ParCFD'2020
32^{nd} International Conference on Parallel Computational Fluid Dynamics
May-11-13 2020, Nice FRANCE

CPU AND GPU PARALLEL EFFICIENCY OF ARM BASED SINGLE BOARD COMPUTING CLUSTER FOR CFD APPLICATIONS

Bastien Di Pierro*, Sarah Hank†

*Lyon Claude Bernard Lyon 1 University
Polytech Engineering School
69622 Lyon, France
e-mail: bastien.di-pierro@univ-lyon1.fr

† Direction générale de l'armement
Essais en vol
13800 Istres
e-mail: sarah.hank@intradef.gouv.fr

Key words: MPI-CUDA, ARM-based SBC, CFD Applications

Abstract. This paper examines the parallel efficiency of an ARM-based single computing cluster made of 16 Raspberry Pi 4 and 8 Nvidia Jetson Nano Single Board Computer, considering both CPU and GPU parallel implementation of CFD applications. It is found that the parallelization scales up to 16 Raspberry Pi 4 cluster and 8 Nvidia Jetson Nano one. Moreover, it is shown that the computation time on these architectures is smaller than the one a computing station with 16 Intel cores and one Nvidia Quadro K5000 GPU.

1 INTRODUCTION

Since the advent of computing and numerical methods, many studied have turned to Computational Fluid Dynamics (CFD) to solve fluid mechanics problem. Indeed, the uprise of computing resources and the democratization of prallelization technique have made the CFD efficient in solving complex configurations. However, the increase in CPU relative performance is much faster than that of memory and according to the "roofline model", many numerical algorithms are then limited by memory capabilities.
Recent development in scientific computing has made the GPGPU very interesting for scientific purposes, especially since the creation of the CUDA paradigm and the CUDA-aware MPI, allowing the use of multiple GPU on multiple nodes [7]. However, the MPI-CUDA implementation sometimes seems disappointing since MPI directives are a bottleneck for data exchange between GPUs.
Another evolution in computer architectures is the appearance of ARM processors (especially the 64-bit ARMv8 version) with their energetic performances. A recent study [5] show that these architectures are adapted to the CFD, showing a very good parallel efficiency.
This paper proposes to examine the parallel performance of a low consumption computing cluster, made from ARM-based Single Board Computer (SBC). The parallel CPU

	CPU Performance	GPU Performance	Memory Bandwidth
RPI4(4 cores)	9.6 Gflop/s	—	4.8 GB/s
NJN(4 cores, 1 GPU)	9.0 Gflop/s	7.368 Gflop/s	7.5 GB/s
REF(16 cores, 1 GPU)	250 Gflop/s	89.27 Gflop/s	12.4 GB/s

Table 1: Measure of computing and memory performance for SBC and REF architecture

implementation is done with a Raspberry Pi 4 cluster and GPU with Nvidia Jetson Nano boards.

2 HARDWARE SETUP AND NUMERICAL METHODS

To establish the parallel efficiency of an ARM base single board (SBC) computing, two kinds of SBC are used. The cluster is built with Raspberry PI 4 boards (Quad core Cortex A-72 64 bit @ 1.5GHz CPU and 4GB SDRAM-LPDDR4) for CPU parallel algorithm (denoted as "RPI4") and Nvidia Jetson Nano (Quad core Cortex A-57 64 bit @ 1.4GHz CPU, Maxwell GPU with 128 CUDA cores and 4GB SDRAM-LPDDR4 shared) for GPU parallel algorithm (denoted as "NJN"). The parallel efficiency of such SBC is estimated by comparison with a classical computing workstation (Bi-Socket Octo core Intel(R) Xeon(R) CPU E5-2680 @ 2.70 GHz CPU, 256 GB DDR4 RAM, NVIDIA Quadro K5000 1536 CUDA cores 4 GB GDDR5 RAM) and denoted as "REF". 16 RPI4 and 8 NJN are connected to a single switch on a gigabit network.
Raw performances are calculated from dedicated libraries: High Performance Linpack (HPL) for floating point operations on CPU [1], HPL-GPU for floating point operations on GPU [2] and STREAM for memory bandwidth measurement [3]. Table 1 shows the measured performances with these libraries. All tests are performed with FP 64 double precision floating point operations.

Two classical numerical methods in computational fluid dynamics are used here to estimate the efficiency of such architecture. The first one a first order finite volume Godunov scheme with HLL flux solver [4] for solving Euler's equations

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} &= 0, \\ \frac{\partial \rho u}{\partial t} + \frac{\partial \rho u^2 + p}{\partial x} &= 0, \\ \frac{\partial \rho e}{\partial t} + \frac{\partial \rho e u + p u}{\partial x} &= 0, \end{aligned} \tag{1}$$

and ideal gas state law is considered (test case 1). The second algorithm solves the Korteweg-De Vries equation

$$\frac{\partial u}{\partial t} + 6u\frac{\partial u}{\partial x} + \delta \frac{\partial^3 u}{\partial x^3} = 0, \tag{2}$$

with a globally second order leapfrog finite difference scheme [5] (test case 2). Both algorithms are parallelised with MPI library only using a decomposition domain method and

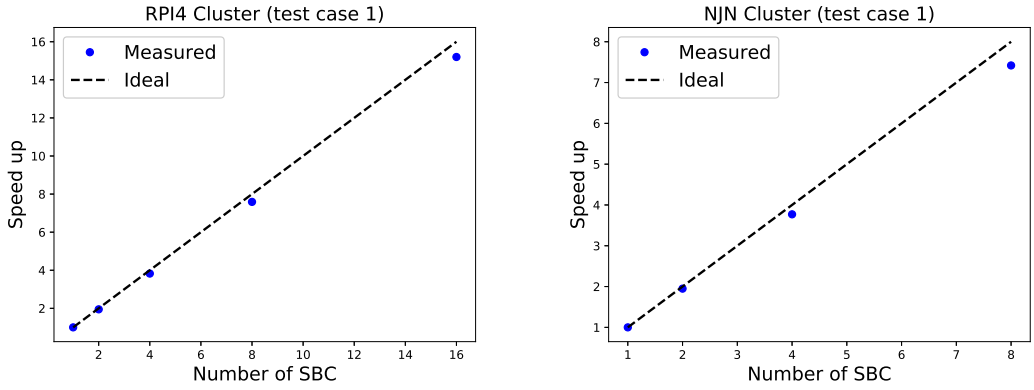

Figure 1: Speedup of the ARM based SBC cluster. Left: RPI4 SBC, right: NJN SBC.

with a hybrid CUDA-MPI domain decomposition (using CUDA-aware MPI implementation).

3 NUMERICAL PERFORMANCES

The tests presented here were done with 10^9 cells and about 10^6 time steps. For a given configuration, the computation time of 1 RPI4 node is very close to the computation time of 1 NJN node, according to table 1. Figure 1 shows the speedup of the present cluster configuration for both CPU and GPU implementations. One can see that the parallel efficiency is about 0.95 up 16 RPI4 and 8 NJN SBC.

To estimate the efficiency of the present cluster, the computation time ratio between the SBC cluster and the reference (baseline) computing station is measured, under the same conditions as figure 1. This is shown on figure 2. The reference (computing station) computations are performed with 16 cores or with the CPU. It appears that for the CPU efficiency, the RPI cluster is approximately three times slower with the same number of cores (4 nodes). This is because the algorithm has an arithmetic intensity of 1. With reference to the roofline model, the computation time is then driven by the bandwidth of the architecture (see table 1). Moreover, one can see that the MPI-CUDA implementation for multiple GPU reach a parallel efficiency of 0.95 up to 8 NJN SBC. The same behaviour is observed for test case 2. Since the computation time on RPI4 and NJN are comparable for a given computation, it is therefore assumed that the performance of the NJN cluster would be higher than that of the reference calculator beyond 12 boards.

4 CONCLUSIONS

This paper examines the parallel efficiency of ARM-based single board computer with both CPU and GPU algorithms. It is found that the RPI4 cluster has a parallel efficiency over 0.95 up to 64 cores, while the NJN ones reach the same efficiency up to 8 boards (1024 CUDA cores). It is also shown that this computing cluster has better performances than a classical computing station for a total consumption of 240 Watt.

The authors aknowledge “LyonCalcul”, “Fédération Lyonnaise de Modélisation et Science

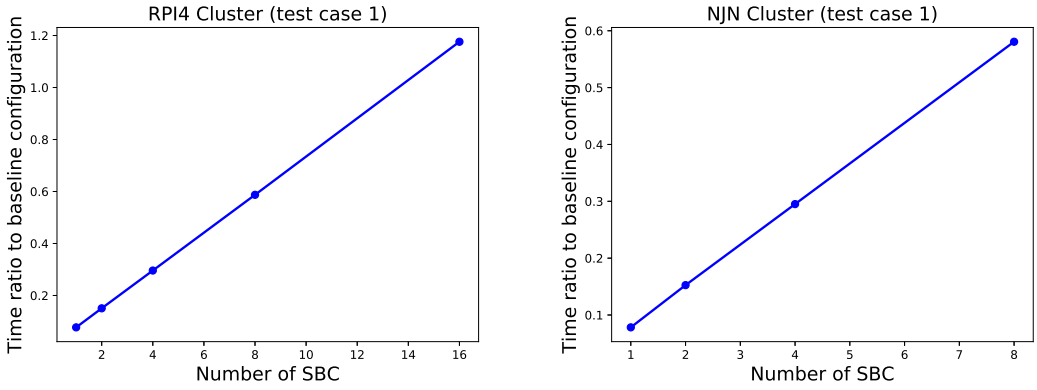

Figure 2: Time ratio to the "reference" configuration. Left: RPI4 cluster (reference with 16 cores), right: NJN cluster (reference with Quadro K-5000).

Numérique", the Mechanical department of Lyon university and the mechanical department of Polytech Lyon engeneering school for providing numerical facilities.

REFERENCES

[1] A. Petitet and R. C. Whaley and J. Dongarra and A. Cleary, HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers (2018), *https://www.netlib.org/benchmark/hpl/*

[2] D. Rohr and M. Bach and M. Kretz and J. Bornschein and D. Demidov, High Performance Computing Linpack Benchmark (HPL-GPU) (2015), *https://github.com/davidrohr/hpl-gpu*

[3] John D. McCalpin, Memory Bandwidth and Machine Balance in Current High Performance Computers, *IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter* (1995) 19–25, *http://www.cs.virginia.edu/stream/ref.html*

[4] E.F. Toro, Riemann solvers and numerical methods for fluid dynamics, *Springer* (2009)

[5] N.J. Zabusky and M.D. Krustal, Interaction of 'solitons' in a collisionless plasma and the recurrence of initial states, *Physics Review Letter* (1965), **15(6)**:240-243

[6] G.Oyarzun and R.Borrella and A.Gorobets and F.Mantovani and A.Oliva, Efficient CFD code implementation for the ARM-based Mont-Blanc architecture, *Future Generation Computer Systems* (2018) **79(3)**:786-796

[7] J. Lai and H. Yu and Z. Tian and H. Li, Hybrid MPI and CUDA Parallelization for CFD Applications on Multi-GPU HPC Clusters, *Scientific programming* (2020) **8862123**