Rpi Cluster DeepLearning

Matrix multiplication benchmark

Matthieu François - Fatou Kiné Sow - Simon Lebeaud - Nicolas Tirel

GreenAI U.P.P.A.

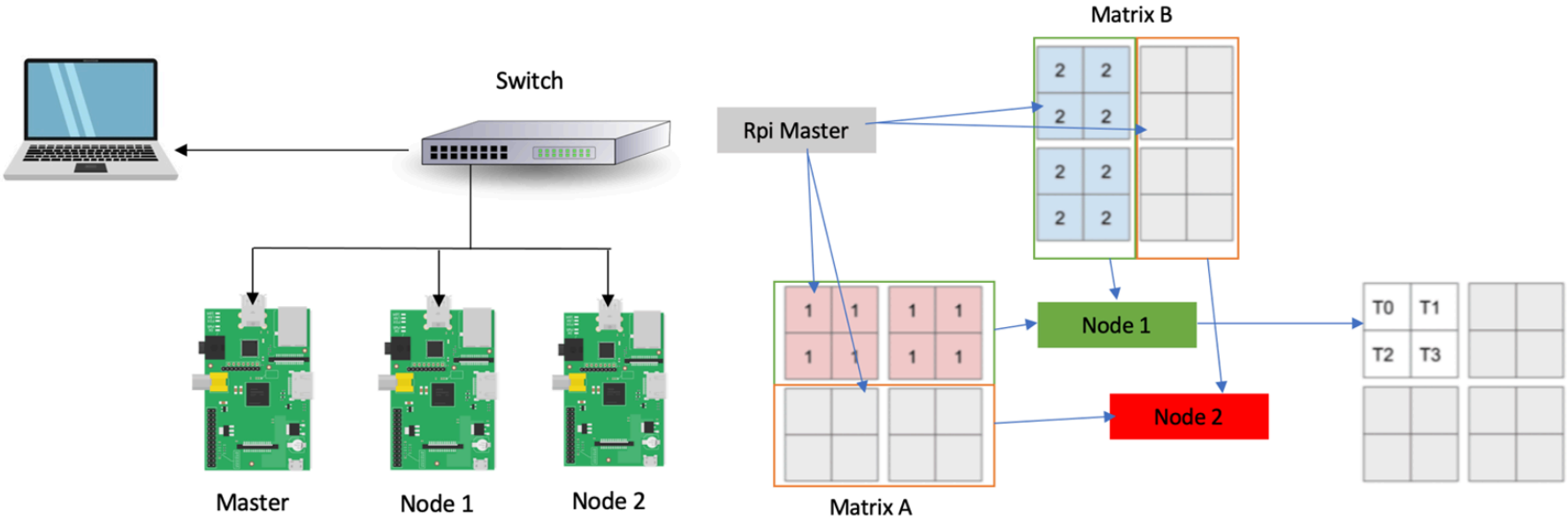
30-05-2022

Distributed Matrix Multiplication Approach

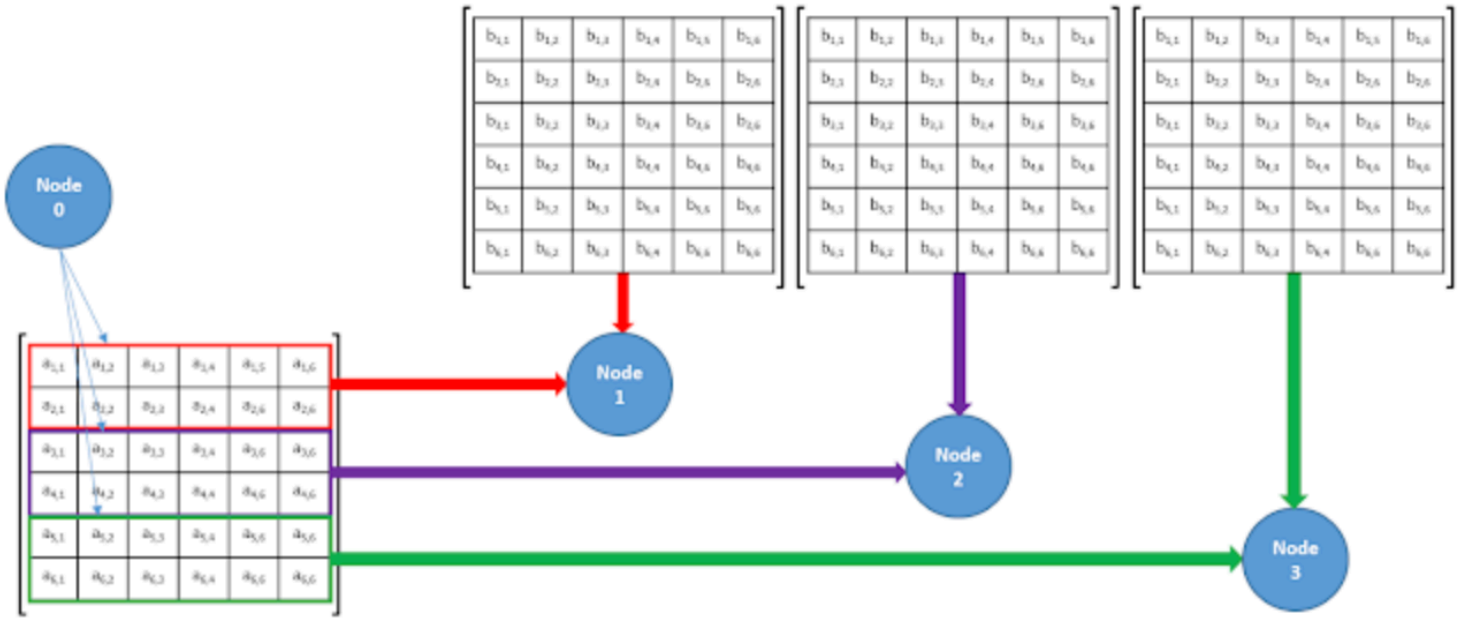
Two matrices, matrix A and a matrix B, are multiplied together to produce a resultant matrix C

- Matrices A and B are divided into tiles/blocks
- The matrix multiplication is divided into tasks that multiply one tile of A to a corresponding tile of B
- Its corresponding data is then allocated to one of the nodes in the cluster

Distributed Matrix Multiplication Approach



Distributed Matrix Multiplication Approach



[Corbilla, 2016]

Benchmark

Run Time Analysis of Matrix Multiplication Using Raspberry Pi Cluster Supercomputer [Parikshit Dubey and Arva Kagdi, EECS, Syracuse University, May 2020]

- Four Raspberry Pi model 4B's (2 GiB).
- High performance and widely portable implementation MPICH of the MPI standard.
- Square matrices benchmarked at different sizes varying from 3 to 150.

Benchmark

Parallel Processing on a single Raspberry Pi.

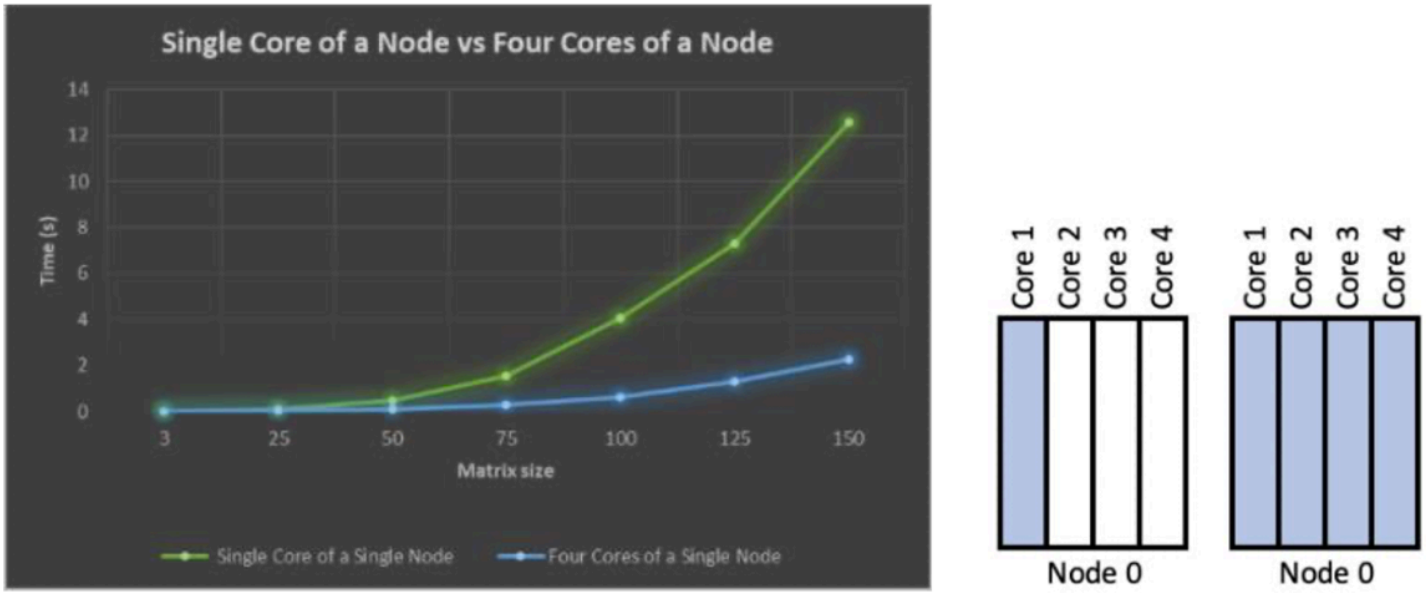


Figure 1: Single core of a node vs four cores of a node

[Dubey and Kagdi, 2020]

Benchmark

Parallel and Distributed Computing using four cores of a single node and single cores of four nodes in a cluster.

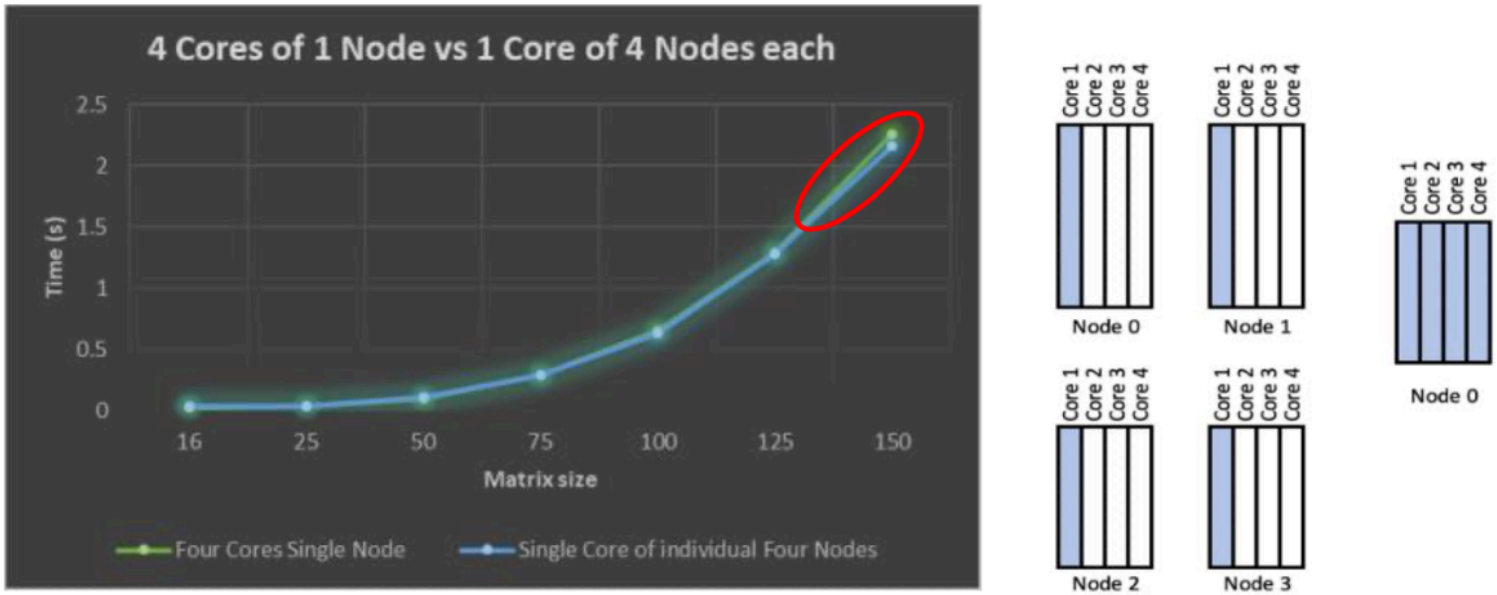


Figure 2: Four cores of a node vs single cores of four nodes each

[Dubey and Kagdi, 2020]

Benchmark

Fully Parallel and Fully Distributed Computing. Sixteen cores of four RPis and four cores of a single Rpi.

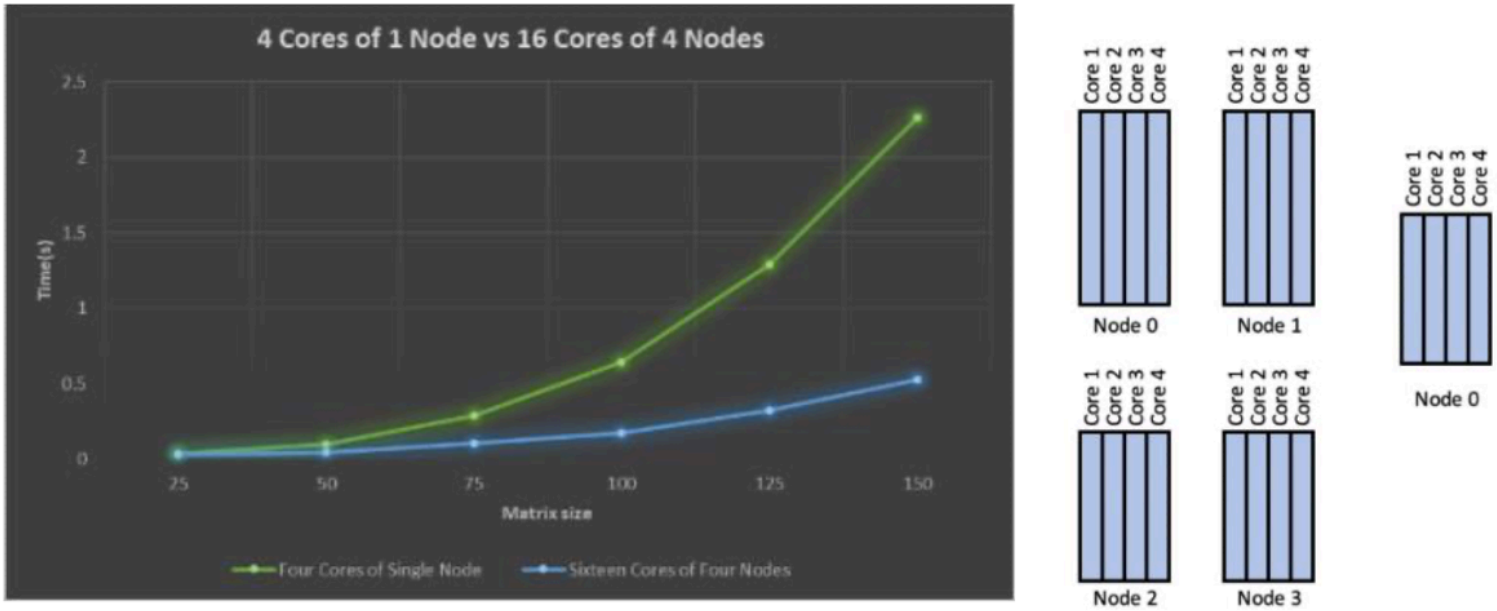


Figure 3: 4 cores of a node vs 16 cores of 4 nodes

[Dubey and Kagdi, 2020]

Benchmark

System	N	GFLOPS	Idle Power	AvgLoad Power	GFLOPS per Watt	MFLOPS per US$
Gumstix Overo	4000	0.041	2.0	2.7	0.015	0.20
Beagleboard-xm	5000	0.054	3.2	4.0	0.014	0.36
Beaglebone Black	5000	0.068	1.9	2.6	0.026	1.51
Raspberry Pi Model B	5000	0.213	2.7	2.9	0.073	6.09
Raspberry Pi Model B+	5000	0.213	1.6	1.8	0.118	6.09
Raspberry Pi Compute Module	6000	0.217	1.9	2.1	0.103	5.43
Raspberry Pi Model A+	4000	0.218	0.8	1.0	0.223	10.9
Raspberry Pi Zero	5000	0.319	0.8	1.3	0.236	63.8
Cubieboard2	8000	0.861	2.2	4.4	0.194	14.4
Pandaboard ES	4000	0.951	3.0	5.8	0.163	4.78
Raspberry Pi Model 2B	10,000	1.47	1.8	3.4	0.432	42.0
Dragonboard	8000	2.10	2.4	4.7	0.450	28.0
Chromebook	10,000	3.0	5.9	10.7	0.277	16.3
Raspberry Pi Model 3B	10,000	3.7 *	1.8	4.4	0.844	106
ODROID-xU	12,000	8.3	2.7	13.9	0.599	49.1
Jetson TX-1	20,000	16.0	2.1	13.4	1.20	26.7
pi-cluster	48,000	15.5	71.3	93.1	0.166	7.75
2 core Intel Atom S1260	20,000	2.6	18.6	22.1	0.149	4.33
16 core AMD Opteron 6376	40,000	122	167	262	0.466	30.5
16 core Intel Haswell-EP	80,000	428	58.7	201	2.13	107

Figure 4: Performance (FLOPS) and power summary for the various ARM boards, with three x86 systems shown for comparison

[Michael F. Cloutier and Weaver, 2016]

Conclusion

- The overall run-time performance improvement from a single-node, single- core serialized implementation to sixteen core, four-node fully distributed, and fully parallel implementation is approximately 2400%.
- for parallelized implementation in a single node compared to the sixteen cores, the overall improvement in run- time using all sixteen cores is approximately 420%.
- However, this speedup can only be achieved when we are working on a significantly large input sample.

Conclusion

- The previous results were obtained with a non-optimized multiplication with a complexity of $O(n^3)$
- The performance can be improved with the use of optimized code for a raspberry cluster based on the General Matrix Multiplication Optimized for Raspberry Pi Clusters repository (https://github.com/alvarezpj/gemm-raspberrypi-cluster)

Thanks!

References I

[Corbilla, 2016] Corbilla, J. (2016).
Raspberry pi 3 cluster test.
.

[Dubey and Kagdi, 2020] Dubey, P. and Kagdi, A. (2020).
Run time analysis of matrix multiplication using raspberry pi cluster supercomputer.

[Michael F. Cloutier and Weaver, 2016] Michael F. Cloutier, C. P. and Weaver, V. M. (2016).
A raspberry pi cluster instrumented for fine-grained power measurement.
.