Vers des mécanismes génériques de communication et une meilleure maîtrise des affinités dans les grappes de calculateurs hiérarchiques

Brice Goglin

15 avril 2014

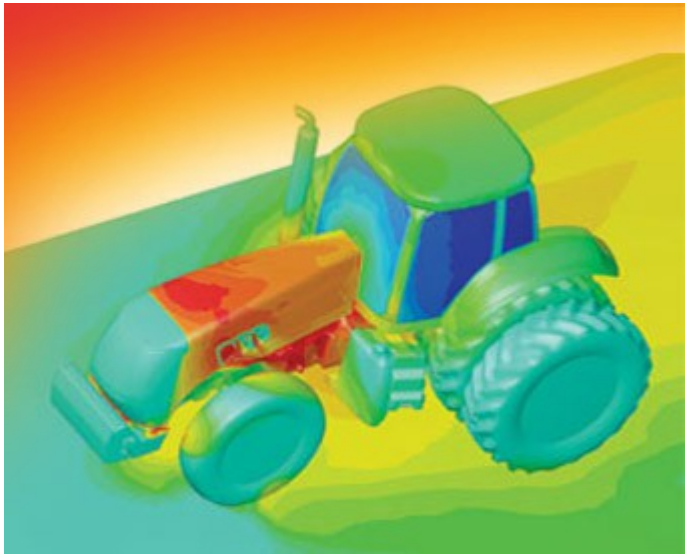
Towards generic Communication Mechanisms and better Affinity Management in Clusters of Hierarchical Nodes

Brice Goglin

April 15th, 2014

Scientific simulation is everywhere

- Used by many industries
  - Faster than real experiments
  - Cheaper
  - More flexible



- Today's society cannot live without it

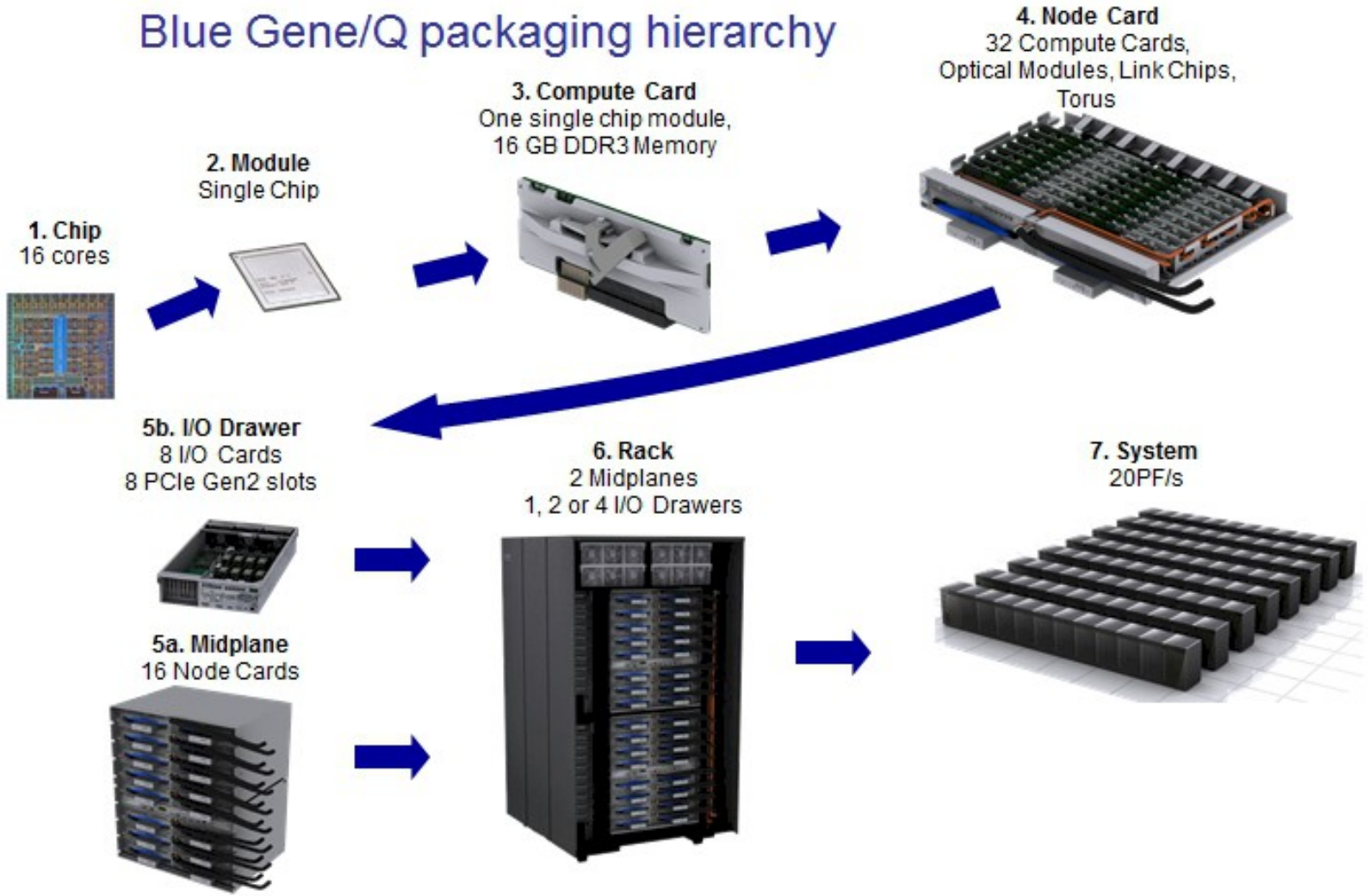
- Used by many non computer scientists

Growing computing needs

- Growing platform performance
  - Multiprocessors
  - Clusters of nodes
  - Higher frequency
  - Multicore processors

- High Performance Computing combines all of them
  - Only computer scientists can understand the details
  - But everybody must parallelize his codes

Hierarchy of computing resources



Increasing hardware complexity

- Vendors cannot keep the hardware simple
  - Multicore instead of higher frequencies
    - You have to learn parallelism
  - Hierarchical memory organization
    - Non uniform memory access (NUMA) and multiple caches
      - Your performance may vary
  - Complex network interconnection
    - Hierarchical
    - Very different hardware features

Background

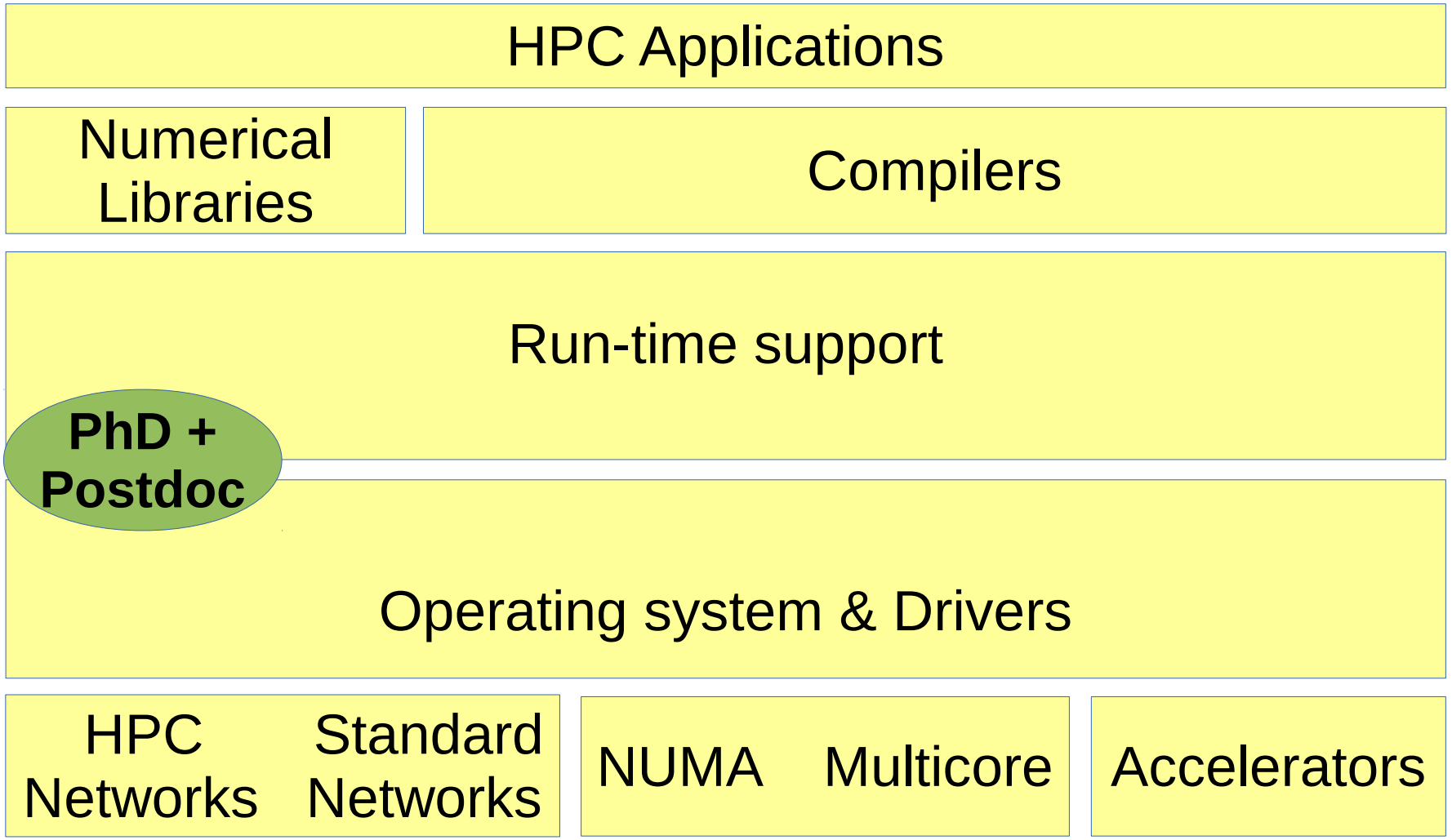
- 2002-2005: PhD
  - Interaction between HPC networks and storage
    - Towards a generic networking API
    - Still no portable API?
- 2005-2006: Post-doc

  - On the influence of vendors on HPC ecosystems
    - Benchmarks, hidden features, etc.
  - Multicore and NUMA spreading
    - Clusters and large SMP worlds merging

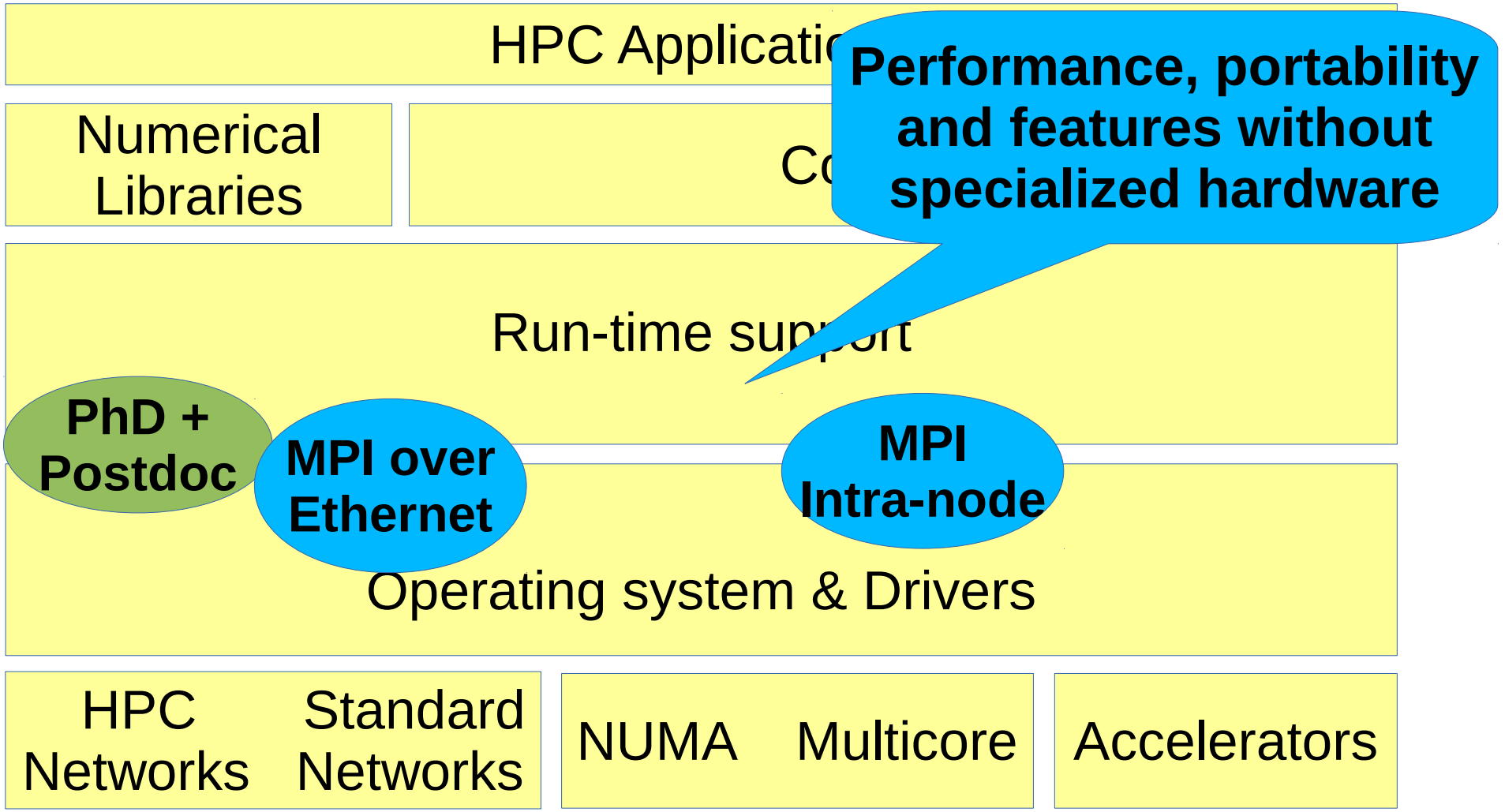
Since 2006

- Joined Inria Bordeaux and LaBRI in 2006
- Optimizing low-level HPC layers
  - Interaction with OS and drivers

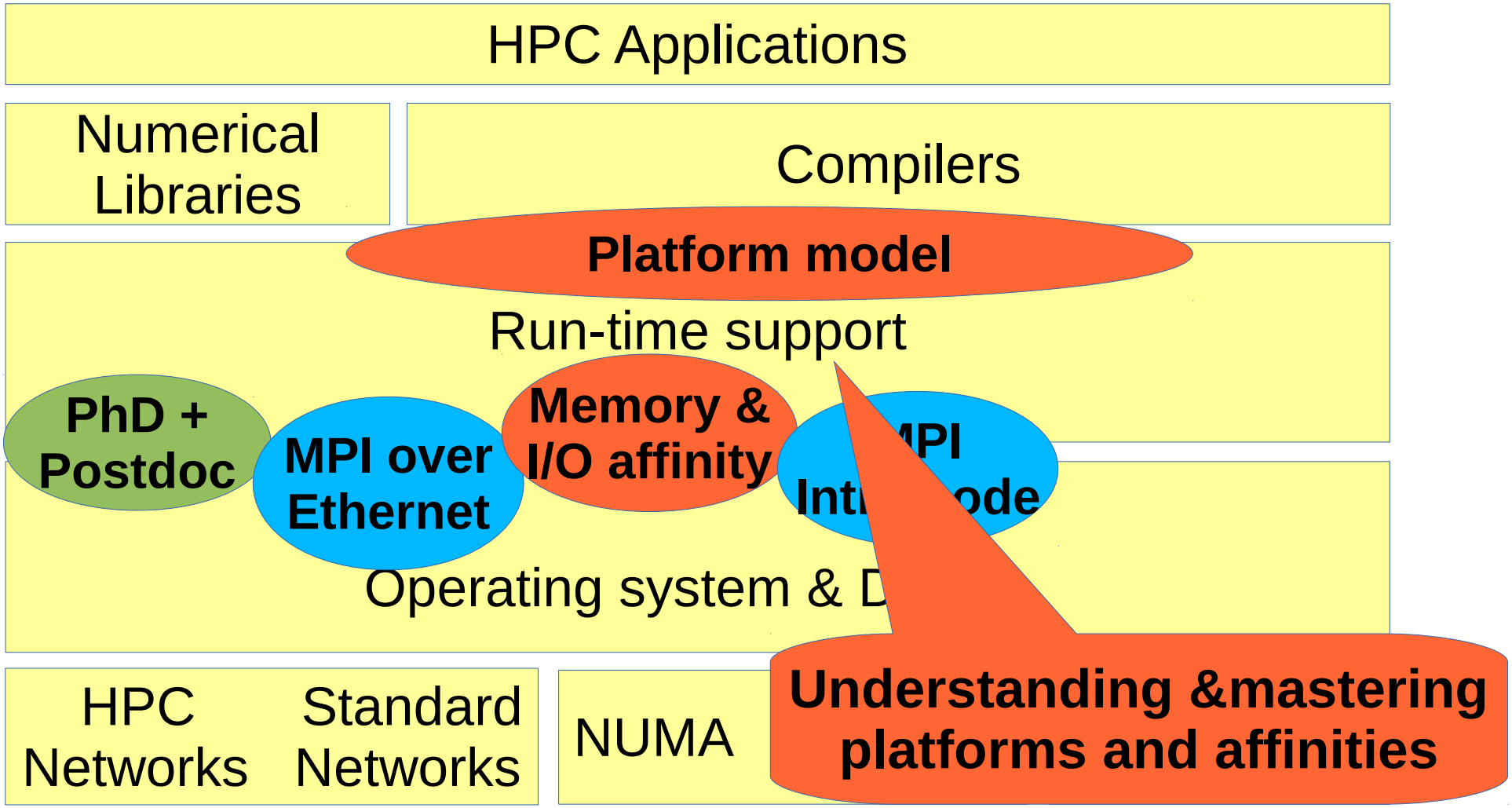
HPC stack


HPC Applications
Numerical Libraries
Compilers
Run-time support
PhD + Postdoc
Operating system & Drivers
HPC Networks
Standard Networks
NUMA
Multicore
Accelerators

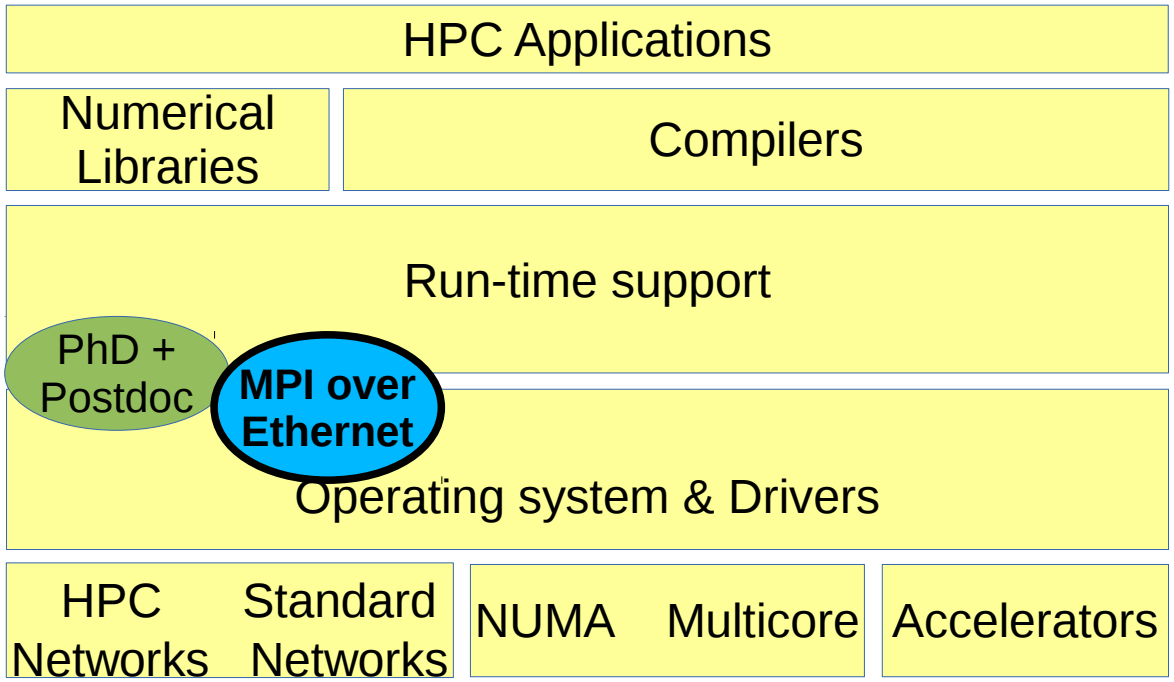
A) Bringing HPC network innovations to the masses



B) Better management of hierarchical cluster nodes



A.1) Bringing HPC network innovations to the masses:
High performance MPI over Ethernet



MPI is everywhere

- De facto standard for communicating between nodes
  - And even often inside nodes
- 20-year-old standard
  - Nothing ready to replace it
  - Real codes will not leave the MPI world unless a stable and proven standard emerges
- MPI is not perfect
  - API needs enhancements
  - Implementations need a lot of optimizations

Two worlds for networking in HPC

Technology	**Specialized (InfiniBand, MX)**	**Standard (TCP/IP, Ethernet)**
Hardware	Expensive, specialized	Any
Performance	Low latency, high throughput	High latency?
Designed for	RDMA, messages	Flows
Data transfer	Zero-copy	Additional copies
Notification	Write in user-space, or interrupt	Interrupt in the kernel

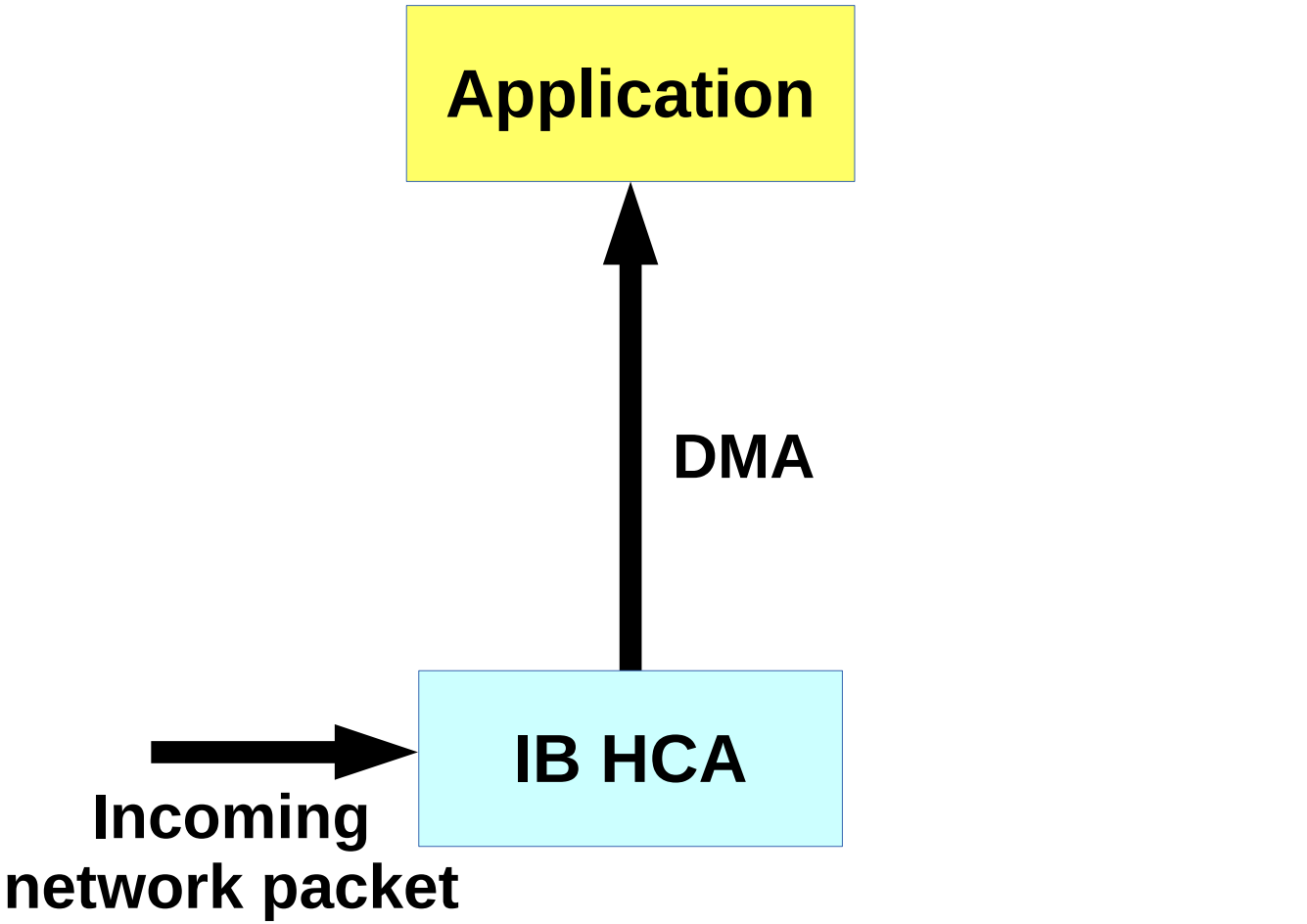
Existing alternatives

- Gamma, Multiedge, EMP, etc.
  - Deployment issues
    - Require modified drivers and/or NIC firmwares
      - Only compatible with a few platforms
    - Break IP stack
      - No more administration network?
  - Use custom MPI implementations
    - Less stable, not feature complete, etc.

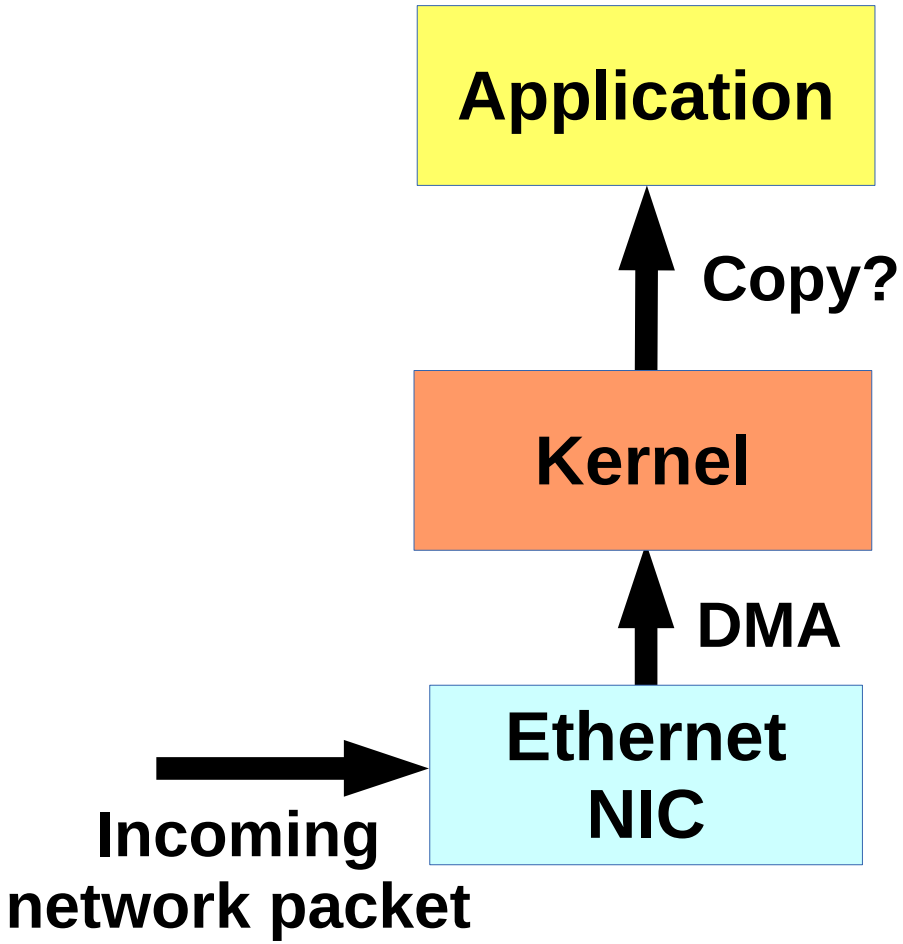
High Performance MPI over Ethernet, really?

- Take the best of both worlds
  - Better Ethernet performance by avoiding TCP/IP
  - Easy to deploy and easy to use
- Open-MX software
  - Portable implementation of Myricom's specialized networking stack (MX)
- Joint work with N. Furmento, L. Stordeur, R. Perier,

MPI over Ethernet Issue #1: Memory Copies

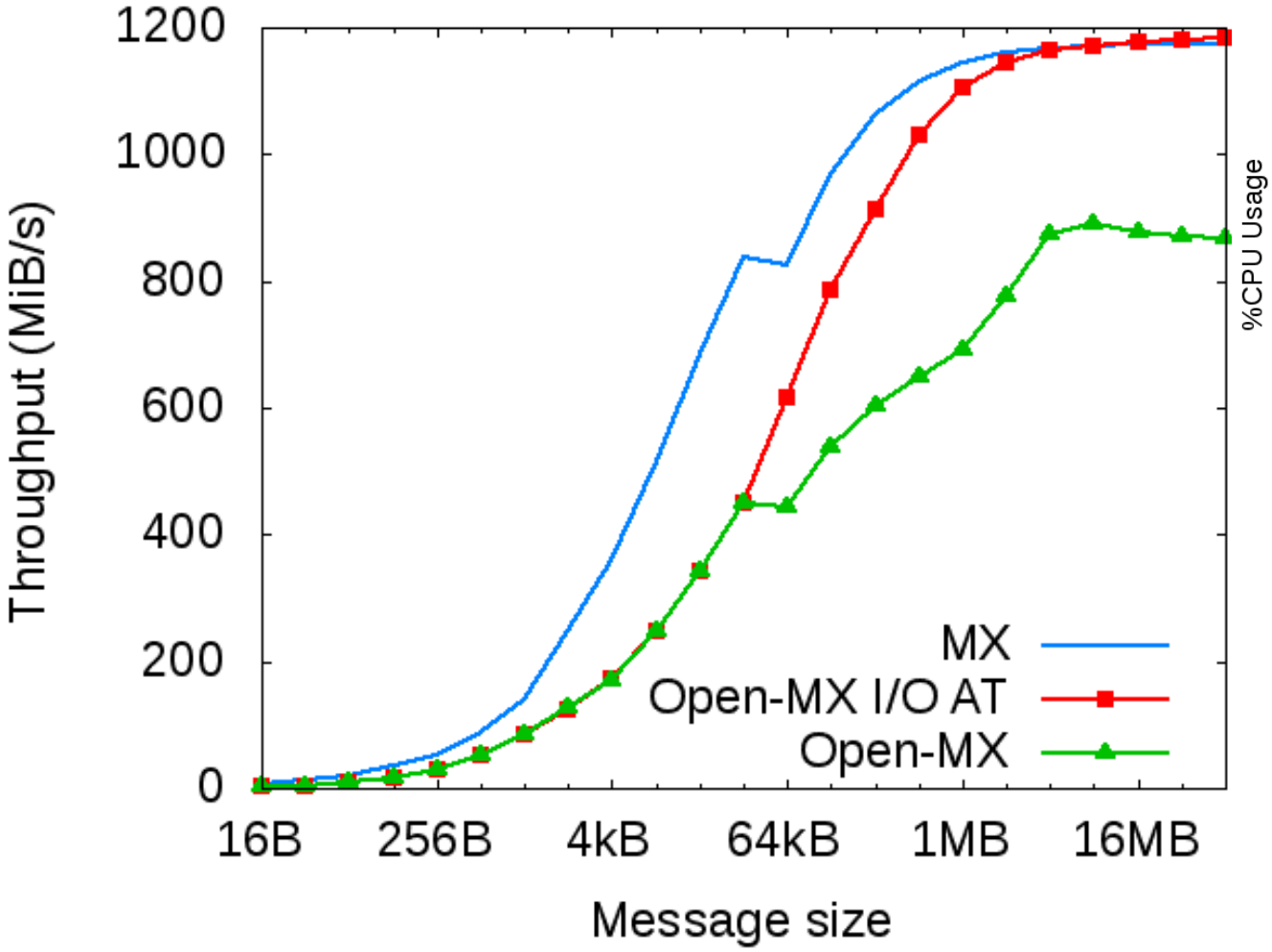


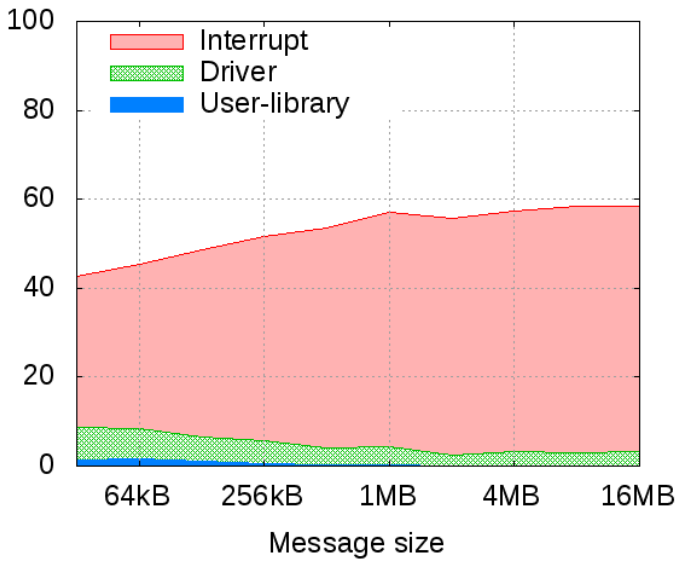
MPI over Ethernet Issue #1: Memory Copies



- Copy is expensive
  - Lower throughput
- Virtual remapping?
  - *[Passas, 2009]*
  - Remapping isn't cheap
  - Alignment constraints
- ➔ I/OAT Copy Offload
  - on Intel since 2006

MPI over Ethernet Issue #1: IMB Pingpong

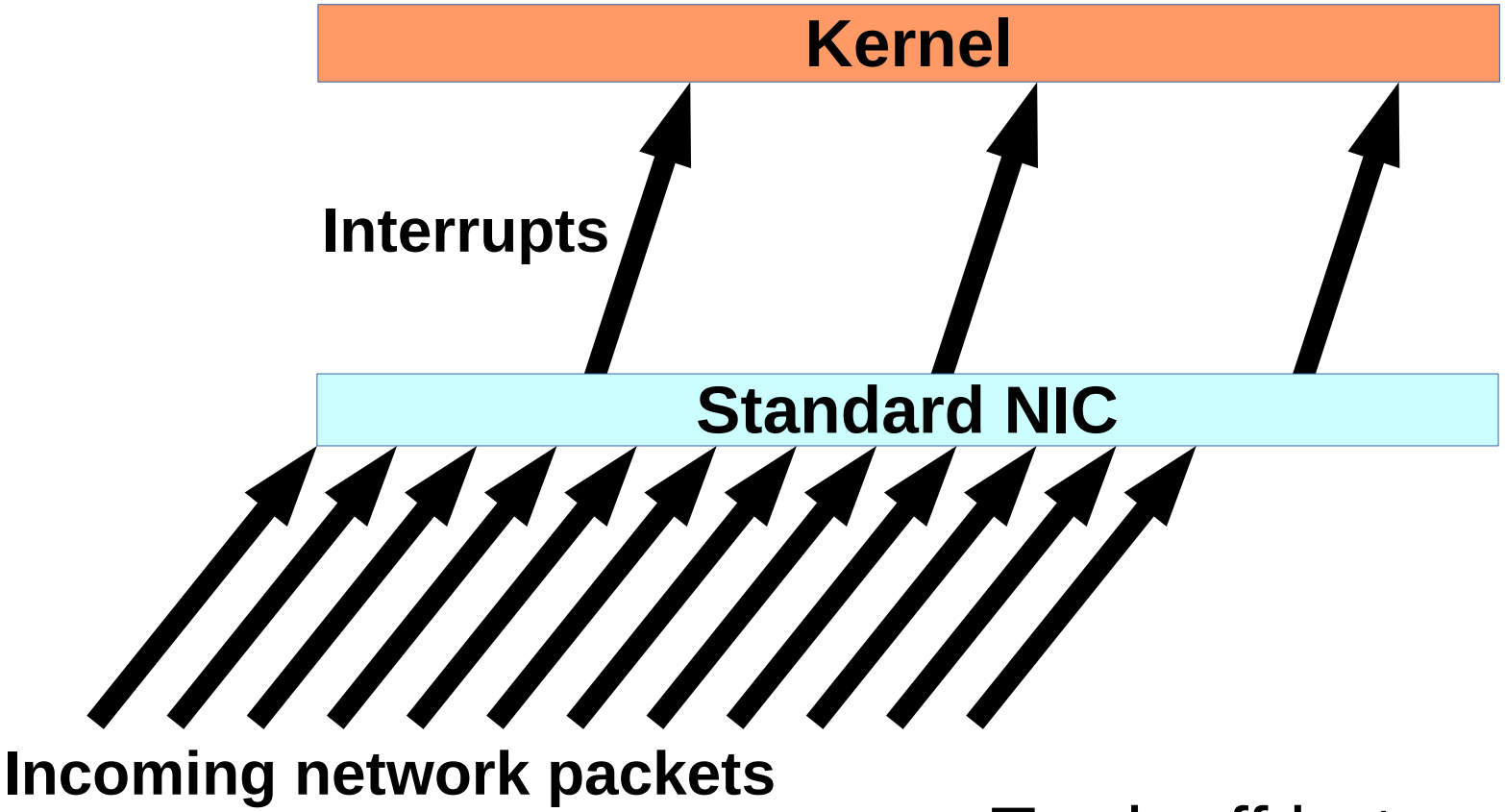




+30% on average for other IMB tests

[Cluster 2008]

MPI over Ethernet Issue #2: Interrupt Latency

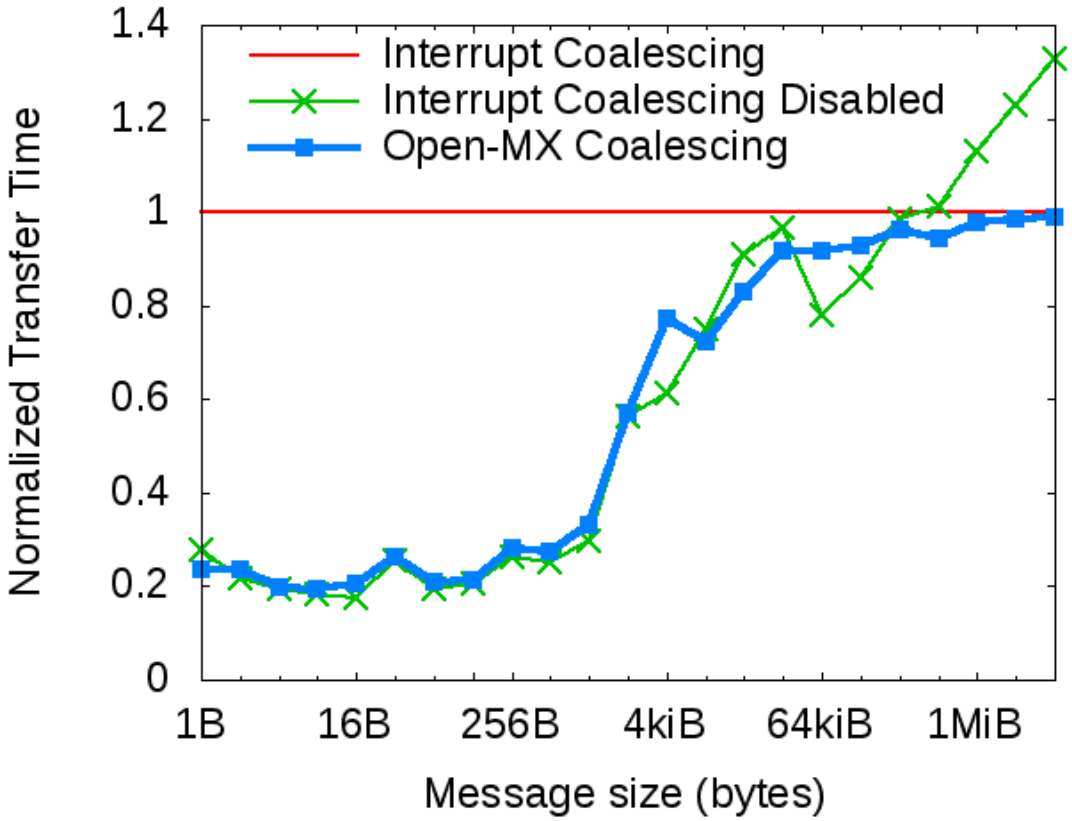


- Tradeoff between reactivity and CPU usage

MPI over Ethernet Issue #2: Interrupt Latency

- Adapt interrupts to the message structure
  - Small messages
    - Immediate interrupt
      - ➔ Reactivity
  - Large messages
    - Coalescing
      - ➔ Small CPU usage

[Cluster 2009]



MPI over Ethernet, summary

- TCP/IP Ethernet features adapted to MPI
  - Interrupt coalescing (and multiqueue filtering)
- Success thanks to widespread API
  - Open-MX works with all MPI implementations

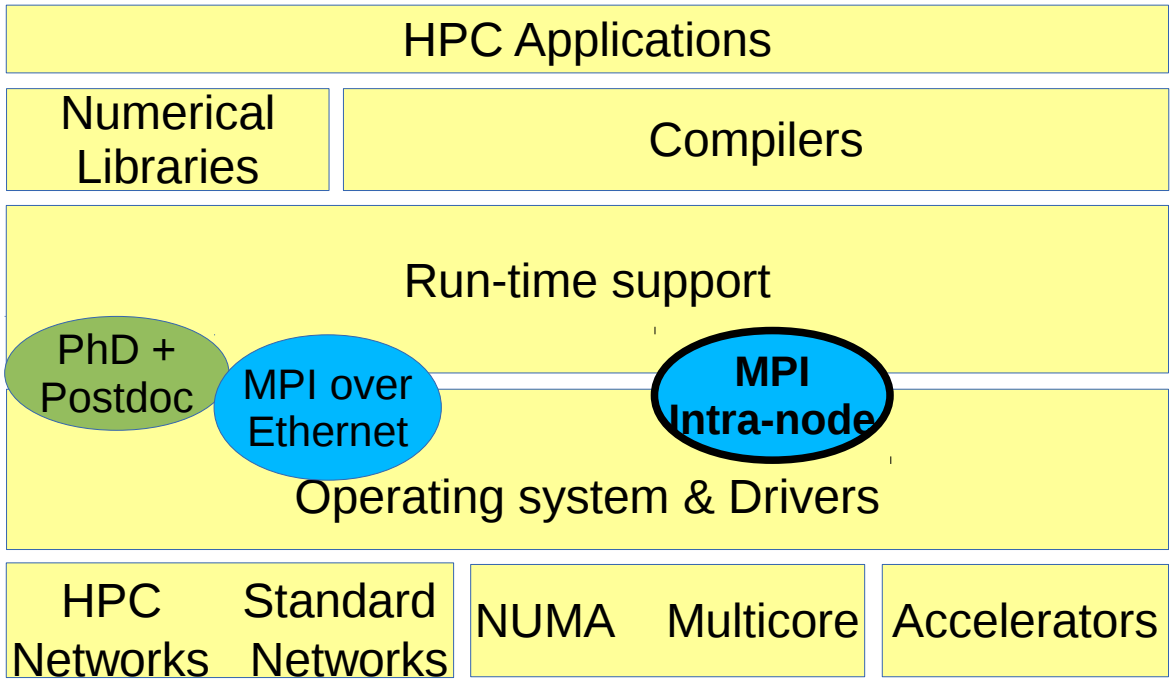
[ParCo 2011]

- But MX is going away
  - Still waiting for a generic HPC network API?

OAK RIDGE National Laboratory

A.2) Bringing HPC network innovations to the masses:

Intra-node MPI communication

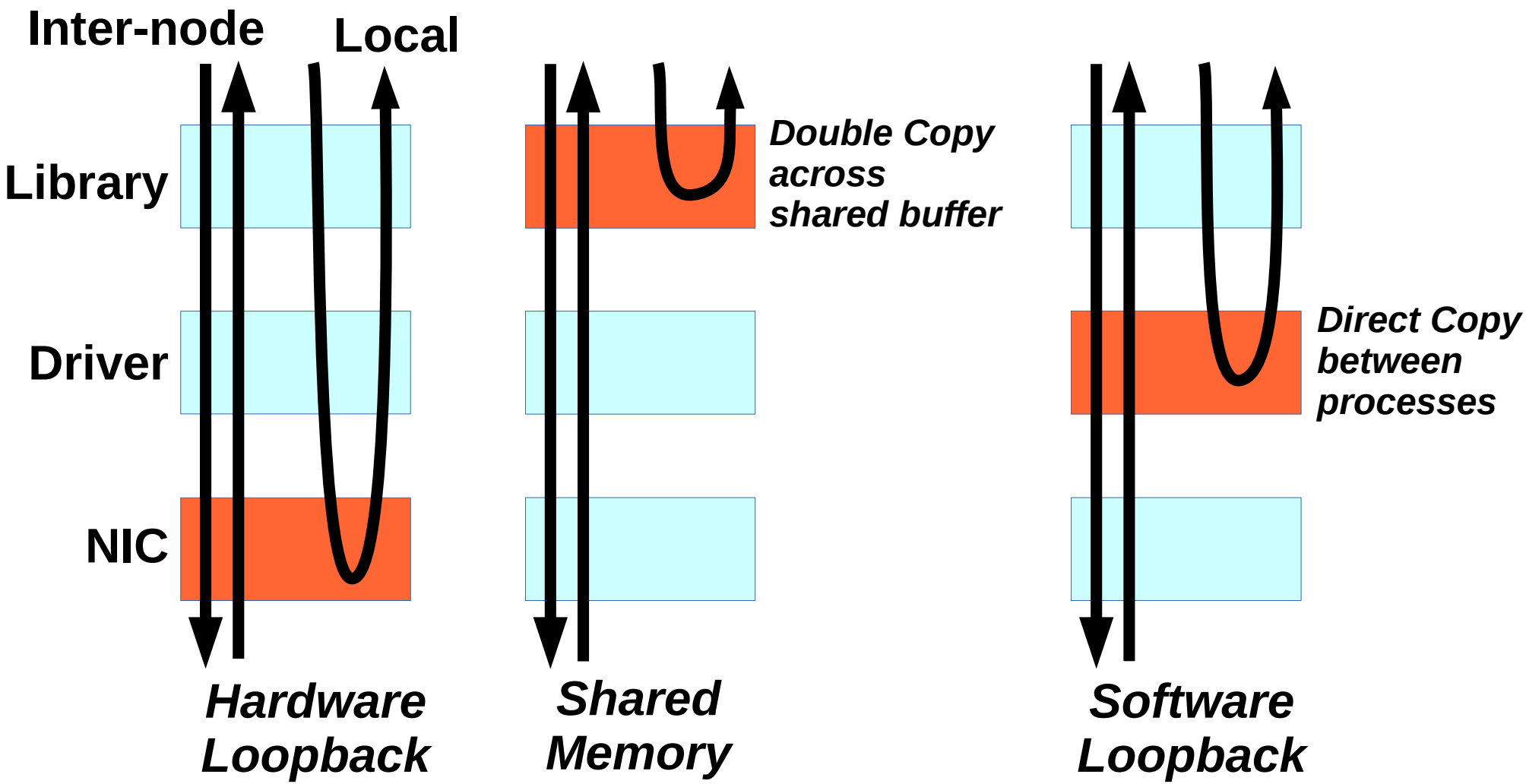


MPI inside nodes, really?

- MPI codes work **unmodified** on multicores
  - No need to add OpenMP, etc.
- Long history of intra-node communication optimization in the Runtime team
  - ➔ Focus on large messages
- KNEM software
  - Joint work with S. Moreaud (PhD), G. Mercier, R. Namyst,

Innovative Computing Laboratory
UNIVERSITY OF TENNESSEE
COMPUTER SCIENCE DEPARTMENT

MPI inside nodes, how?
or how HPC vendors abuse drivers



Portability issues

Solution	**Shared-memory**	**Direct-copy**
Latency	OK	High
Throughput	Depends	OK
Features	Send-receive OK Collectives OK RMA needs work	Send-receive only
Portability	OK	Network-specific or Platform-specific
Security	OK	None

KNEM (*Kernel Nemesis*) design

- RMA-like API
  - Out-of-bound synchronization is easy
- Fixes existing direct-copy issues
  - Designed for send-recv, collectives and RMA
  - Does not require specific network/platform driver
  - Built-in security model

Argonne NATIONAL LABORATORY

[ICPP 2009]

Applying KNEM to collectives

- OpenMPI collectives directly on top on KNEM
  - No serialization in the root process anymore
  - Much better overlap between collective steps
  - e.g. MPI_Bcast 48% faster on 48-core AMD server

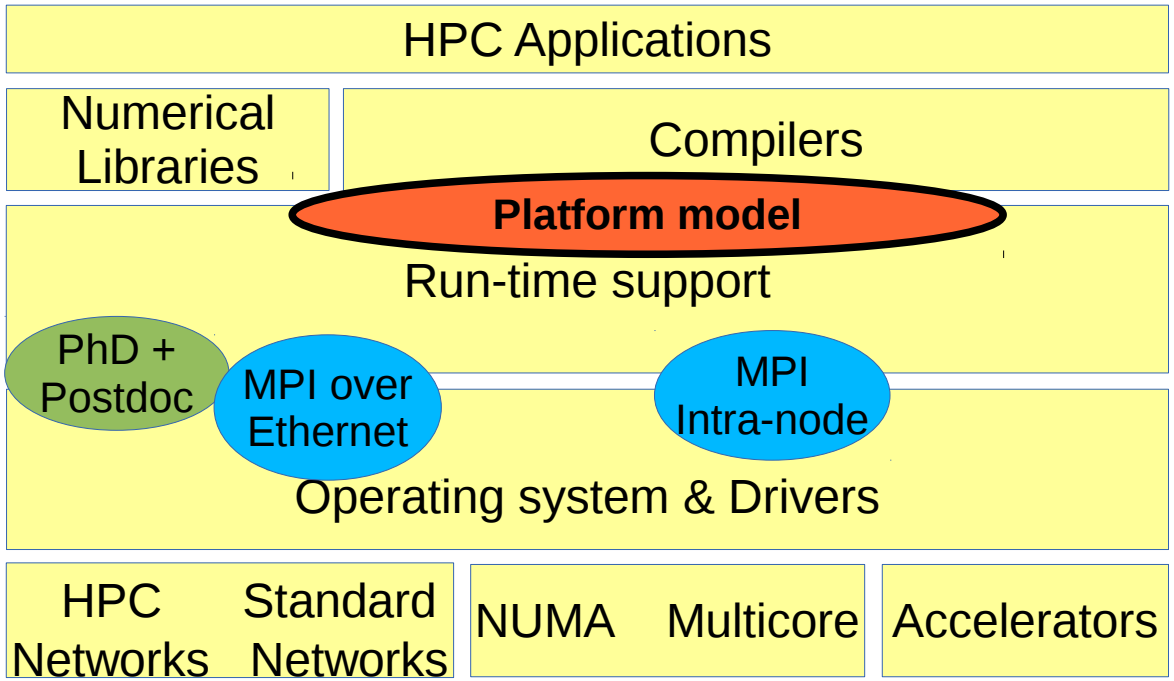
[ICPP 2011, JPDC 2013]

Innovative Computing Laboratory
UNIVERSITY OF TENNESSEE
COMPUTER SCIENCE DEPARTMENT

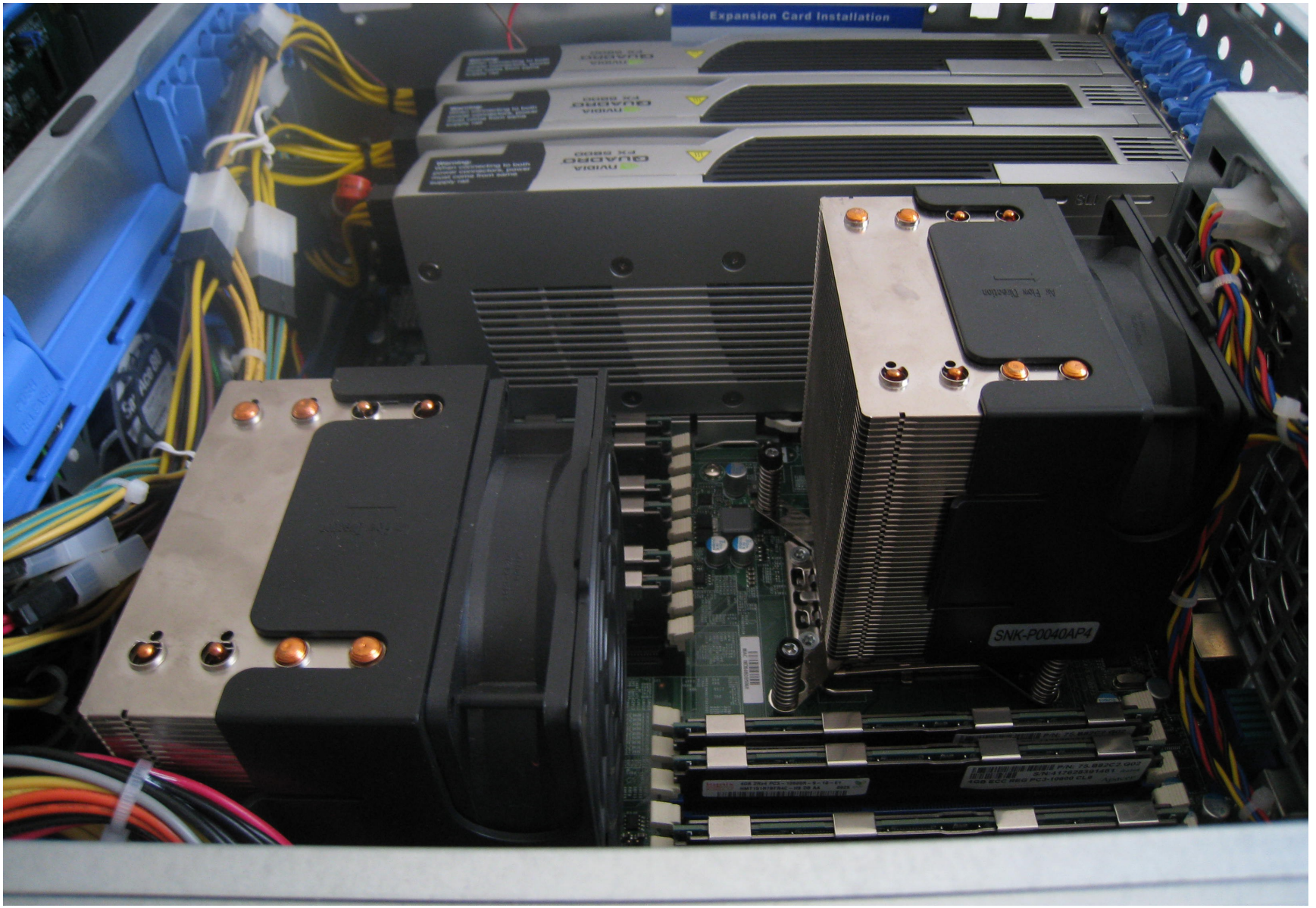
MPI intra-node, summary

- Pushed kernel-assistance to the masses
  - Available in all MPI implementations, for all platforms
  - For different kinds on communication, vectorial buffer support, and overlapped copy offload
- Basic support included in Linux (CMA)
  - Thanks to IBM
- When do we enable which strategy?
  - High impact of process locality

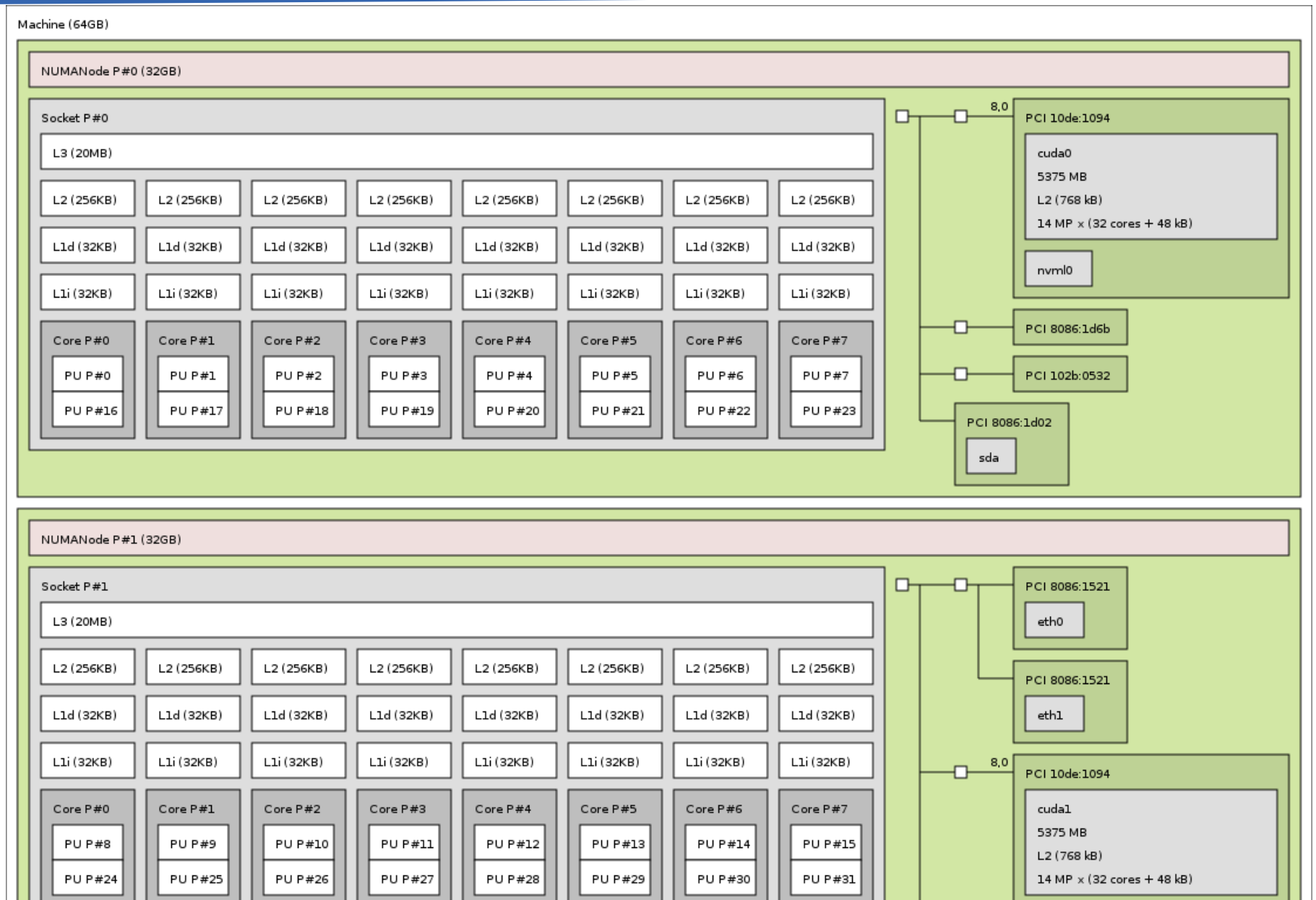
B.1) Better managing hierarchical cluster nodes : Modeling modern platforms



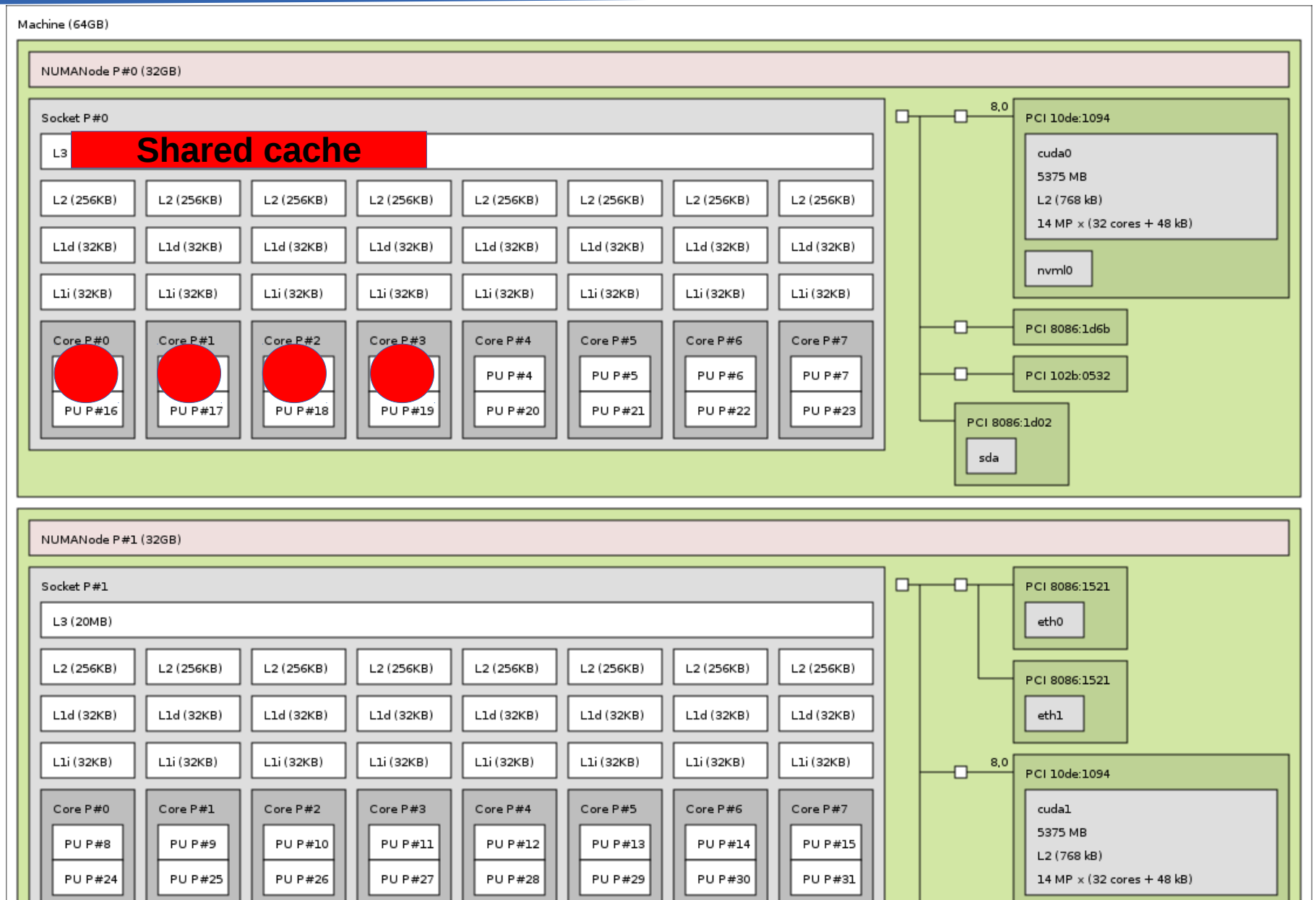
View of server topology



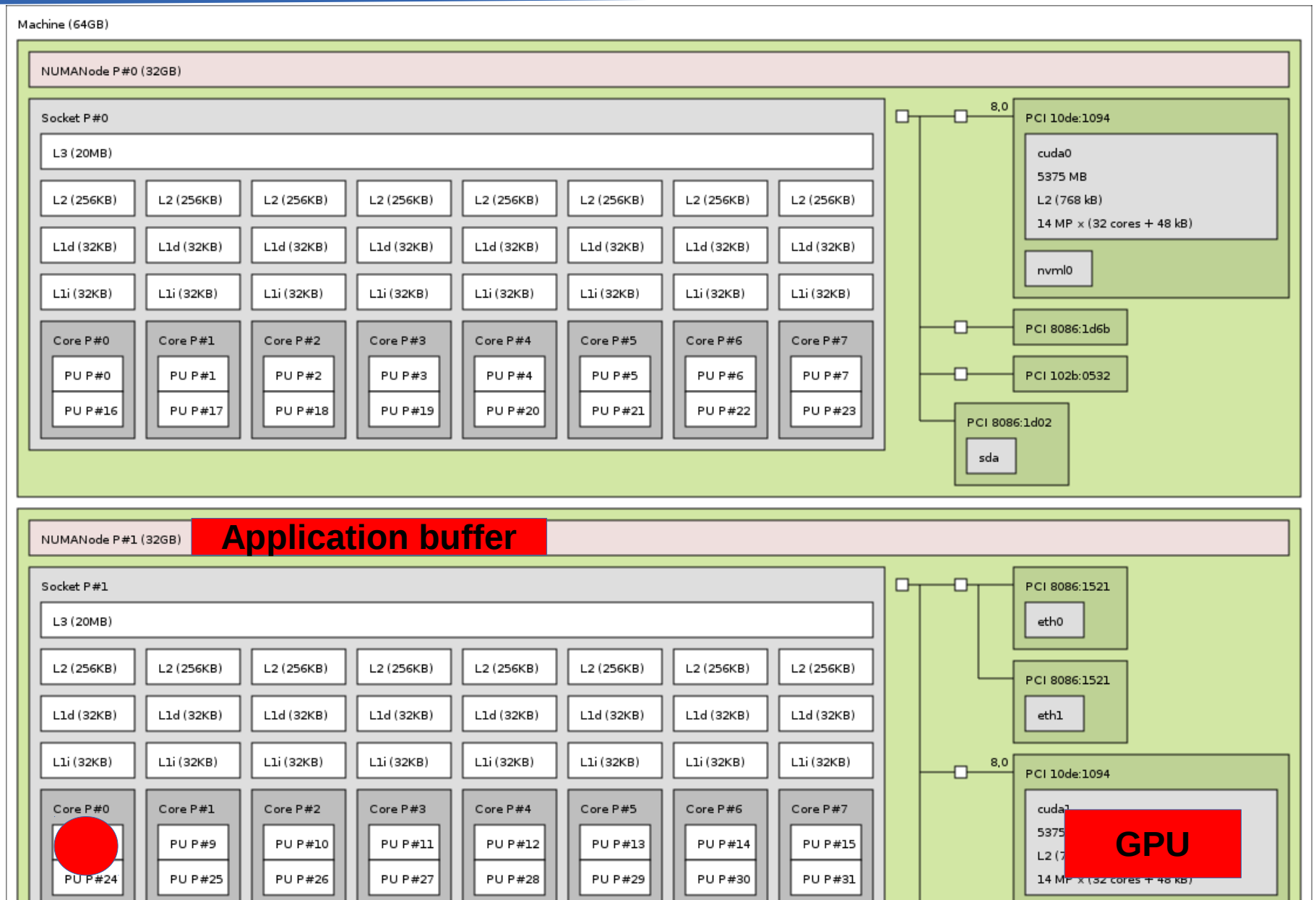
Servers' topology is actually getting (too) complex



Using locality for binding: Binding related tasks



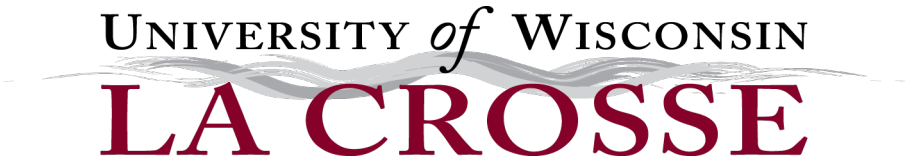
Using locality for binding: Binding near involved resources



Using locality AFTER binding: Adapting hierarchical barriers

Modeling platforms

- Static model (hwloc software) + memory model
- Joint work with J. Clet-Ortega (PhD), B. Putigny (PhD), A. Rougier, B. Ruelle, S. Thibault,



and many other academics and vendors contributing to hwloc.

Static platform model with Hardware Locality (hwloc)

- De facto standard tool for server topology discovery and binding
  - C programming API + tools
  - Used by most MPI implementations, many batch schedulers, parallel libraries, etc.
- Tree of resources based on inclusion+locality
  - Cores #3 and #6 share a 256kB cache in socket #1
  - eth0 NIC is near socket #0

[PDP 2010]

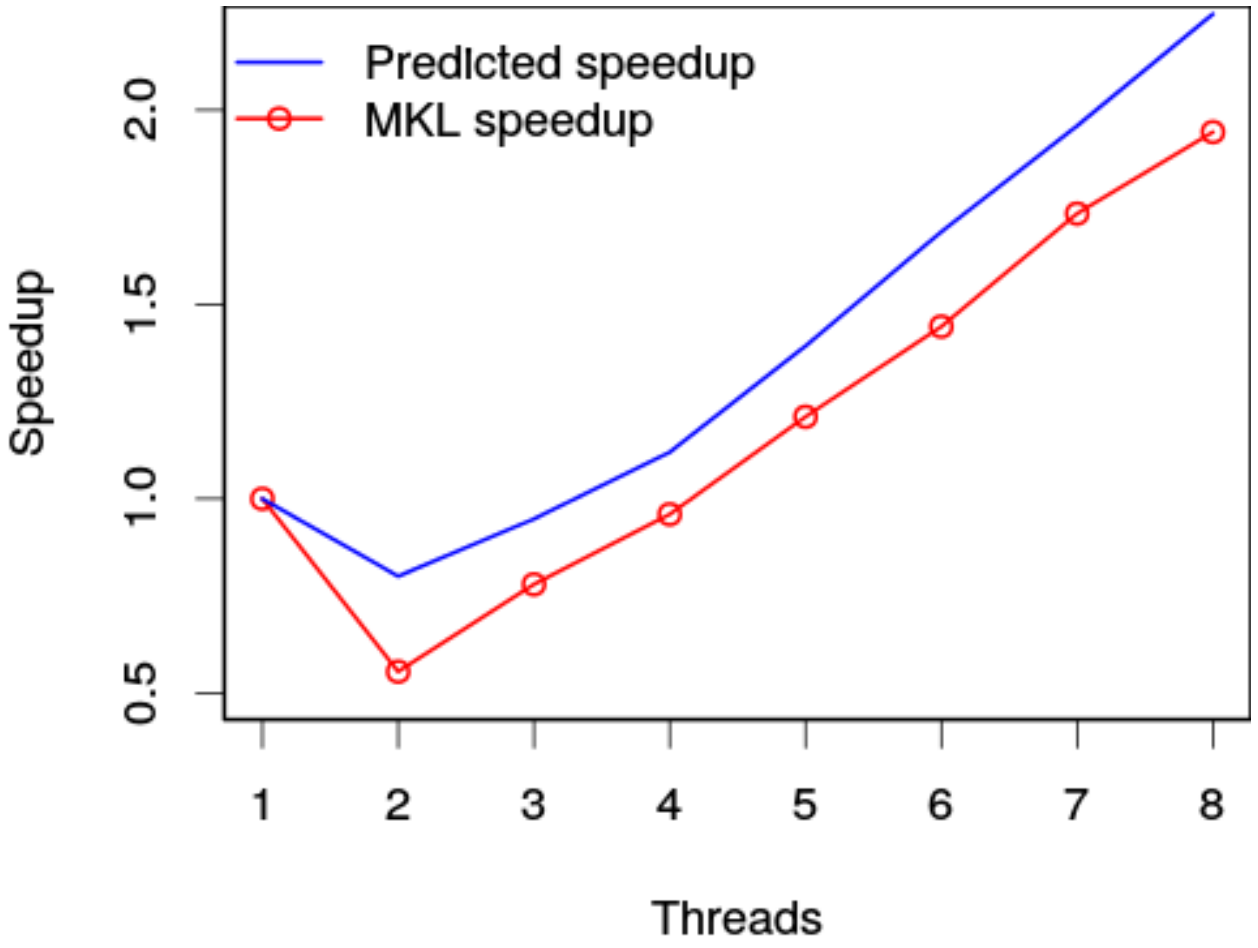
- Extension to networks

[ICPP 2014]

Modeling memory to find bottlenecks

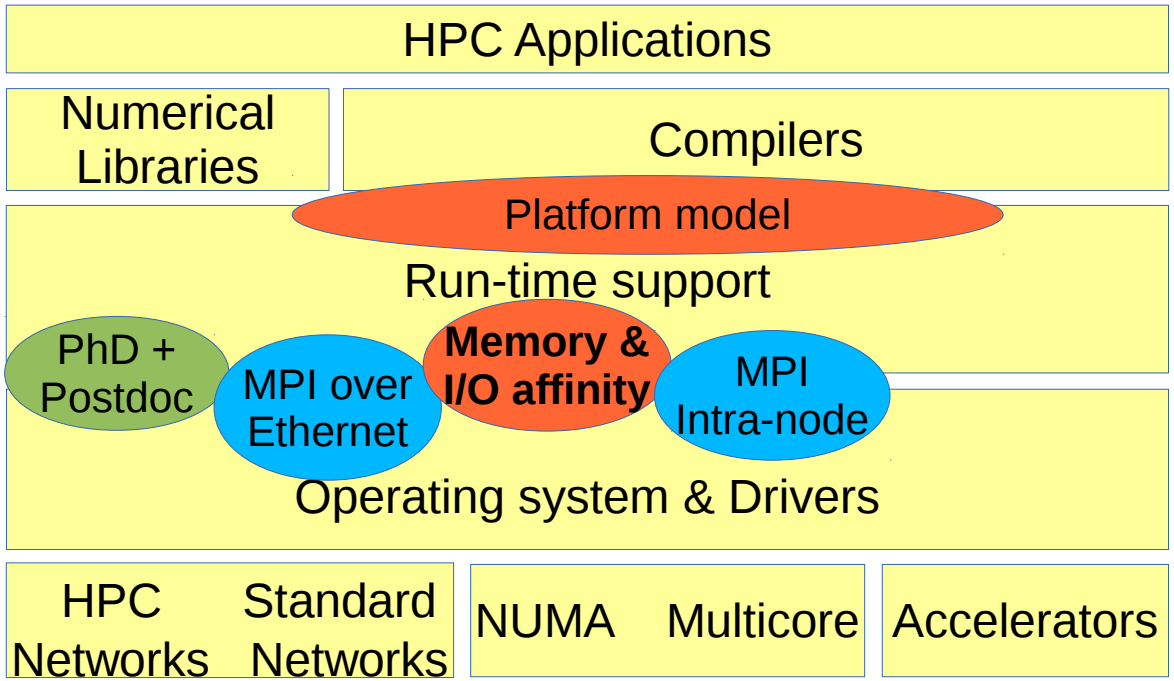
- Memory and caches are the main locality issue
  - Need quantitative numbers
- Capture platform performance characteristics with micro-benchmarks
- Extract the memory access skeleton of the application
- Combine both to predict performance, scalability, etc.
  - Or select intra-node MPI communication strategy

Cache-coherence overhead on dotproduct scalability



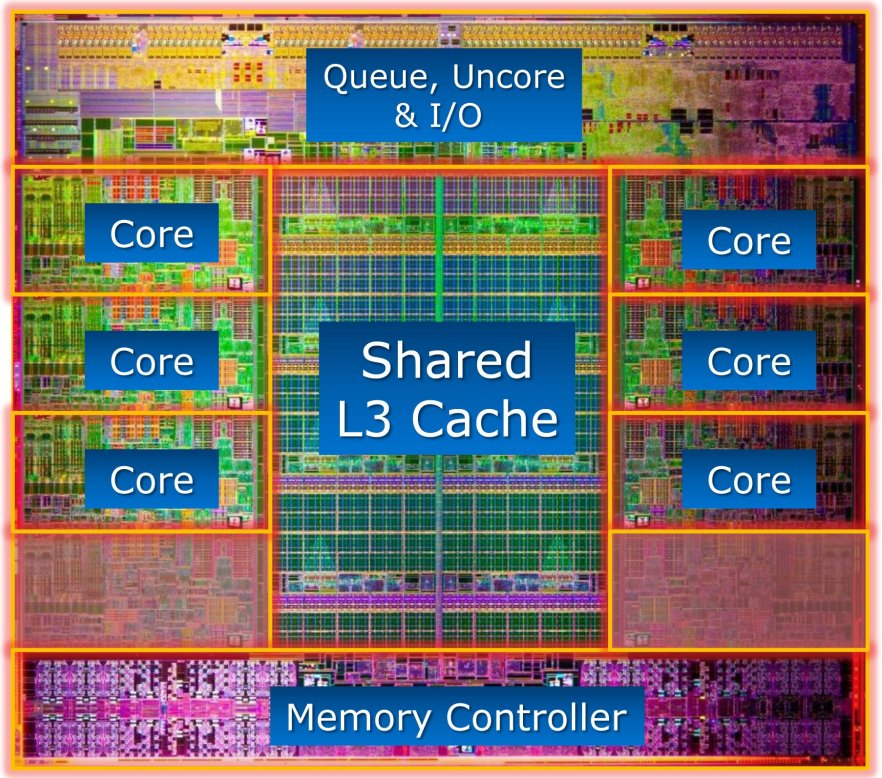
[HPCS 2014]

B.2) Better managing hierarchical cluster nodes : Memory and I/O affinities



Locality matters to more resources

- Vendors integrating more components in the processor



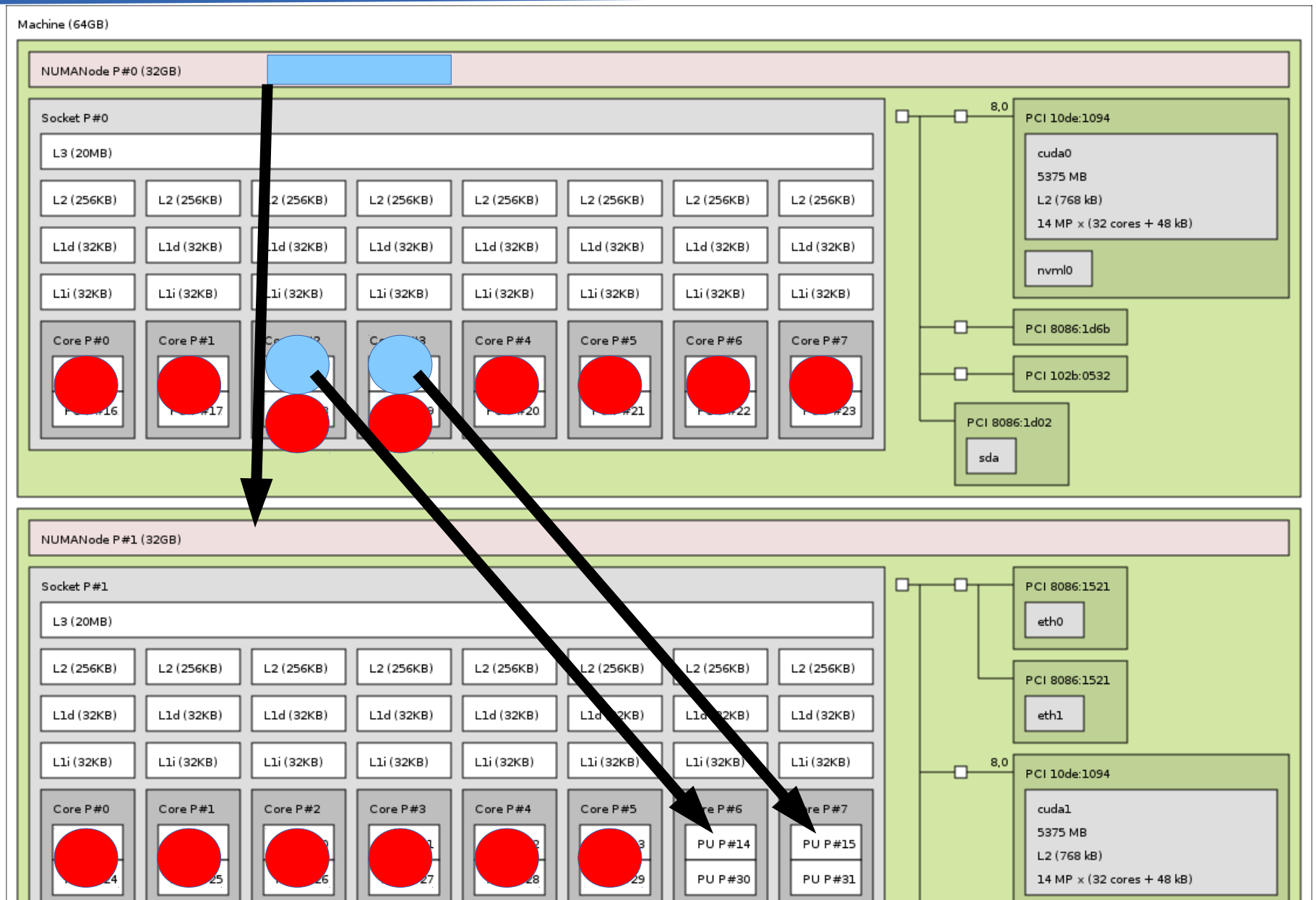
- Locality becoming even more critical

Need for ways to manage memory and I/O affinities

- Enhanced memory migration for NUMA affinity in OpenMP thread scheduling
- Pioneered I/O affinity MPI communication strategies

- Joint work with F. Broquedis (PhD), N. Furmento, S. Moreaud (PhD), P.A. Wacrenier, R. Namyst

Joint threads+memory scheduling



Application buffers must follow tasks

- Needs relevant memory migration techniques
  - Improved Linux migration performance
  - Added lazy migration API
    - No need to detect which buffer needs to move and where
- Applied to OpenMP

NAS BT-MZ class C on 4x4 cores

[IJPP 2011]

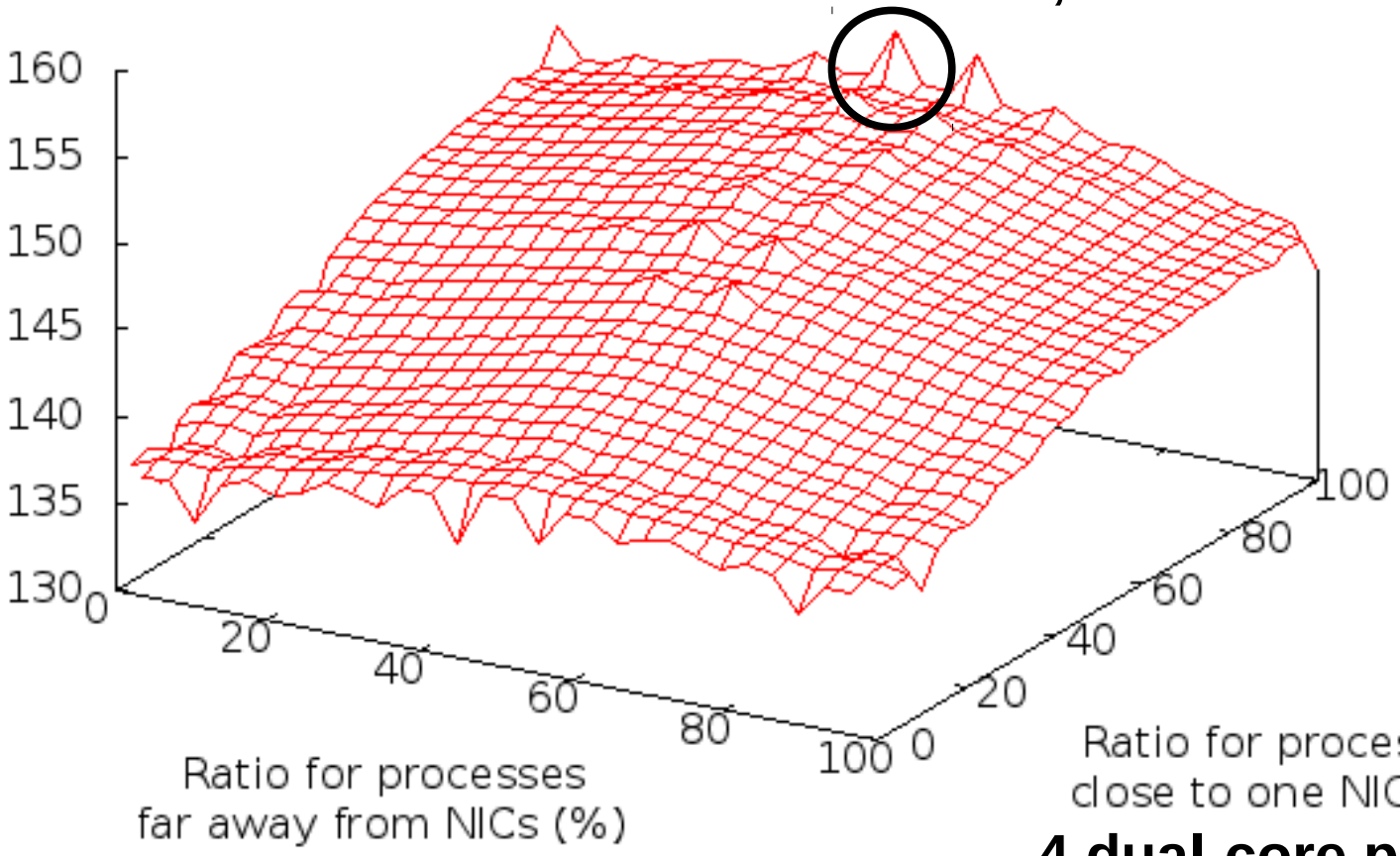
	Speedups		
Threads	**GCC**	**ICC**	**ForestGOMP**
4x4	**9.4**	**13.8**	**14.1**
16x1	**14.1**	**13.9**	**14.1**
16x8	**11.5**	**4.0**	**14.4**
32x8	**10.9**	**2.8**	**14.5**

I/O locality

- Application buffers must be close to GPUs, NICs, etc.
  - 40% DMA Write performance discrepancy
  - ➔ *Non Uniform Input/Output Access*
- Can adapt placement to I/O affinities
  - Or adapt I/Os to placement

NUIOA multirail MPI: Which of my NICs should I use ?

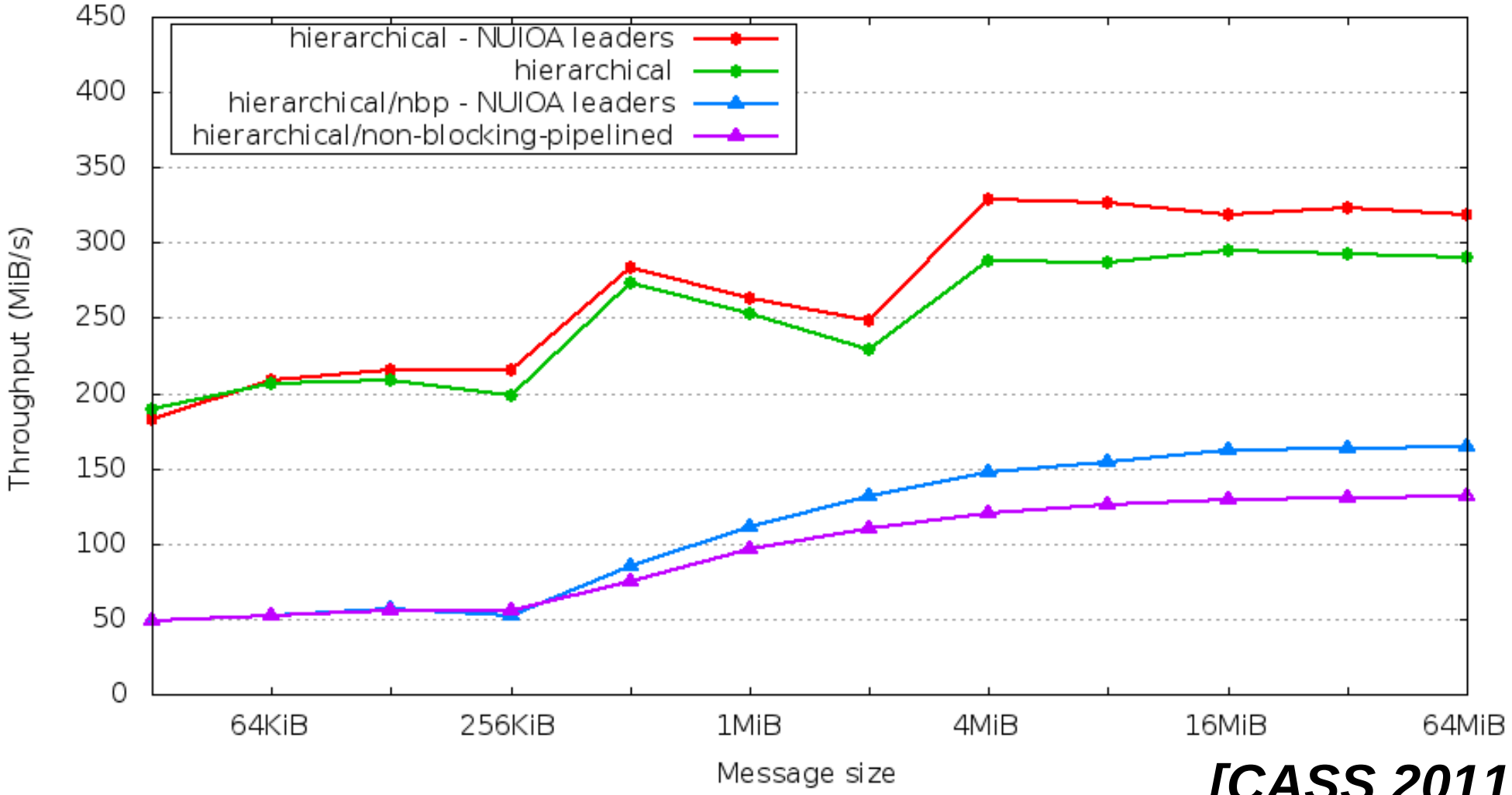
Processes should only use the local NIC if any.
Otherwise, send half to each NIC.



[EuroMPI 2010]

2 IB NICs
4 dual-core processors x 2 nodes
IMB Alltoall between 16 processes

Hierarchical collectives: Choice of the local leader?

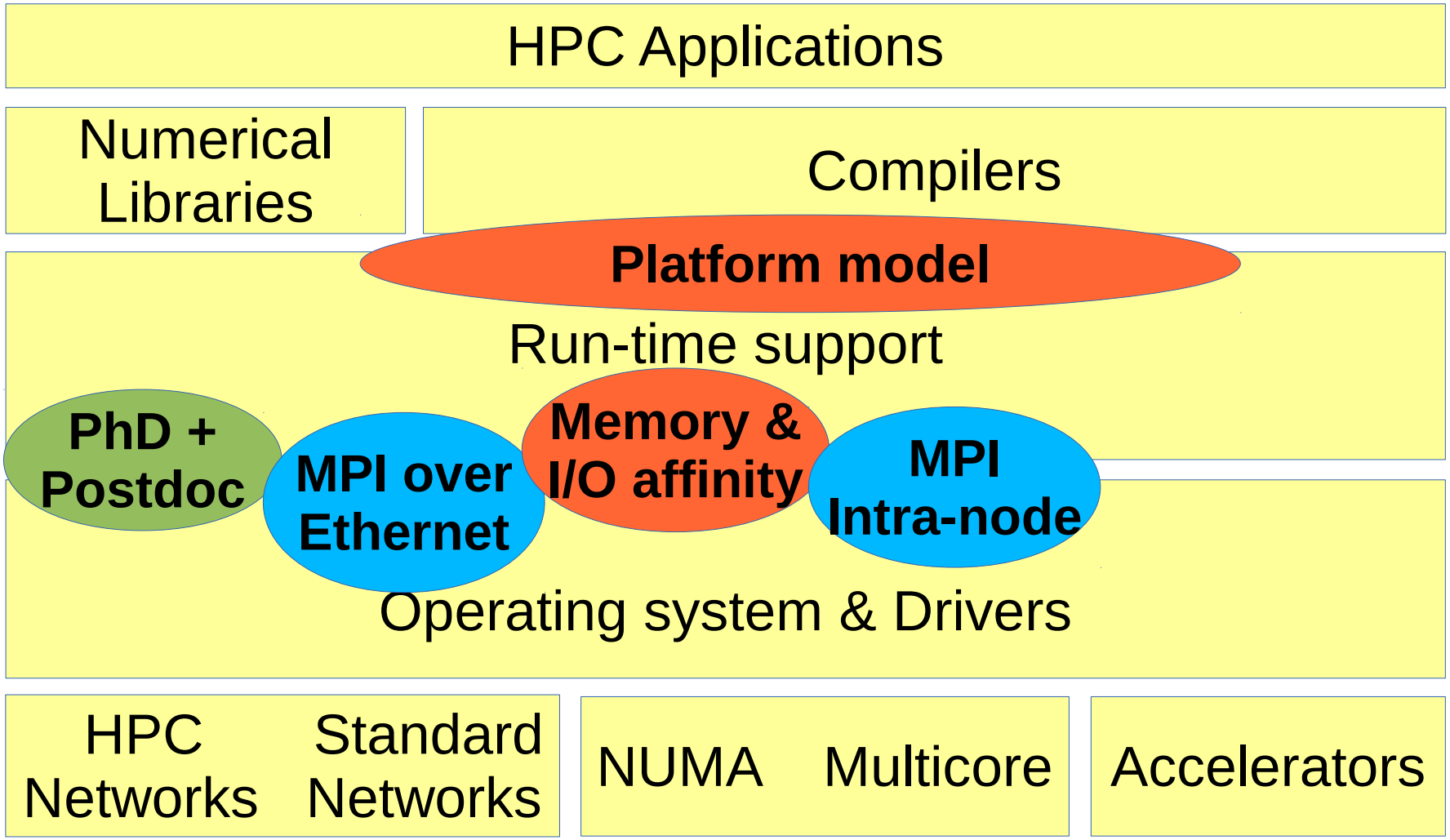


[CASS 2011]

Conclusion

Future Work

Contributions



Contributions to low-level HPC layers

- 90k lines of C, 20k in the Linux kernel
- Influenced MPI implementations
  - Several software pieces integrated in major projects
- Thanks to 2 PhD students, 5 master students, 2 engineers, and many collaborations

Collaborations

- Industrial

- Academic



- ANR projects PARA, NUMASIS, SONGS
- STIC-AmSud SEHLOC project

Other activities

- Many other contributions to the Linux kernel
- Almost 300 hours of operating system teaching at ENSEIRB engineering school
- A lot of science outreach

In the middle of numerous communities

- Applications are from Mars, Hardware is from Venus
  - Big gap to bridge
- HPC standardization boards
  - Communities often look too small
    - MPI misses vendors feedback
    - OpenMP focuses on compilers only
    - Who's designing Exascale programming model?
- HPC and Linux

Next research challenges: Operating systems

- Do we really want Linux as OS for HPC?
  - Depends on the programming model used for Exascale?
- Can HPC work with Linux people?
  - Very different but connected worlds
    - Academics vs vendors?
  - Collaboration could be improved
    - Networking: likely?
    - Scheduling and Memory: unlikely?
- Vendors are of great help

Next research challenges: Networking

- MPI is here to stay
  - No next programming model/language soon?
  - Need locality improvements
- Generic low-level HPC networking API?
  - Depends on IB and CCI future

Next research challenges: Complexity still increasing

- Memory wall
  - Locality even more important?
- Millions of cores?
  - Can we even represent the full topology at scale?
    - Needs multiple levels of precision/factorization
- End of cache coherence?
  - Just another level between shared-memory and distributed?
  - Manual management of non-cache coherence?

Next research challenges: Dealing with complexity

- Too many possible runtime configurations?
  - No way to compare them at runtime
- Mix static and dynamic decisions
  - Compiler-based general execution scheme
  - Refined at runtime
    - Feedback from performance counters
    - Compiler-envisioned bottlenecks?
  - Need strong collaborations between all layers

Thank you