Towards Simulation of an Unified Address Space for 128-bit Massively Parallel Computers

Eduardo Tomasi [1,2], César Fuguet [1], Christian Fabre [1], Frédéric Pétrot [2]

[1] Univ. Grenoble Alpes, CEA, List, F-38000 Grenoble, France
[2] Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, F-38000 Grenoble, France

This work was funded by the French National Research Agency (ANR) under grant agreement ANR-21-CE25-0016 (MAPLURINUM)

CONTEXT

Distributed Architectures

A cluster is a multicore computer that has its own memory hierarchy (multi-level caches and DRAM) and accelerators (such as GPUs). They are connected to each other through a high-performance communication network.

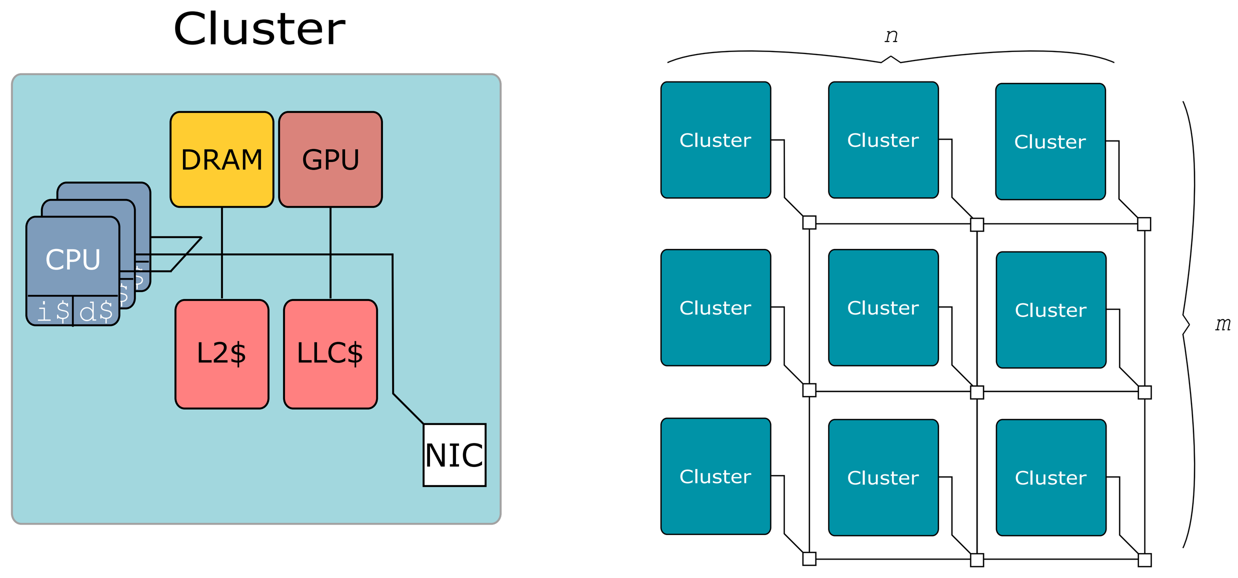


Fig 1. An example of a typical clustered architecture, as can be found in high performance computing (HPC).

Programming Models

Within a cluster, memory coherence is ensured by hardware protocols, such that CPUs and accelerators communicate through a **shared memory.**

Beyond a cluster, coherence cannot be efficiently guaranteed. Processes in different clusters do not have access to each other's address space. Data can only be shared by **message passing** through the communication network.

A **global address space** offers a simplified programming model for distributed systems. It provides a high-level abstraction of the memory, hiding the complexity associated to its management.

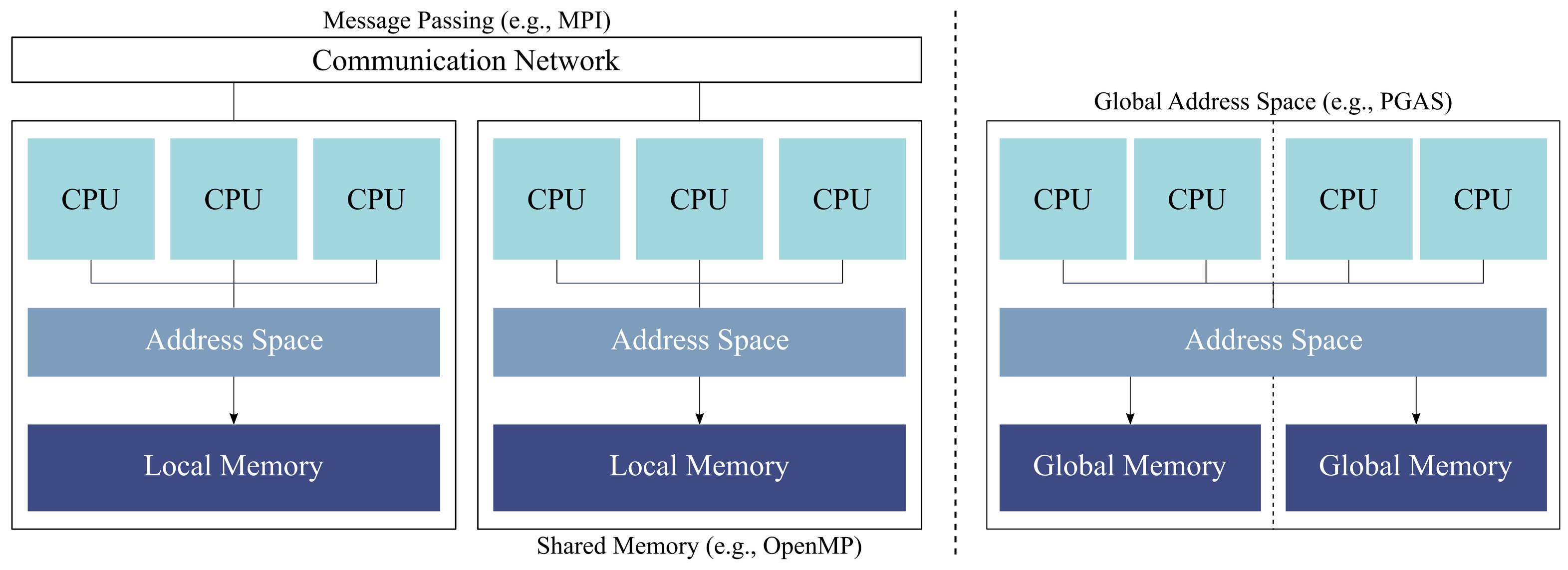


Fig 2. Shared Memory is used for parallelism inside a cluster, and Message Passing for parallelism between clusters.

MPI requires developers to expllictly define the communication and synchronization points. Also, pointers cannot be shared, which complexifies the transfer of structured data from one process to another. PGAS, on the other hand, eliminates the need for explicit communication, but introduces challenges related to data consistency and synchronization.

SIMULATION OF A DISTRIBUTED SYSTEM

QEMU

QEMU is an open-source machine emulator. Multiple architectures can be executed in a single host. It provides a virtual model of an entire machine (CPU, memory and devices). We chose it for three main reasons:

1 — IT IS FAST

It uses dynamic binary translation (DBT) to reach very high simulation speed. Also, QEMU's scalability on SMP machines is good [1].

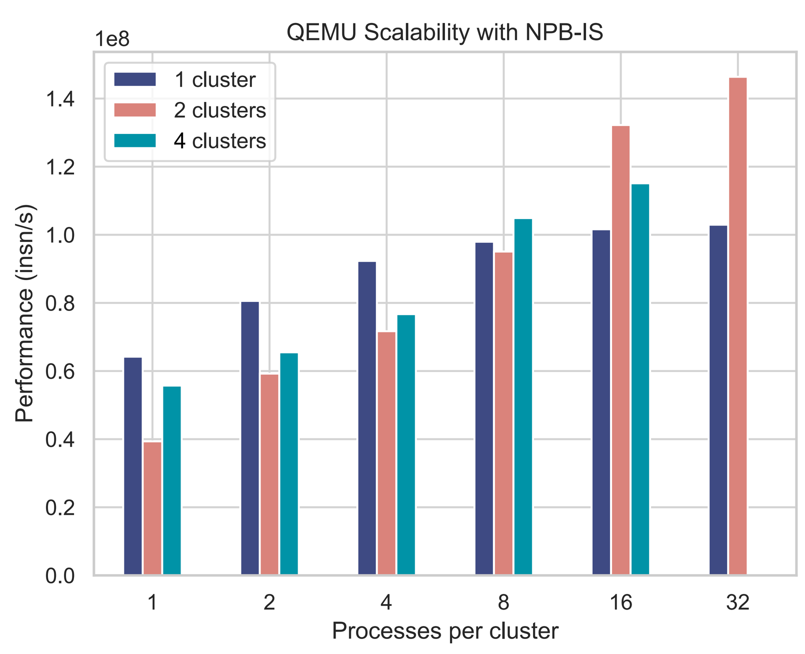
2 — TCG PLUGIN

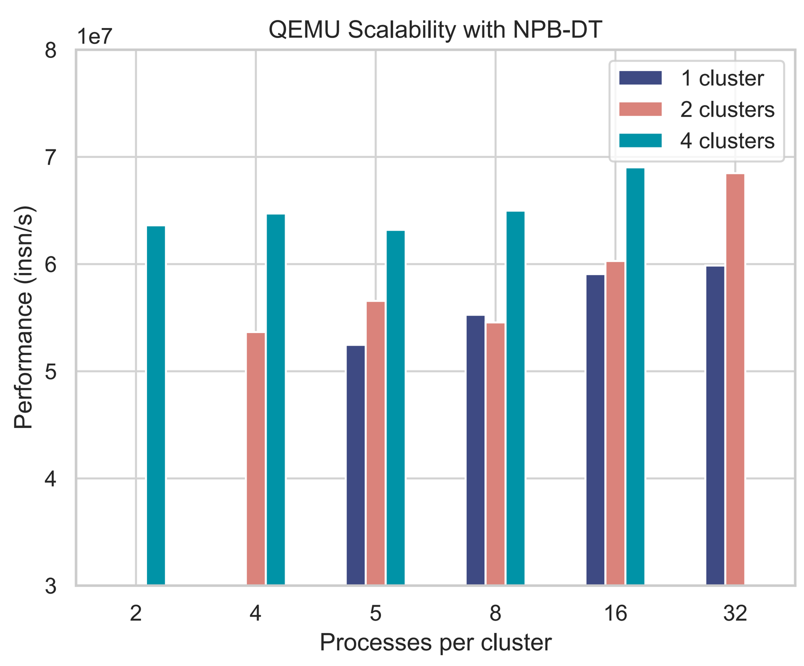
QEMU plugins provide interfaces to extend the simulator and add proper instrumentation. Plugins provide a mechanism to subscribe to events during translation and execution of instructions [2].

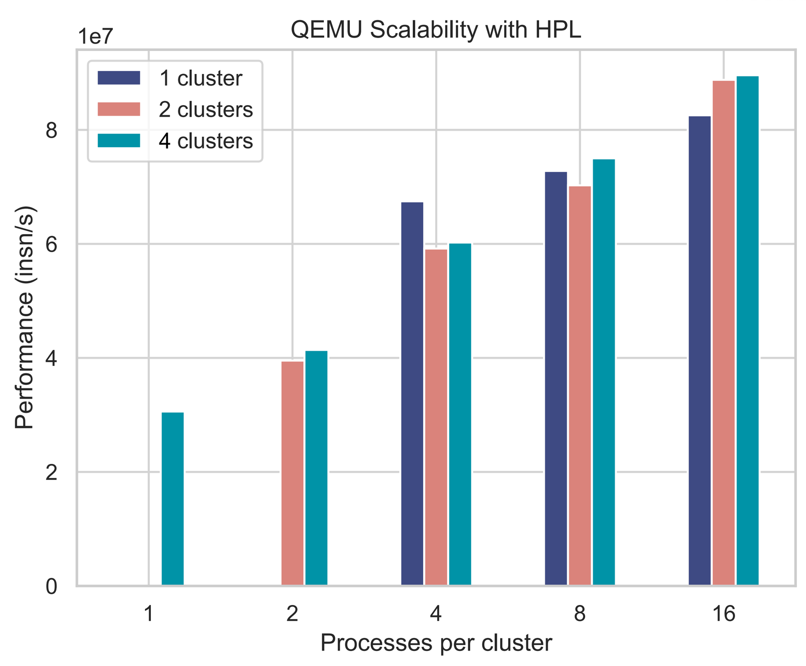
3 — RISC-V 128

QEMU already has support for RV128 [3]. Global memory in HPC might, in the next decade, exceed 2^{64} bytes. It allows us to rethink the software architecture of supercomputers, including memory virtualization [4].

Scalability







NON-INTRUSIVE WORKLOAD ANALYSIS

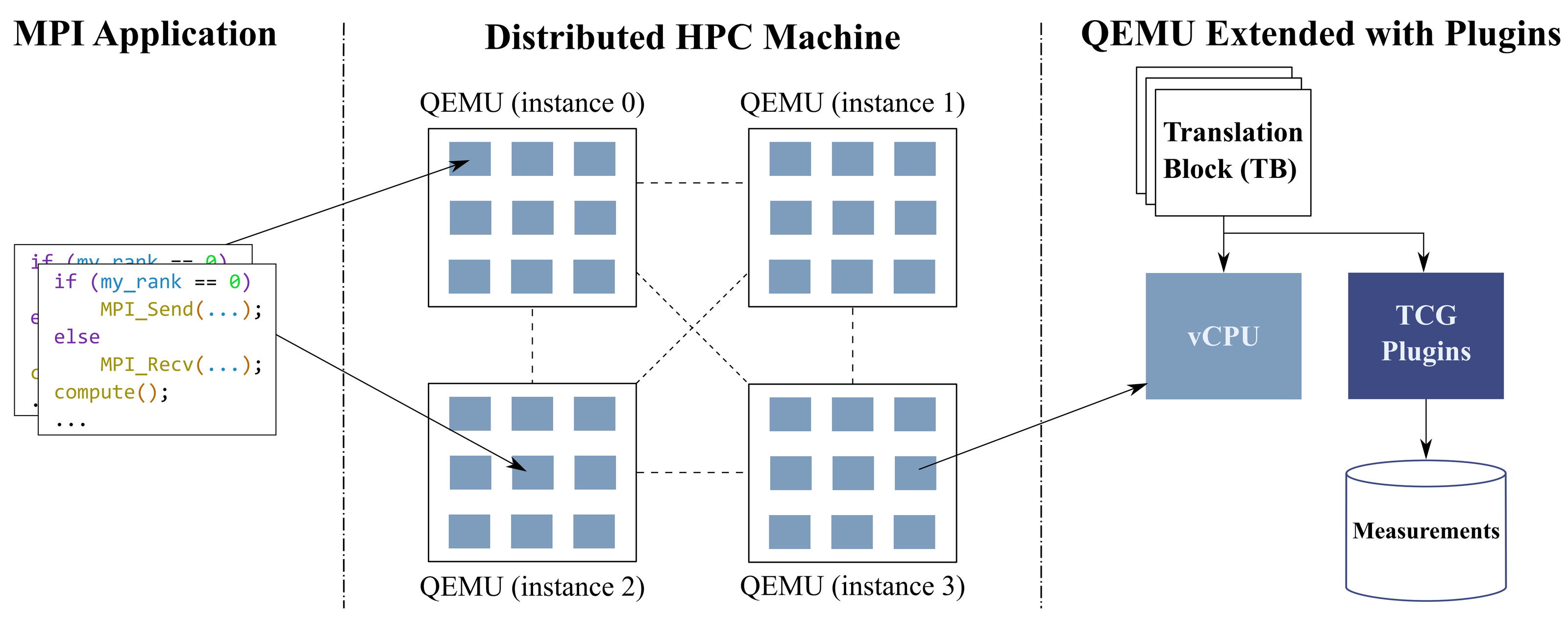


Fig 3. How to use QEMU to profile MPI calls.

To retrieve the metrics of interest, we need to monitor the code during execution. We need to know beforehand the virtual addresses of the MPI functions in the binary code. Through a QEMU plugin, we can monitor the translation blocks executed and profile the calls.

Results

We can gather metrics such as: number of instructions during a MPI call, number of system calls and number of memory accesses.

Function	n_p	Syscalls 1 cluster	Syscalls 2 clusters	Syscalls 4 clusters
MPI_Allreduce	1	0	4	634
	2	93	41	888
	4	78	3656	2379
	8	80	5056	2178
	16	85	4203	126
	32	144	294	-
MPI_Alltoallv	1	37	4763	5488
	2	59	7627	5809
	4	51	8529	6792
	8	55	8762	7242
	16	65	9756	4361
	32	109	10666	-

Table I. Number of syscalls during MPI calls of NPB-IS.

Function	n_p	Syscalls 1 cluster	Syscalls 2 clusters	Syscalls 4 clusters
MPI_Send	5	9	9	10
	8	10	10	10
	16	11	12	15
	32	19	22	-
MPI_Recv	5	4	4	4
	8	5	5	5
	16	11	7	8
	32	9	9	-

Table II. Number of syscalls during MPI calls of NPB-DT.

	No plugin	insn plugin		MPI plugin	
n_p	MIPS	MIPS	Slowdown	MIPS	Slowdown
1	74.3	64.2	13.6%	40.0	46.1%
2	101.6	80.6	20.7%	48.7	52.0%
4	126.2	92.3	26.9%	53.7	57.4%
8	155.3	97.9	36.9%	60.0	61.3%
16	170.0	101.6	40.2%	64.5	62.1%
32	203.2	102.9	49.3%	72.3	64.4%

Table III. Impact of the plugins during execution of NPB-IS.

References

[1] M. Badaroux, S. Miroddi, and F. Petrot, "To Pin or Not to Pin: Asserting the Scalability of QEMU Parallel Implementation," in 2021 24th Euromicro Conference on Digital System Design (DSD), Palermo, Italy: IEEE, Sep. 2021, pp. 238–245. doi: 10.1109/DSD53832.2021.00045.
[2] E. G. Cota and L. P. Carloni, "Cross-ISA machine instrumentation using fast and scalable dynamic binary translation," in Proceedings of the 15th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, Providence RI USA: ACM, Apr. 2019, pp. 74–87. doi: 10.1145/3313808.3313811.
[3]F. Portas and F. Petrot, "Fast simulation of future 128-bit architectures," in 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium: IEEE, Mar. 2022, pp. 1131–1134. doi: 10.23919/DATE54114.2022.9774706.
[4] A. Waterman and K. Asanović, "Chapter 6, RV128I Base Integer Instruction Set, Version 1.7," in The RISC-V Instruction Set Manual - Volume I: Unpriviliged ISA, 20191213th ed.The RISC-V Foundation, 2019. [Online]. Available: https://riscv.org/technical/specifications/

UGA Université Grenoble Alpes — anr agence nationale de la recherche