Chapter 15
On Performance of Commodity Single Board Computer-Based Clusters: A Big Data Perspective

Basit Qureshi and Anis Koubaa

15.1 Introduction

Big data technologies are becoming ever more popular and are currently a focus of both science and industry. The amount of data generated by scientific as well as business applications has increased manifolds in the last few years. A key framework for processing large datasets is the MapReduce framework which allows data to be divided into fixed-size chunks that are processed in parallel on the cloud infrastructure. Several open source MapReduce frameworks have been developed in the last years with the most popular one being Hadoop. Hadoop has been deployed on physical servers across data centers around the globe and continues to provide the realization of on-demand resource availability, scalability with reliability for big data analyses. Figure 15.1 shows the coupling of various technologies for big data analysis in cloud computing infrastructure.

A leading motivation for cloud computing is the reduction of installation and operational cost for small businesses and enterprises. On the other hand, it is immensely important for students in universities to be exposed to real cloud computing infrastructure. Indeed, universities and academic institutions need to provide hands-on experience in this area, which means that universities need

B. Qureshi (✉)
Prince Sultan University, Riyadh, Saudi Arabia
e-mail: qureshi@psu.edu.sa

A. Koubaa
Prince Sultan University, Riyadh, Saudi Arabia

Gaitech Robotics, Shanghai, China

CISTER, INESC-TEC, ISEP, Polytechnic Institute of Porto, Porto, Portugal
e-mail: akoubaa@psu.edu.sa

R. Mehmood et al. (eds.), *Smart Infrastructure and Applications*,
EAI/Springer Innovations in Communication and Computing,
https://doi.org/10.1007/978-3-030-13705-2_15

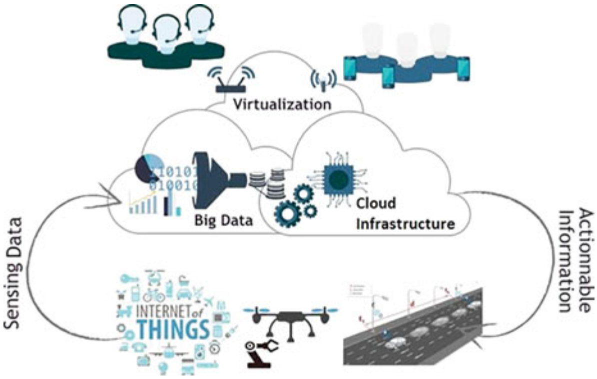

Fig. 15.1 Role of cloud infrastructure in big data analysis [2]

to provide access to a suitable cloud computing infrastructure that can be used for experimentation, research, and teaching. Setting up cloud infrastructure in universities could be a very costly endeavor [24]. Although most universities do not reveal the actual costs of setting up and running the infrastructure, the cost of Ukko Cloud Computing Cluster with 240 Dell PowerEdge M610 nodes, each with 32 GB of RAM and 2 Intel Xeon E5540 2.53 GHz quad-core CPUs at University of Helsinki Finland was reported to be over 1 million Euros [13]. Expedient, a private cloud data center construction organization for small businesses, estimates the cost of installation of a tier III data center with ten racks to be upwards of 1 million US Dollars [12].

In order to build a low-cost effective cloud computing cluster with low energy consumption requirements resulting in near-zero carbon footprint, researchers have investigated the use of SBCs. Indeed, an SBC is a complete computer built on a single circuit board that incorporates a microprocessor(s), memory, I/O as well as multitude of other features required by a functional computer [3]. Typically, an SBC is ideally priced at (35–80 US$), with power requirements set to be as low as 2.5 W and designed in small form factors comparable to a credit card or pocket size. These computers are portable and are capable of running a wide range of platforms including Linux distributions, Unix, Microsoft Windows, Android, etc. A cluster of single board computers has very limited resources and cannot compete with the performance of higher value systems. But despite these drawbacks, useful application scenarios exist, where clusters of single board computers are a promising option. This applies in particular to small- and medium-sized enterprises as well as for academic purposes like student projects or research projects with limited financial resources.

The Beowulf cluster created at Boise State University [7] was perhaps the earliest attempt at creating a cluster consisting of multiple nodes of SBCs. This cluster is composed of 32 Raspberry Pi Model B computers and offers an alternative in case if the main cluster is unavailable. The Bolzano Raspberry Pi cloud cluster experiment implemented a 300 node Pi cluster [8]. The main goal of this project was to study the process and challenges of building a Pi cluster on such a large scale.

The Iridis-Pi project implemented a 64 node Raspberry Pi cluster [9]. Tso et al. [10] built a small-scale data center consisting of 56 RPi Model B boards. The Glasgow Raspberry Pi Cloud offers a cloud computing testbed including virtualization management tools. Whitehorn [11] presented the first ever implementation of a Hadoop cluster using five Raspberry Pi Model B nodes. In 2016, C. Baun in [14] presented the design of a cluster geared towards academic research and student scientific projects building an eight-node Raspberry Pi Model 2B cluster. All of these works demonstrate constructing a cluster using SBCs at an affordable cost to researchers and students. However, none of these works provide detailed performance analysis of computing tasks, memory, storage utilization, and network throughput. Indeed effective Hadoop deployment depends on efficient utilization of resources available onboard cluster nodes as well as network traffic management. The lack of performance evaluation of SBC-based cloud computing clusters as well as energy efficiency provides motivation for this work.

In this chapter, we present a detailed study on design and deployment of two SBC-based clusters using Raspberry Pi Model 2 B and HardKernel Odroid Model Xu-4. The objectives of this study are in three folds: (1) To provide a detailed analysis of the performance of Raspberry Pi and Odroid XU-4 SBCs in terms of power consumption, processing/execution time for various tasks, storage read/write as well as network throughput; (2) To study the viability and cost-effectiveness of the deployment of SBC-based Hadoop clusters against virtual machine-based Hadoop clusters deployed on personal computers and (3) To contrast the power consumption and performance aspects of SBC-based Hadoop clusters for Big Data Applications in academic research. To this end, three clusters were constructed and deployed for extensively studying the performance of individual SBCs as well as a cluster deployment to provide a detailed comparison. Furthermore, Hadoop was deployed on these clusters to study the performance aspects of the environment using popular and widely used performance benchmarks. Power consumption, task execution time, I/O read/write latencies as well as network throughput were studied. In addition to the above, we provide analysis of energy consumption in the clusters, the energy efficiency, and cost of operating these clusters. Results from this study show that it is possible to deploy a cost-effective Hadoop cluster with reasonable performance for low yield workloads; however for larger workloads, the operation cost would significantly increase.

The contribution of this chapter is as follows:

- Design and compact layout for two clusters using SBCs are presented in addition to a PC-based cluster running in the virtual environment. Performance evaluation of task execution time, storage utilization, network throughput as well as power consumption are detailed.
- Popular Hadoop benchmark programs such as Pi Computation, Wordcount, TestDFSIO, TeraGen, and TeraSort are executed on these clusters and results are compared against a virtual machine-based cluster using workloads of various sizes.

The remainder of this chapter is organized as follows. Section 15.2 presents related works with details on the ARM-based computing platforms used in this study as well as a review of recent applications of SBCs in high-performance computing and Hadoop-based environments. Section 15.3 presents the design and architecture of the RPi, Xu20, and HDM Clusters used in this study. Section 15.4 deals with a comprehensive performance evaluation study of these clusters based on popular benchmarks. Section 15.5 provides details on the deployment of Hadoop environment on these clusters with a detailed presentation of performance aspects of Hadoop benchmarks for the clusters. Section 15.6 provides summary and discussion followed by conclusions in Sect. 15.7.

15.2 The Single Board Computers

Advanced RISC Machine (ARM) is a family of Reduced Instruction Set Computing (RISC) architectures for computer processors that are commonly used nowadays in tablets, phones, game consoles, etc. [4]. The ARM is the most widely used instruction set architecture in terms of quantity produced [6]. Since October 2011, the ARM has started to support 64-bit address space and instruction set in the ARM v8 architecture. Currently, ARM Cortex cores architecture is popular and widely used in smartphones, single board computers, etc. An SBC is a complete computer built on a single circuit board. An SBC incorporates a microprocessor(s), memory, I/O as well as host of other features required by a functional computer. While keeping the manufacturing costs to the lowest (25–80 US$), various companies have developed SBCs in small form factors comparable to a credit card or pocket size. These computers are capable of running a wide range of platforms including Linux distributions, Unix, Microsoft Windows, Android, etc. In what follows, we briefly describe the two popular SBCs using ARM-based CPUs and their features.

The Raspberry Pi Model 2B The Raspberry Pi Foundation [1] developed a credit card-sized SBC called Raspberry Pi (RPi). This development was aimed at creating a platform for teaching computer science and relevant technologies at the school level. Raspberry Pi 2B version was released in February 2015 improving the previous development platform by increased processor speed, larger onboard memory size as well as newly added features. Figure 15.2 shows RPi Model 2B. Table 15.1 summarizes the hardware specifications of RPi Model 2B. Although the market price, as well as the cost of energy consumption of an RPi, is low, the computer itself has many limitations in terms of shared compute and memory resources. Raspberry Pi uses a 32-bit quad-core ARM Cortex A7 processor clocked at 0.7 GHz with 256 KB L2 cache memory, which is shared with the GPU. While it is possible to overclock the processor and tune the performance, the results may reduce the overall lifespan of the computer. For data storage, RPi relies on solid state flash memory. The SD memory reads and writes in 128 KB blocks of data, i.e., even for reading/writing one byte, the entire block of memory needs to be read from

Fig. 15.2 Raspberry Pi 2 B

Table 15.1 Features of Raspberry Pi Model 2B and HardKernel Odroid Xu-4

	RPi Model 2B	Odroid XU-4
Processor (CPU)	0.9 GHz quad core ARM Cortex-A7	Samsung Exynos5 Octa ARM Cortex-A15 (@ 2.0 GHz) and Cortex-A7 (@1.3 GHz) CPUs
GPU	Broadcom Video Core IV Multimedia Graphics co-processor	Mali T628 Open GL 3.0
Onboard RAM	256 KB L2 cache 1 GB SDRAM at 400 MHz	2 GB LPDDR3 at 933 MHz
Ethernet/Network	10/100 MB Ethernet RJ45 Jack	10/100/1000 MB Ethernet RJ45 Jack
Storage	Micro SD Card	Micro SD Card and eMMC 5.0 flash storage
Audio/Video	3.5 mm jack and HDMI	HDMI (standard) supports 1080p video
Power Consumption	3.2 W (idle) 3.8 W (under load)	2.5 W (idle) 4.5 W (under load)
USB Ports	4 USB 2.0	1x USB 2.0, 2× USB 3.0
Released	February 2015	2015
Price (US$)	35$	79 $

or written to. Furthermore, the lifespan of the SD card is reduced significantly with very frequent write operations. In summary, the RPi is a very affordable platform with low cost and low energy consumption [3, 4]. The major drawback is the compute performance. Recent experiments in distributed computing have shown that this can be rectified by building a cluster of many RPi computers. Further details about configuration in the cluster would be provided in the next section.

The Hardkernel Odroid platform ODROID-XU-4 [5] is a newer generation of single board computers offered by HardKernel. Offering open source support, the

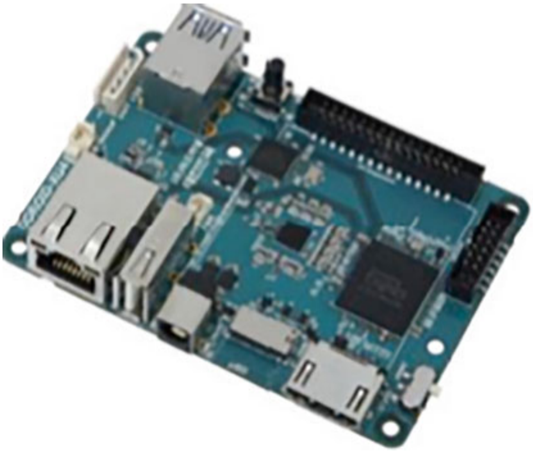

Fig. 15.3 Hardkernel Odroid XU-4

board can run various flavors of Linux, including Ubuntu 15.04, Ubuntu MATE, Android 4.4 Kit Kat, and 5.0 Lollipop. XU-4 uses Samsung Exynos5 Quad-core ARM Cortex™-A15 Quad 2 GHz and Cortex™-A7 Quad 1.3 GHz CPUs with 2 Gbyte LPDDR3 RAM at 933 MHz. The Mali-T628 MP6 GPU supports OpenGL 3.0 with 1080p resolution via standard HDMI connector. Two USB 3.0 ports, as well as a USB 2.0 port, allows faster communication with attached devices. The power-hungry processor demands 4.0 A power supply with power consumption of 2.5 W (idle) and 4.5 W (under load). By implementing the eMMC 5.0, the ODROID C1 and XU-4 boast improved I/O transfer speeds over Class 10 SD card flash memories. XU-4 comes with an onboard heat sink as well as a fan. With heavy computation loads, the temperature can increase resulting in increased power consumption due to cooling. We noticed that the temperature doubled under increased computation stress resulting in the constant running of the fan creating excessive noise. Odroid XU-4 priced at $79 is slightly expensive compared to Raspberry Pi 3B; nevertheless, the improved processing power although demanding more power provides tradeoff with improved performance, task execution time as well as better I/O read and write operations. Table 15.1 shows a summary of Odroid XU-4 SBC (Fig. 15.3).

The low-cost aspect of an SBC makes it attractive for students as well as researchers in academic environments. As pointed out in the literature, it is possible to deploy a Hadoop cluster using SBCs such as Raspberry Pi computers. Although the Raspberry Pi computers are cheap and widely available, the limitations in terms of processing power, available onboard memory and reliance on SD cards for external storage with slow I/O operations, yield performance with much to be desired. Thanks to increased interest in SBCs, newer single board computers with better design and faster operations speeds are becoming available. It remains to be seen how the improved SBCs perform when deployed in Hadoop clusters. In this chapter, we present a detailed study on design and deployment of Hadoop on two SBC-based clusters using Raspberry Pi Model 2 B as well as HardKernel Odroid

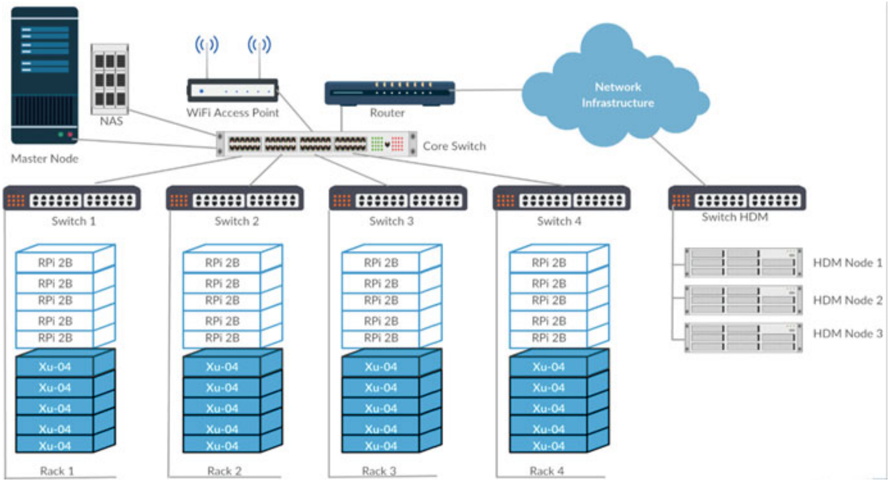

Fig. 15.4 Network topology diagram for RPi, Xu20, and HDM clusters

Model Xu-4. The Odroid XU-4 is an SBC with the faster processor, larger onboard memory, and faster I/O storage.

15.3 Design and Architecture of the DM-Clusters

This section presents the architecture and configuration of the clusters deployed in this experimental study. For the purpose of benchmarking cluster performance as well as comparatively analyzing their performance, we built three clusters.

The first cluster, called RPi Cluster, is composed of 20 Raspberry Pi Model 2B Computers connected to a network. The second cluster, called Xu-20, is composed of 20 Odroid XU-4 devices in the same network topology. The third cluster HDM is composed of four regular PCs running Ubuntu in the virtual environment using VMware Workstation [28]. To maintain similarity in network configuration, all the clusters follow the same star topology with a 24-port Giga-bits-per-second smart managed switch acting as the core of the network as can be seen in Fig. 15.4. Each node (RPi, XU-4, or PC) connects a 16-port Ethernet switch that connects to the core switch. Currently, five nodes connect to each switch allowing further scalability of the cluster. The master node, as well as the uplink connection to the Internet through a router, is connected to the core switch. The current design allows easy scalability with up to 60 nodes connected in the cluster that can be extended up to 300 nodes. Table 15.2 presents a summary of the cluster characteristics.

Table 15.2 Configuration of the DM-Clusters

	RPi Cluster	Xu20 Cluster	HDM Cluster
Master Node	Intel i7 at 3.00 GHZ 64Bit Win 10	Intel i7 at 3.00 GHZ 64Bit Win 10	Intel i7 at 3.00 GHZ 64Bit Win 10
Number of Data Nodes	20	20	4
Slave Node Device	Raspberry Pi Model 2 B	HardKernel Odroid Xu-4	Intel i7 at 3.00 GHZ 64Bit Win 10
Data Node Clock Speed	1000 MHz	2000 MHz	3000 MHz
OS	Raspbian OS	Ubuntu MATE 15 OS	Ubuntu 14.4 LTE
Storage (GB)	16 GB	32 GB	40 GB
Storage Medium	Class 10 SD Card	eMMC 5.0 module	Kingston Solid State Disk (SSD)
RAM	856 MB (available)	1024 MB (available)	3 GB (available)
Virtual Machine	Only Master Node runs OS in VM		All nodes on VM

15.3.1 Components and the Design of the DM-Clusters

Each cluster is composed of a set of components including SBCs, power supplies, network cables, storage modules, connectors, and cases. Each SBC is carefully mounted with storage components. All the Raspberry Pi computers are equipped with 16 GB Class-10 SD cards for primary bootable storage. The Odroid XU-4 devices are equipped with 32 GB eMMCv5.0 modules and can be seen in Fig. 15.3. All the SBCs are housed in a compact layout racks using M2/M3 spacers, nuts, and screws. The racks are designed to house 5 SBCs per rack for easy access and management. Figure 15.5a shows the Raspberry Pi computers organized in racks with 5 computers per rack, Fig. 15.5b shows the Odroid XU-4 computers organized in racks with 5 computers per rack.

Currently, each Raspberry Pi computer is individually supplied by the 2.5 A power supply; each Odroid XU-4 computer is supplied by a 4.0 A power supply that provides ample power for running each node. All the power supplies are connected to the Wattsup Pro .net power supply meter for measuring power consumption. These power meters are then connected to a voltage regulator connected to the main supply. The Wattsup Pro .net power meter can be seen in Fig. 15.6a.

Each SBC's network interface is connected to a Cat6e Ethernet cable through the RJ-45 Ethernet connector. All Ethernet cables connect to the 16-port Cisco switches which connect to a Gigabit Core switch. An Internet router, as well as the Master PC running Hadoop namenode, is connected to the network. Figure 15.6b shows the network connectivity. The HDM Cluster is composed of four PCs all connected in the same network topology as of the other clusters. Each PC is equipped with an Intel i7 4th Gen Processor with 3.0 GHz Clock speed, 8 GB RAM, and 120 GB Solid State Disk Drive for storage. Each PC is equipped with a 400 W power supply

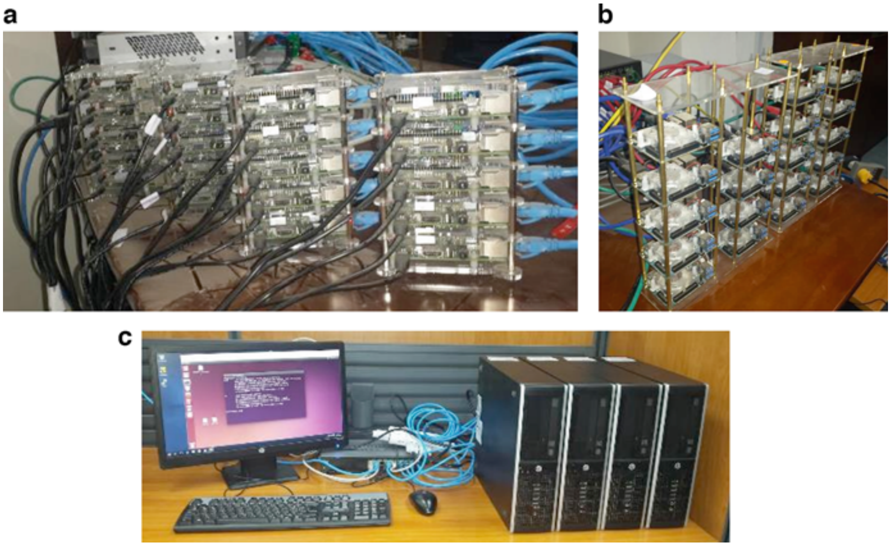

Fig. 15.5 Hardware installation; (**a**) The RPi Cluster composed of 20 RPi Model 2B computers; (**b**) The Xu20 Cluster composed of 20 Odroid XU-4 computers; (**c**) The HDM Cluster composed of 4 Intel 7, 3.0 GHz PCs

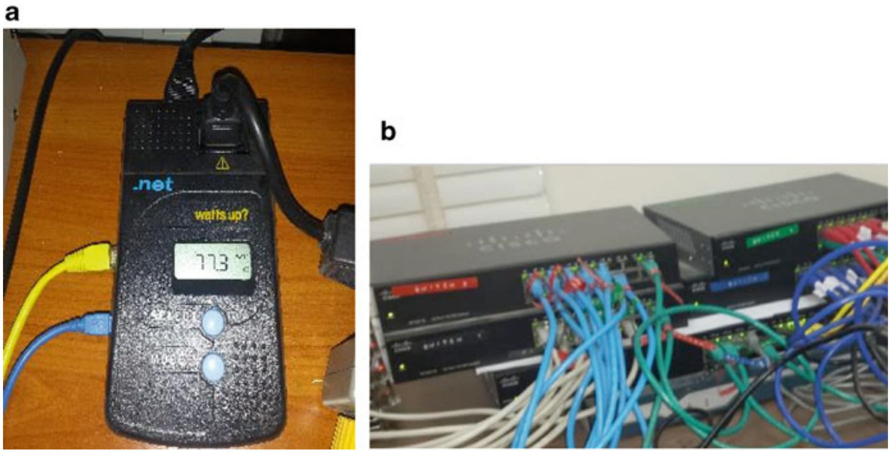

Fig. 15.6 (**a**) Wattsup Pro .net power meter (**b**) Cisco Core switch, Cisco Internet Router, and 4 × 16 port switches

and connects to the Ethernet Switch. Figure 15.5c shows the HDM Cluster. The purchase cost of all components of the RPi, Xu20, and HDM Clusters was \$1300, \$2700, and \$4200, respectively. The Network and Power reading equipment cost is approximately \$450.

15.3.2 Raspbian and Ubuntu MATE Image Installation

For the RPi Cluster, we built the RPi Image. The Raspbian OS image is based on Debian that is specifically designed for ARM processors [29]. Using Raspbian OS for RPi is easy with minimal configuration settings requirements. Each individual RPi is equipped with a SanDisk Class 10, 16 GB SD card capable of up to 45 MB/s read as well as up to 10 MB/s write speeds available at a cost of US$15. We created our own image of the OS which was copied on the SD cards. Additionally, Hadoop 2.6.2 is installed on the Image with Java JDK 7 for ARM platform. When ready, these SD cards are plugged into the RPi systems and mounted. The Master node is installed on a regular PC running an Ubuntu 14.4 virtual machine on Windows 10 as the host operating system.

For the Xu20 Cluster, we built another image based on Ubuntu MATE 15.10. Ubuntu MATE is an open source derivate of the Ubuntu Linux distribution with MATE desktop. HardKernel provides Ubuntu MATE 15.10 pre-installed on the Toshiba eMMCv5.0 memory module which is preconfigured for Odroid XU-4 single board computers at a price of US$43. The eMMCv5.0 is capable of reading and write speeds of 140 MB/s and 40 MB/s, respectively. Apache Hadoop 2.6.2 along with Java JDK 7 for ARM platform was installed on the image. These modules were inserted into eMMC socket on the Odroid XU-4 boards and connected to the network. Similar to the RPi Cluster, the Hadoop master node was installed on a regular PC running Ubuntu 14.4 VM.

The final cluster HDM is composed of four PCs all connected in the same network topology as of the other clusters. A virtual machine in the VMware workstation was built to run Hadoop 2.6.2 with Java JDK 7 for 64-bit architecture. One of the VMs serves as the master node and runs Hadoop namenode only. The rest of the VM run the data nodes of the cluster.

15.4 Performance Evaluation of DM-Clusters

In this section, we present a performance evaluation study of DM-Clusters in terms of energy consumption, processing speed, storage read/write, and networking.

15.4.1 Energy Consumption Approximation

Energy consumption in data centers is a major concern for green cloud computing research. The Greenpeace [26] in 2012 estimated the global energy consumption for data centers to be over 31 GW. Recently, the NRDA [27] estimated in 2013, in the USA alone, the data centers consumed 91 billion kiloWatts hours (kWh) of energy, which is estimated to increase by 141 billion kWh every year until 2020, costing

Table 15.3 Power consumption of clusters in idle and stress modes with power cost per year

	Idle mode		Stress mode	
	Power consumption (E) (W)	Power cost in USD	Power consumption (E) (W)	Power cost in USD
RPi CLUSTER (20 NODES)	34.1	$ 14.94	46.4	$ 20.33
XU20 CLUSTER (20 NODES)	56.2	$ 24.63	78.7	$ 34.49
HDM CLUSTER (4 NODES)	108.4	$ 47.51	197.7	$ 86.66

Table 15.4 CPU execution time (s) for individual nodes with n threads

			CPU execution time with n threads				
	CPU cores	Clock rate GHz	1	2	4	8	16
Raspberry Pi 2B	4	1.0	448.2	225.1	113.8	113.7	113.7
Odroid Xu-4	8	2.0	83.3	41.68	25.33	17.66	18.02
Intel i7 4th Gen	4	3.0	8.51	4.272	2.22	2.27	2.23

businesses $13 billion annually in electricity bills and emitting nearly 100 million metric tons of carbon pollution per year. Resource over-provisioning and energy non-proportional behavior of today's servers [25] are two of the most important reasons for high energy consumption of data centers. On the other hand, use of low-end computers is increasingly becoming popular due to low cost and low energy consumption. In this section, we analyze the power consumption of SBCs used in this study.

The energy consumption for the DM-Clusters was measured using the Wattsup Pro .net power meters. These meters provide consumption in terms of Watts for 24 h a day and log these values in local memory for accessibility. To estimate the approximate power consumption over a year, we measured the power consumption in two modes, Idle mode and stress mode for each DM-Cluster. In idle mode, the clusters were deployed without any application/task running for a period of 24 h. In stress mode, the clusters ran a host of computation intensive applications for a period of 24 h. Observing the logs, the upper-bound wattage usage within a period of 23 h was taken as power consumption in the idle mode as well as the stress mode. Table 15.3 shows the power consumption for DM-Clusters in idle and stress modes.

The cost of energy for the cluster is a function of power consumption per year and the cost of energy per kiloWatts hour [23]. An approximation of energy consumption cost per year (C_{y}) can be given by Eq. (15.1) where E is the specific power consumption for an event for 24 h a day and 365.25 days per year. The approximate cost for all the clusters computed based on values given in Table 15.4, whereas the cost per kilowatt-hour (P) is assumed to be 0.05 US$.

$$C_{\mathrm{y}} = E \times 24\frac{\text{hour}}{\text{day}} \times 365.25\frac{\text{days}}{\text{year}} \times \frac{P}{\text{kWh}} \tag{15.1}$$

The Bolzano Experiment [8] reports Raspberry Pi cluster built using Raspberry Pi Model B (first generation) where each node is consuming 3 W in stress mode. In RPi Cluster, the Raspberry Pi Model 2B consumes slightly less power with 2.4 W in stress mode. We observe that this slight difference in power consumption is due to the improved design of the second-generation Raspberry Pi. The Cardiff Cloud testbed reported in [30] compared two Intel Xeon-based servers deployed in the data center with each server consisting of 2 Xeon e5462 CPU (4 cores per processor), 32 GB of main memory, and 1 SATA disk of 2 TB of storage each. The researchers in this study used similar equipment to measure power consumption as presented in this study. Their work reports that each server on average consumes 115 W and 268 W power in idle and stress modes, respectively. The power consumption for the RPi Cluster with 20 nodes is 5 times better compared to a typical server in a cluster.

In a scenario where the RPi Cluster runs an application in stress mode (i.e., 46.4 W) for the whole year, the cost for power usage is approximately $20.33. For Xu20 and HDM Clusters, the yearly cost would be $34.49 and $86.66, respectively. It is clear that using low-cost low-power devices enable a greener computing environment in terms of energy consumption.

15.4.2 CPU Performance

In this section, we analyze the performance of the DM-Clusters using various benchmark. The objective of this study is to investigate and compare the processing speed of the three platforms under consideration to understand their intrinsic performance.

The benchmark suite Sysbench[1] was used to measure the CPU performance. Sysbench provides benchmarking capabilities for Linux and supports testing CPU, memory, File I/O, mutex performance in clusters. We execute the Sysbench benchmark[2] testing each number up to value 10,000 if it is a prime number for *n* number of threads [22]. Since each computer has a quad-core processor, we run the sysbench CPU test for 1, 2, 4, 8, and 16 threads. We measure the performance of this benchmark test for Raspberry Pi Model 2B, Odroid XU-4 as well as Intel i7 fourth-generation computers used in the three DM-Clusters. Table 15.4 shows the average CPU execution time for nodes with *n* threads.

As can be seen from Fig. 15.7, all the tested devices had four cores, the CPU execution times scale well with the increased number of threads. Sysbench test runs

[1] https://wiki.gentoo.org/wiki/Sysbench

[2] Using `sysbench --test=cpu --cpu-max-prime=10000 --num-threads=`*n* `run`

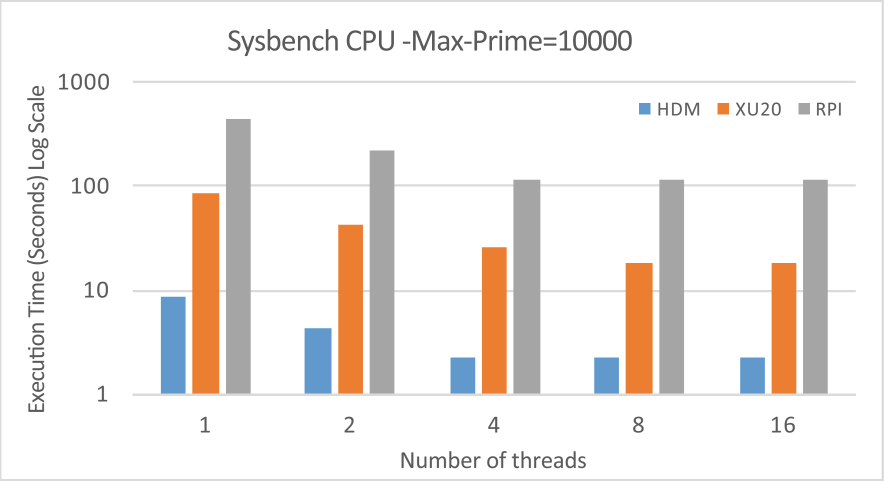

Fig. 15.7 Sysbench CPU execution times for SBCs (logarithmic scale)

with $n = 2$ and $n = 4$ threads significantly improve the execution times performance for all processors by 50%. With $n = 8$ and $n = 16$ threads, the test results yield almost similar execution times with little improvement in performance. It can also be noted from Fig. 15.7 that the execution times for Odroid XU-4 are 10 times better as compared to Raspberry Pi Model 2B. The increased number of threads does not provide gain in performance of Odroid XU-4 over Raspberry Pi; furthermore, the execution time for Raspberry Pi is further extended with larger n. The HDM Cluster nodes run 4.42 times faster compared to Odroid Xu-4. These results clearly illustrate the handicap of SBC onboard processors when compared to a typical PC.

The Raspberry Pi Model 2B allows the user to overclock the CPU rate to 1200 MHz, in our experiments with the over-clocked CPU we did not observe significant improvement using the sysbench benchmark.

15.4.3 Storage Performance

Poor storage read/write performance can be a bottleneck in clusters. Compared to server machines, an SBC is handicapped in terms of availability of limited storage options. SBCs are typically restricted to external storage connected through the USB interface with bootable flash disks or SD cards are primary storage devices. In this section, we compare the storage performance of the DM-Clusters nodes and analyze the performance of three different mediums for storage.

The small scale of the SBCs of Odroid Xu-4, as well as Raspberry Pi Model 2B, provides few options for external storage. Both SBC is equipped with SD Card Memory slots that come with bootable versions of Linux distributions. In addition

Table 15.5 Read and write throughput (KB/s) for individual devices in the clusters using FIO[a]

	Read throughput (KB/s)		Write throughput (KB/s)	
	Buffered	Non-buffered	Buffered	Non-buffered
Raspberry Pi 2B with 16 GB Class 10 SanDisk SDCard	7135	4518	2701	2537
Odroid Xu-4 with 32 GB eMMCv5.0 Module	14,318	13,577	6421	5118
Intel i7 4th Gen with 120 GB SanDisk Solid State Disk	164,521	93,608	96,987	62,039

[a] Measured using fio –name = randread –ioengine = libaio –iodepth = 1 –bs = 4 k –size = 512 M –runtime = 240

to the SD Card Memory slot, the Odroid XU-4 is also equipped with eMMCv5.0 connector. Apart from these, both devices are equipped with USB 2.0 interfaces with Raspberry Pi having 4, XU-4 having only one. The XU-4 is also equipped with two USB 3.0 ports for faster data transfer. Additional storage devices can be mounted using these USB ports. The Raspberry Pi's were equipped with 16 GB SanDisk Class 10 SD cards, whereas the XU-4 devices were equipped with 32 GB eMMC memory cards. Both of these memory cards were loaded with bootable Linux distributions. For comparison purposes, we used 128 GB SanDisk Solid State Disks on the HDM Cluster machines and used flexible IO (FIO) which is commonly used to benchmark IO performance of storage in various Linux distributions.

FIO[3] allows benchmarking of sequential read and write as well as random read and write with various block sizes. NAND memory is typically organized in pages and groups with sizes 4, 8, or 16 Kilobytes. Although it is possible for a controller to overwrite pages, the data cannot be overwritten without having to erase it first. The typical erase block on SD cards is typically 64 or 128 KB. In newer SD cards, the small number of erase blocks are combined into larger allocation units or segments with a size 4 MB. The controllers of the SD cards implement a translation layer maintaining the mapping and translation of virtual and physical memory addresses. As a result of these design features, the random read and write performance of SD cards depends on the erase block, segment size, the number of segments, and controller cache for address translations.

Table 15.5 shows the comparison of buffered and non-buffered random read and write from all the three devices with block size 4 KB. FIO was used to measure the random read and write throughput with eight threads each working with a file of size 512 MB with a total 4 GB of data. These parameters were set specifically to avoid buffering and caching in RAM issues which are managed by the underlying operating systems that can distort the results, i.e., the data size (4 GB) selected is larger than the onboard RAM available on these devices. As can be seen from Table 15.5, the read throughput (buffered) of Odroid with eMMC memory is at least twice as fast as the Class 10 SD card on the Raspberry Pi whereas the non-buffered

[3] https://www.openhub.net/p/fio

read is more than three times better. Similarly, for buffered write operations, Odroid XU-4 with eMMC module throughput is more than twice better when compared to the Class 10 SD card in Raspberry Pi. Table 15.5 also shows the comparison of the throughput of the SSD Storage on the PC in the HDM Cluster against the throughput of these devices. The buffered read throughput for SSD storage is at least 10 times better compared to eMMC module in Odroid XU-4 computers whereas the buffered write throughput of SSD storage is 15 times better. These experimental observations clearly imply the benefit of using SSDs with higher throughput when compared to Class 10 SD cards as well as eMMC v5.0 memory modules. When deployed in a distributed environment such as Hadoop that requires frequent read and write operations, the SD cards with slower read/write throughput can increase the task completion rate. On the other hand, faster memories such as eMMC or SSD Drives can have a pivotal role in improving performance for the applications.

15.4.4 Network Performance

When data are being processed in a cluster, servers need to transfer data with a certain amount of network bandwidth for the data to be delivered quickly and processed efficiently. If the network cannot allocate bandwidth properly, the speed of delivering and processing data will suffer because of unnecessary network congestion among many other reasons. Major factors that can have an impact on data processing and task execution time includes not only the speed of CPU, size of main memory, the speed of storage I/O, but also the allocation of network resources. Figure 15.4 shows the network topology for various networking components in the three clusters. In this section, we provide the comparative analysis of network performance using network throughput and latency using various payload sizes of data over the TCP protocol using Linux-based benchmark tools.

The network performance was measured using the popular Linux-based command line tool iperf v3.13 with the NetPIPE benchmark version 3.7.2. Through various sets of runs, iperf states the network throughput to be 82–88 Mbits per second for the RPi and XU20 Clusters. NetPIPE [15, 16], on the other hand, provides more details considering performance aspects for network latency, throughput, etc. over a range of messages with various payload size in bytes. For this study, we executed the benchmark within the clusters for various payload sizes over the TCP end-to-end protocol. The NPtcp, NetPIPE benchmark using TCP protocol, involves running transmitter and receiver on two nodes in the cluster. In our experimentation, we executed the receiver on the cluster namenode with 1000 KB as maximum transmission buffer size for a period of 240 ms. The transmitter was executed on the individual SBCs one by one.

As can be seen from Fig. 15.8, the network latency for all clusters with small payload is almost similar. As the payload increases, we observe a slight increase in network latency between the three clusters. On the other hand, we observe a spike in throughput at message size 1000 bytes; this indicates that the smaller a message is, the more is the transfer time dominated by the communication layer overhead.

For larger messages, the communication rate becomes bandwidth limited by a component in the communication subsystem that may include the data rate at the network link, utilization of the communication medium at the time, or the traffic on the network switch. In the context of Hadoop installation in the cluster, the namenode frequently communicates with data nodes using heartbeat messages with smaller payloads, whereas the data blocks typically larger than the 128 MB need to be copied from one data node to another. We present detailed network performance using Hadoop benchmarks in the next section.

We also note that the throughput at the HDM Cluster is lowest compared to the other clusters, this is mainly due to the proximity of the HDM Cluster. This cluster is physically located in a farther area and requires an extra switch to connect to the namenode of the clusters. The physical proximity and the longer distance yields degradation in throughput performance for the HDM Cluster. Contrasting the performance of XU-4 and RPi SBCs, we note the visible difference in throughput between the two, this is due to the poor overall Ethernet performance of the Raspberry Pi probably caused by design. On the Raspberry Pi, 10/100 Mbps Ethernet controller is a component of the LAN9512 controller which contains the USB 2.0 hub as well as the 10/100 Mbit Ethernet controller. On the other hand, the Odroid XU-4 is equipped with an onboard Gigabit Ethernet controller which is part of the RTL8153 controller. The coupling of faster Ethernet port with high-speed USB 3.0 provides better network performance. Figure 15.8 shows comparatively the throughput on the Xu20 Cluster is 1.52 times better when compared to the RPi Cluster.

15.5 Performance of Hadoop Benchmark Tests on Clusters

Apache Hadoop is an open source framework that provides distributed processing of large amounts of data in a data center. The Hadoop framework scales well for thousands of machines allowing processing of petabytes of data. It offers high availability options for detection and recovery from failures in software as well as hardware thus making it a very reliable distributed ecosystem. Hadoop uses the map/reduce programming model for big data processing over multiple nodes. The map/reduce model is composed of two steps, the map step performs filtering and sorting of data, the reduce step provides further processing of data from map step usually summarizing the outcomes. Depending on the application, the map/reduce tasks can be parallelized. Hadoop 2 introduced Yet Another Resource Negotiator (YARN) as a new resource management layer allowing for better resource management and monitoring.

On all three clusters, Hadoop version 2.6.2 was installed due to the availability of YARN daemon which improves the performance of the map/reduce jobs in the cluster. To optimize the performance of these clusters, yarn-site.xml and Mapred-

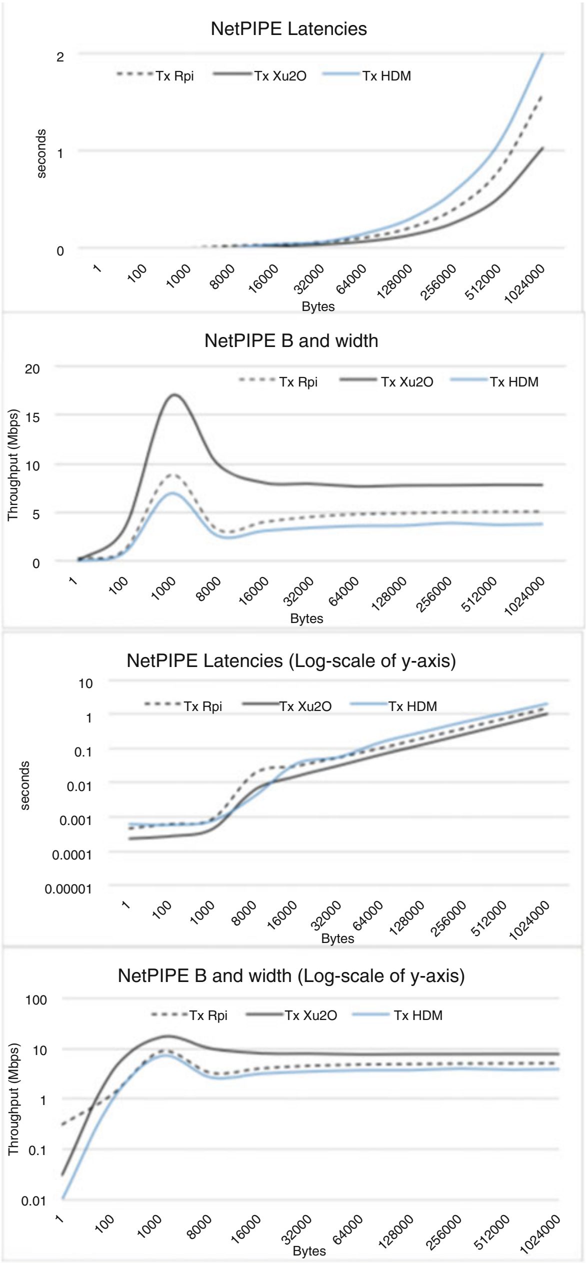

Fig. 15.8 NetPIPE benchmark results for all clusters considering latencies and bandwidth with data size in terms of bytes on the *x*-axis

Table 15.6 Properties in mapred-site.xml

Property	Value
yarn.app.mapreduce.am.resource.mb	852
mapreduce.map.cpu.vcores	1
mapreduce.reduce.cpu.vcores	1
mapreduce.map.memory.mb	852
mapreduce.reduce.memory.mb	852
mapreduce.input.fileinputformat.split.minsize	8 MB

Table 15.7 Properties in YARN-site.xml

Property	Value
yarn.nodemanager.resource.memory-mb	1024
yarn.nodemanager.resource.cpu-vcores	1
yarn.scheduler.minimum-allocation-mb	256
yarn.scheduler.maximum-allocation-mb	852
yarn.scheduler.minimum-allocation-vcores	1
yarn.scheduler.maximum-allocation-vcores	1
yarn.nodemanager.vmem-pmem-ratio	2

Table 15.8 Properties in hdfs-site.xml

Property	Value
dfs.replication	2

site.xml were configured with 852 MB of resource size allocation. The primary reason for this is the limitation in the RPi Model 2B which has 1 GB of onboard RAM out of which 852 MB is available; the rest is used by the Operating System as well as the CPU Memory Bus. The default container size on the Hadoop Distributed File System (HDFS) is 128 MB. Each SBC node was assigned a static IPv4 address based on the configuration and all slave nodes were registered in the Master node. YARN and HDFS containers and interfaces could be monitored using the web interface provided by Hadoop. Tables 15.6, 15.7, and 15.8 provide details of important configuration properties for the Hadoop environment. It must be noted that maximum memory allocation per container is 852 MB; this is set on purpose so that the performance of all clusters could be measured and contrasted. Additionally, the replication factor for HDFS is 2 which means only two copies of each block would be kept on the file system.

These clusters were tested extensively for performance using Hadoop benchmarks for Quasi-Random Pi generation and word count applications.

15.5.1 The Pi Computation Benchmark

Hadoop provides its own benchmarks for performance evaluation over multiple nodes. One of the simplest benchmarks is the computation of the value of π using Quasi-Monte Carlo Method and map/reduce. We execute the compute Pi program

Table 15.9 CPU execution times for Pi computation benchmark on clusters

		Average CPU execution times (s)		
Map tasks	Samples	RPi Cluster	Xu20 Cluster	HDM Cluster
10	10^3	98.469	37.37	22.86
10	10^4	99.13	37.69	20.5
10	10^5	97.902	36.97	18.92
10	10^6	100.629	37.87	25.35
100	10^3	465.675	49.62	17.84
100	10^4	461.4	49.7	19.35
100	10^5	470.264	49.43	20.12
100	10^6	486.48	49.89	21.24

that computes exact m binary digits of the mathematical constant π using a quasi-Monte Carlo method and MapReduce. The precision value m is provided at the command prompt with values ranging from 1×10^3 to 1×10^6 increased at an interval of 1×10^1. Each of these is run against a number of map tasks set at 10 and 100. We study the impact of the value of m versus the number of map tasks assigned and compute the difference in time consumption (execution time) for completion of these tasks. Each experiment is repeated at least 10 times for significance of statistical analysis. In this experimentation, the Pi computation benchmark's goal is to observe the CPU bound workload of all the three clusters. Table 15.9 shows average CPU execution times for various runs of the Pi computation program with 10 and 100 map tasks. Figure 15.9a, b show the box-whisker plot with upper and lower quartiles for each sample set with 10 and 100 map tasks. With 10 maps, the average execution time for RPi Cluster with 10 + E06 number of samples is 100.8 s, whereas for XU20 and HDM Cluster the average execution time is 38.2 and 25.1 s, respectively. As the number of maps increases to 100, we observe significant degradation in performance of RPi Cluster with average execution time at 483.7 s for 10 + E06 number of samples. Comparatively, the execution times for Xu20 and HDM Clusters are 50.1 and 21.8 s, respectively. This clearly shows the significant difference in the computation performance between the RPi Cluster and the Xu20 Cluster. Figure 15.9c shows the ratio of performance degradation of RPi and XU20 Clusters compared to HDM Cluster for Pi program CPU execution times with 10 and 100 maps.

15.5.2 *The Wordcount Benchmark*

The Wordcount program contained in the Hadoop distribution is a popular micro-benchmark widely used in the community [15]. The Wordcount program is representative of a large subset of real-world MapReduce jobs extracting a small amount of interesting data from a large dataset. The **Wordcount** program reads text files and counts how often words occur within the selected text files. Each mapper takes a line

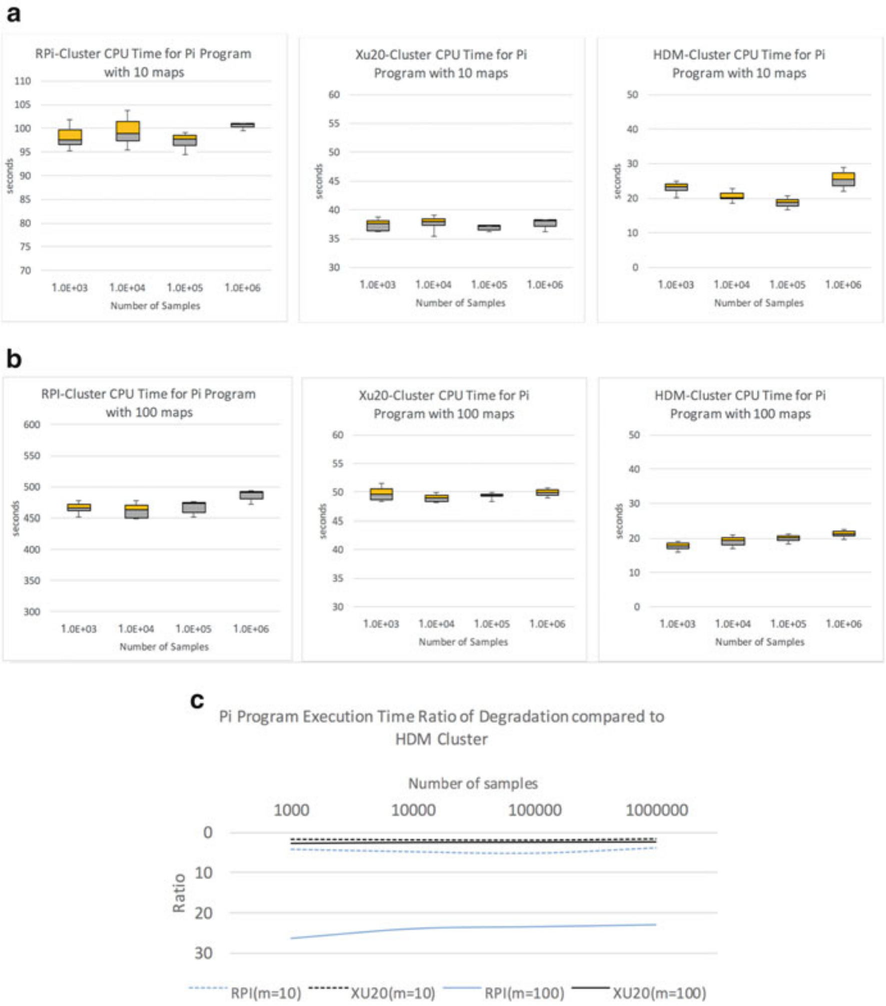

Fig. 15.9 CPU execution time versus number of *m* samples for computation of Pi benchmark in all clusters with (**a**) 10 maps (**b**) 100 maps (**c**) ratio of execution time for Rpi and Xu20 Cluster against HDM Cluster

from a text file as input and breaks it into words. It then emits a key/value pair of the word and a count value. Each reducer sums the count values for each word and emits a single key/value pair containing the word itself and the sum that word appears in the input files. For optimization, the reducer also imitates as a combiner on the map outputs to reduce the amount of data sent across the network by combining each word into a single record. In our experimentation, the Wordcount benchmark's goal is to observe the CPU bound workload of the three clusters.

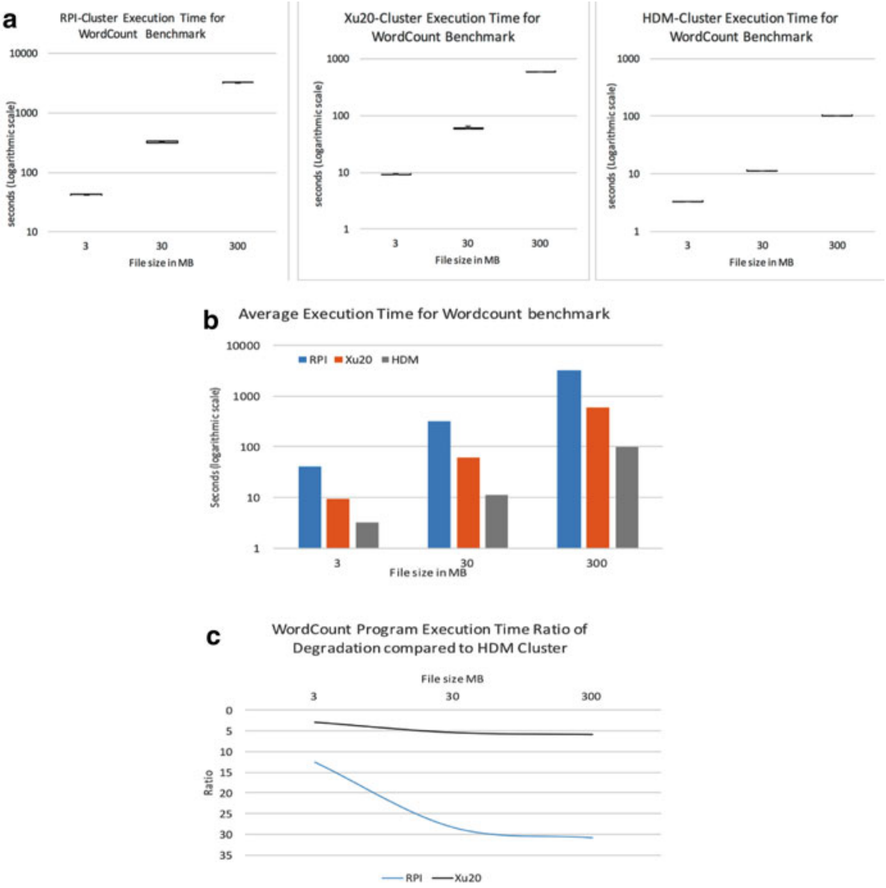

Fig. 15.10 (**a**) CPU execution time for the Wordcount benchmark for all clusters against input files sizes 3, 30, and 300 MB. (**b**) Average execution time for Wordcount on all clusters. (**c**) Ratio of performance degradation against HDM Cluster

In our experimentation, we generated three large files of sizes 3, 30, and 300 Megabytes, respectively. The Wordcount program was executed in the Hadoop environment for all the three clusters. Depending on the initial dataset size, Wordcount generates mappers for every HDFS container associated with the input files. For the datasets provided Wordcount generated a single mapper, four mappers, and 36 mappers, respectively. Each experiment was run on the clusters separately at least 10 times for statistical accuracy. Figure 15.10a shows the performance of CPU execution time, for the Wordcount benchmark for all clusters against input files sizes 3, 30, and 300 MB, in seconds on a logarithmic scale. Again, RPi Cluster performs four times worse (Fig. 15.10c) compared to Xu20 Cluster and 12.5 times worse compared to HDM Cluster due to the relatively slower processor clock speeds, slower memory read/write, and network latency. The effect of the slower clock speed of the processor in the RPi nodes is clearly evident with smaller input file sizes of 3 MB. The average execution times of RPi and XU20 should be comparable since Wordcount generates only one mapper for each run resulting in a single container read by the mapper; however, the slower storage throughput with SD cards adds to the overall latency. With input file size 30 MB, Wordcount generates four mappers reading four containers from different nodes in the cluster, increasing the degree of parallelization thus reducing the overall CPU execution time.

Finally, with 300 MB as input file size, we observe execution time performance correlating with smaller datasets although the increased numbers of mappers should have improved the overall execution time. This is due to the fact that Wordcount generated 36 mappers for the job since there are only 19 nodes available (1 reserved for reducing job) in the Xu20 and RPi Clusters, the rest of the mappers would queue for the completion of previous mapper jobs resulting in increased overhead and reduced performance. Figure 15.10b shows the average CPU execution times for all three clusters with different input file sizes. Furthermore, we observe that the Wordcount program executing on Xu20 is 2.8 times slower compared to HDM Cluster for file size 3 MB. For larger file sizes, Xu20 is over five times slower compared to HDM Cluster. RPi Cluster, on the other hand, performs worse from 12 to 30 times slower compared to the HDM Cluster.

15.6 Discussion

In this chapter, we conducted an extensive study with varying parameters on the Hadoop cluster deployed using ARM-based single board computers. An overview of popular ARM-based SBCs Raspberry Pi, as well as HardKernel Odroid XU-4 SBCs, was presented. The work also detailed the capabilities of these devices and tested them using popular benchmarking approaches. Details on requirements, design, and architecture of clusters built using these SBCs were provided. Two SBC clusters based on RPi and XU-4 devices were constructed in addition to a PC-based cluster running in the virtual environment. Popular Hadoop benchmark programs such as Wordcount, TestDFSIO, and TeraSort were tested on these clusters and their performance results from the benchmarks were presented. This section presents a discussion of our findings and main lessons learned.

- Deployment of Clusters: Using low-cost SBCs is an amicable way of deploying a Hadoop cluster at a very affordable cost. The low-cost factor would encourage students to build their own clusters and to learn about installation, configuration, and operation of a cloud computing testbeds. The cluster also provides a platform for developers to build applications, test, and deploy in public/private cloud environments. The small size of the SBCs allows installation of up to 32 nodes in a single module for a 1 U rack mounting form factor. Further to this, these small clusters can be packaged for mobility and can be deployed in various emergency and disaster recovery scenarios.

- Hadoop configuration optimization: Section 15.4(a) comparison of CPU execution times using sysbench for both SBCs considered in this chapter. XU-4 devices in Xu20 Cluster perform better due to higher clock speeds and larger onboard RAM. Using sysbench we observed that increasing the number of cores in the CPU intensive benchmark, the execution time decreased. In Hadoop deployment configuration, we noticed that increasing the number of cores resulted in RPi Cluster to be irresponsive for heavier workloads. On the other hand, XU-4 boards performed well with an increased number of cores (up to 4). A possible explanation for this behavior is the Hadoop deployment setting where each core is assigned 852 MB of memory, additional cores running Hadoop tasks would have to request virtual memory from the slower SD cards resulting in poor performance leading to responsiveness. Although RPi devices are equipped with quad-core processors, due to the poor performing SD cards, it is inadvisable to use multiple cores for Hadoop deployment.

  In Hadoop deployment, not all of the available RAM onboard SBCs was utilized since we only allow one container to execute in YARN Daemon. The size of the container was set to 852 MB which is the maximum available onboard memory in a Raspberry Pi node. This was intentionally done in order to study the performance correlation with the similar amount of resources in both kinds of SBCs. In further experimentation, we notice that XU-4 devices are capable of handling up to four containers in each core at a time, resulting in better performance. We will further investigate the performance of all cores on the SBCs using Hadoop deployment of larger replication factors and a large number of YARN containers executing per node. On the HDM Cluster running Hadoop environment in a virtual machine, we note that higher replication factors resulted in a large number of errors due to replication overheads resulting in Hadoop stuck in an unrecoverable state. The SD cards are slow and the storage provided per node in the cluster is distributed over the network degrading the overall performance of the cluster. Raspberry Pi with slower network port at speeds 10/100 Mbps also poses a considerable degradation in network performance.

  On the other hand, Xu20 Cluster performed well comparatively with faster eMMC memory modules onboard the XU-4 devices. The SSD storage used in the HDM Cluster on the PCs provide the best performance in terms of storage IO although the network configuration of this cluster was a hindrance. We will consider using Network Attached Storage (NAS) attached to the master node where every rack would have a dedicated volume managed by Logical Volume Manager (LVM) that would be shared by all SBCs in the clusters.
- Power efficiency: A motivation for this study was to analyze the power consumption of SBC-based clusters. Due to their small form factor, SBC devices are inherently energy efficient, it is worth investigating if a cluster comprising of SBCs as nodes provides a better performance ratio in terms of power consumption and dollar cost. Although we did not measure the FLOPs per watt efficiency of either of our clusters, we notice wide inconsistencies in energy consumption results reported in the literature [17–22] for similar devices. This is due to the power measurement instruments varying results and inconsistencies in the design

of power supplies. RPi, as well as XU-4 devices, has no standard power supply and micro-USB-based power supply with unknown efficiency can be used. Since the total power consumed in the cluster is small, the efficiency of power supplies can make a big difference in overall power consumption. Nonetheless, WattsUp meters were effectively used to observe and analyze the power utilization for each task over the period of its execution in all experimentation.

It is difficult to monitor and normalize the energy consumption for every test run over a period of time. It was observed that the MapReduce jobs, in particular, tend to consume more energy initially while map tasks are created and distributed across the cluster, while a reduction in power consumption is observed towards the end of the job. For the computation of power consumption, we assumed max power utilization (stress mode) for each job, during a test run in the clusters. Based on the power consumption of each cluster and the dollar cost of maintaining the clusters (given in Table 15.4), a summary of average execution times, energy consumption, and cost of running various benchmark tasks is presented in Table 15.10.

15.7 Conclusions and Future Work

In this chapter, we investigated the Hadoop deployment on low-cost low-power ARM-based single board computers. We consider two kinds of popular platforms Raspberry Pi 2B and Odroid XU-4 using ARM Cortex Processors connected in a tree network topology. We perform various performance benchmarking tests on these two platforms testing performance metrics for CPU task execution times, removable memory modules, energy consumption, and network performance. We present the power consumption and estimate cost of power per year. Further to this, we configure and deploy Hadoop 2.6.2 on these clusters considering the limited capabilities of the SBCs. Various CPU-intensive and IO-intensive Hadoop benchmarks including computation of Pi using Monte Carlo method, Wordcount, TestDFSIO, and TeraSort were executed and performance results obtained. We carried out an in-depth analysis of energy consumption of these clusters and correlate performance with low-cost low-energy capabilities of these clusters.

Results from these studies show that while SBC-based clusters are energy efficient overall, the operation cost to performance ratio can vary based on the workload. In terms of power efficiency, for smaller workloads, the Xu20 Cluster outperforms the other clusters; however, with larger workloads, the Xu20 Cluster performance is comparable to HDM with the exception of TeraGen and TeraSort benchmarks. Similarly, in terms of dollar cost of operation for these clusters, the results heavily depend on execution time. For low-intensity workloads, the Xu20 Cluster outperforms the HDM Cluster; however, the TeraGen and TeraSort heavy workloads yield poor performance for Xu20 Cluster when compared to HDM

Table 15.10 Summary of execution time, energy consumption, and cost of running per job for all benchmarks

	CPU execution time (s)			Energy consumption (W) per job			Cost (Dollars) per job		
Test parameters	RPi Cluster	Xu20 Cluster	HDM Cluster	RPi Cluster	Xu20 Cluster	HDM Cluster	RPi Cluster	Xu20 Cluster	HDM Cluster
10 maps10^3 samples	98.47	37.37	22.86	1.27E + 0	8.17E − 1	1.26E + 0	6.35E − 2	4.08E − 5	6.28E − 5
10^4 samples	99.13	37.69	20.50	1.28E + 0	8.24E − 1	1.13E + 0	6.39E − 2	4.12E − 5	5.63E − 5
10^5 samples	97.90	36.97	18.92	1.26E + 0	8.08E − 1	1.04E + 0	6.31E − 2	4.04E − 5	5.20E − 5
10^6 samples	100.63	37.87	25.35	1.30E + 0	8.28E − 1	1.39E + 0	6.48E − 2	4.14E − 5	6.96E − 5
100 maps10^3 samples	465.68	49.62	17.84	6.00E + 0	1.08E + 0	9.80E − 1	3.00E − 1	5.42E − 5	4.90E − 5
10^4 samples	461.40	49.70	19.35	5.95E + 0	1.09E + 0	1.06E + 0	2.97E − 1	5.43E − 5	5.31E − 5
10^5 samples	470.26	49.43	20.12	6.06E + 0	1.08E + 0	1.10E + 0	3.03E − 1	5.40E − 5	5.52E − 5
10^6 samples	486.48	49.89	21.24	6.27E + 0	1.09E + 0	1.17E + 0	3.14E − 4	5.45E − 5	5.83E − 5
Filesize = 3 MB	41.12	9.25	3.29	5.30E – 1	2.02E − 1	1.81E − 1	2.65E − 5	1.01E − 5	9.03E − 6
30 MB	318.75	59.75	11.22	4.11E + 0	1.31E + 0	6.16E − 1	2.05E − 4	6.53E − 5	3.08E − 5
300 MB	3131.60	588.45	101.38	4.04E + 1	1.29E + 1	5.57E + 0	2.02E − 3	6.43E − 4	2.78E − 4

Cluster. The RPi Cluster consistently was outperformed by the other two clusters regardless of the variation in workloads.

For heavier workload application, such as big data applications, due to the inefficient performance of these devices, the SBC-based clusters may not be an appropriate choice. The overall cost of operation can be expensive mainly due to the inefficient onboard SBC resources resulting in larger execution times for job completion effectively ensuing increased operation costs. It is, however, possible to tweak Hadoop configuration parameters to adjust with given resources to improve the overall performance. At the moment, we intend to use these clusters for academic research and teaching. In the future, we will consider the use of NAS for RPi Cluster to improve the storage performance since the currently installed SD card storage provides a bottleneck. We will also study the effect of replication factor and containers per node in the Xu20 Cluster to tweak the performance on that cluster. Further, we intend to study newer SBC boards deployed in similar configurations with reliable power measurement and energy consumption analysis.

Acknowledgements This work is supported by the Robotics and Internet of Things Unit at the Research and Innovations Center at Prince Sultan University.

References

1. The Raspberry Foundation. https://www.raspberrypi.org (2015). Accessed online 20 Nov 2015
2. Chaari, R., Ellouze, F., Koubaa, A., Qureshi, B., Pereira, N., Youssef, H., Tovar, E.: Cyber-physical systems clouds: a survey. Comput. Netw. **18**, 260–278 (2016)
3. Gómez, A., Cuiñas, D., Catalá, P., Xin, L., Li, W., Conway, S., Lack, D.: Use of single board computers as smart sensors in the manufacturing industry. Proc. Eng. **132**, 153–159 (2015)
4. Grigoriev, S.N., Martinov, G.M.: An ARM-based multi-channel CNC solution for multi-tasking turning and milling machines. Proc. CIRP. **46**, 525–528 (2016)
5. Fernandes, S.L., Bala, G.J.: ODROID XU4 based implementation of decision level fusion approach for matching computer-generated sketches. J. Comput. Sci. **16**, 217–224 (2016)
6. Grisenthwaite, R.: ARMv8-A Technology Preview. Online. Accessed 31 Oct 2011
7. Kiepert, J.: Creating a Raspberry Pi-Based Beowulf Cluster. Boise State University, pp. 1–17 (2013). http://coen.boisestate.edu/ece/files/2013/05/Creating.a.Raspberry.Pi-Based.Beowulf.Cluster_v2.pdf
8. Abrahamsson, P., Helmer, S., Phaphoom, N., Nicolodi, L., Preda, N., Miori, L., Angriman, M., Rikkila, J., Wang, X., Hamily, K., Bugoloni, S.: Affordable and energy-efficient cloud computing clusters: the Bolzano raspberry pi cloud cluster experiment. In: 2013 IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom), vol. 2, pp. 170–175 (2013)
9. Cox, S.J., Cox, J.T., Boardman, R.P., Johnston, S.J., Scott, M., O'Brien, N.S.: Iridis-pi: a low-cost, compact demonstration cluster. Clust. Comput. **17**(2), 349–358 (2014)
10. Tso, F.P., White, D.R., Jouet, S., Singer, J., Pezaros, D.P.: The Glasgow raspberry pi cloud: a scale model for cloud computing infrastructures. In: 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 108–112. IEEE, Philadelphia, PA (2013)
11. Whitehorn, J.: Raspberry flavored Hadoop. On-Line access: http://www.idatasci.com/uploads/1/4/6/6/14661274/jamiewhitehorn_raspberryflavouredhadoop_annotated.pdf

12. Expedient Data Center Cost Estimator. https://www.expedient.com/data-center-build-vs-buy-calculator/
13. Toor, S., Osmani, L., Eerola, P., Kraemer, O., Lindén, T., Tarkoma, S., White, J.: A scalable infrastructure for CMS data analysis based on OpenStack Cloud and Gluster file system. J. Phys. Conf. Ser. **513**, 062047 (2014). https://doi.org/10.1088/1742-6596/513/6/062047
14. Baun, C.: Mobile clusters of single board computers: an option for providing resources to student projects and researchers. SpringerPlus. **5**, 360 (2016). PMC. Web. 16 Aug 2016
15. Huang, S., Huang, J., Dai, J., Xie, T., Huang, B.: The HiBench benchmark suite: characterization of the MapReduce-based data analysis. In: International Conference on Data Engineering Workshops (ICDEW), March 2010
16. Ivanov, T., Niemann, R., Izberovic, S., Rosselli, M., Tolle, K., Zicari, R.V.: Performance evaluation of enterprise big data platforms with HiBench. In: IEEE Trustcom/BigDataSE/ISPA, Helsinki (2015)
17. Ge, R., Feng, X., Song, S., Chang, H.C., Li, D., Cameron, K.W.: PowerPack: energy profiling and analysis of high-performance systems and applications. IEEE Trans. Parallel Distrib. Syst. **21**(5), 658–671 (2010)
18. Cloutier, M.F., Paradis, C., Weaver, V.M.: Design and analysis of a 32-bit embedded high-performance cluster optimized for energy and performance. In: Hardware-Software Co-Design for High-Performance Computing (Co-HPC), 2014, New Orleans, LA (2014)
19. Psaroudakis, I., Kissinger, T., Porobic, D., Ilsche, T., Liarou, E., Tözün, P., Ailamaki, A., Lehner, W.: Dynamic fine-grained scheduling for energy-efficient main-memory queries. In: The Tenth International Workshop on Data Management on New Hardware (DaMoN'14) (2014)
20. Schöne, R., Treibig, J., Dolz, M.F., et al.: Tools and methods for measuring and tuning the energy efficiency of HPC systems. J. Sci. Program. **22**(4), 273–283 (2014)
21. Piga, L., Bergamaschi, R.A., Breternitz, M., Rigo, S.: Adaptive global power optimization for Web servers. J. Supercomput. **68**, 1088 (2014)
22. Ilsche, T., Hackenberg, D., Graul, S., Schöne, R., Schuchart, J.: Power measurements for compute nodes: improving sampling rates, granularity and accuracy. In: Sixth International Green Computing Conference and Sustainable Computing Conference (IGSC), Las Vegas, NV (2015)
23. Hackenberg, D., Ilsche, T., Schöne, R., Molka, D., Schmidt, M., Nagel, W.E.: Power measurement techniques on standard compute nodes: a quantitative comparison. In: 2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Austin, TX, pp. 194–204 (2013)
24. Divakaran, D., Le, T., Gurusamy, M.: An online integrated resource allocator for guaranteed performance in data centers. IEEE Trans. Parallel Distrib. Syst. **25**(6), 1382–1392 (2014)
25. Dalvandi, A., Gurusamy, M., Chua, K.C.: Time-Aware VMFlow Placement, Routing, and Migration for Power Efficiency in Data Centers. IEEE Trans. Netw. Serv. Manag. **12**(3), 349–362 (2015)
26. Cook, G.: How clean is your cloud? Catalysing an energy revolution. Greenpeace International, Amsterdam (2012)
27. Data Center Efficiency Assessment: Scaling up energy efficiency across the data center industry. Natural Resources Defense Council, August 2014. https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IP.pdf
28. Xiao, Z., Song, W., Chen, Q.: Dynamic resource allocation using virtual machines for cloud computing environment. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1107–1117 (2013)
29. Qureshi, B., Koubaa, A.: Power efficiency of a SBC based Hadoop cluster. In: Proceedings of EAI Smart Societies, Infrastructure, Technologies and Applications (SCITA 2017), Jeddah, Saudi Arabia, pp. 52–60, 27–29 Nov 2017
30. Conejero, J., Rana, O., Burnap, P., Morgan, J., Caminero, B., Carrión, C.: Analyzing Hadoop power consumption and impact on application QoS. Futur. Gener. Comput. Syst. **55**, 213–223 (2016)