Introduction to the Parallel Execution Environment

Pau Adell (pau.adell@estudiantat.upc.edu)

October 2024

Node architecture and memory

Complete the following table with the relevant architectural characteristics of the different node types available in boada:

	boada-11
Number of sockets per node	2
Number of cores per socket	10
Number of threads per core	2
Maximum core frequency	3200Mhz
L1-I cache size (per-core)	32KB
L1-D cache size (per-core)	32KB
L2 cache size (per-core)	1024KB
Last-level cache size (per-socket)	14MB
Main memory size (per socket)	47GB
Main memory size (per node)	94GB

Table 1: P-values for different data transformations

Include in the document the architectural diagram for one of the nodes boada-11 to boada-14.

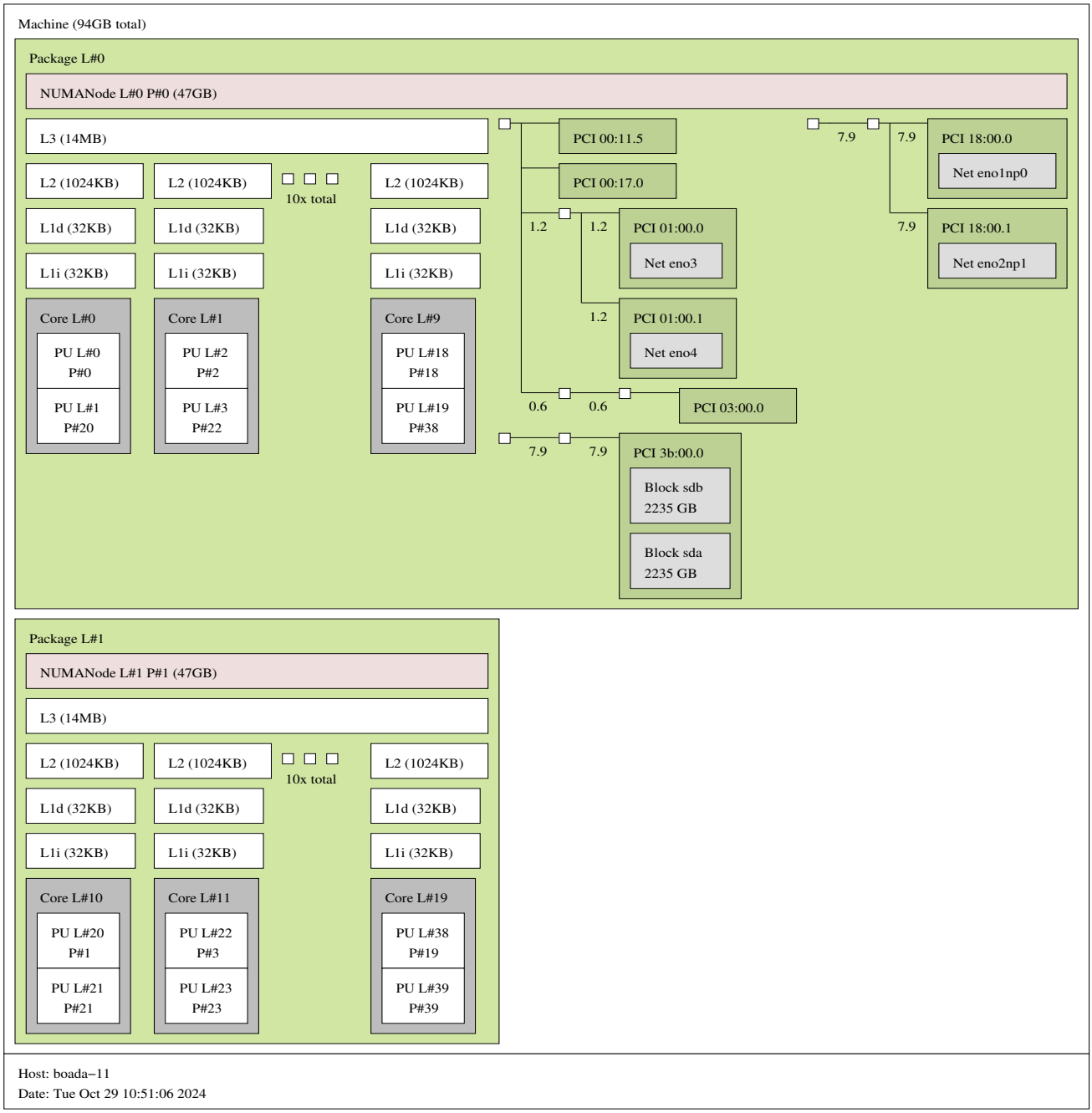

Figure 1: Architectural diagram of Boada-11.

Timing sequential and parallel executions

Plot the execution time and speed–up that is obtained when varying the number of threads (strong scalability) by submitting the jobs to the execution queue (section 1.4.3).

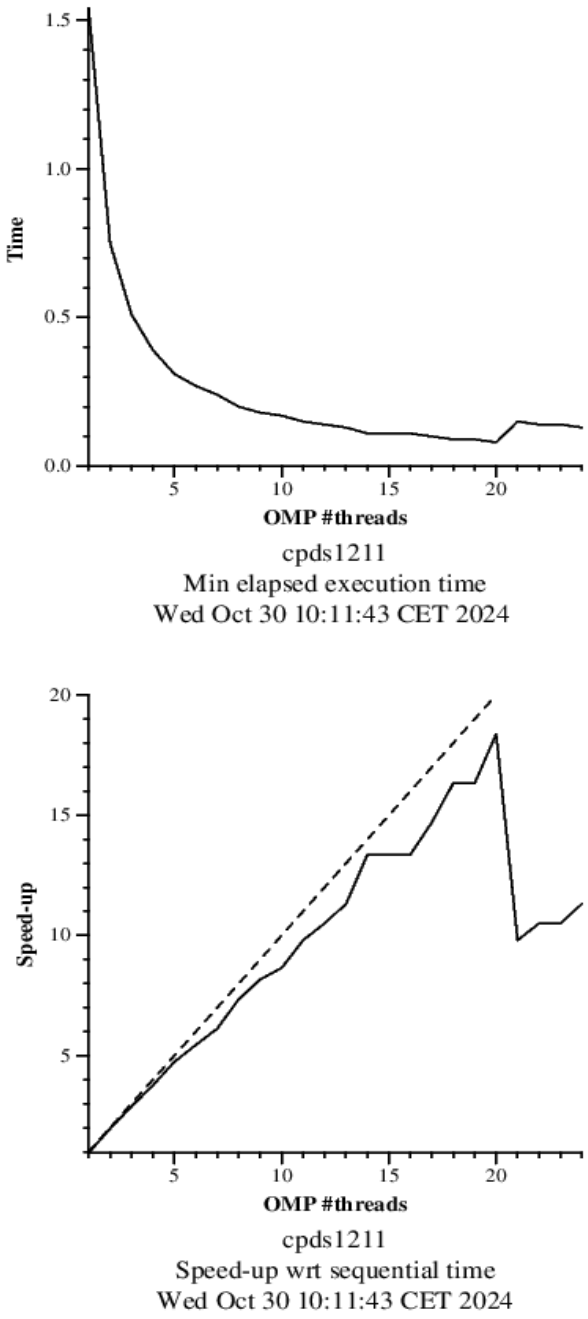

Figure 2: Execution time and speed-up for OMP strong scalability

Strong scalability refers to the ability reducing the execution time for a fixed size problem when the number of threads is increased.

As we can see at 21 threads we loose speedup and it takes more time, this could be due to a synchronization bottleneck. As the number of threads increase and the time spent waiting for these synchronization points can grow, it seems that using more than 20 threads has to use another node (as there is 20 threads per node) and this may increase the synchronization time.

Show the parallel efficiency obtained when running the weak scaling test.

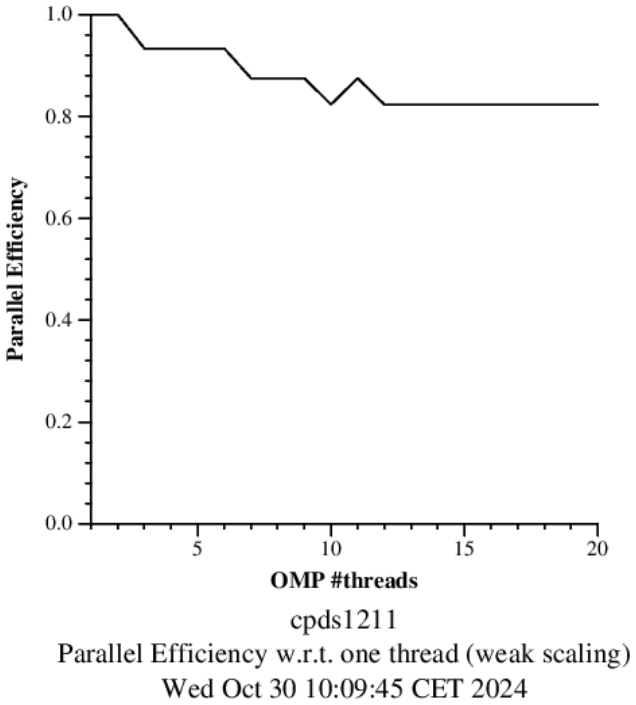

Figure 3: Efficiency of weak scaling test

On the other hand, weak scalability measures system's efficiency when both the problem size and the number of threads increase proportionally. The focus is to maintain a constant execution time as workload and resources scale together. In this plot as it remains hight, we can see a "strong" weak scalability.

Plot the execution time and speed–up that is obtained when varying the number of MPI processes from 1 to 20 (strong scalability) by submitting the jobs to the execution queue (section 1.5.1).

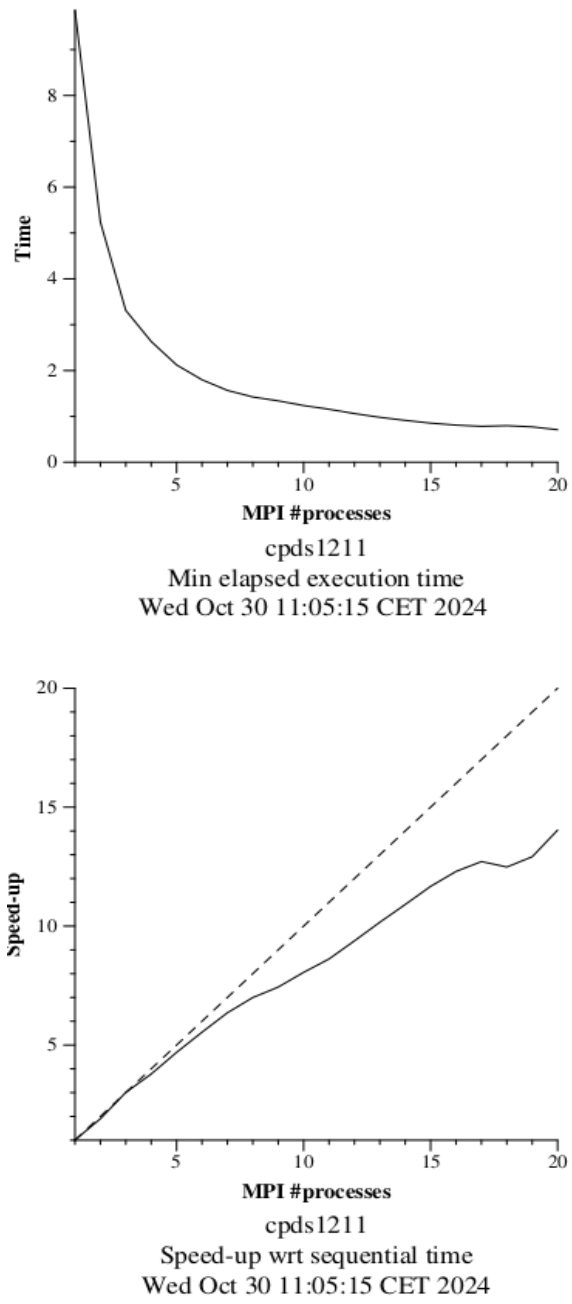

Figure 4: Execution time and speed-up for MPI strong scalability

Here we can see the original sequential time is (when num.thread = 1) 9,86 seconds, while the time when we use 20 threads is 0.71 seconds.

In addition, show in a table the elapsed execution time when executed with 2 MPI processes when varying the number of threads from 1 to 20

Number of threads	1	2	3	4	5	6	7	8	9	10
Execution time (sec)	4.931	2.473	1.753	1.259	1.053	0.879	0.754	0.680	0.605	0.545

Table 2: Execution time for 2 MPI processes from 1-10 threads

Number of threads	11	12	13	14	15	16	17	18	19	20
Execution time (sec)	0.573	0.525	0.489	0.454	0.424	0.397	0.374	0.355	0.336	0.320

Table 3: Execution time for 2 MPI processes from 11-20 threads

While using 2 MPI processes the time with 20 threads is 0.320 as we can see in Table 3.