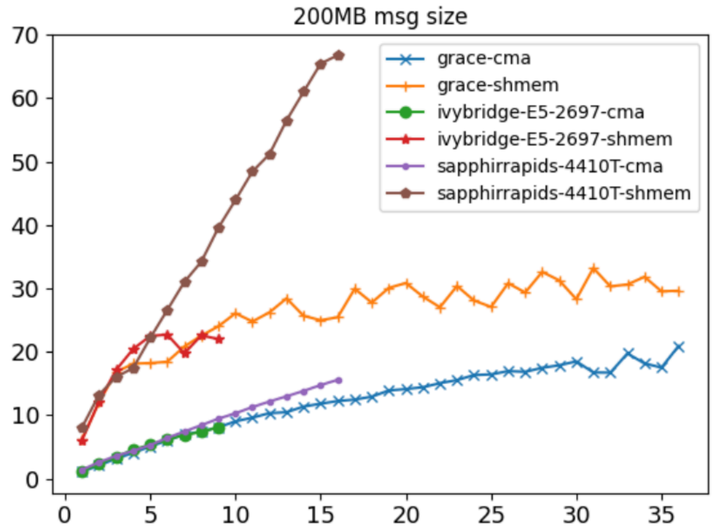
Core to core bandwidth

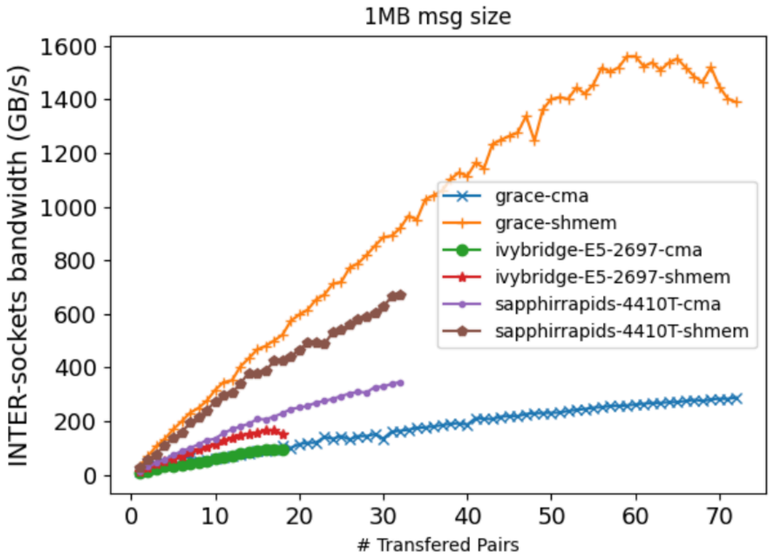
Performance in Shared-Memory

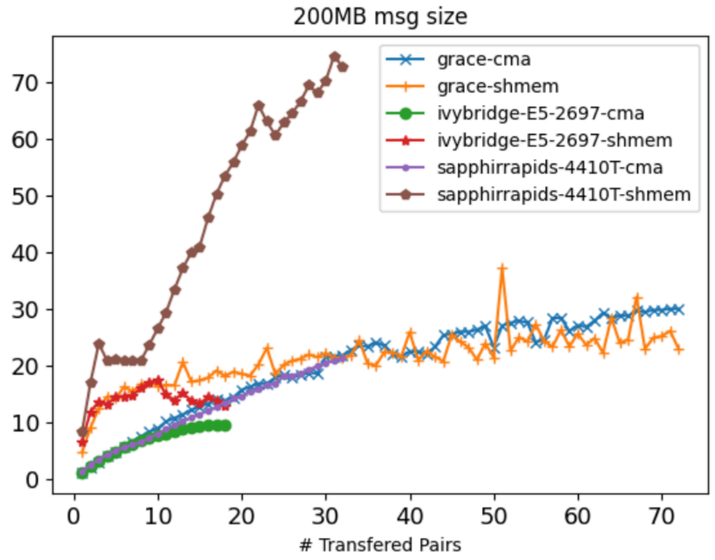


- Shared-memory transfers give the best performance compared to CMA when doing trivial benchmarks
- Note XPMEM was not deployed in test environment





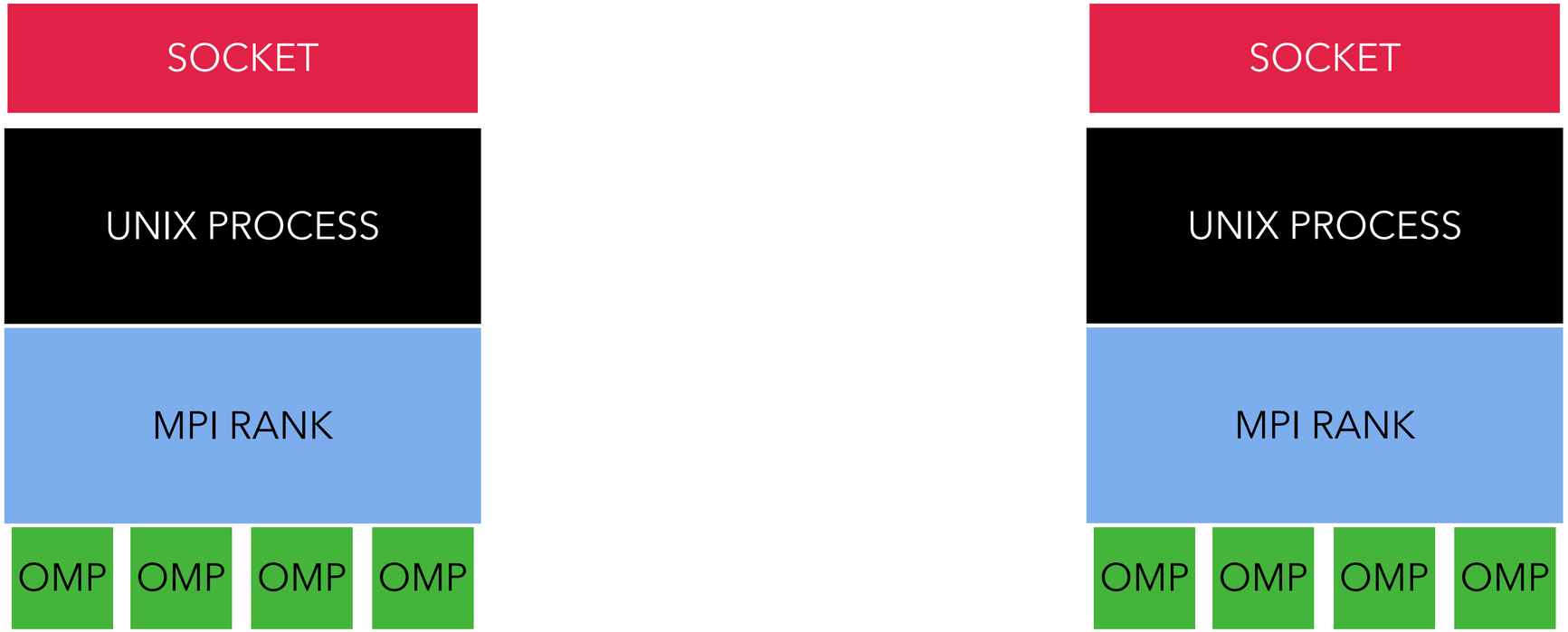




MPI and Compute Topology

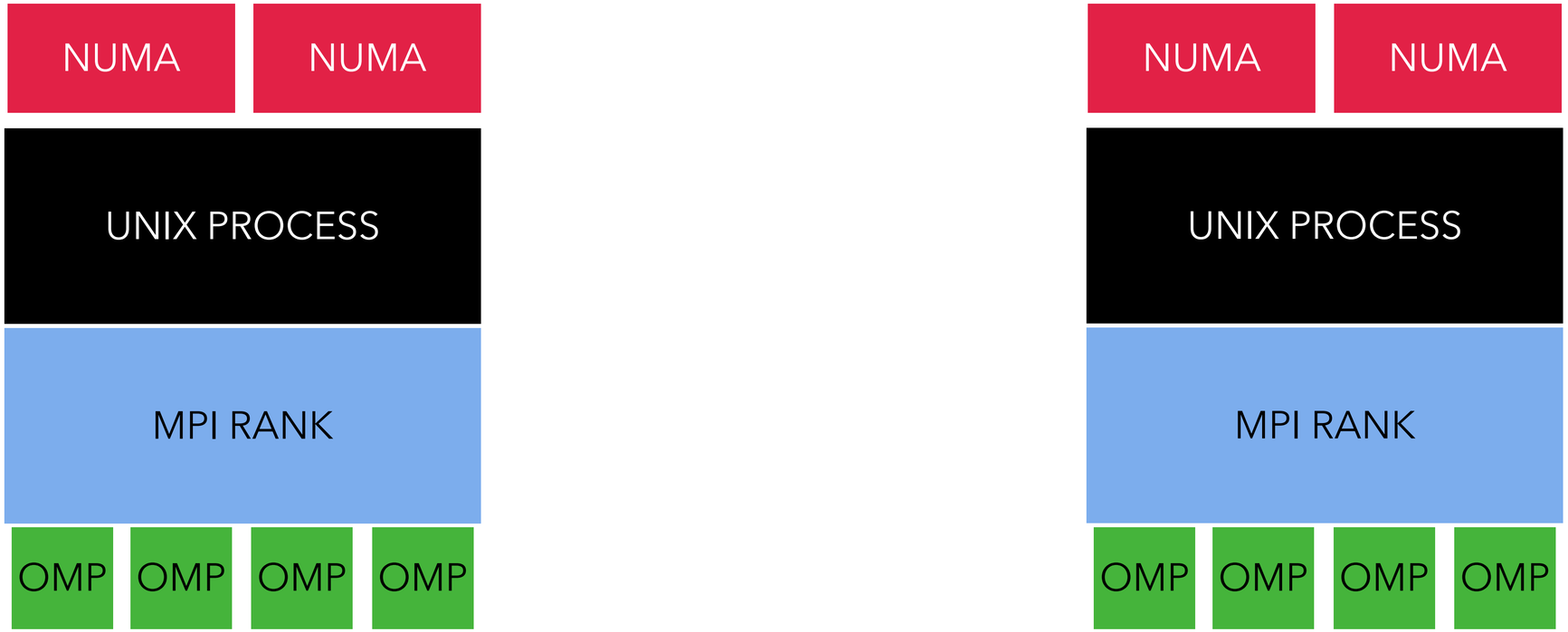
EuroMPI/Australia 2024 — 09/26/2024

Seeing MPI as the Embedding Model



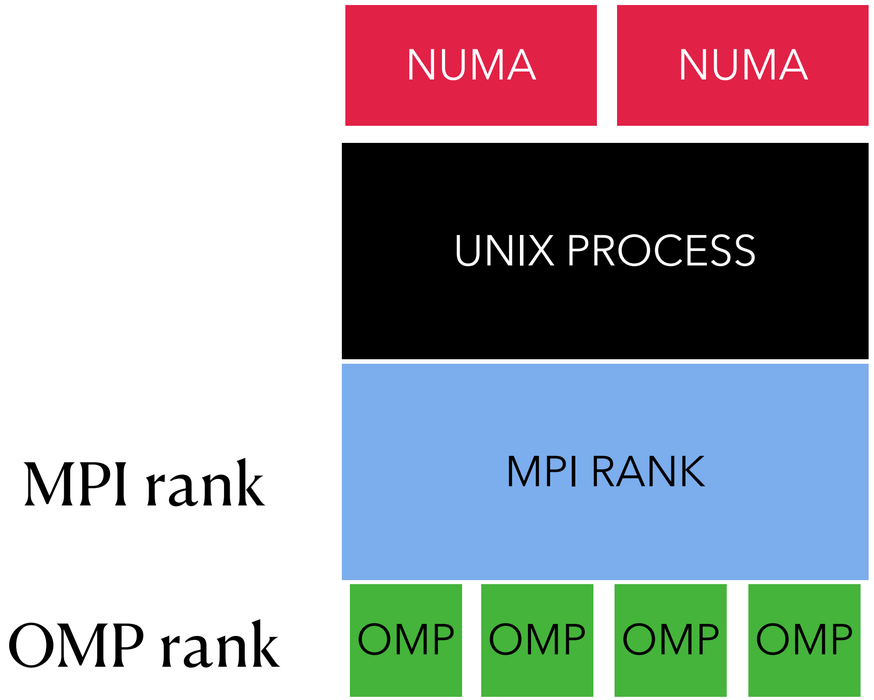
MPI is in charge of **Mapping** and **Addressing** the distributed ranks.

Seeing MPI as the Embedding Model

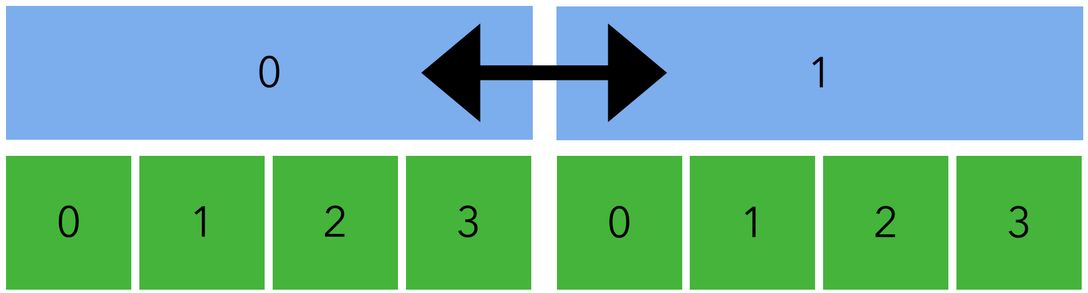


OMP Pinning is generally inherited from MPI

Seeing MPI as the Embedding Model

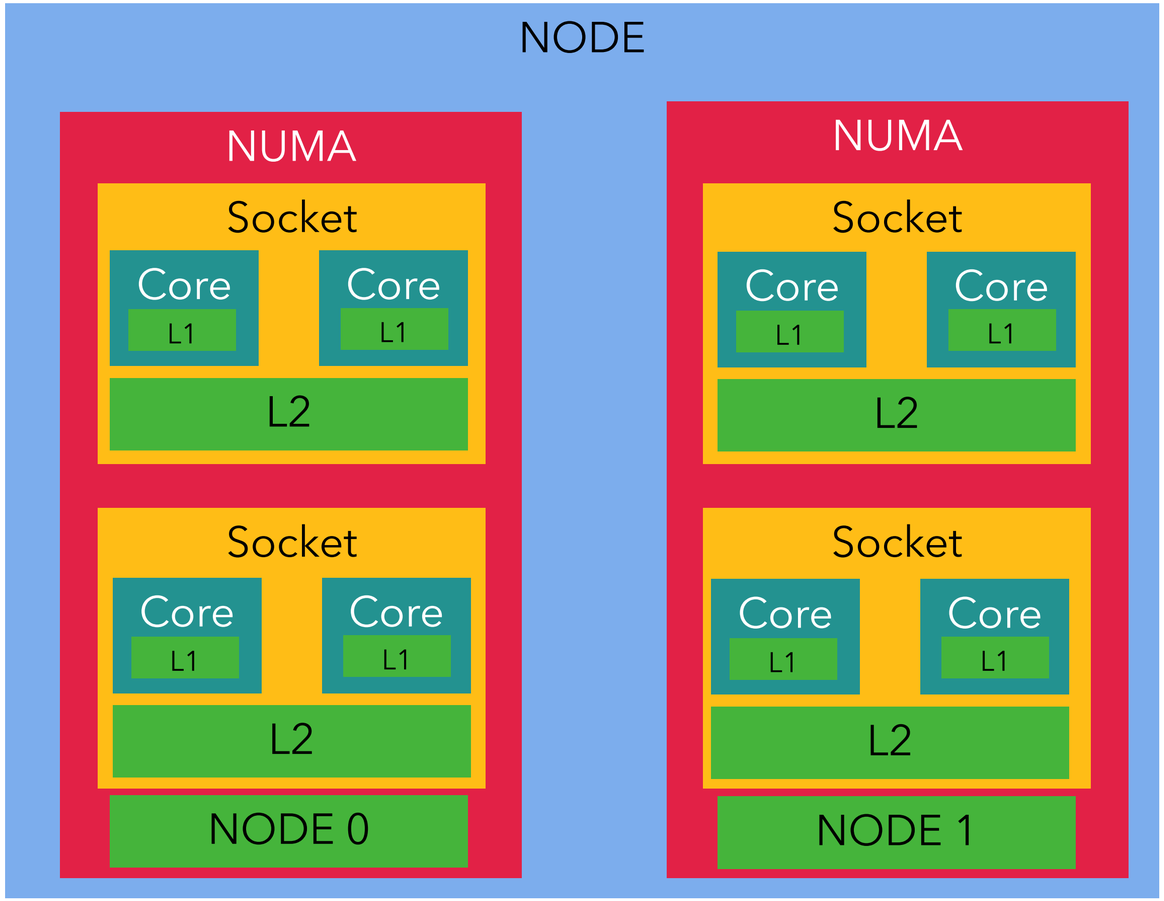


Data Mapping



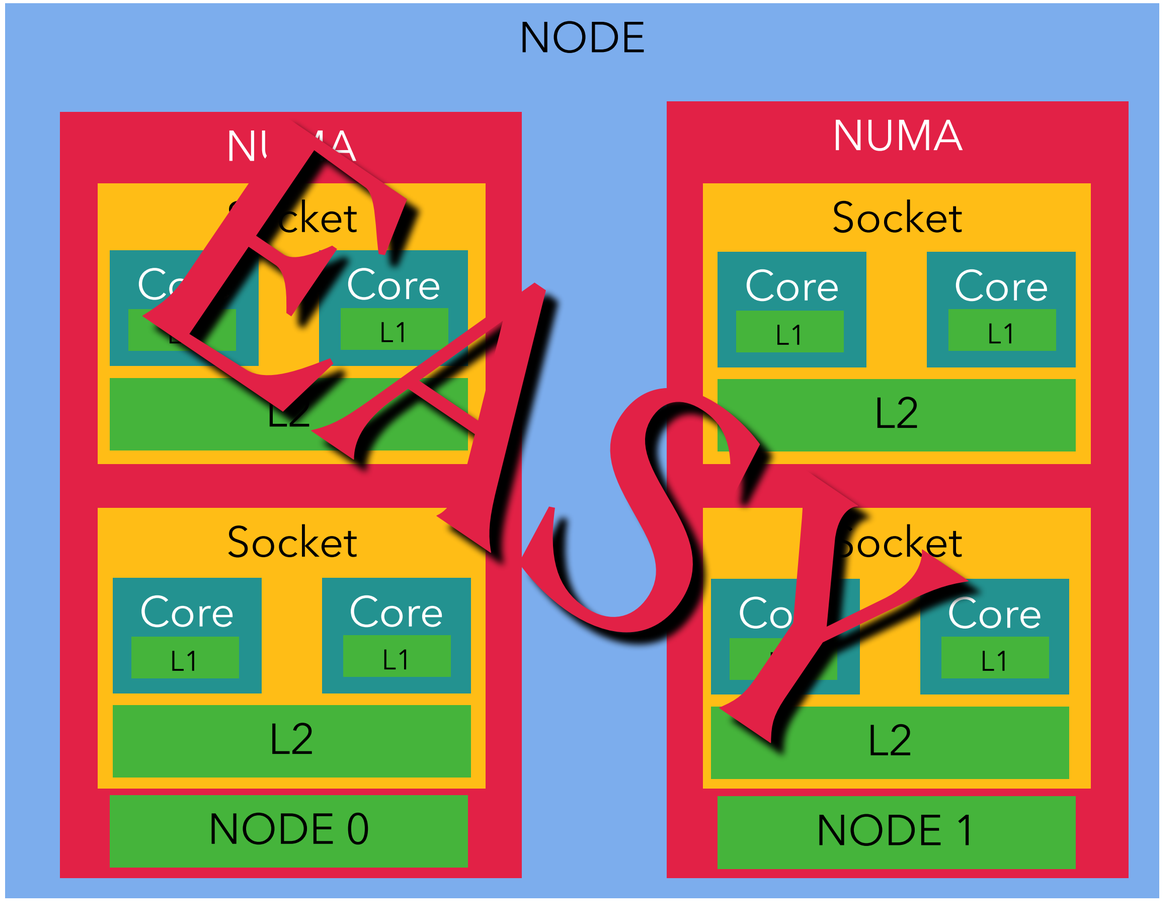
Parallel Programs 'Spatialization

- With differenciated hardware the program does not only live in **time** but also in **space**
  - Nesting programming models creates a hierarchy for locality with different indexing (MPI rank, OMP rank, Cuda indexes)
  - Data must follow these indexes as they are used to define computational split
  - Locality is increasingly a key performance aspect of current systems



Parallel Programs 'Spatialization

- With differenciated hardware the program does not only live in **time** but also in **space**
  - Nesting programming models creates a hierarchy for locality with different indexing (MPI rank, OMP rank, Cuda indexes)
  - Data must follow these indexes as they are used to define computational split
  - Locality is increasingly a key performance aspect of current systems

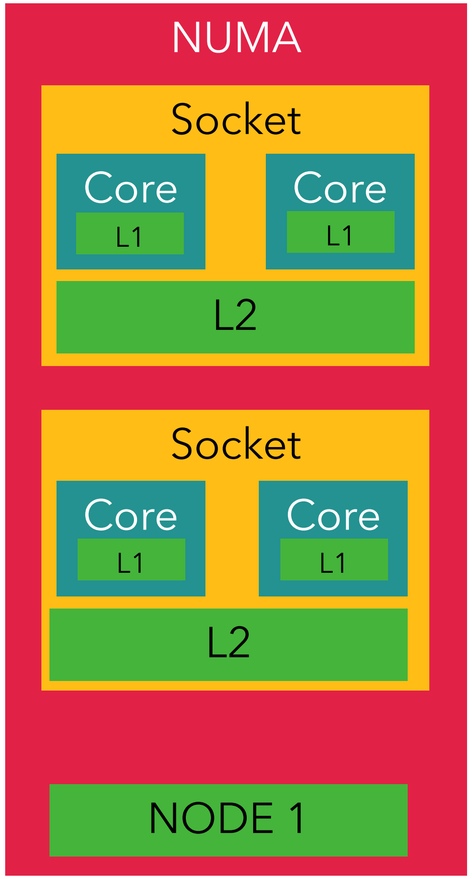


Parallel Programs 'Spatialization

Nvidia GH100

8 Compute Graphic Cluster (GPC); 72 Texture Processing Clusters (TPC); 144 Streaming Multiprocessor (SM); 18 432 Cuda Cores

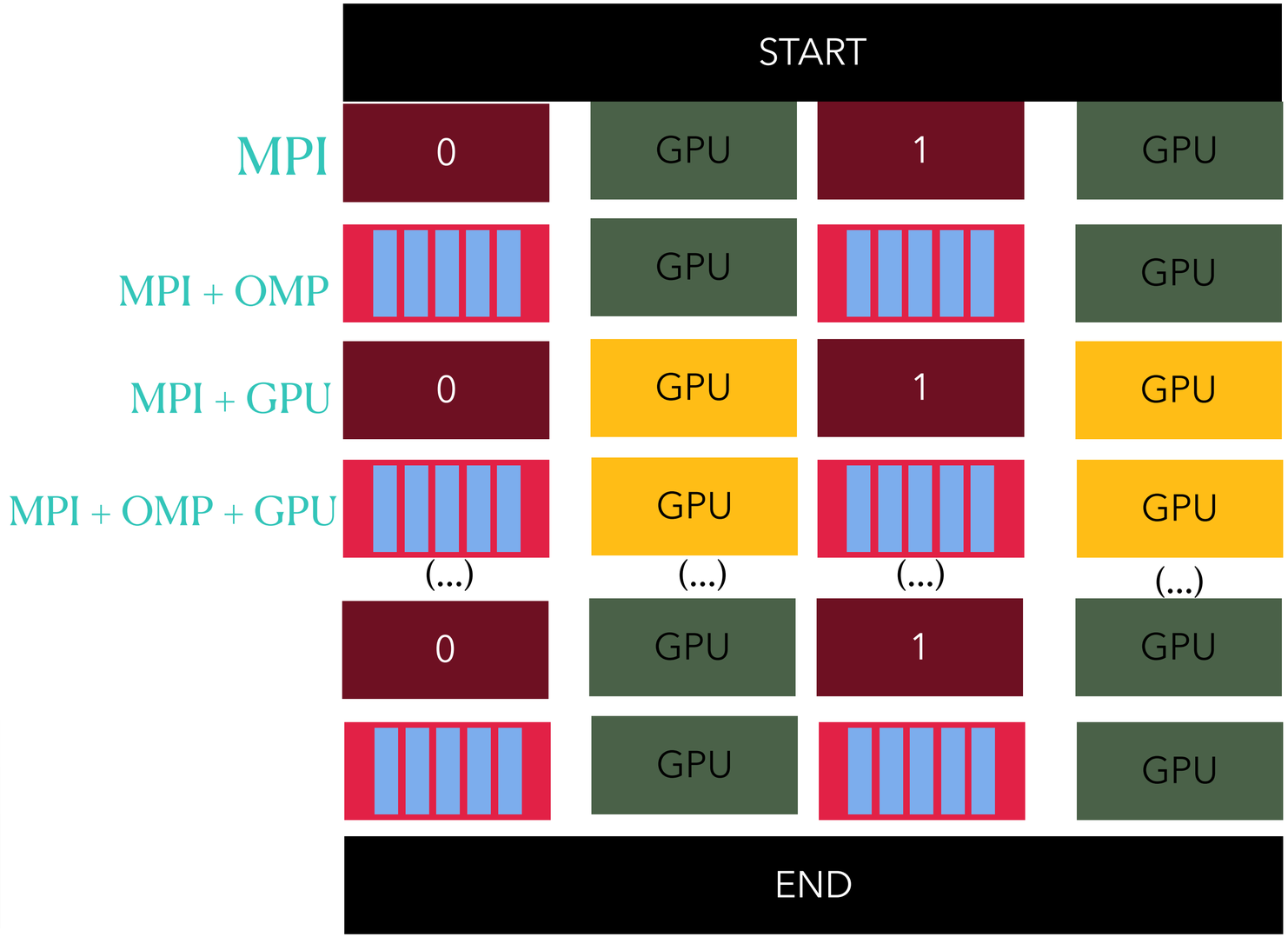
Parallel Programs 'Spatialization



Current architectures also features a regular CPU over coherent memory

Towards the Limits of Model-Nesting ?

- Node-level parallelism is becoming increasingly important
  - Not only NUMA and OpenMP
  - But also multiple memory kinds and different cores (GPUs)
- Model nesting MPI + X + Y is heading towards a complexity barrier
  - Creates fork-join overheads
  - Adds layers over layers

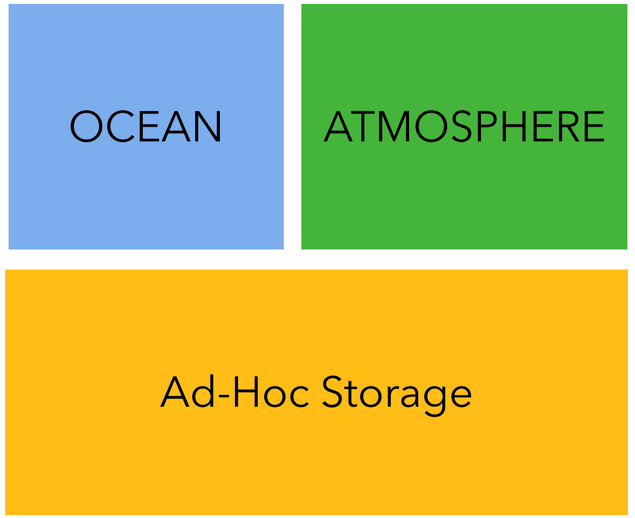


Horizontalization of Compute

EuroMPI/Australia 2024 — 09/26/2024

On Software Differenciation

- Maintaining programs in an SPMD model is increasingly difficult:
  - Hardware is not symetrical
  - Leads to nested states and programming models
- There is a need for **software differenciation**:
  - Collaborating programs require communication
  - Horizontalization of compute and not only data (including programs)

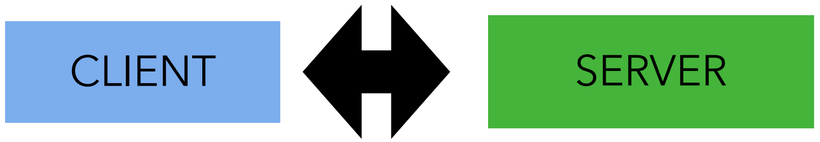


On Software Differenciation

MPI is not bulk synchronous, but ...

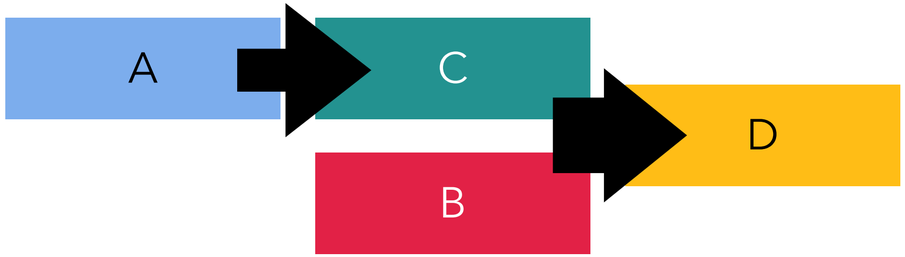
SPMD

Client-Server

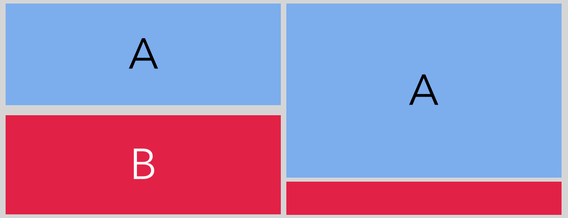


MPMD

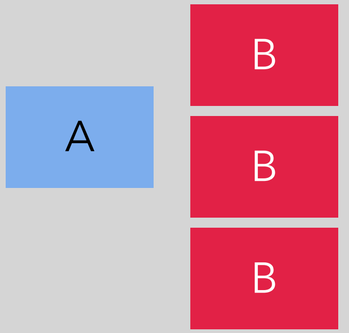
Systolic Architectures / Workflows



Malleability



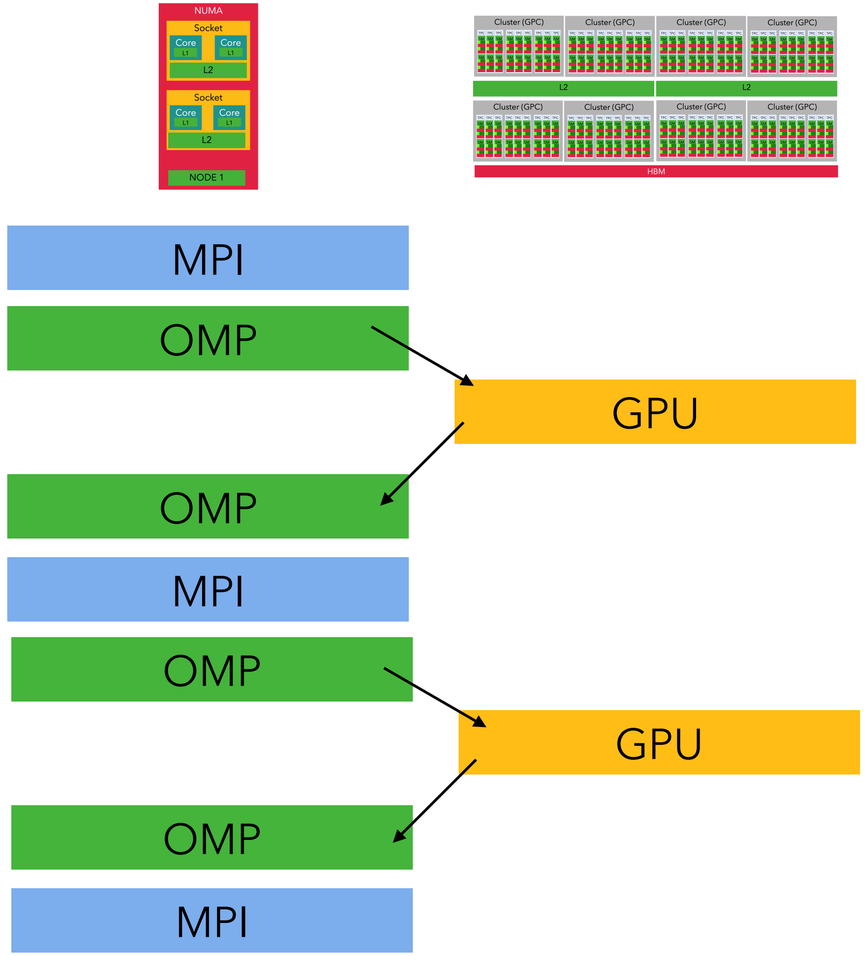
Invasive Computing



On Software Differenciation

CURRENT NESTING

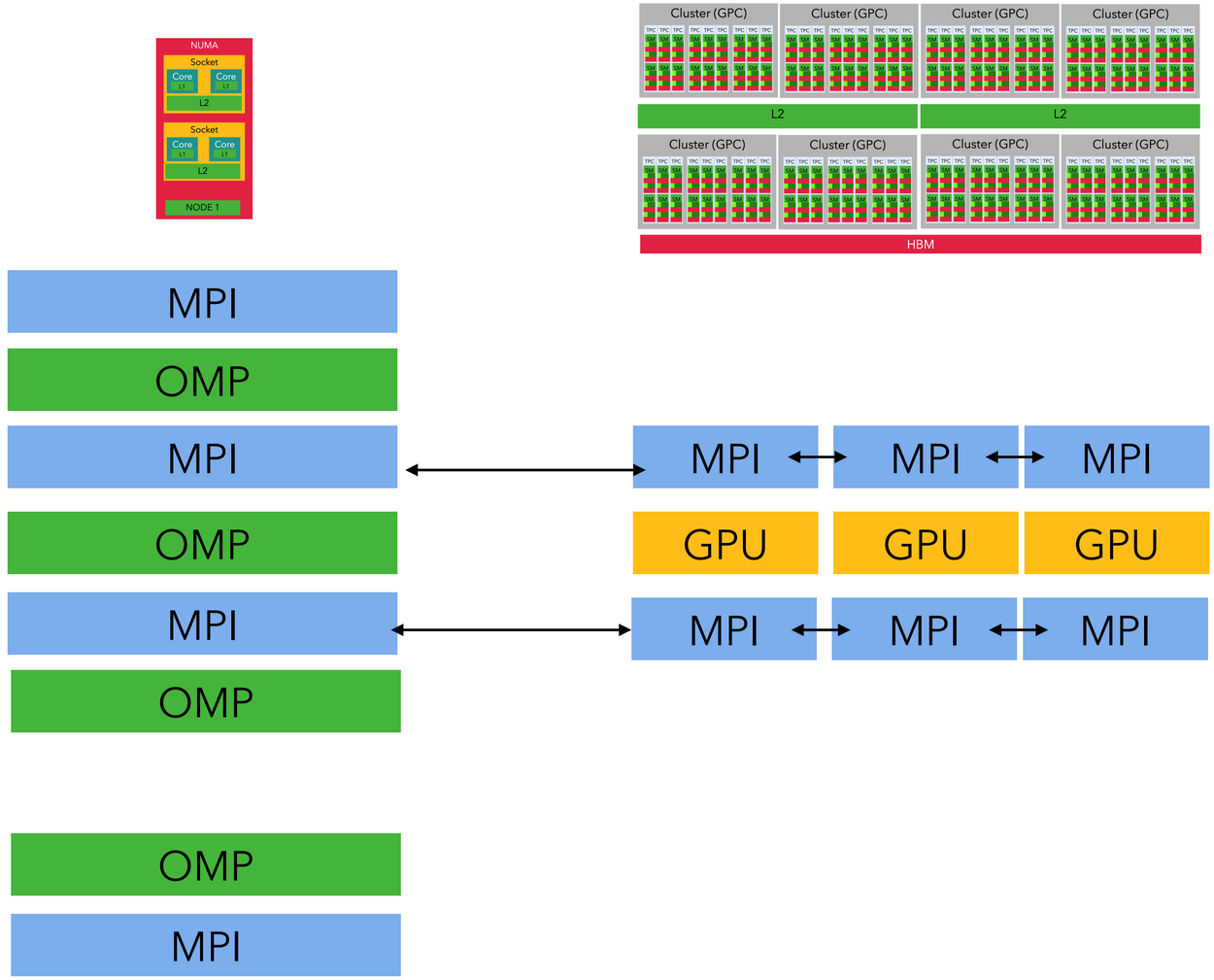
- Alternating behavior
- Difficult to light all the silicium at once
- Some of the **mapping** is outside of MPI's view (CUDA memcpy, CUDA kernels)



On Software Differenciation

DIFFERENCIATION

- Functions / Kernels are spatially bound to a given MPI rank
- Already practical with CCL
- Single indexing scheme (MPI rank)
- MPI is the coordinator (more than doing data-movements)



Current Support of MPI in SHMEM

EuroMPI/Australia 2024 — 09/26/2024

Shared-Memory Support in MPI

- Evolutions of the standard:
  - Shared-memory windows
  - Finepoints -> Partitioned Communications
  - Endpoints
  - Support for RPCs (active messages)
  - Continuations
  - Ownership passing
- Patterns of interest:
  - RPCs (for kernel calls and work dispatch)
  - Shared-memory queues (mostly in language at this point)

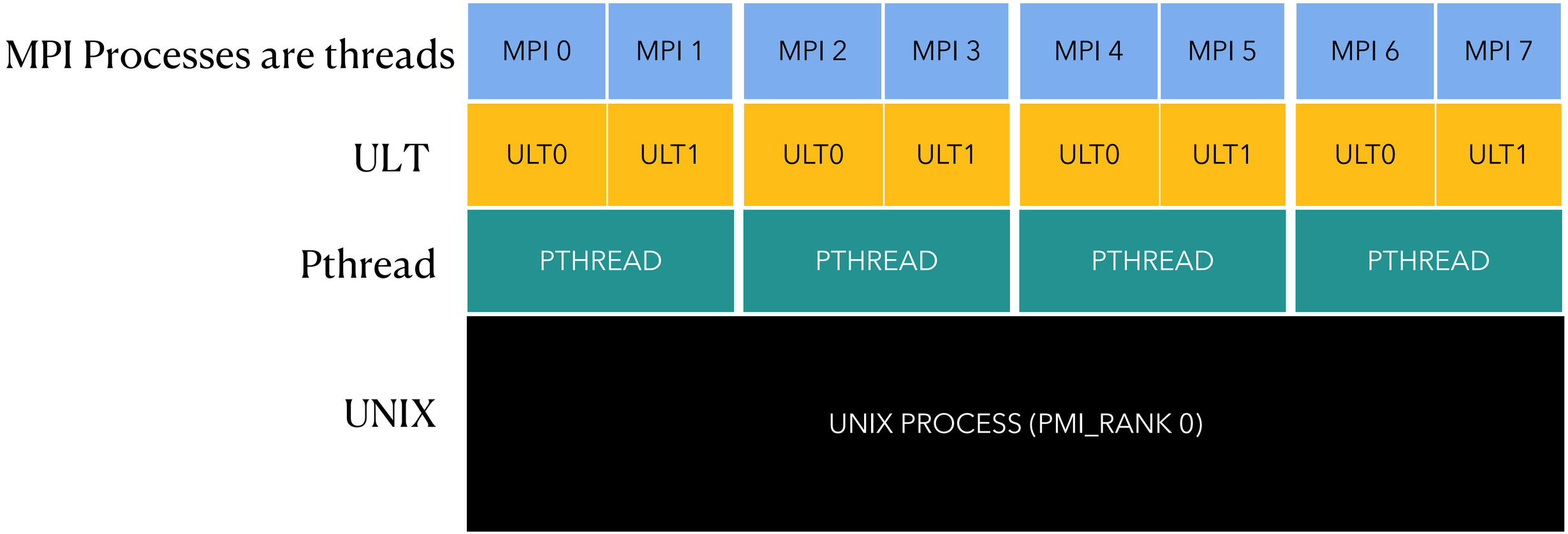
Shared-Memory Support in Runtimes

- User-Level:
  - HMPI
  - AMPI
  - Process-In-Process (PiP)
  - Multi-Processor Computing (MPC)
- Kernel level:
  - SMARTMAP
  - PVAS

Shared-Memory Support in Runtimes

- User-Level:
  - HMPI
  - AMPI
  - Process-In-Process (PiP)
  - **Multi-Processor Computing (MPC)**
- Kernel level:
  - SMARTMAP
  - PVAS

Multi-Processor Computing



MPC and Existing Codes

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

int rank = -1;

int main( int argc, char ** argv )
{
    MPI_Init( &argc, &argv );

    MPI_Comm_rank( MPI_COMM_WORLD, &rank );

     printf("My rank %d", rank );

    MPI_Finalize();

    return 0;
}
```

$ mpcrun -p=4 -n=4 ./a.out

```
My rank 2
My rank 3
My rank 0
My rank 1
```

It's working with n == p ⟶ Process-based MPI

$ mpcrun -p=1 -n=4 ./a.out

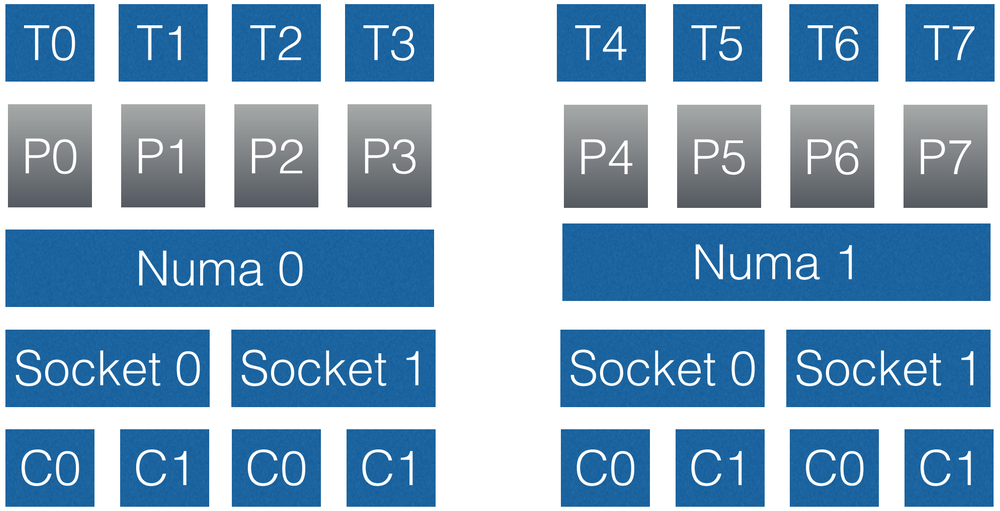
```
My rank 1
My rank 1
My rank 1
My rank 1
```

Overwritten global variable ⟶ Thread-based MPI

Need for a privatisation mechanism extending TLS[1]

[1] Jean-Baptiste Besnard, Julien Adam, Sameer Shende, Marc Pérache, Patrick Carribault, Julien Jaeger, and Allen D. Maloney. 2016. Introducing Task-Containers as an Alternative to Runtime-Stacking. In Proceedings of the 23rd European MPI Users' Group Meeting (EuroMPI '16). Association for Computing Machinery, New York, NY, USA, 51–63. https://doi.org/10.1145/2966884.2966910

MPC Launch Configuration



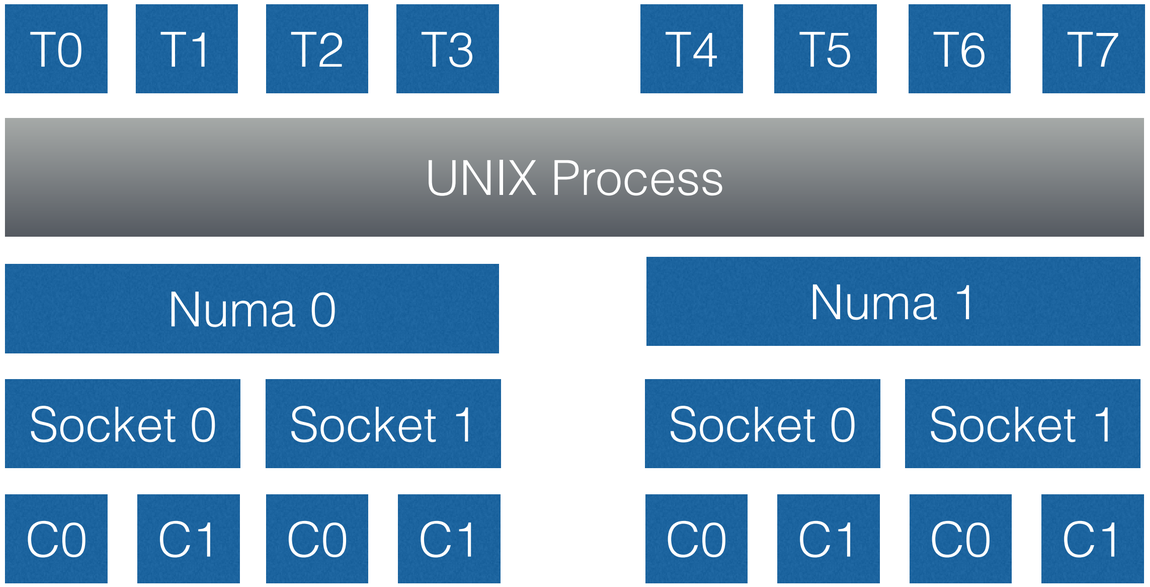
Process-Based MPI

mpcrun -N=1 -p=8 -n=8 ./a.out

Launch is described with four parameters (replacing -np):

- N: Number of nodes
- p: Number of system processes
- n: Number of MPI tasks
- c: Number of cores per process

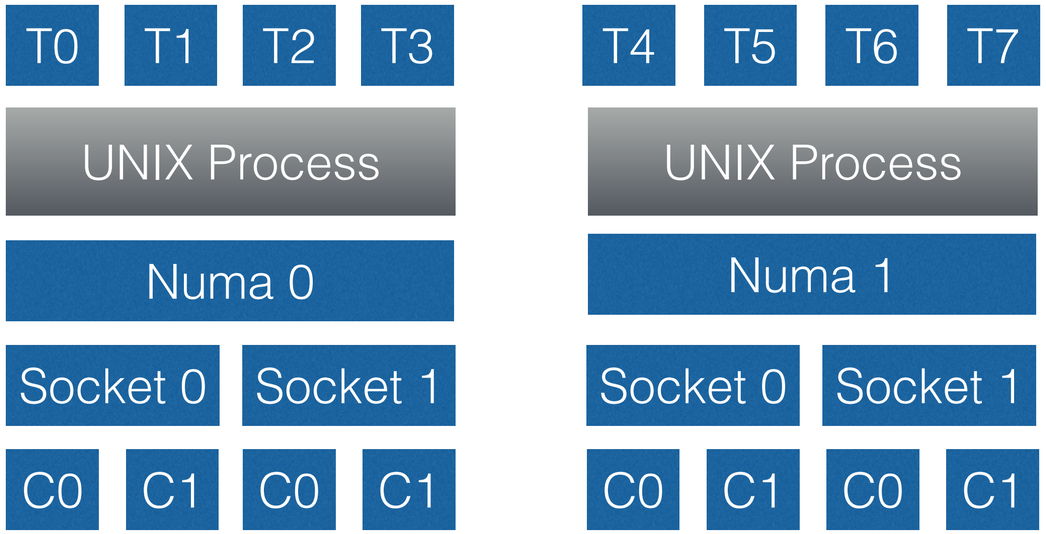
MPC Launch Configuration



Thread-Based MPI

mpcrun -N=1 -p=1 -n=8 ./a.out

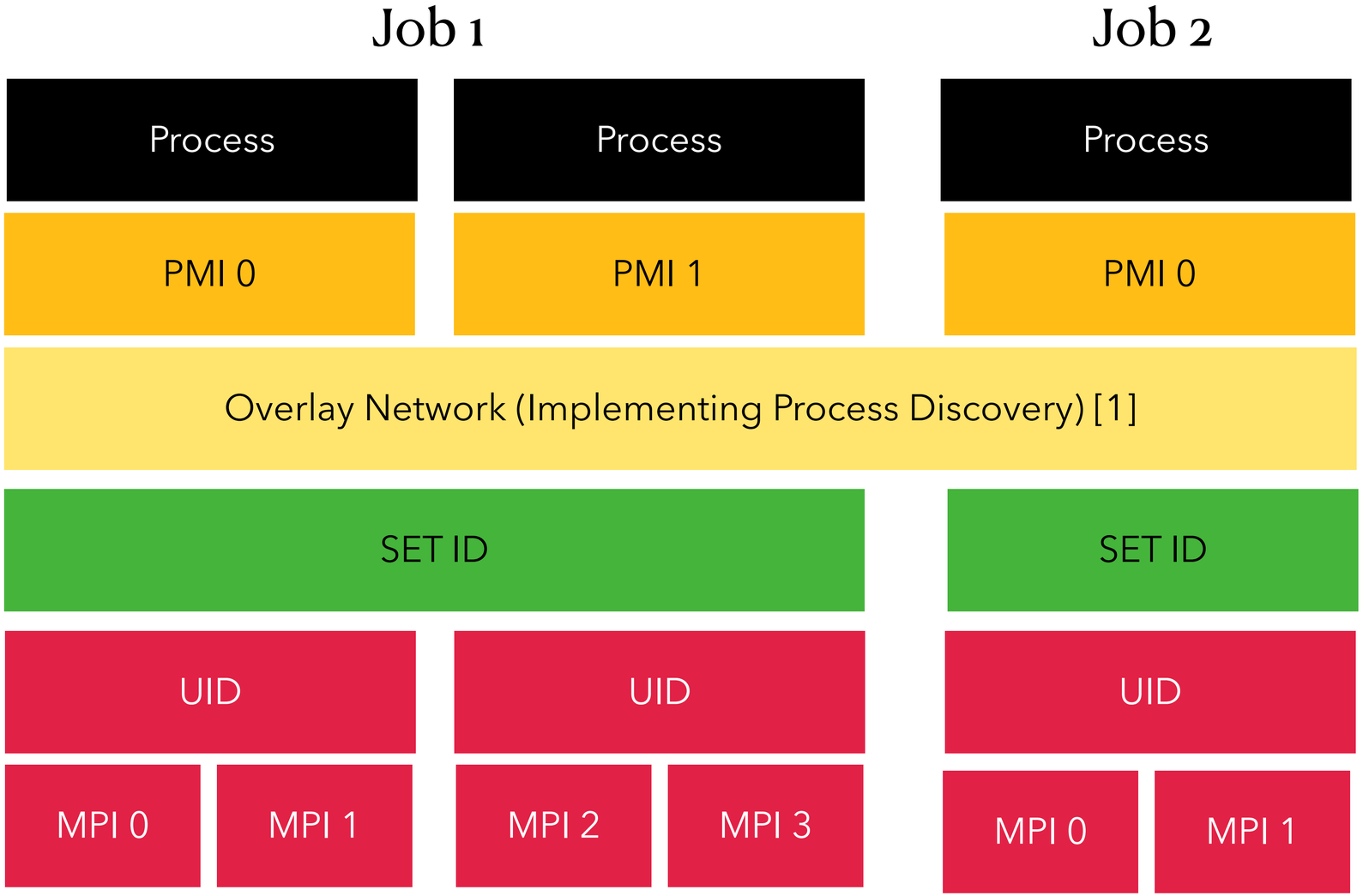
MPC Launch Configuration

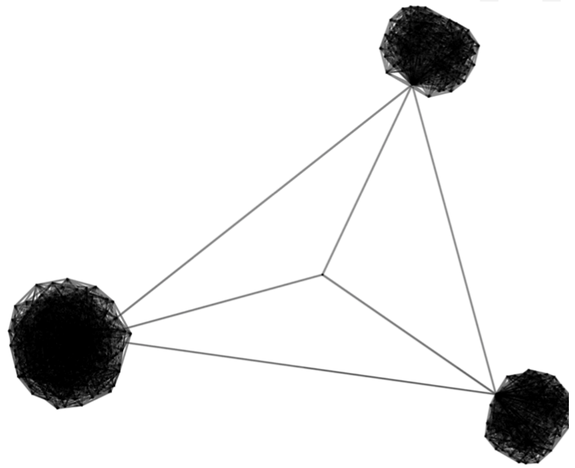


Mixed Approach

mpcrun -N=1 -p=2 -n=8 -c=4 ./a.out

MPC Launch





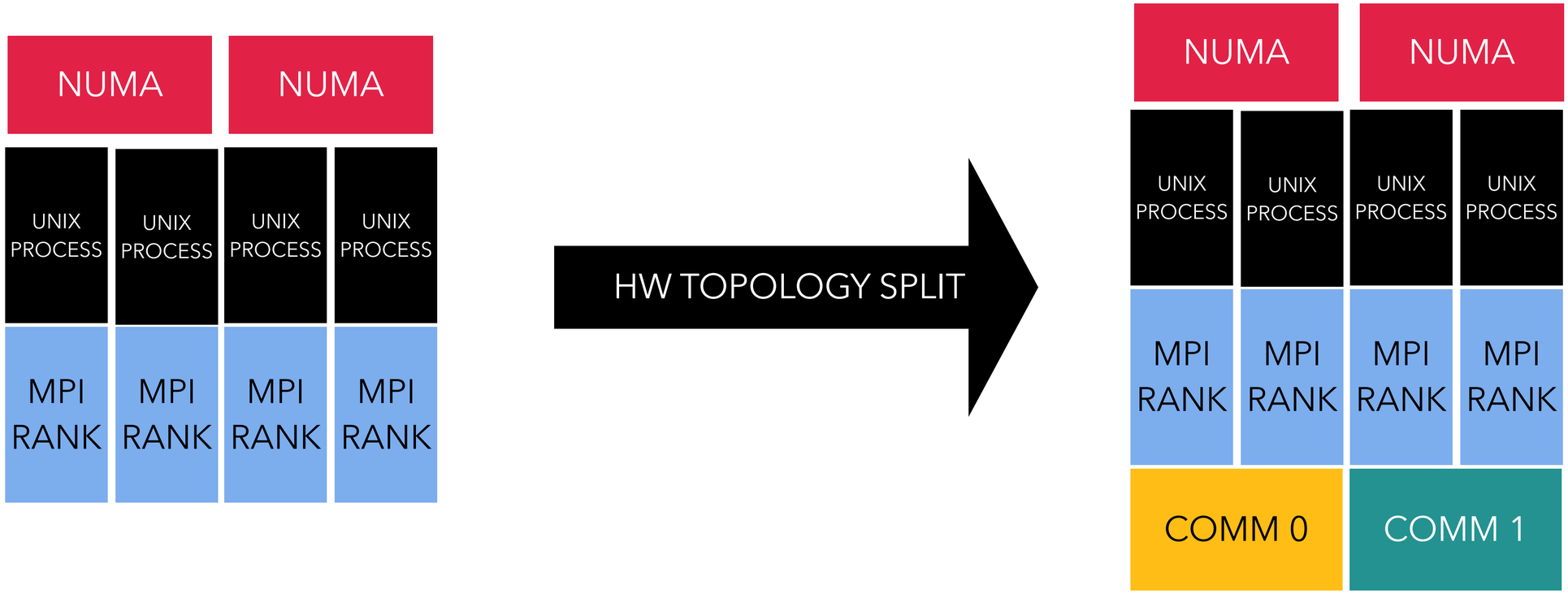
[1] Jean-Baptiste Besnard, Sameer Shende, Allen Malony, Julien Jaeger, and Marc Perache. 2022. Enabling Global MPI Process Addressing in MPI Applications. In Proceedings of the 29th European MPI Users' Group Meeting (EuroMPI/USA '22). Association for Computing Machinery, New York, NY, USA, 27–36. https://doi.org/10.1145/3555819.3555829

Sessions at the Rescue

EuroMPI/Australia 2024 — 09/26/2024

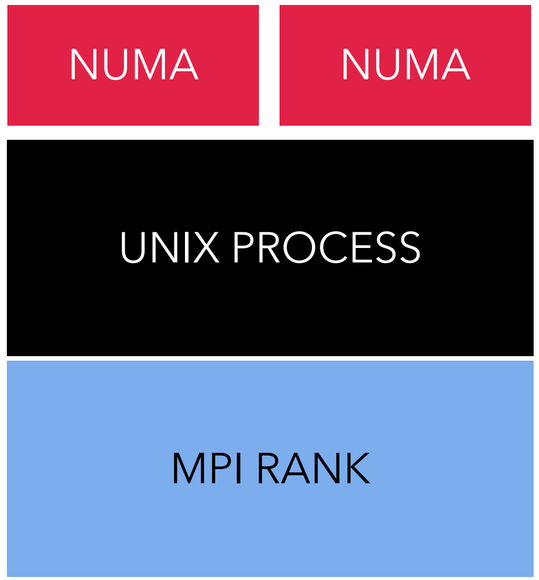
Topological Split on MPI Processes

MPI's role is to **MAP** and **ADDRESS** the distributed MPI processes



Topological Split Inside Processes

WORLD MODEL

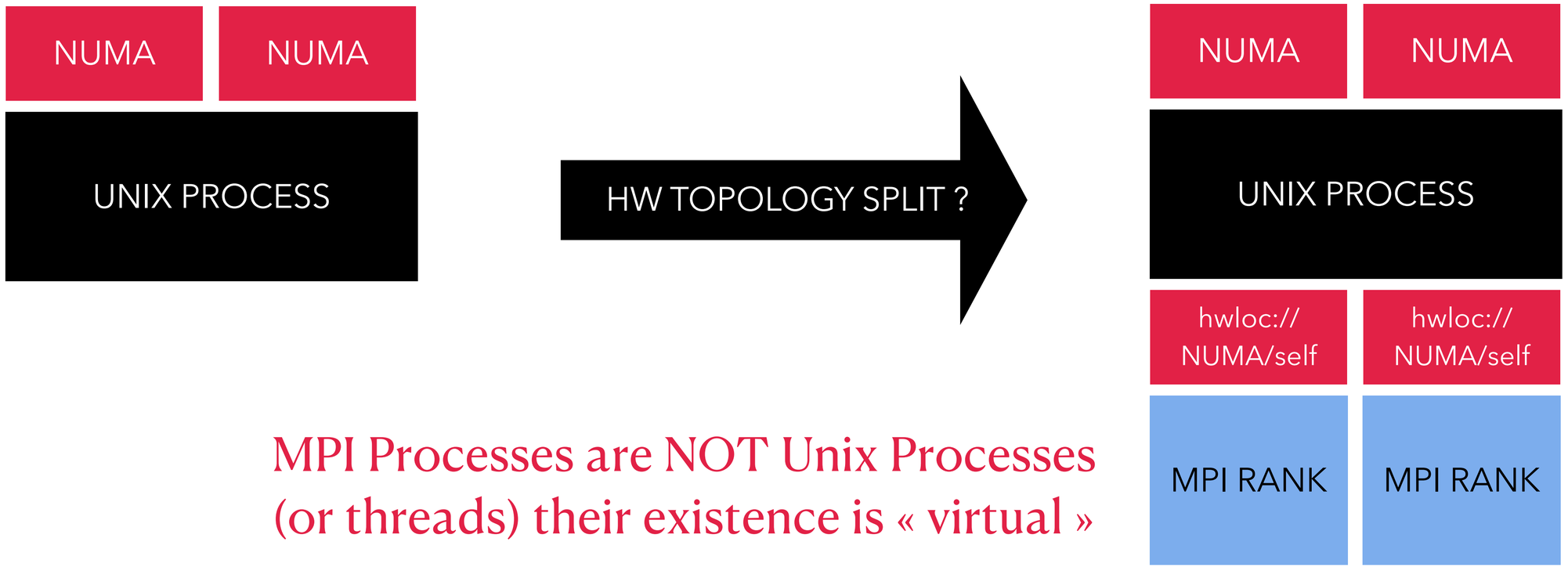


?

Rank is bound to proc

Topological Split Inside Processes

SESSIONS MODEL



MPI Processes are NOT Unix Processes
(or threads) their existence is « virtual »

Context inherited by handle nesting

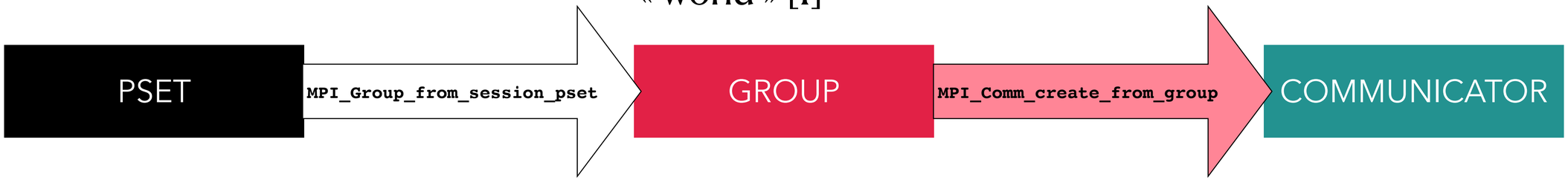
Topological Split Inside Processes

SESSIONS MODEL

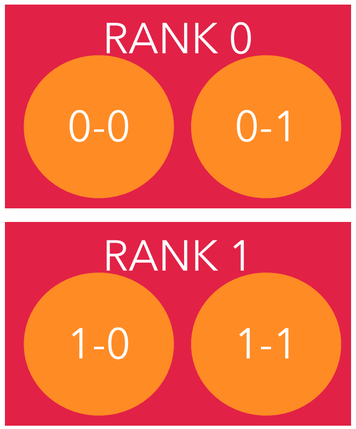
Groups with processes that are initially not from « world » [1]

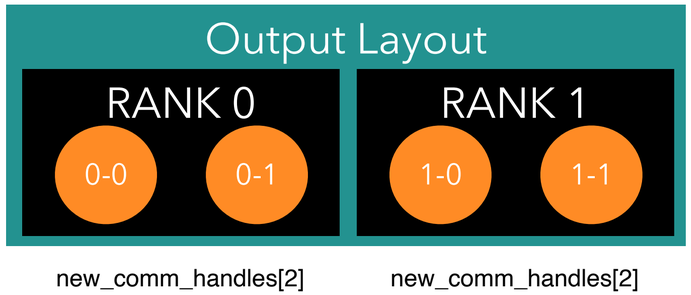
Instead, communicators created in a non-collective manner (not bound to executors) == **Endpoints**

```
int MPI_Comm_endpoints_from_group(MPI_Group group,
        MPI_Info info, MPI_Comm *new_comm_handles,
        int max_len, char *tag)
```



endpoints://world/2





new_comm_handles[2] new_comm_handles[2]

[1] Jean-Baptiste Besnard, Sameer Shende, Allen Malony, Julien Jaeger, and Marc Perache. 2022. Enabling Global MPI Process Addressing in MPI Applications. In Proceedings of the 29th European MPI Users' Group Meeting (EuroMPI/USA '22). Association for Computing Machinery, New York, NY, USA, 27–36. https://doi.org/10.1145/3555819.3555829

To Share or Not to Share: a case for MPI in Shared-Memory

Julien Adam[1], Jean-Baptiste Besnard[1], Adrien Roussel[2,3], Julien Jaeger[2,3], Patrick Carribault[2,3], Marc Pérache[2,3]

1. ParaTools SAS, Bruyères-le-Châtel, France
2. CEA, DAM, DIF, F91297 Arpajon, France
3. Université Paris-Saclay, CEA, Laboratoire en Informatique Haute Performance pour le Calcul et la simulation
91680 Bruyères-le-Châtel, France