IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 16, 2021, accepted October 4, 2021, date of publication October 15, 2021, date of current version October 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3120660

Big Data Processing on Single Board Computer Clusters: Exploring Challenges and Possibilities

EUNSEO LEE, HYUNJU OH, AND DONGCHUL PARK, (Member, IEEE)
Department of Software, Sookmyung Women's University, Seoul 04310, South Korea

Corresponding author: Dongchul Park (dpark@sookmyung.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) through the Korean Government [Ministry of Science and ICT (MSIT)] under Grant 2020R1F1A1048485, and in part by Sookmyung Women's University Research under Grant 1-2003-2006.

ABSTRACT For more than a decade, "big data" has been an industry and academia buzz phrase. Over this time, many companies adopted Apache Hadoop and Spark frameworks for their massive data storage and analysis efforts, using powerful, energy-hungry, general-purpose server as their big data processing platforms. But not all industry or academic fields want, or even need, such large systems. Moreover, capital costs aside, power consumption has also become a primary data center concern. Consequently, lower-cost, lower-power microservers have emerged as viable alternatives in many settings. Now, the latest generation Raspberry Pi (RPi), model 4B, exhibits significant computational performance improvements over its predecessors, and is presently considered a sufficiently powerful single board computer (SBC) to run many mainstream operating systems and accommodate heavy workloads. This paper reexamines SBC cluster big data processing possibilities by integrating the most powerful (presently) RPi model–the RPi 4B with 4 Gigabytes (GB) main memory. We examine external storage's performance impact on such an SBC cluster's big data processing performance by employing three different external storage solutions with measurably distinct performance characteristics. Moreover, we discuss challenges we encountered and identify further SBC cluster performance optimizations. We perform several representative big data application benchmarks and measure various key performance metrics such as execution time, power consumption, throughput, performance-per-dollars, etc. Our extensive experiments and comprehensive studies conclude this current, fourth-generation RPi has evolved to become the first generation to effectively run massive (i.e., more than 100GB) workloads in big data processing applications.

INDEX TERMS Raspberry Pi, big data, Hadoop, Spark, UFS, SBC, single board computer, cluster.

I. INTRODUCTION

Widespread high-speed Internet has exacerbated data production, driving efficient big data processing platforms in a "big data era". For the last decade, the big data has been a buzzword and various big data technology advances have made a crucial impact on our daily life as well as numerous industries [1]. Both the Apache Hadoop and Spark platforms provide the foundation for this big data revolution. Many companies employed Hadoop and Spark for storing and analyzing their big data. Thus, Hadoop and Spark have undoubtedly become representative hallmarks of big data [2], [3]. Industries have built powerful servers and clusters to best exploit these big data software platforms and they have been industries' general-purpose big data processing or analysis workhorses (Figure 1). However, because of their high cost and high power consumption, not all industries and academic fields need or can afford such powerful servers.

Today, server architecture is transitioning from general-purpose rack servers to purpose-built servers such as blade servers and microservers [4]. A principal goal of these purpose-built servers is to reduce power and space requirements. Particularly, microservers have been in the spotlight

The associate editor coordinating the review of this manuscript and approving it for publication was Ali Kashif Bashir.

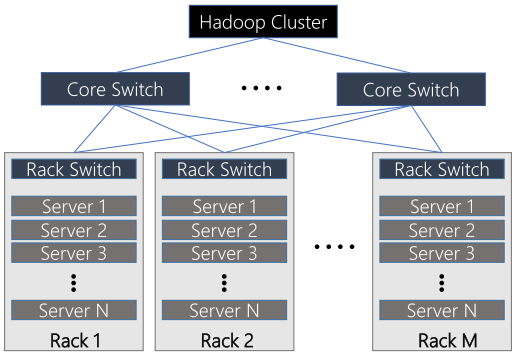


FIGURE 1. A general-purpose rack server architecture for big data processing (Hadoop cluster example).

since 2010s. Microservers are server class computers based on a system on a chip (SoC) with the goal of integrating all server motherboard functions onto the SoC except main memories and power circuits. Multiple microservers generally can be built together in a small package to construct dense data centers [4]. Thus, they typically had denser architectures than blade servers and adopted ultra-low-powered processors. They tried re-balancing computing and performance to achieve extreme energy conversation, space, and cost efficiencies [4].

A single board computer (SBC) is a complete computer built on a single circuit board. It has a microprocessor, memory, general-purpose I/O ports (GPIO), and other features for processing [3]. An SBCs' reduced power consumption and lower cost enable them to bridge the gap between controller boards and personal computers (PCs). That is, both characteristics enable SBCs to address situations where PCs are inappropriate, but where controller boards cannot meet processing requirements [5]. However, though one SBC's processing power may not be powerful enough to effectively run heavy big data processing or scientific computing applications, it is possible to link multiple SBCs together to build low-power, low-cost small clusters.

The advent of the Raspberry Pi (RPi) has significantly changed the SBC market since 2012. Even though similar SBCs (e.g., Gumstix and BeagleBone) had existed since 2003, RPi now dominates the SBC market and has become the third most popular personal computer to date, following the Apple Macintosh and the Windows PC [1], [5]. Early RPi-based clusters were mostly built to support educational challenges or practical use cases where traditional general-purpose server-based clusters would be inappropriate, such as hands-on educational and learning experiences [23]–[25]. As RPi's computing power, including CPU and RAM, evolved, some researchers have examined RPi clusters for more compute-intensive applications such as big data processing, micro data center, and edge computing [5], [6]. However, their workloads were typically not realistic for big data processing (for instance, mostly hundreds of Megabytes) because previous generation RPis were not powerful enough to accommodate 'real' big data (that is, more than hundreds of Gigabytes to Terabytes).

Compared to the previous generations, the latest generation RPi (i.e., RPi 4B) exhibits significantly improved computational capabilities by adopting a fully-redesigned quad-core CPU, up to 4GB RAM,[1] USB 3.0 ports, dual 4K video support, 1Gb Ethernet, etc. Consequently, more users started to employ this RPi 4B as a personal home computer alternative.

In this paper, we explore the challenges and possibilities of the latest generation RPi for cluster-based big data processing. We build a cluster of 5 RPi 4B nodes with a separate master node (6 nodes total) and install Apache Hadoop and Spark (please refer to table 5). We then perform several representative benchmarks (for example, Wordcount, TeraGen/TeraSort, TestDFSIO, and Pi computation) to evaluate single RPi performance as well as cluster performance. Moreover, we compare the RPi 4B cluster to both an RPi 3B cluster and a desktop PC for a more objective performance evaluation. Furthermore, we study storage media performance impact on RPi cluster-based big data processing platforms by adopting three distinctly different storage media with different performance characteristics (i.e., typical, fast, fastest storage media available for RPi 4B). To the best of our knowledge, this is the first extensive study that not only addresses storage performance impact on RPi cluster performance, but also comprehensively explores the big data processing possibilities using the most powerful, latest generation RPi-based processing cluster.

The main contributions of this paper are as follows:

A. STORAGE PERFORMANCE IMPACT STUDY

Most RPi research studies do not consider storage performance and only employ a default media card (i.e., microSD cards) with typical performance. However, this paper explores how much cluster performance improves for big data processing according to different storage media with distinctly different performance characteristics. To this end, we adopt a fast storage media card, a Universal Flash Storage (UFS) card, that exhibits up to 500MB/s read performance versus a typical microSD card (up to 95MB/s read) and the fastest microSD card (up to 170MB/s read) available to date. We evaluate their big data processing performance improvement impacts on these clusters (Section IV).

1) SUMMARIES

Adopting faster storage media is a significantly effective way to improve cluster performance, achieving 1.3× to 7.07× big data processing performance improvements.

B. EXTENSIVE PERFORMANCE EVALUATION

The RPi 4B is the most powerful RPi as of now. Its computational capabilities have been unprecedentedly improved. To explore big data processing possibilities on an RPi cluster using this processor, we perform popular Hadoop and Spark benchmarks including Wordcount, TeraGen/TeraSort,

[1] An 8GB model has been recently introduced on the market.

TestDFSIO and Pi computation. In addition, we evaluate diverse RPi performance metrics by measuring CPU, network, storage performance, and power consumption. Further, comparing a latest generation RPi cluster performance to both a previous generation RPi cluster and a desktop PC performance informs a useful intuition for an RPi 4B's objective performance (Section IV and V).

1) SUMMARIES

The RPi 4B cluster exhibited, on average, 6.65× to 9.56× improved performance than the RPi 3B cluster particularly under the Spark platform and a typical microSD card. We also observed the 5 node RPi 4B cluster exhibited close big data processing performance (on average 1.45×) to one desktop PC.

C. CHALLENGES AND SUGGESTIONS

An RPi cluster does not have abundant computing resources. Thus, big data processing on the RPi 4B cluster introduces unexpected challenges (e.g., CPU throttling due to overheating). We also provide further suggestions for RPi cluster big data processing that conserve resources (e.g., Cyclic Redundant Check (CRC) disablement and master node separation). We discuss RPi performance in terms of performance-per-watt and performance-per-dollar. Finally, we conclude our discussion section by verifying 'real' big data processing possibilities using the latest generation RPi-based cluster with 1TB data (Section VI).

1) SUMMARIES

We observed the RPi 4B's CPU throttling mechanism (due to inadequate cooling) severely reduced performance by up to 1.97×. Disabling CRC saved 7% CPU capacity. Master node separation achieved 29% I/O traffic reduction and 42.2% memory saving. Unlike previous generations, the RPi 4B finally competes with a desktop PC for big data processing in terms of performance-per-watt and per-dollar. Further, we concluded RPi 4B is the first generation RPi to handle 'real' big data successfully and efficiently (i.e., with low-cost and low-power).

The remainder of this paper is organized as follows. Section II gives a Raspberry Pi overview and describes various storage media for Raspberry Pi. In addition, it presents related and previous research studies. Section III explains our Raspberry Pi cluster architecture and configurations. Sections IV and V provide a variety of experimental results and analyses. Section VI discusses diverse challenging problems and suggestions. Section VII concludes the discussion.

II. BACKGROUND AND RELATED WORK

A. RASPBERRY PI

Raspberry Pi is a small single board computer (SBC) family developed by the Raspberry Pi (RPi) Foundation since 2012. Originally it was intended to promote teaching basic computer science in schools by inspiring students to engage in electronics and programming [3]. The Raspberry Pi's popularity has far exceeded the Foundation's expectation and it has become the third most popular personal computer, following the Apple Macintosh and the Windows PC. It is now widely adopted for various applications such as home automation, manufacturing, robotics, and mobile devices [1].

TABLE 1. Specification comparison for the Raspberry Pi model 3B and model 4B [8]. "Power" stands for power consumption.

	Raspberry Pi 3B	Raspberry Pi 4B
CPU	ARM Cortex-A53 @ 1.2GHz (4 cores)	ARM Cortex-A72 @ 1.5GHz (4 cores)
RAM	1GB LPDDR2 SDRAM	1,2,or 4GB LPDDR4 SDRAM
GPU	VideoCore IV @ 1.2GHz	VideoCore VI @ 1.5GHz
Bluetooth	Bluetooth 4.1	Bluetooth 5.0
Ethernet	100Mbps Gigabit Ethernet	Native Gigabit Ethernet
USB	4x USB 2.0	2x USB 3.0 + 2x USB 2.0
Power	1.4W(idle),3.7W(full-load)	2.7W(idle),6.4W(full-load)
SDIO speed	up to 25MB/s	up to 50MB/s
Release	February 2016	June 2019
Price	$35	$35(1GB), $45(2GB), $55(4GB)

The Raspberry Pi 4 Model B (4B) was released in June 2019 and exhibits a tremendous performance improvement over the previous model due to its full-chip redesign, the first in Raspberry Pi history: more powerful processing cores, the first graphics processor upgrade, vastly improved memory and external hardware bandwidth, including the first UBS 3.0 ports, full-speed Gigabit Ethernet, micro HDMI ports for 4K displays, up to 4GB RAM, etc. [21], [22]

Raspberry Pi is a very low-cost and low-energy consuming computing platform [19], [20]. The major historic drawback was the computing performance. Table 1 compares hardware specifications of both the Raspberry Pi 3B and 4B.

B. STORAGE MEDIA FOR RASPBERRY PI

This section describes various storage media types for Raspberry Pi. A microSD (Secure Digital) card is the standard Raspberry Pi storage media and uses an integrated microSD card slot. A Universal Flash Storage (UFS) card provides for highest Raspberry Pi 4B storage performance through its USB 3.0 port with a UFS adapter.

1) microSD CARD

The Raspberry Pi has one microSD (Secure Digital) card slot for loading an operating system and data storage. This microSD card is the most widely used Raspberry Pi storage media. The microSD card is a removable, miniaturized SD flash memory card, developed primarily for mobile devices such as smart phones that need a smaller, lighter form factor than the original SD card form factor.

There are four capacity types of SD cards and three different form factors (table 2). The four capacity types include: the original Standard-Capacity (SDSC), a High-Capacity (SDHC), an eXtended-Capacity (SDXC), and an Ultra-Capacity (SDUC) [16]. The two form factors are the original size (SD) and the micro size (microSD). Electrically

TABLE 2. SD card capacity choices. An SD card with a higher capacity class is not interoperable with host devices with a lower capacity class.

	SDSC	SDHC	SDXC	SDUC
Capacity	up to 2GB	> 2GB up to 32GB	> 32GB up to 2TB	> 2TB up to 128TB
File system	FAT 12, 16	FAT 32	exFAT	exFAT
Form factors	SD (32×24×2.1mm), microSD (11×15×1.0mm)			

TABLE 3. SD card speed classes. Speed class symbols with a number indicate minimum writing speed (MB/s).

Minimum sequential write speed (MB/s)	Original speed class	UHS speed class	Video speed class
90	-	-	V90
60	-	-	V60
30	-	U3	V30
10	C10	U1	V10
6	C6	-	V6
4	C4	-	-
2	C2	-	-

passive adapters allow smaller cards to fit and operate in devices supporting a larger card.

In addition, SD cards have three speed ratings: an original speed class, a UHS (Ultra High Speed) speed class, and a video speed class according to a minimum sustained write speed (please refer to table 3) [16]. For example, the SD card with UHS speed class U1 and U3 support a minimum write performance of 10MB/s and 30MB/s respectively. We employ both U1 and U3 class microSD cards for our evaluations.

2) UNIVERSAL FLASH STORAGE (UFS) CARD

Universal Flash Storage (UFS) is a Flash storage specification for mobile systems requiring low power consumption and high data transfer speed, such as mobile phones, consumer electronic devices, and recently automotive systems [17]. It also uses NAND Flash memory. UFS was originally developed to replace SD cards and considered an eMMC (embedded Multi-Media Card) successor that has been predominantly adopted for built-in storage in most Android mobile OS-based smart phones. Unlike eMMC that supports 104MB/s (eMMC v4.4) and up to 400MB/s (eMMC v5.0), UFS cards bring up to 600MB/s (UFS card v1.0) and 1.2GB/s (UFS card v2.0) (please refer to table 4). Unlike SD or eMMC, UFS adopted latest technologies such as command queuing, parallel and out-of-order execution, advanced asynchronous I/O protocol, etc. [17]. To study storage performance impacts on big data processing performance improvement on the clusters, we adopt this UFS card in addition to microSD cards.

C. RELATED WORK

Abrahamsson and Helmer implemented a 300 node Raspberry Pi Model B (i.e., the first generation Raspberry Pi) cluster [25] to explore the challenges of building such a large scale Raspberry Pi cluster. They described each procedure of hardware and software setup and configuration. Although this work did not provide any performance numbers, it tackled several challenges such as supplying power, connecting and housing a large scale RPi cluster, and installing and configuring system software.

TABLE 4. UFS card version comparison. Here, BW stands for bandwidth.

UFS card	Version 1.0	Version 1.1	Version 2.0	Version 3.0
Release	Mar. 2016	Jan. 2018	Sep. 2018	Dec. 2020
BW / lane	600MB/s	600MB/s	600MB/s	1200MB/s
No. of lane	1	1	2	2
Max. BW	600MB/s	600MB/s	1200MB/s	2400MB/s

The Glasgow Raspberry Pi Cloud (PiCloud) Raspberry Pi clusters, a miniature cloud data center, provided a low-cost cloud computing testbed [23] to overcome limited software simulation. PiCloud consists of 56 Raspberry Pi model B devices which are divided into 4 racks with 14 Raspberry Pis each. This work presented the design and implementation of the PiCloud to emulate a cloud data center, from its overall architecture to the software stack on each individual machine. However, they did not provide detailed performance evaluations.

Kaewkasi and Srisuruk built a Hadoop cluster of 22 Cubieboards each of which is an ARM Cortex-A8 processor-based single board computer (SBC) [26]. They conducted Apache Spark over Hadoop Distributed File System (HDFS) experiments on the Hadoop cluster. They claimed the cluster could process a 34GB Wikipedia article file in acceptable time, while generally consumed the power of 0.061-0.322kWh for all benchmarks. The authors concluded cluster processing bottlenecks resulted from both I/O and CPU's processing power deficiencies.

Baun built a cluster of 8 Raspberry Pi Model B nodes for academic purposes such as student projects or scientific projects because typical clusters consisting of servers, workstations or personal computers as nodes may not be appropriate for academic projects [27]. He conducted performance experiments including computation time, I/O, and network throughput, and concluded the SBC cluster's performance and energy-efficiency could not compete with higher-value systems. However, the RPi clusters were still useful for academic purposes and research projects due to lower purchase costs and operating costs.

Morabito investigated Docker container virtualization performance on a low-power SBC cluster to provide insights for optimally using SBCs during virtualized instance execution [28]. He quantified the overhead introduced by the virtualization layer under compute-intensive and networking-intensive workloads by deploying Docker-based containers on the systems. The author concluded employing container virtualization technologies on SBCs produced an almost negligible impact in terms of performance when compared

to native (non-virtualized) execution [28]. Noronha *et al.* conducted a similar study [29]. They deployed Docker containers on various embedded microprocessor systems including the Raspberry Pi and made an extensive experiments on CPU, memory, and network performance. They also came to a similar conclusion to the aforementioned Morabito's work.

Johnston *et al.* investigated diverse SBC clusters including the Raspberry Pi and explored their use cases [5]. They outlined the broad domains where SBC clusters might be deployed including education, edge compute, expendable compute, resource constrained compute, portable clusters, and next-generation data centers. They claimed SBCs could run mainstream operating systems and workloads. Furthermore, SBC clusters became a game changer in pushing application logic towards the network edge. However, this work did not provide performance evaluations.

Recently, Qureshi and Koubaa addressed the energy efficiency of a small scale data center by building two different ARM-based clusters (RPi 2B and Odroid XU-4 platforms) of 20 SBC nodes and one regular cluster of 4 desktop PCs [3]. They deployed Apache Hadoop on the clusters and performed extensive testing to analyze the clusters' performance by adopting popular benchmarks measuring task execution time, memory and storage utilization, network throughput, and energy consumption. Their studies showed, for lower workloads, the Odriod XU-4 cluster outperformed other clusters in terms of cost-effectiveness and power efficiency. For heavy workloads, it consumed 2.41× more energy than the PC cluster. They claimed the RPi cluster exhibited poor performance results compared to the other clusters for all performance benchmarks. However, this is mainly because they employed a very old generation RPi.

We found diverse research studies on SBC clusters. However, no research studies specifically investigated storage media performance impacts on cluster performance for big data processing or adopted the Raspberry Pi 4B–the most recent and the most significantly upgraded generation Raspberry Pi model.

III. CLUSTER ARCHITECTURE AND CONFIGURATIONS

Figure 2 exhibits the overall Raspberry Pi (RPi) cluster architecture for our big data processing platform. Our single board computer (SBC) cluster is composed of five Raspberry Pi Model 4B (RPi 4B) and one separate master node (for cluster configurations, please refer to table 5). Each node (the five RPis and the master node) connects to a Gigabit Ethernet (GbE) switch via each node's RJ45 Gigabit Ethernet (GbE) port that fully supports a native Gigabit network connection.

To provide readers with meaningful hints, we also built another cluster (i.e., a 5 node RPi 3 model B cluster) and compared its performance with our RPi 4B cluster's performance to identify performance improvement. Please note that since the RPi 3B does not support USB 3.0 ports and limits its maximum SDIO (Secure Digital Input Output) bandwidth to 25MB/s, it is inappropriate to evaluate various

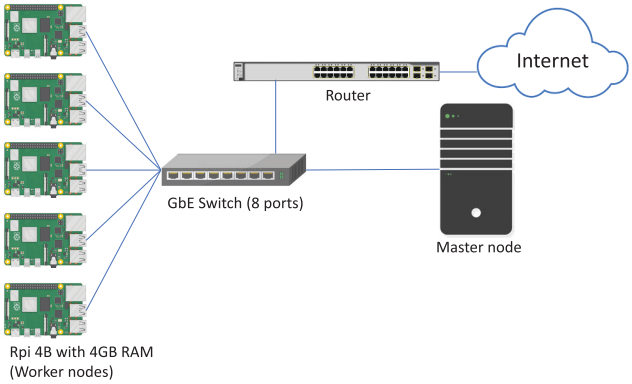


FIGURE 2. Our latest generation Raspberry Pi 4B cluster architecture. The Raspberry Pi 3B cluster configuration is identical.

TABLE 5. Cluster configurations. Note: in addition to the RPi 4B cluster, we built a 5 RPi 3B node cluster for reference and evaluation purposes.

	Master node: Desktop	Worker node: RPi 4B
CPU	Intel i5-9600K@3.7GHz (6 cores)	ARM Cortex-A72@1.5GHz (4 cores)
RAM	8GB DDR4	4GB LPDDR4
Storage	500GB SSD	MicroSD (32, 64GB), UFS (256GB)
Node no.	1 node	5 nodes
Network	1G bit Ethernet RJ45 Jack, 8 port GbE switch	
OS	Ubuntu 18.04 LTS 64 Bit	
Software	Apache Hadoop V3.1.3, Spark V2.3.0	

storage performance impact on the RPi 3B cluster by adopting higher-performance storage media cards such as the UFS card (i.e., Samsung UFS card) and fast microSD card (i.e., Sandisk Extreme Pro). Thus, we could only use a typical microSD card (32GB) as storage media for both clusters and we necessarily omitted both storage media cards for this evaluation. Consequently, for fair evaluation, we built both RPi 4B and 3B clusters with a typical microSD card, and performed Hadoop and Spark TeraSort benchmarks to measure average execution time. Our main research goal lies in exploring big data processing possibilities on the most powerful RPi-based cluster to date, not to provide a one-to-one performance comparison between an RPi 3B cluster and an RPi 4B cluster.

Due to stability and compatibility problems, we installed the Ubuntu 18.04 LTS (Long Term Support) 64-bit OS (Operating System) on each microSD card instead of the Raspberry Pi OS (previously called ''Raspbian'') that is an officially supported OS for RPi. At the time we built our RPi 4B cluster, Raspberry Pi OS for RPi 4B had just been released for experimentation. However, extensive testing indicated it was not stable and had Apache-based open source software compatibility problems at that time. In addition, Ubuntu is one of the third-party OSes the RPi foundation officially suggested a Raspberry Pi OS alternative. To conserve RPi computing resources, we adopted Ubuntu 18.04 LTS *without* a GUI (Graphic User Interface) such as GNOME desktop.

TABLE 6. Features of the three storage media with distinctly different performance characteristics employed for our experiments. Performance numbers are excerpted from their product specifications.

	microSD1	microSD2	UFS card
Model code	MB-MP64HA/KR	SDSDXXG-064G	MB-FA256G/APC
Model name	Samsung EVO	Sandisk Ext. Pro	Samsung UFS
Speed class	C10, U1	C10, U3, V30	N/A
Capacity	32GB	64GB	256GB
Form factor	microSD (SDHC)	microSD (SDXC)	UFS
Read (seq.)	Up to 95MB/s	Up to 170MB/s	Up to 500MB/s
Write (seq.)	Up to 30MB/s	Up to 100MB/s	Up to 200MB/s

For big data processing and analysis, we installed Apache Hadoop version 3.1.3 and Spark version 2.3.0.

Table 6 presents three representative RPi storage media cards and their specifications. Samsung EVO microSD card (referred to as microSD1) is one of the best selling microSD cards in the market [7]. Thus, we employed it as a typical (i.e., widely used) microSD card for our Raspberry Pi cluster nodes. We adopted the Sandisk Extreme Pro microSD card (referred to as microSD2) as a faster storage media for our Raspberry Pi 4B cluster. This Sandisk Extreme Pro microSD card was specifically selected as one of the best microSD card for Raspberry Pi among 10 different brand microSD cards because it exhibits all-around excellent performance [18]. It performed 2× to 3× faster than the Samsung Evo microSD card. As a fastest storage media card for our Raspberry Pi cluster, we employed Samsung UFS card because it dominates all microSD cards in every performance metric. Please note this UFS card form factor fits standard microSD card slots, but its interface is not compatible with the microSD interface. Therefore, we installed it through the RPi USB 3.0 port with a UFS adapter since the USB 3.0 interface has sufficient bandwidth (5Gbps) to support the UFS card's bandwidth (500MB/s). Unlike previous Raspberry Pi generations, the Raspberry Pi 4B supports USB 3.0 ports. Thus, we can utilize the high speed USB 3.0 port to provide a faster storage media in the Raspberry Pi 4B cluster.

For storage performance impact studies, we employed each identical storage media cards to all RPi 4B nodes in the cluster and set up identical Hadoop and Spark platforms. Then, we performed various representative benchmarks to measure average execution time (seconds), I/O rate (Mb/s), and throughput (Mb/s).

IV. INDIVIDUAL RASPBERRY PI PERFORMANCE

Individual RPi node performance directly affects overall cluster performance. Thus, we first evaluate the computational capabilities of the latest RPi. For comparison, the previous generation (RPi 3B) is also evaluated to verify RPi 4B's performance improvements. This section evaluates each Raspberry Pi (RPi)'s performance in CPU processing, network, storage performance, and energy consumption.

TABLE 7. Power consumption of the RPi 3B and 4B in idle and stress modes.

Modes	RPi 3B	RPi 4B
Idle mode	280mA (1.4W)	540mA (2.7W)
Stress mode (stress −−cpu 4)	740mA (3.7W)	1,280mA (6.4W)

TABLE 8. Average CPU performance (events per second) for each RPi with *n* threads.

Threads	1	2	4	8	16
RPi 3B	4,862.90	9,735.55	19,417.18	19,400.11	19,395.49
RPi 4B	12,109.77	24,206.92	48,441.63	48,395.37	48,428.20

A. POWER CONSUMPTION

To measure individual RPi (i.e., 4B and 3B) power consumption, we used a Bplug S01 power meter which provides both real-time and accumulated power consumption in terms of watts. For precise measurement, nothing was plugged into the USB ports. Two power consumption modes were measured: an idle mode and a stress mode. In an idle mode, the RPi node remained without any application running for 5 hours. In a stress mode, we assigned a 400% CPU load (*stress −−cpu 4*) for 1 hour. We then measured the RPi 4B's power consumption. The RPi 3B's power consumption was similarly evaluated and added for reference.

As table 7 indicates, an RPi 4B consumes more power (1.9× in idle mode and 1.7× in stress mode) than the previous generation RPi 3B. This is because RPi 4B's advanced computing capabilities requires more power. Thus, considering the RPi 4B as a desktop PC alternative may not be the best choice for battery-powered and portable applications. However, plugged into a wall and sitting on the desk as a cluster node for big data processing, RPi 4B can exploit its more powerful computing capabilities at the cost of portability [9]. Section V describes this tradeoff in more detail.

B. CPU PERFORMANCE

For CPU performance measurement, we adopted *sysbench* which has been in the MySQL ecosystem for a long time. *sysbench* was originally developed to run synthetic MySQL benchmarks and the hardware (CPU, RAM, and I/O) it runs on. Now it is widely used to perform Linux file I/O, CPU, and memory performance tests. We ran *sysbench* (Version 1.0.11) CPU tests for 1, 2, 4, 8, and 16 threads (e.g., *sysbench −−num-threads = 4 −−test = cpu −−cpu-max-prime = 2000 run*)

Table 8 presents average CPU performance for each model RPi and demonstrates that the RPi 4B's CPU performance is approximately 2.49× better than the RPi 3B's CPU performance. Based on [21], RPi 3B's CPU performance improved approximately 1.5× compared to the RPi 2B. Considering this, our CPU performance measurement indicates the RPi 4B's CPU performance unprecedentedly improved due to its fully-redesigned CPU architecture. *sysbench* CPU

TABLE 9. Average on-board network (both Ethernet and WiFi) performance of RPi 3B and 4B.

Modes	RPi 3B	RPi 4B
On-board Ethernet (Mbps)	94.8	943
On-board WiFi (Mbps)	39.8	92.8

performance of both RPi 4B and 3B keeps increasing with the number of threads. That is, CPU performance with 2 threads shows 2× faster than the one with 1 thread. Similarly, 4-thread performance doubles 2-thread performance. However, we did not observe notable performance gain with 8 and 16 threads because both RPis have CPUs with 4 physical cores without hyperthreading.

C. NETWORK PERFORMANCE

RPi 4B is the first generation that natively supports a 1Gb Ethernet network with a maximum theoretical throughput of 1,000Mbps. In contrast, the RPi 3B uses a 100Mbps Ethernet network. In the real world, utilizations vary, tending to reach about 90–95% of the theoretical maximum.

Network performance is one of the most straightforward aspects to benchmark. However, it also has a caveat-laden aspect. For example, file upload/download performance or network file copy performance is dependent on other RPi components such as memory I/O, bandwidth, disk I/O, USB bus speed, etc [9]. Therefore, we focus on raw network throughput.

iperf is an active measurements tool for a maximum achievable raw network throughput. We used *iperf* to test on-board raw network (e.g., *iperf -c 10.0.10.1 -i1*). Table 9 presents the average network throughput after several runs. Please note we disabled WiFi power management (e.g., *sudo iwconfig wlan0 power off*) to improve WiFi connection stability. Otherwise, many packets can get dropped, corrupting network throughput measurements [9].

As indicated in table 9, with the help of the RPi 4B's native gigabit Ethernet support, the RPi 4B exhibits almost 10× higher on-board LAN performance than the RPi 3B. For on-board WiFi performance, the RPi 4B shows a 2.3× performance improvement over the RPi 3B due to the RPi 4B's new WiFi chipset.

D. STORAGE MEDIA PERFORMANCE

In general, there is an order-of-magnitude performance difference between the inexpensive storage media cards and the slightly more expensive ones especially for small (e.g., 4KB) random I/O performance [10]. Thus, we employed three different storage media cards (please refer to table 6) and performed benchmarks to evaluate their performance.

For objective evaluation, we employed three widely used storage performance benchmark tools: *hdparm*, *dd*, and *iozone*. *hdparm* gives basic raw throughput statistics for buffered disk reads (e.g., *sudo hdparm -t /dev/mmcblk0*). *dd* simply copies data from one place to another. Please note the count parameter value must be sufficiently large to avoid file system cache effects (e.g., *sudo dd if = /dev/zero of = /drive/output bs = 8k count = 50k conv = fsync; sudo rm -f /drive/output*).

TABLE 10. Storage media card performance on a desktop, RPi 4B, and RPi 3B (MB/s). 4K read and 4K write correspond to 4K random read and 4K random write performance respectively.

Storage media	Node	Read	Write	4K read	4K write
Samsung EVO	Desktop	90.19	24.6	9.45	2.09
	RPi 4B	43.09	19.0	9.56	2.14
	RPi 3B	21.8	17.3	5.61	1.8
Sandisk Extreme Pro	Desktop	88.9	77.4	8.92	2.27
	RPi 4B	43.36	31.1	6.59	3.18
	RPi 3B	22.35	18.8	4.82	2.89
Samsung UFS	Desktop	340.35	194.0	24.22	40.0
	RPi 4B	336.77	117.0	17.83	20.01
	RPi 3B	30.09	30.8	7.13	7.99

iozone is a popular file system benchmark tool that provides a broad overview of read and write performance with various block sizes and situations. We adopted this especially for its small (typically 4KB) block random I/O performance test. This 4KB random I/O performance has a critical performance impact on many big data or database operations such as logging, bulk data loading, writing to RDBMS (Relational Database Management System), etc [10] (e.g., *iozone -e -I -a -s 100M -r 4k -r 512k -r 16M -i 0 -i 1 -i 2 [-f /path/to/file]*).

As indicated in table 10, even identical storage media cards exhibit very different performance according to the host node. For reference, we added a desktop environment (Intel i5-8400 CPU @2.8GHz, 8GB RAM) in addition to the exiting both RPi 4B and 3B comparison. All media cards show dominant performance on the desktop environment over the other two RPi environments. Particularly, a UFS card outperforms other microSD cards in all performance aspects. Please note UFS card performance on the RPi 3B is unreasonably lower than the RPi 4B because RPi 3B does not support USB 3.0 ports. In general, each storage media card on the RPi 4B shows approximately 2× better performance than that on RPi 3B.

Interestingly, unlike the product specifications (up to 95MB/s vs. 170MB/s in table 6) for both the Samsung EVO and the Sandisk Extreme Pro, read performance of both media cards did not show notable difference (i.e., they were nearly identical). That is, the Sandisk Extreme Pro microSD card never reached its maximum read performance (170MB/s), not even 100MB/s. For a more objective evaluation, we adopted other widely-known benchmarks such as IOmeter,[2] CrystalDiskMark,[3] and ATTO disk benchmark,[4] in addition to *hdparm, dd,* and *iozone*. However, no benchmark program exhibited a meaningful read performance difference between the two cards.

[2]Iometer, http://www.iometer.org/.
[3]CrystalDiskMark, https://crystalmark.info/en/software/crystaldiskmark/.
[4]ATTO technologies, https://www.atto.com/disk-benchmark/.

TABLE 11. Properties in Hadoop configurations.

mapred-site.xml	Value
yarn.app.mapreduce.am.resource.mb	3072
mapreduce.map.cpu.vcores	1
mapreduce.reduce.cpu.vcores	1
mapreduce.map.memory.mb	2048
mapreduce.reduce.memory.mb	4096
mapreduce.input.fileinputformat.split.minsize	4MB
YARN-site.xml	**Value**
yarn.nodemanager.resource.memory-mb	4096
yarn.nodemanager.resource.cpu-vcores	8 (default)
yarn.scheduler.minimum-allocation-mb	1024
yarn.scheduler.maximum-allocation-mb	4096
yarn.scheduler.minimum-allocation-vcores	1 (default)
yarn.scheduler.maximum-allocation-vcores	4 (default)
yarn.nodemanager.vmem-pmem-ratio	2.1 (default)
hdfs-site.xml	**Value**
dfs.replication	3 (default)

V. RASPBERRY PI CLUSTER PERFORMANCE FOR BIG DATA PROCESSING

Table 11 presents major Hadoop configuration properties for our Hadoop and Spark cluster performance evaluation. Please note that, for 'large' data (500GB and 1TB) test on our RPi 4B cluster, we changed an HDFS replication factor from 3 to 1 to get enough working space to run benchmarks (please refer to Section VI).

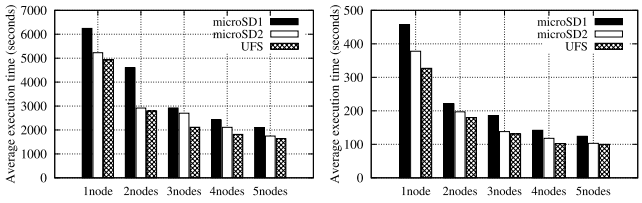
A. WORDCOUNT

''Wordcount'' is one of the most popular and representative micro-benchmarks to evaluate a MapReduce job. It counts the number of separate word occurrences from input data (i.e., text or a sequence file). That is, it splits input data into each word (in terms of key-value pair) via a Map function. The Map function then generates intermediate data which correspond to Reduce function's input data. The Reduce function aggregates each map's intermediate data to final word count data [11].

For objective evaluation, we adopted public data (2006.csv, 678MB) from the American Statistical Association (ASA) website which consist of flight arrival and departure details for all commercial flights within the USA from October 1987 to April 2008.[5] We prepared four large input files (i.e., 500MB, 1GB, 2GB, and 4GB) based on the public data. Please note we did not manipulate any data except simple file cropping or appending to meet data sizes. On our Raspberry Pi (RPi) 4B cluster, we ran the Wordcount benchmark five times on both popular big data processing/analysis platforms (i.e., Apache Hadoop and Spark) with different storage media (i.e., typical microSD, faster microSD, and the UFS card). The number of cluster nodes varies from one to five.

Figure 3 presents average Wordcount benchmark execution time with 2GB data on our RPi 4B cluster for each different number of nodes. As the number of nodes increases, Wordcount execution time decreases for both Hadoop and Spark. In Hadoop Wordcount, a single node using UFS and microSD2 is 1.3× and 1.2× faster than a single node with microSD1 respectively (Figure 3 (a)). Similarly, for Spark Wordcount on a single node, both UFS and microSD2 show 1.4× and 1.2× faster execution time than microSD1 respectively (Figure 3 (b)).

[5]American Statistical Association, Data Expo 2009 - Airline on-time performance



(a) Hadoop Wordcount (b) Spark Wordcount

FIGURE 3. Average wordcount benchmark execution time with 2GB data on the Raspberry Pi 4B cluster.

TABLE 12. Hadoop wordcount benchmark results (seconds).

Data size	Storage	1 node	2 nodes	3 nodes	4 nodes	5 nodes
500MB	UFS	1314.6	694.1	563.1	539.8	530.9
	microSD2	1387.2	709.5	633.7	615.2	562.7
	microSD1	1569.1	828.8	888.8	862.9	821.8
1GB	UFS	2565.0	1306.0	1060.0	856.1	819.5
	microSD2	2649.0	1541.8	1279.9	1033.6	884.1
	microSD1	2975.5	1725.6	1526.3	1142.4	1103.0
2GB	UFS	4930.2	2791.8	2113.3	1814.5	1637.2
	microSD2	5230.3	2916.1	2699.2	2111.2	1748.8
	microSD1	6246.0	4602.7	2919.3	2424.2	2102.9
4GB	UFS	9943.1	5109.1	4173.8	3529.9	3270.0
	microSD2	10556.0	7333.5	4484.2	4066.0	3524.7

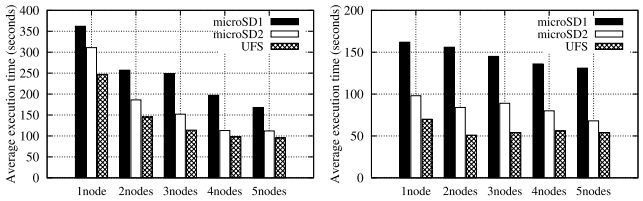
TABLE 13. Spark wordcount benchmark results (seconds).

Data size	Storage	1 node	2 nodes	3 nodes	4 nodes	5 nodes
500MB	UFS	115.0	70.0	58.2	46.9	45.4
	microSD2	127.0	79.8	64.6	58.2	52.6
	microSD1	164.7	97.4	85.4	78.9	69.0
1GB	UFS	181.3	106.6	82.7	66.3	65.2
	microSD2	198.6	116.8	83.5	71.2	68.7
	microSD1	215.5	146.1	116.0	102.6	91.2
2GB	UFS	327.2	180.2	131.3	102.5	109.0
	microSD2	378.6	197.0	138.6	118.2	103.7
	microSD1	457.5	221.2	186.9	142.8	124.0
4GB	UFS	620.5	330.9	227.2	175.5	163.9
	microSD2	711.6	367.4	264.1	189.8	176.0
	microSD1	851.7	449.5	329.8	259.2	201.9

Table 12 presents all experimental results of our Hadoop Wordcount benchmark with diverse configurations. As we increase the number of RPi 4B nodes and adopt faster storage media, it exhibits noticeably improved performance. In addition, as the data size increases from 0.5GB to 4GB, total execution time also increases almost linearly. Please note the table omits the experimental results of a microSD1 with 4GB data because the RPi 4B cluster with these configurations did not work properly.

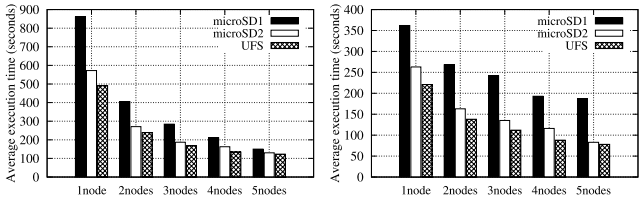
B. TeraSort

Originally, TeraSort benchmark goal was to sort 1TB data as quickly as possible. TeraSort was named after this 1TB data sorting. The TeraSort benchmark combines testing the HDFS (Hadoop Distributed File System) and MapReduce layers of a Hadoop cluster. Thus, it is often used to compare



(a) Hadoop TeraGen (b) Spark TeraGen

FIGURE 4. Average execution time of TeraGen benchmark with 2GB data on Raspberry Pi 4B cluster.



(a) Hadoop TeraSort (b) Spark TeraSort

FIGURE 5. Average execution time of TeraSort benchmark with 2GB data on Raspberry Pi 4B cluster.

the results of one cluster with another. It measures file distribution and processing capabilities (i.e., map and reduce functions), and consists of 3 components: TeraGen, TeraSort, and TeraValidate [12], [13]. TeraGen generates random data. TeraSort sorts these random data using a MapReduce processing framework. Finally, TeraValidate ensures the TeraSort output is correct (i.e., data were sorted correctly). During the map phase, TeraSort reads input files and sorts them. Additionally, it writes HDFS output files during the reduce phase. We performed both TeraGen and TeraSort benchmarks on our RPi cluster with various data sizes (500MB, 1GB, 2GB, and 4GB) by varying the node numbers from one to five.

Unlike Wordcount, that typically generates final data of a reduced size, TeraSort does not change its data size. It simply rearranges (i.e., sorts) the data sequence in ascending or descending order. Therefore, both read and write performances of storage media cards have a significant impact on overall performance (i.e., execution time).

As shown in figure 5 (a), the performance difference between microSD1 (i.e., typical SD) and UFS significantly increases by up to 1.75×. Compared to the Wordcount benchmark, TeraSort noticeably differs in performance, particularly according to storage media card write performance. Based on our performance evaluation of storage media cards, UFS exhibits 7.88× and 6.15× improved write performance compared to a microSD1 on the desktop and RPi 4B node respectively (please refer to table 10). Spark TeraSort enlarges the performance gap between microSD1 and UFS by an average of 1.96×. UFS also exhibits on average 1.64× better performance than microSD2 (Figure 5 (b)).

C. TestDFSIO READ AND WRITE

The TestDFSIO benchmark is an I/O (i.e., read and write) test to stress Hadoop Distributed File System (HDFS) on the clusters [14]. It measures cluster I/O speed and is therefore helpful to explore cluster performance bottlenecks. TestDFSIO creates one mapper for one file (i.e., one map task per file). Since a TestDFSIO read test does not generate its own input test files, first we should run a write test (as input data for the subsequent read test) and a read test then should follow it by subsequently reading these test file. Meanwhile, the benchmark measures average I/O (Mb/s), throughput (Mb/s) and execution time (seconds) [3].

We performed this benchmark with 10 files of sizes 1GB, 5GB, 10GB, and 20GB on the RPi 4B cluster of 5 nodes. We also used a value of 3 as an HDFS replication factor. Both table 16 and 17 respectively present TestDFSIO read and write benchmark results of Hadoop and Spark.

In the Hadoop DFSIO write benchmark, the UFS card showed better performance than microSD2 and microSD1 by an average of 1.87× and 4.37× respectively. On average, the microSD2 was 2.33× faster than microSD1. Similarly, in the Spark DFSIO write benchmark, we observed the UFS card exhibited on average 1.9× and 7.07× better performance than microSD2 and microSD1 respectively. The microSD2 showed on average 3.69× better performance than microSD1.

In the DFSIO read benchmark, we did not observe notable performance differences between microSD1 and microSD2 for both Hadoop and Spark. This microSD2 read performance problem was previously addressed in the subsection IV-D and this DFSIO read benchmark confirmed that issue. Undoubtedly, UFS dominated both microSD1 and microSD2 by an average of 3.2× on Hadoop and 3.7× on Spark.

TABLE 14. Hadoop TeraSort benchmark results (seconds).

Data size	Storage	1 node	2 nodes	3 nodes	4 nodes	5 nodes
500MB	UFS	358.1	174.9	127.5	101.3	95.1
	microSD2	366.2	181.6	137.6	110.9	98.1
	microSD1	425.6	205.8	148.4	132.4	107.9
1GB	UFS	395.1	192.9	143.7	110.4	106.3
	microSD2	440.0	198.7	147.4	120.0	110.5
	microSD1	559.6	233.0	197.7	168.6	131.2
2GB	UFS	491.1	239.6	168.9	134.6	123.7
	microSD2	572.1	271.7	187.6	163.5	130.5
	microSD1	860.9	406.1	284.6	211.3	150.5
4GB	UFS	867.6	412.0	281.1	231.8	190.8
	microSD2	1050.9	527.2	395.4	320.2	211.3
	microSD1	1675.8	961.6	569.5	468.1	418.4

TABLE 15. Spark TeraSort benchmark results (seconds).

Data size	Storage	1 node	2 nodes	3 nodes	4 nodes	5 nodes
500MB	UFS	72.3	49.2	49.2	35.6	37.0
	microSD2	72.9	55.3	49.8	39.2	34.9
	microSD1	87.2	55.8	58.4	47.0	48.8
1GB	UFS	127.4	79.0	68.4	55.6	52.2
	microSD2	133.5	83.7	78.4	59.3	55.3
	microSD1	141.6	121.6	105.7	107.6	87.1
2GB	UFS	221.0	138.3	112.8	88.4	78.5
	microSD2	263.3	163.6	135.6	116.2	83.8
	microSD1	362.1	268.2	242.5	193.5	187.0
4GB	UFS	432.6	263.5	197.0	157.4	132.3
	microSD2	587.8	332.6	290.6	216.3	173.0
	microSD1	834.7	649.6	507.7	471.1	402.9

TABLE 16. Hadoop TestDFSIO read and write performance on the Raspberry Pi 4B cluster of 5 nodes. SD1 and SD2 stand for microSD1 and microSD2 respectively.

Operations	Size	Average throughput (Mb/s)			Average I/O rate (Mb/s)			Average execution time (seconds)		
		SD1	SD2	UFS	SD1	SD2	UFS	SD1	SD2	UFS
Read	1GB	16.46	14.46	36.22	18.08	15.67	42.01	69.1	64.11	56.02
	5GB	13.3	14.37	46.55	13.93	15.84	47.91	100.26	104.3	64.27
	10GB	12.97	14.42	50.42	13.22	14.66	52.26	145.87	133.8	77.1
	20GB	13.38	14.61	54.96	13.57	14.69	55.71	215.23	203.36	93.28
Write	1GB	5.47	9.38	13.82	7.59	9.77	14.8	101.98	80.18	72.86
	5GB	2.32	5.72	10.96	2.4	6.06	11.02	318.29	161.53	105.94
	10GB	2.02	5.11	11.27	2.08	5.16	11.3	640.09	271.54	146.64
	20GB	1.16	5.03	11.14	1.18	5.06	11.18	1988.7	474.3	239.31

TABLE 17. Spark TestDFSIO read and write performance on the Raspberry Pi 4B cluster of 5 nodes. SD1 and SD2 stand for microSD1 and microSD2 respectively.

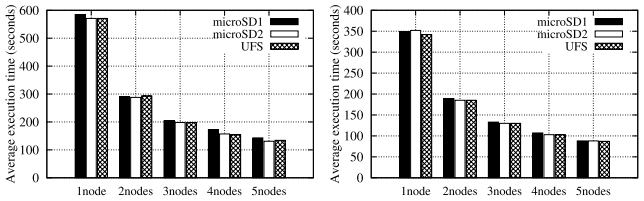
Operations	Size	Avg. throughput (Mb/s)			Avg. I/O rate (Mb/s)			Avg. execution time (s)		
		SD1	SD2	UFS	SD1	SD2	UFS	SD1	SD2	UFS
Read	1GB	31.21	24.98	79.93	33.27	27.71	105.4	23.47	24.06	17.17
	5GB	29.44	22.65	86.86	30.02	23.03	97.05	51.24	60.98	27.39
	10GB	25.54	29.69	111.53	26.66	29.95	126.99	99.94	85.06	35.25
	20GB	26.79	24.38	105.22	27.62	24.62	117.11	183.25	180.31	59.38
Write	1GB	5.55	18.22	24.37	12.06	19.64	25.91	52.62	14.34	10.49
	5GB	2.82	10.53	23.83	3.14	10.99	24.34	406.34	106.51	46.08
	10GB	3.39	11.15	23.0	3.48	11.41	23.25	652.61	196.1	90.61
	20GB	2.32	9.42	22.54	2.38	9.46	23.02	1780.43	438.86	187.54

Please note, because DFSIO read benchmarks are performed after completing DFSIO write benchmarks, a cache must be first eliminated for objective read performance evaluation before the DFSID read tests. Without cache elimination, we observed both microSD1 and microSD2 DFSIO read performance exhibited performance very similar to UFS, particularly with small test data (i.e., 1GB). As test data size increases, their DFSIO read performance (without cache dropping) decreases because cache effects also decrease. However, when we initially dropped cache and performed the DFSIO read benchmarks, each storage media card's read performance showed nearly consistent performance, irrespective of test data size.

D. PI COMPUTATION

Mathematically, Pi is the ratio of the circumference of any circle to the diameter of that circle. Regardless of the circle's size, this ratio is always equal to Pi. The Pi computation benchmark is a MapReduce program that estimates Pi using a quasi-Monte Carlo method. Its main goal is to measure CPU performance by observing a compute-bound CPU. First, the Monte Carlo method presents a inscribed circle of a square. It then generates a large number of random points within the square and counts how many fall in the inner circle [3]. If we divide the area of the circle by the area of the square, we get $\pi/4$. The same ratio can be applied to the number of points within the square and the number of points within the inner circle. Hence, we can use the following formula to estimate Pi: $\pi \approx 4\times$ (number of points in the circle / total number of points).

In the Pi computation benchmark, a mapper generates random points in a unit square and then counts points inside and outside of the inner circle of the square. A reducer accumulates points inside and outside results from the mappers [3]. The goal of this Pi computation benchmark is to observe the CPU computation-bound workload of the RPi cluster because this Pi computation is a heavy CPU-intensive workload.



(a) Hadoop Pi computation (b) Spark Pi computation

FIGURE 6. Average execution time of Pi computation benchmark on Raspberry Pi 4B cluster.

TABLE 18. Hadoop Pi benchmark results (seconds). Numbers stand for the number of nodes on the cluster.

Configurations	Storage	1	2	3	4	5
100 Maps 10K Samples	UFS	571.1	293.3	198.4	154.7	134.9
	microSD2	571.3	288.5	198.0	157.6	131.2
	microSD1	595.6	290.3	201.9	173.3	143.3

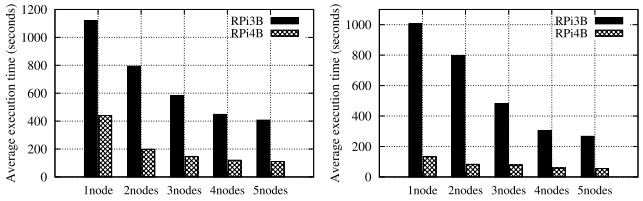
For this benchmark, we adopted 100 maps that each uses 10,000 samples per map. Figure 6 displays average execution time of the Pi computation benchmark on our Raspberry Pi 4B cluster. Unlike other benchmark tests, performance gap among the three different media cards is ignorable because the Pi computation benchmark is very CPU-intensive. Consequently, storage performance does not affect test results. However, as the number of nodes increases, each completion (i.e., execution) time decreases since it benefits from more distributed computation.

E. RASPBERRY PI 3B VS. RASPBERRY PI 4B

This subsection determines the Raspberry Pi 4B's performance improvement by comparing it to a Raspberry 3B.

TABLE 19. Spark Pi benchmark results (seconds). Header numbers indicate the number of cluster nodes.

Configurations	Storage	1	2	3	4	5
10K Samples	UFS	342.0	185.9	130.5	103.8	87.4
	microSD2	352.8	185.8	130.4	103.0	88.0
	microSD1	349.9	189.3	133.6	107.2	88.7



(a) Hadoop TeraSort (b) Spark TeraSort

FIGURE 7. TeraSort benchmark with 1GB data on Raspberry Pi 3B cluster vs. 4B cluster.

TABLE 20. Hadoop TeraSort benchmark results (seconds): RPi 3B vs. RPi 4B.

Data size	RPi	1 node	2 nodes	3 nodes	4 nodes	5 nodes
500MB	3B	889.7	790.7	485.0	361.2	352.5
	4B	366.2	181.6	137.6	110.9	98.1
1GB	3B	1121.9	793.0	580.3	448.9	407.2
	4B	440.0	198.7	147.4	120.0	110.5
2GB	3B	1683.0	1183.3	832.7	785.2	585.8
	4B	572.1	271.7	187.6	163.5	130.5
4GB	3B	4623.7	3593.5	2552.2	2399.3	2309.0
	4B	1050.9	527.2	395.4	320.2	211.3

Since the main goal of our work is not 1:1 performance comparison between RPi 3B and RPi 4B, we provide one benchmark result which suggests meaningful hints for their performance differences.

We performed TeraSort benchmark on both the RPi 4B cluster and the RPi 3B cluster using various data sizes (500MB, 1GB, 2GB, and 4GB) and varying the node numbers from one to five. As subsection V-B mentioned, unlike other benchmark programs, TeraSort is more appropriate to test comprehensive computing capabilities.

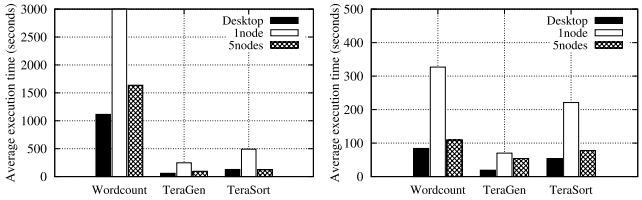
Figure 7 demonstrates that RPi 4B performance dominates RPi 3B performance. In the Hadoop framework (Figure 7 (a)), RPi 4B TeraSort shows better performance than RPi 3B by an average of 3.58× and up to 4.1×. Similarly, in the Spark framework (Figure 7 (b)), RPi 4B is even faster than RPi 3B by an average of 6.65× and up to 9.56×. This mainly results from Spark's computing model (i.e., in-memory processing). Apache Spark effectively exploits main memory, so DRAM capacity significantly affects overall performance. Our RPi 4B has 4GB main memory. On the other hand, the RPi 3B has 1GB DRAM. Consequently, Spark TeraSort presents a more pronounced performance difference than Hadoop TeraSort.

F. RASPBERRY PI 4B VS. DESKTOP PC

Subsection V-E identified the substantial improvement of the RPi 4B's computational capabilities. The RPi 4B's performance compared to a desktop PC' performance suggests the possibilities of the RPi 4B as a desktop PC alternative or replacement for big data processing. We performed

TABLE 21. Spark TeraSort benchmark results (seconds): RPi 3B vs. RPi 4B.

Data size	RPi	1 node	2 nodes	3 nodes	4 nodes	5 nodes
500MB	3B	829.8	789.1	299.7	305.2	199.0
	4B	72.9	55.3	49.8	39.2	34.9
1GB	3B	1005.9	794.6	482.4	305.2	276.3
	4B	133.5	83.7	78.4	59.3	55.3
2GB	3B	1572.4	2097.1	1766.1	1215.7	589.8
	4B	263.3	163.6	135.6	116.2	83.8
4GB	3B	3636.8	3277.8	3343.39	2528.7	1920.5
	4B	587.8	332.6	290.6	216.3	173.0



(a) Hadoop benchmarks (b) Spark benchmarks

FIGURE 8. Performance comparison between Raspberry Pi 4B vs. desktop PC. Both 1node and 5nodes stand for RPi 4B cluster with 1 node and 5 nodes respectively.

TABLE 22. Performance comparison between RPi 4B and desktop PC with various benchmarks (seconds).

Platforms	Computing nodes	Wordcount	TeraGen	TeraSort
Hadoop	Desktop PC	1114.3	59.7	128.0
	RPi4B (1 node)	4930.2	247.4	491.1
	RPi4B (5 nodes)	1637.2	95.9	123.7
Spark	Desktop PC	84.6	19.6	54.1
	RPi4B (1 node)	327.2	70.0	221.0
	RPi4B (5 nodes)	109.0	54.5	78.5

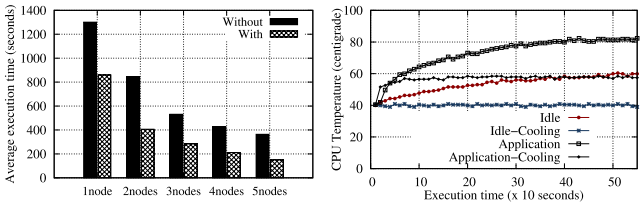
three representative Hadoop and Spark benchmarks on both the RPi 4B node and the desktop PC (table 5 presents both specifications). Figure 8 shows the benchmark results. We observed the desktop PC respectively exhibited 4.3× and 3.9× better performance than one RPi 4B in Hadoop and Spark. However, when we added 4 more RPi 4B nodes to the cluster (i.e., total 5 nodes), the performance gap between one desktop PC and the 5 node RPi 4B cluster noticeably decreased to an average 1.4× and 1.5×.

VI. DISCUSSION

The latest generation RPi's substantially improved computational capabilities enabled us to re-visit and re-explore the possibilities of heavy computing-intensive applications on RPi clusters. Based on our studies, this section discusses challenges and suggestions for big data processing on Raspberry Pi (RPi) clusters.

A. CPU THERMAL THROTTLING

The RPi 4B's new A72 series 64bit CPU consumes 1.7× more power than the previous ARM processor (i.e., A53 series 64bit CPU) in the RPi 3B. This increases requirements for thermal dissipation. When we put the RPi 4B in an official RPi case without a cooling fan, we found a significant performance drop. This implies the RPi 4B clearly engaged CPU thermal throttling. On the other hand, when we opened the case and applied an active cooling with a fan,



(a) Hadoop TeraSort (b) CPU Temperature

FIGURE 9. Cooling effect: TeraSort benchmark with 2GB data on Raspberry Pi 4B cluster (without cooling vs. with cooling).

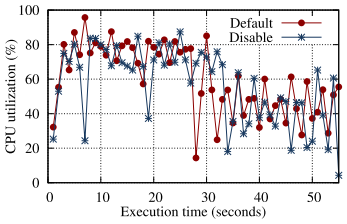


FIGURE 10. CRC disablement: CPU utilization of TeraSort benchmark with 2GB data on Raspberry Pi 4B (disable vs. default).

our RPi 4B cluster exhibited best performance. Therefore, we strongly suggest installing an active cooling system (such as a fan) into the RPi 4B systems or using a metal case that passively dissipates heat. Moreover, based on [9], the author recommended to invest a heatsink or some ventilation even for the previous model (RPi 3) because RPi 3 is hotter than its predecessors. All in all, a stronger cooling system is necessary for the RPi 4B.

Figure 9 (a) presents the CPU cooling effect of RPi 4B (i.e., without cooling vs. with active cooling). RPi 4B with active cooling shows better performance than RPi 4B without the cooling system by an average of 1.97× and 1.57× in Hadoop and Spark TeraSort respectively. This is because RPi 4B without the cooling activated CPU thermal throttling. Please note we omitted the Spark TeraSort chart because it showed a similar result to Hadoop TeraSort benchmark. For clarification, we measured CPU temperatures over time (Figure 9 (b)). In an idle state with a cooling system, its temperature looks very stable over time (around 40 degrees centigrade). In an idle state without the cooling system, the temperature keeps increasing from 48 degrees to 62 degrees. We also measured CPU temperature while running an application (TeraSort). The CPU temperature of RPi 4B with active cooling slightly increases but soon stabilizes. However, the RPi 4B without cooling displayed as high as 87 degrees centigrade. If any heavier program runs on the RPi 4 node without an active cooling system, the CPU expected temperature will definitely be higher than 87 degrees. Our experiments demonstrate, for maximum performance, effective cooling is necessary (not optional) for the RPi 4B.

B. CRC DISABLEMENT

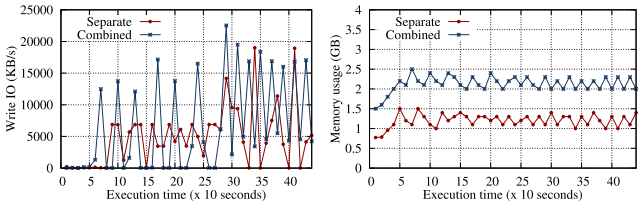
HDFS (Hadoop Distributed File System) generates checksums of all data written to it and verifies checksums when reading data. A separate checksum is generated for each data of 512 bytes (*io.bytes.per.checksum*) by default. Hadoop datanodes are responsible for verifying the data they receive before storing both the data and their checksums. At rest, they continuously verify data against stored checksums to detect and repair, via other means, bit errors. When clients read data from datanodes, they also verify checksums by comparing them with the ones stored at the datanodes. These HDFS data integrity processes can cause extra overhead, particularly for RPi clusters because the RPi does not have relatively strong computing capabilities. To save RPi's computing resources, we can consider disabling the checksum verification of Hadoop by setting the setVerifyChecksum() method to false on FileSystem before using the *open()* method to read a file.

Figure 10 depicts TeraSort CPU utilization with 2GB data on one RPi 4B node. We performed the same TeraSort benchmark with default CRC (i.e., enabled) vs. disabled CRC configurations, and measured average CPU utilization over time. TeraSort with a disabled CRC configuration consumes 52.5% CPU in total. On the other hand, the default (i.e., CRC enabled) TeraSort shows on average 59.5% CPU utilization. Therefore, disabling HDFS CRC checksums helps reduce CPU consumption when running big data processing applications on RPi nodes.

C. MASTER NODE SEPARATION

Namenode is a master node in the Apache Hadoop architecture. It manages and maintains the data blocks on the datanodes (i.e., slave nodes) in the cluster. That is, it manages the file system name space and controls client access to files by recording the metadata of all HDFS data files the cluster stores. Additionally, the namenode regularly receives both heartbeats and block reports from all datanodes in the cluster to ensure each datanode is alive. On the other hand, the datanodes are in charge of worker nodes that actually store big data and process them. Since the namenode is not involved in data computation and processing jobs, it does not consume high computing resources. Thus, in many Hadoop cluster configurations, we can find such a configuration that the master daemon runs together on one of the datanodes. If the master daemon runs on a separate, non-datanode, it can help conserve RPi's datanode (i.e., worker node) resources.

Figure 11 presents TeraSort benchmark resource usage on the RPi 4B cluster. We measured I/O, CPU, and memory consumption on a datanode which was configured to run a master daemon on the datanode together (referred to as *Combined* in the figure) and on a separate namenode (referred to as *Separate* in the figure) respectively. In figure 11 (a), the datanode with a separate namenode configuration (i.e., *Separate*) generates on average 28.5% less write traffic than the datanode with a combined namenode configuration (i.e., *Combined*). Similarly, the *Separate* generates less read traffic than the *Combined* by an average of 29.5% (we omitted this figure). Figure 11 (b) shows memory consumption of a datanode with both configurations. We observed *Separate*



(a) Write I/O (b) Memory consumption

FIGURE 11. Master node separation effect: Resource usage of TeraSort benchmark with 1GB data on Raspberry Pi 4B cluster.

consumes on average 42.2% less memory than *Combined*. Interestingly, unlike both I/O traffic and memory usage, we did not find notably different CPU usage between both configurations: *Separate* consumes slightly less CPU than *Combined* by an average of 2% (we omitted this figure).

D. DIFFERENT PERFORMANCE VIEWS

Subsection V-F compared the RPi 4B's performance to a desktop PC's performance. We observed the 5 node RPi 4B cluster exhibited close big data processing performance (on average 1.4× to 1.5×) to the PC. This implies a cluster of about seven RPi 4B nodes is likely to show comparable performance to the desktop PC. This hypothesis introduces two different performance metrics: performance-per-dollar and performance-per-watt. Our desktop PC costs about $600 (US) and each RPi 4B node with 4GB RAM costs $55 (US). Other considerations such as storage cards and a network switch hub (approximately $200 (US) total), the seven RPi 4B node cluster costs approximately $585 (US) total. This implies the RPi 4B essentially matches the desktop PC in terms of performance-per-dollar. We also measured the power consumption of both systems during benchmarks. The PC consumed about 102 watts and the cluster required approximately 42 watts. The RPi 4B outperformed the PC in terms of performance-per-watt. Please note the power consumption and the total price of a computer system totally depend on hardware configurations. However, as shown in table 5, our experimental desktop PC was not inappropriately configured to manipulate our performance results.

A primary goal of this discussion is to provide useful hints of the latest RPi's performance as well as explore RPi possibilities for big data processing. Based on our previous studies, the previous generation (i.e., RPi 3B) never matched desktop PCs in terms of performance-per-dollar as well as performance-per-watt because its overall performance is significantly lower than the RPi 4B. However, through extensive experiments, we observed the latest generation of RPi with large RAM (e.g., 4GB or more) exhibited this possibility.

E. DATA SIZES

For a single board computer (SBC) cluster to be adopted for practical big data processing applications, the data size the cluster can process in practice is a crucial factor. Most previous studies simply evaluate processing hundreds of Megabytes data (mostly less than 1GB). This is because previous generations were not powerful enough to accommodate 'real' big data (i.e., more than 100s GB to TB). We also employed a few GB data for our evaluations, not because an RPi 4B cluster cannot process larger data amounts, but because the objective is to provide various hints at our performance results by varying the data size from 0.5GB to 4GB.

This subsection addresses how large data can be practically processed on our 5 node RPi 4B cluster. For this, we used UFS cards (256GB each) for storage media to accommodate as much data as possible (i.e., approximately 1.2TB storage space in total). To test both 500GB and 1TB data, we changed the Hadoop configuration replication factor from 3 to 1 to get enough working space to perform Spark Wordcount benchmarks. Our RPi 4B cluster successfully processed them in 17509 seconds and 32884 seconds respectively. For reference, we built a 5 node RPi 3B cluster with identical configurations within hardware limitations for the same benchmarks. However, the RPi 3B cluster often failed to simply put dozens of GB data to HDFS. We observed the RPi 3B cluster frequently met with failure in putting just 10GB data to the cluster via HDFS due to each datanode's slow data processing.

Based on our experiments, we expect larger data can be processed on our RPi 4B with 4GB RAM cluster if we can access a larger storage space. Specifically, we observed large RAM significantly assists big data processing capabilities by notably reducing Hadoop intermediate data amount such as data spills. We conclude RPi 4B is the first generation to finally address the possibilities of 'real' big data processing with the help of its unprecedented computational capabilities.

VII. CONCLUSION

Big data has been spotlighted for the last decade and various big data technologies have been emerged. Today, the big data industry has a significant impact on our daily life. Many companies adopted Apache Hadoop and Spark as the core of their big data revolution, storing and analyzing increasingly massive data on powerful servers. However, now, server architecture is transitioning from general-purpose rack servers to purpose-built servers such as blade servers and microservers in an attempt to reduce power consumption and space requirements. The advent of single board computers (SBCs) has changed the computing environment by bridging the gap between controller boards and personal computers. Importantly, the Raspberry Pi (RPi) family now leads the SBC market for several good reasons.

This paper comprehensively explored RPi cluster-based big data processing challenges and possibilities. We built a 5 node RPi cluster (Model 4B with 4GB RAM, the most powerful RPi as of now) and installed Apache Hadoop and Spark on it. We evaluated storage media performance impact on RPi clusters by utilizing three different storage solutions with distinctly different performance (i.e., typical, fast, fastest storage media available for RPi 4B). To our knowledge,

this is the first study that addresses storage performance impact on RPi cluster, but also revisits the possibilities of big data processing on the Raspberry 4B with 4GB RAM-based clusters.

In the individual RPi performance evaluation, we observed RPi 4B' performance has unprecedentedly improved in many respects compared to the previous generation RPi (i.e., RPi 3B): about 2.5× faster CPU, 10× faster on-board LAN, 4× larger DRAM, 10× faster USB, etc. at the cost of 1.7–1.9× higher power consumption.

We also extensively performed popular benchmarks (Wordcount, TeraSort/TeraGen, DFSIO read/write, Pi computation) to evaluate big data processing platforms on RPi clusters with the aforementioned three storage media cards. These representative benchmarks demonstrate that adopting faster storage media is a very effective way in substantially improving the cluster performance by showing 1.3× to 7.07× big data processing performance improvement. We compared RPi 4B cluster performance to RPi 3B cluster performance by running the TeraSort benchmark. This benchmark showed the RPi 4B cluster exhibited on average 6.65× to 9.56× better performance than the RPi 3B cluster, particularly with the Spark platform. Performance comparison to a desktop PC also showed very promising results–RPi 4B exhibited comparable or better performance than the PC in terms of performance-per-dollar and performance-per-watt.

In summary, unlike previous generation RPis, we observed RPi 4B's computational capabilities have been substantially improved so that it now supports big data processing on SBC clusters by running mainstream operating systems and accommodating heavy workloads (more than 100s GB to TBs). Our comprehensive studies suggest some optimization techniques to further reduce RPi 4B cluster resource consumption. If a future generation RPi arrives with a substantially larger DRAM capacity, it would significantly improve big data processing performance on the cluster.

ACKNOWLEDGMENT

The authors would like to thank David Schwaderer (QuantumSafe Ciphers LLC, USA) for his valuable comments and proofreading.

REFERENCES

[1] R. Scolati, I. Fronza, N. El Ioini, A. Samir, and C. Pahl, "A containerized big data streaming architecture for edge cloud computing on clustered single-board devices," in *Proc. 9th Int. Conf. Cloud Comput. Services Sci.*, 2019, pp. 1–13.

[2] A. K. Tripathi, K. Sharma, M. Bala, A. Kumar, V. G. Menon, and A. K. Bashir, "A parallel military-dog-based algorithm for clustering big data in cognitive industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 2134–2142, Mar. 2021.

[3] B. Qureshi and A. Koubaa, "On energy efficiency and performance evaluation of single board computer based clusters: A Hadoop case study," *Electronics*, vol. 8, no. 2, pp. 1–26, 2019.

[4] M. Feldman. (2018). *Evolving Server Architectures Challenge Standardization Efforts*. [Online]. Available: https://www.snia.org/sites/default/files/SSIF/2018-05-31/Serverstandardsarticle.pdf

[5] S. J. Johnston, P. J. Basford, C. S. Perkins, H. Herry, F. P. Tso, D. Pezaros, R. D. Mullins, E. Yoneki, S. J. Cox, and J. Singer, "Commodity single board computer clusters and their applications," *Future Gener. Comput. Syst.*, vol. 89, pp. 201–212, Dec. 2018.

[6] C. Pahl, S. Helmer, L. Miori, J. Sanin, and B. Lee, "A container-based edge cloud PaaS architecture based on raspberry Pi clusters," in *Proc. IEEE 4th Int. Conf. Future Internet Things Cloud Workshops (FiCloudW)*, Aug. 2016, pp. 117–124.

[7] Souvik. *Top 10 Best Selling Micro SD Memory Cards*. Accessed: May 2021. [Online]. Available: https://www.rswebsols.com/reviews/product-reviews/top-10-best-selling-micro-sd-memory-cards

[8] J. Geerling. *Power Consumption Benchmarks*. Accessed: Jun. 2021. [Online]. Available: https://www.pidramble.com/wiki/benchmarks/power-consumption

[9] J. Geerling. (2016). *Review: Raspberry Pi Model 3 B, With Benchmarks Vs Pi 2*. [Online]. Available: https://www.jeffgeerling.com/blog/2016/review-raspberry-pi-model-3-b-benchmarks-vs-pi-2

[10] J. Geerling. (2016). *Raspberry Pi microSD Card Performance Comparison*. [Online] Available: https://www.jeffgeerling.com/blogs/jeff-geerling/raspberry-pi-microsd-card

[11] N. Ahmed, A. L. C. Barczak, T. Susnjak, and M. A. Rashid, "A comprehensive performance analysis of Apache Hadoop and Apache spark for large scale data sets using HiBench," *J. Big Data*, vol. 7, no. 1, pp. 1–18, Dec. 2020.

[12] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1107–1117, Jun. 2013.

[13] Y. Tarutani, K. Hashimoto, G. Hasegawa, Y. Nakamura, T. Tamura, K. Matsuda, and M. Matsuoka, "Temperature distribution prediction in data centers for decreasing power consumption by machine learning," in *Proc. IEEE 7th Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Nov. 2015, pp. 635–642.

[14] T. Ivanov, R. Niemann, S. Izberovic, M. Rosselli, K. Tolle, and R. V. Zicari, "Performance evaluation of enterprise big data platforms with HiBench," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Aug. 2015, pp. 120–127.

[15] T. White, *Hadoop Definitive Guide*, 3rd ed. Sebastopol, CA, USA: O'Reilly Media, 2012, pp. 97–99.

[16] O. Elkeelany and V. S. Todakar, "Data archival to SD card via hardware description language," *IEEE Embedded Syst. Lett.*, vol. 3, no. 4, pp. 105–108, Dec. 2011.

[17] W. Jeong, Y. Lee, H. Cho, J. Lee, S. Yoon, J. Hwang, and D. Lee, "Improving flash storage performance by caching address mapping table in host memory," in *Proc. USENIX Conf. Hot Topics Storage File Syst.*, Santa Clara, CA, USA, Jul. 2017, pp. 19–24.

[18] A. Piltch. (2021). *Best microSD Cards for Raspberry Pi*. [Online]. Available: https://www.tomshardware.com/best-picks/raspberry-pi-microsd-cards

[19] A. Dalvandi, M. Gurusamy, and K. C. Chua, "Time-aware VMFlow placement, routing, and migration for power efficiency in data centers," *IEEE Trans. Netw. Service Manage.*, vol. 12, no. 3, pp. 349–362, Sep. 2015.

[20] S. Toor, L. Osmani, P. Eerola, O. Kraemer, T. Lindén, S. Tarkoma, and J. White, "A scalable infrastructure for CMS data analysis based on OpenStack cloud and Gluster file system," *J. Phys., Conf. Ser.*, vol. 513, no. 6, Jun. 2014, Art. no. 062047.

[21] R. Zwetsloot. (2019). *Raspberry Pi 4 Specs and Benchmarks*. [Online]. Available: https://magpi.raspberrypi.org/articles/raspberry-pi-4-specs-benchmarks

[22] L. Hattersley. (2019). *Raspberry Pi 4 VS Raspberry Pi 3B+*. [Online]. Available: https://magpi.raspberrypi.org/articles/raspberry-pi-4-vs-raspberry-pi-3b-plus

[23] F. P. Tso, D. R. White, S. Jouet, J. Singer, and D. P. Pezaros, "The Glasgow raspberry Pi cloud: A scale model for cloud computing infrastructures," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst. Workshops*, Jul. 2013, pp. 108–112.

[24] P. Basford, G. Bragg, J. Hare, M. Jewell, K. Martinez, D. Newman, R. Pau, A. Smith, and T. Ward, "Erica the rhino: A case study in using raspberry Pi single board computers for interactive art," *Electronics*, vol. 5, no. 4, p. 35, Jun. 2016.

[25] P. Abrahamsson, S. Helmer, N. Phaphoom, L. Nicolodi, N. Preda, L. Miori, M. Angriman, J. Rikkila, X. Wang, K. Hamily, and S. Bugoloni, "Affordable and energy-efficient cloud computing clusters: The Bolzano raspberry Pi cloud cluster experiment," in *Proc. IEEE 5th Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2013, pp. 170–175.

[26] C. Kaewkasi and W. Srisuruk, "A study of big data processing constraints on a low-power Hadoop cluster," in *Proc. Int. Comput. Sci. Eng. Conf. (ICSEC)*, Khon Kaen, Thailand, Aug. 2014, pp. 267–272.
[27] C. Baun, "Mobile clusters of single board computers: An option for providing resources to Student projects and researchers," *SpringerPlus*, vol. 5, no. 1, pp. 1–20, Dec. 2016.
[28] R. Morabito, "Virtualization on Internet of Things edge devices with container technologies: A performance evaluation," *IEEE Access*, vol. 5, pp. 8835–8850, 2017.
[29] V. Noronha, E. Lang, M. Riegel, and T. Bauschert, "Performance evaluation of container based virtualization on embedded microprocessors," in *Proc. 30th Int. Teletraffic Congr. (ITC)*, Sep. 2018, pp. 79–84.

EUNSEO LEE received the bachelor's degree in computer science from Sookmyung Women's University, Seoul, South Korea, in 2021, where she is currently pursuing the master's degree in computer science.

From 2019 to 2020, she was a Research Assistant with the Big Data Storage Systems Lab under the advice of Prof. D. Park, and studied on big data processing on single board computer (SBC) clusters. Her research interests include big data processing platforms (Apache Hadoop and Spark), storage systems, and non-volatile memory (NVM), such as Intel Optane persistent memory and NAND Flash memory.

HYUNJU OH received the bachelor's degree in computer science from Sookmyung Women's University, Seoul, South Korea, in 2020, where she is currently pursuing the master's degree in computer science.

In 2020, she was a Research Assistant with the Big Data Storage Systems Lab under the advice of Prof. D. Park, and did research on a behavior-based ransomware detection technology using I/O distribution. She is interested in big data processing, storage systems, and next generation non-volatile memories (NVM), such as Intel 3D Xpoint memory, designing a new big data processing platform based on new memory technologies.

DONGCHUL PARK (Member, IEEE) received the Ph.D. degree in computer science and engineering from the University of Minnesota–Twin Cities, Minneapolis, USA, in 2012.

He was a member of the Center for Research in Intelligent Storage (CRIS) group under the advice of Prof. David H. C. Du. From 2012 to 2016, he was a Senior Research Engineer with the Memory Solutions Laboratory (MSL), Samsung Semiconductor Inc., San Jose, CA, USA, and a Senior Staff Research Engineer with the Storage Technology Group (STG) at Intel, Hillsboro, OR, USA, in 2017. From 2017 to 2019, he was an Assistant Professor in computer science and engineering at the Hankuk University of Foreign Studies (HUFS), Yongin, South Korea. Since 2019, he has been an Assistant Professor with the Department of Software, Sookmyung Women's University, Seoul, South Korea. His research interests focus on storage system design and applications, including non-volatile memories (NVM), in-storage computing, data deduplication, key-value store, and shingled magnetic recording (SMR) technology. He is also interested in big data processing, Hadoop MapReduce, cloud computing, and next generation NVM, such as Intel Optane Memory.