Capabilities of Raspberry Pi 2 for Big Data and Video Streaming Applications in Data Centres

Nick J. Schot, Paul J.E. Velthuis, Björn F. Postema*

Centre for Telematics and Information Technology,
University of Twente, the Netherlands
{n.j.schot, p.j.e.velthuis}@student.utwente.nl, b.f.postema@utwente.nl
http://www.utwente.nl/ewi/dacs/

Abstract. Many new data centres have been built in recent years in order to keep up with the rising demand for server capacity. These data centres require a lot of electrical energy and cooling. Big data and video streaming are two heavily used applications in data centres. This paper experimentally investigates the possibilities and benefits of using cheap, low power and widely supported hardware in the form of a micro data centre with big data and video streaming as its main application area. For this purpose, multiple Raspberry Pi 2 Model B (RPi2)'s have been used in order to build a fully functional distributed Hadoop and video streaming setup that has acceptable performance and extends to new research opportunities. We experimentally validated the new setup to fit in a data centre environment by analysis of its performance, scalability, energy consumption, temperature and manageability. This paper proposes a high concurrency and low power setup in a small 1U form factor with an estimated number of 72 RPi2's as an interesting alternative to traditional rack servers.

Keywords: Micro data centre, Raspberry Pi 2, benchmarking, Hadoop, big data, video streaming, cloud computing

1 Introduction

In data centres, the density of servers increased significantly in the past years [16]. New technologies emerge, e.g., blade servers, that not only decrease the physical appearance of what used to be an entire rack full of servers, but also decrease power consumption by implementing new technologies. ARM processors, another relatively new technology, might actually fit the increasing demand for modularity in data centres. Since Raspberry Pi's are fully functional servers, that have an ARM processor, a relatively powerful graphical chip onboard and use little energy, these should be considered as a serious alternative. The main

* The work in this paper has been supported by the Dutch national STW project Cooperative Networked Systems (CNS), as part of the program "Robust Design of Cyber- Physical Systems" (CPS).

challenge of this paper is to fit Raspberry Pi 2's with ARM on-board in a data centre. Two major data centre applications elaborated in this paper are (i) big data; and (ii) video streaming. Big data solutions distribute data processing among various servers often with a high demand on storage devices. Big data's main task is to query large chunks of data to retrieve valuable information. On the other hand, demands with video streaming require more network capabilities, since the main task of video streaming is to seamlessly deliver data for the duration of the video. Many small tasks are processed in the case of video streaming, while big data applications perform rather large tasks.

This paper contributes by investigating the capabilities of Raspberry Pi's for micro data centres, thereby focussing on benchmarks and measurements of power, performance, temperature and hardware allocation of an experimental setup with a data centre ready Raspberry Pi cluster. These aspects allow us to analyse three main design criteria of a flexible future proof data centre [4, p.6], namely: scalability, performance and manageability.

First, background information on cloud computing with big data and video streaming is elaborated in Section 2. A few cloud projects based on RPi hardware are described in Section 3. Then our own proposed RPi2 cluster will be discussed, and thoroughly tested with Hadoop and video streaming in order to investigate the possibilities of the RPi2 in a micro data centre.

2 Two key applications for data centres

In this section background is given for the two main applications big data and video streaming. Big data is about data too large and complex to be processed by normal data applications. In Section 2.1, a solution to big data is discussed that allows to distribute processing of these large chunks of data. Since the video streaming service Netflix is responsible for approximately 30 % of the downstream traffic in the US [2], this relevant application domain is elaborated in Section 2.2 by a short description of its operations and approach inspired by the existing video streaming service Netflix.

2.1 Big data

Apache Hadoop [25] is an open source framework which offers necessary components for the distributed processing of large amounts of distributed data, using simple programming models like map/reduce. It has been designed to scale well from one to thousands of machines. Hadoop offers high-availability options for detecting and recovering from failures in both hard- and software. Hadoop is used for applications like risk modelling and recommendation engines which have petabytes of data to be analysed.

Map/reduce [9] as implemented in Hadoop is a programming model to allow for simple distributed processing of large data sets. A map/reduce program consists of two steps. The map step performs filtering and sorting. The reduce step can then do further computations on the output of the maps, which is usually

a summarizing operation. Depending on the program the map and/or reduce tasks can be parallelised.

In Hadoop 2 *Yet Another Resource Negotiator* (YARN) was introduced as a new resource management layer. YARN handles workload management and monitoring, manages high availability features and allows for more programming models next to just map/reduce.

Big server manufacturers like Dell, HP and SuperMicro offer all kinds of servers for Hadoop applications. Hadoop usually runs on a multitude of 1U rack servers containing eight or more storage drives. 1U rack servers are relatively cheap, but bring a lot of space and energy overhead when you place a lot of servers compared to more expensive, but more efficient solutions like blade servers which are usually 10U [20]. Blade servers house vertically placed blades combined with a single power supply and network access for all blades combined, which makes them more space efficient [20] than traditional 1U servers. Hadoop setups can start with just a single server and be scaled to thousands of servers.

2.2 Video streaming

Large video streaming providers often require various operations to deliver videos to their clients in an uninterrupted and fast manner. For this reason, the buffer time of a video is minimized, such that videos are accessible at any given time. These videos are then delivered by the server that has the best latency for the client. Large video streaming providers like Netflix require the following three operations for their services:

1. Content ingestion, which means that the studio master version of the films are received and uploaded to the cloud.
2. Content processing, which means that in the cloud many different formats are created for each video (e.g. AVI, MP4 and MKV format). These formats are uploaded to the content distribution network (CDN), which is a network of several data centres to spread the content to users. This means that all the formats been made are distributed over the CDN.

Netflix has its own CDN allowing better analysis of the network and improvements of load balancing and video streaming algorithms. In order to store all the video data, Netflix uses the file storage systems Amazon's AWS, simpleDB, S3 and Cassandra [2].

Video streaming heavily relies on data storage, most of the time spinning hard drives (HDD) are used. If a video is accessed frequently then a U1 SSD server is used to make faster streaming possible. This means there are two types of servers. The servers with HDD normally take 4U of server space and the SSD variant with servers consumes 1U server space [27].

3 Related Work

There have been several cluster projects with the Raspberry Pi Model B(+)(RPi).

The *Iridis-pi* cluster with 64 RPi's was built by Cox et al. [7]. A Message Passing Interface was used to communicate between the Raspberry Pi's. The research was done to investigate what the performance of a low-power high performance cluster was. It was designed as a portable and passively cooled cluster for educational purposes.

Tso et al. [26] built a data centre consisting of 56 RPi's that offers a cloud computing testbed including virtualisation management tools called the *Glasgow Raspberry Pi Cloud*. It was built for practical research on cloud computing without the limitations of simulation.

Kiepert [17] created a *Beowulf* cluster for a PhD assignment. It was built for collaboratively processing sensor data in a wireless sensor network. The Raspberry Pi cluster offers an alternative in case of the main cluster is unavailable.

The *Bolzano Raspberry Pi* cluster consists of 300 RPi's and was made as an affordable energy-efficient computer cluster by Abrahamsson et al [1]. Applications such as a green research testbed and as a mobile data center are evaluated. Their main goal was to introduce a cluster of RPi's on a larger scale.

The RPi clusters described are for research, application performance and cluster mobility. The projects described above applied the first generation RPi which offers significantly lower performance than the newer second generation RPi. This research distinguishes itself from other RPi clusters by providing benchmarks of the temperature, power consumption and performance of the Hadoop and video streaming applications.

4 System Description

In the system description the software and experiment setup are elaborated. First short summary is given of the device used in the micro data centre, namely the RPi2 in Section 4.1. In the experimental setup our own micro data centre is elaborated for the Hadoop and video streaming variant in Section 4.2. The Hadoop software needed for distributed processing is discussed in Section 4.3. Then the video streaming software required for a large streaming service is elaborated in Section 4.4.

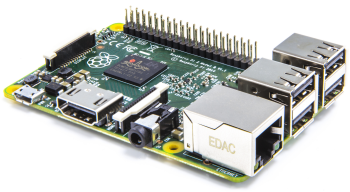

Fig. 1. Raspberry Pi 2 model B [24]

Table 1. Raspberry Pi 2 Model B specifications [24]

System on Chip	Broadcom BCM2836
Ethernet	Onboard 10/100 Ethernet RJ45 jack
USB	Four USB 2.0
Video out	HDMI 1.4
Audio	2 x analog
CPU	900MHz quad-core ARM Cortex-A7
GPU	Dual Core VideoCore IV Multimedia Co-Processor
Card slot	Micro SD

4.1 Raspberry Pi 2

The Raspberry Pi 2 Model B [24] is a small, cheap yet feature packed computer. It is based on the Broadcom BCM2836 system on a chip which offers a 900MHz quad-core ARMv7 CPU combined with 1 GB of RAM and can currently be bought for about $35. Detailed specifications can be found in Table 1. 16GB Adata Premier Pro UHS-I microSD cards are used as the storage solution.

4.2 Experimental setup

A total number of eight RPi2's is used in our experimental setup. A setup diagram for Hadoop and video streaming is displayed in Figure 2 and Figure 3, respectively. The numbers in the setup diagrams correspond to the physical setup shown in Figure 4. The number (2) indicates a small router/switch that is connected to the power supply. The number (1) shows the eight RPi2's. In case of video streaming a load balancer and several video streamers are installed. For Hadoop there is one masternode and several slavenodes.

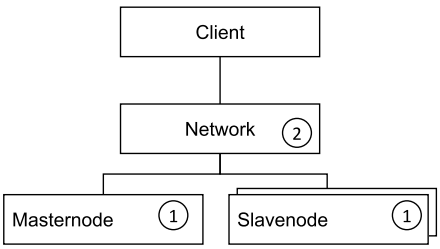

Fig. 2. Hadoop design

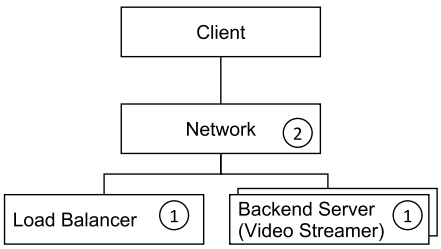

Fig. 3. Video streaming design

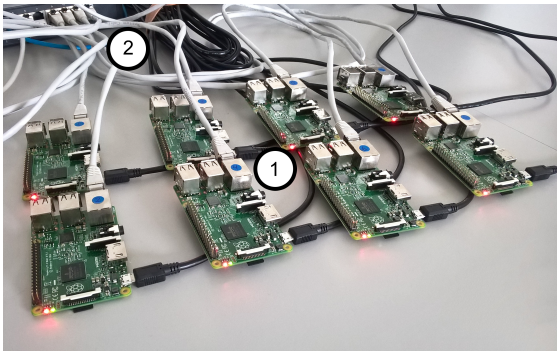

Fig. 4. Project Setup

Dietpi [8] is used as the operating system for the individual nodes. It is a lightweight version of Raspbian which is the Linux distribution specifically tailored for the RPi2.

4.3 Hadoop software

A basic Hadoop installation consists of three main parts: HDFS, YARN and the JobHistoryServer.

HDFS is the Hadoop distributed file system and consists of a couple of processes. The NameNode is the main process which keeps track of where all files are distributed and replicated. It is the main access point for all clients and processes and runs on the master node. The SecondaryNameNode keeps a recent backup of the NameNode so the NameNode can be restored if it might go down. The DataNode processes run on the remaining slave nodes and handle data storage and retrieval.

YARN consists of a ResourceManager, which manages all jobs in the system, and on each slave node a NodeManager. The NodeManager process handles the execution of jobs allocated to a slave node.

Finally, the JobHistoryServer keeps track of all completed jobs and their logs.

A natively compiled version of Hadoop 2.6.0 with YARN was configured in conjunction with Oracle Java 7 ARM HF. Because there are only eight available RPi2's, a single master node runs the NameNode, Secondary NameNode, ResourceManager and the JobHistoryServer. The other (scalable) amount of nodes act as slaves and each runs a NodeManager and a DataNode.

The setup has 91 GB of distributed storage available with the replication factor of two, which resulted in an effective amount of roughly 45 GB. YARN has been configured so that two containers can run concurrently on a single slave node. This gives 14 available container slots for Hadoop to allocate tasks to in the test setup.

4.4 Video streaming software

This video streaming software consist of four software programs: nginx, FFmpeg, JW Player and Cassandra. For load balancing and streaming over HTTP, nginx [23] is used. nginx has an efficient algorithm for HTTP load balancing. The *Real Time Messaging Protocol* (RTMP) module from Arut for nginx is used to make a media streaming server over HTTP [5]. This has an efficient algorithm to transfer the HTTP with RTMP encapsulated data to the users. FFmpeg is a cross-platform solution to record, convert and stream audio and video [11]. Using this software makes adaptive streaming and streaming in different formats possible. JW Player is a HTML5/flash embedded media player [18]. JW Player makes load balancing possible dependable on the bit rate that is coming from the video. It supports dynamic streaming, that consists of multiple single streams with the same content, all in a different quality [18]. Cassandra is a database that helps replicating data across multiple data centres [6]. The data can automatically be replicated across the nodes for fault-tolerance. Therefore, the data is still available when a node crashes.

5 Cluster Benchmarking

This section elaborates benchmarks and measurements on power, temperature, storage, memory and network to test the cluster as if in a data centre environment.

5.1 Storage and memory performance

For basic system benchmarks, the SysBench suite [19] has been used. It serves as a tool to quickly determine system performance without setting up any complex software.

Table 2. SysBench Storage & Memory

Benchmark	**Transfer speed**
random storage read	9.9718 MB/s
random storage write	1.2604 MB/s
random storage read/write	3.4046 MB/s
sequential storage read	17.7400 MB/s
sequential storage write	6.3972 MB/s
sequential storage rewrite	13.0000 MB/s
sequential memory read	207.5000 MB/s
sequential memory write	177.0200 MB/s

The SD card storage was tested by running random and sequential storage tests. 4 GB of test data was prepared with SysBench. The benchmarks were run with a maximum execution time of 300 seconds. The memory test sequentially read and wrote 512 MB of data to memory.

Table 2 shows that the write performance of the SD cards is low. Read performance is below what was expected from the SD card, which promised 40 MB/s for sequential read operations but achieved barely half that speed. The RPi2's memory is sequentially read at 207 MB/s while its write speed is 177 MB.

5.2 Energy consumption

The energy consumption is measured with a simple setup. A prototyping PCB with two USB connectors and some jumper wires are used to allow for a multimeter (Elro M990) to connect for voltage and current measurements of a single RPi2. This way the actual power usage of the RPi2 is measured, because the (in)efficiency of the power supply is not taken into account. When a measurement would be done at the wall outlet, power usage is expected to be higher.

The power consumption was measured under several workloads to find out what effect different kind of operations have on the power consumption of the RPi2. SysBench is used to consistently stress different parts of the board.

Table 3. Benchmarks to test energy consumption of Raspberry Pi 2 without power supply losses

	Current (A)	**Voltage (V)**	**Power (W)**
CPU 1 core	0.340	4.84	1.65
CPU 2 cores	0.365	4.79	1.75
CPU 3 cores	0.392	4.77	1.87
CPU 4 cores	0.415	4.78	1.99
Memory test	0.440	4.79	2.11
Storage read	0.442	4.77	2.11
Storage write	0.395	4.77	1.89
Idle	0.315	4.78	1.51

The RPi2 has a power consumption of at most 2.1 W in this test as shown in Table 3. A normal server needs about 500 W [21], so 238 RPi2's take as much power on one server.

5.3 Network performance

Iperf3 [10] was used to find out whether the network, the storage or the memory is a bottleneck by reading/writing from/to the different mediums [10]. The RPi2 uses a 100 Mbit Ethernet connection which is connected via a combined USB 2.0/Ethernet chip [22]. This is important as Hadoop shuffles a large amount of data around the network, video streaming needs to transport a lot of data to the user and in between the servers. To find out if there is a bottleneck, 60 second iperf3 benchmarks with a congestion windows of 133 KB have been executed from memory to memory, memory to storage and from storage to memory.

Table 4. Ethernet throughput benchmark with RPi2 memory and SD card storage

Write direction	**Avg. bandwidth (Mbit)**
Memory → memory	93.4
Memory → storage	24.3
Storage → memory	94.2

For every throughput benchmarks the congestion window is 133 KB. From the results in Table 4 the write performance of the Raspberry Pi 2 and/or the SD card forms a bottleneck with only 3 MB/s. This number is in line with the results from the SysBench write tests which were between 1.26 MB/s and 6.4 MB/s for random and sequential writes respectively. USB 2.0/Ethernet causes some overhead, therefore it has a throughput of around 94 Mbit.

5.4 Temperatures

CPU temperature measurements were taken under SysBench CPU stressing with different numbers of threads. During this benchmark the temperature is measured by logging the operating systems data on temperatures with a shell

script. The temperature is measured on the CPU. Results are shown in Figure 5. The room temperature during this benchmark was around 23 °C. The cooldown phase, that occurs after the benchmark has finished, is shown in Figure 6. The room temperature during the cooldown phase was around 21 °C and has been measured during a separate benchmark run. By default, the RPi2 is a passively cooled board without any heat sink or fan.

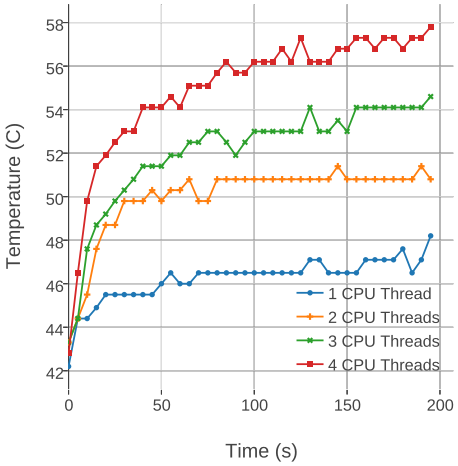

Fig. 5. Temperature benchmark

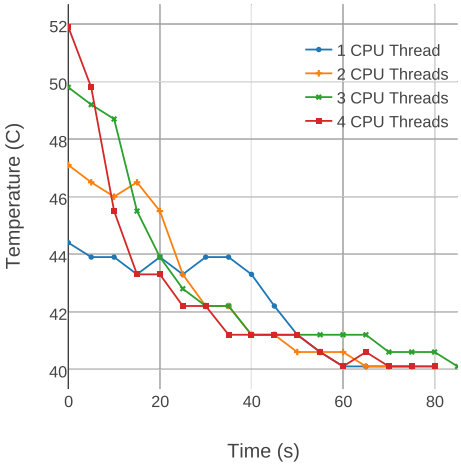

Fig. 6. Temperature cooldown

When running a four-thread CPU benchmark the maximum temperature is 60 °C and the temperature is 42 °C when idle, see Figure 5. In Figure 5, after a short period of time, temperatures converge to an upper bound. After benchmark completion, the CPU cools down quickly to idle temperature, as can be seen in Figure 6. Data centres require a temperature of around 26 °C and in order to do this, additional energy is required for cooling [12]. The most common workload for Hadoop and video streaming would be two CPU threads for which the temperature stays around 50 °C. So, if multiple RPi2's are used, some cooling is required in order to keep them working at optimal performance temperature.

6 Application Benchmarking

In this section, several Hadoop and video streaming benchmarks are analysed to show that in a data centre environment the proposed setup has acceptable performance.

6.1 Hadoop benchmarks

A selection of Hadoop benchmarks is made to cover the most important aspects of a Hadoop cluster. The benchmarks are part of the HiBench benchmark suite

[13], the standard Hadoop test suite and cover CPU bound computation and generic computation on distributed big data. A comparison is made with the CTIT cluster of the University of Twente where Hadoop runs on 32 Dell R415 servers.

Terasort is a benchmark which measures sort speed on large distributed files. The benchmark consists of a map/reduce job which creates and sorts a multiple of 100 byte rows and validates the results. A replication factor of one for the output files was forced instead of the cluster default. This way the replication of data throughout the cluster does not affect actual map/reduce performance.

Table 5. TeraSort benchmark

	Raspberry Pi 2						**CTIT**		
Nodes	5	8	5	8	5	8	-	-	-
Slots	8	14	8	14	8	14	-	-	-
Maps	16	16	64	64	80	80	16	64	80
Reduces	8	8	8	8	8	8	8	8	8
Data (GB)	1	1	7	7	10	10	1	7	10
Total (s)	366	230	3584	1747	-	341	22	49	67
Avg. map (s)	70	72	144	141	-	261	7	10	11
Avg. shuffle (s)	70	88	-	698	-	830	4	19	24
Avg. reduce (s)	48	49	1741	406	-	550	2	15	21

The CTIT cluster has far more container slots and nodes than the RPi2 cluster. Enough slots were available to allocate all map/reduces at once in the CTIT cluster and thus available slots are not mentioned in Table 5.

An inherent problem to a smaller cluster showed up in the 7 GB run on five nodes and is caused by one of Hadoops optimizations for bigger clusters. When a map task finishes on a node, Hadoop starts a reduce task on that same node since the necessary data is already there. The nodes are configured to run two concurrent tasks. With seven nodes available, this gives a total of 14 container slots of which one is the Application Master. With more map tasks than the amount of available containers, part of the tasks will run sequentially. The problem is that as soon as the first batch of map tasks finishes, reduce tasks get started on the nodes, so only few containers are available for the relatively high amount of map tasks to be completed. The reduce tasks will have a lot of idle time, because input data from the map tasks becomes available at a low pace. Adding more nodes would solve this problem as enough slots should be available to allocate the map jobs. This would bring the total running time closer to the average map time.

Since the 7 GB run allocated 64 map tasks, it took a total of 1747 seconds to complete all jobs. The average reduce time is high, because the reducers were still waiting for new input data. The shuffle time is the time to get the required data as output by a map task to the correct reducer. As there are usually many more map tasks then there are reduce tasks, this is a vital number for fast Hadoop operations. The reducers were able to retrieve data from other nodes with a reported speed of about 11 MB/s. This means Hadoop is most of the

time writing into memory, as iperf3 showed that the write speed to the SD card is much lower over the network.

The last problem showed up for the first time when TeraSort ran with 10GB of data on 5 nodes. If Hadoop assigns two reduce tasks to a single node, they have a lot of data to process, so the reduce tasks will use too much memory when writing their results to HDFS causing the DataNode process to crash and get kicked out of memory causing the reduce task to fail. Hadoop may then decide to start two copies of the same job to the cluster. This amplifies the problem with a small cluster, making the chance that two are running on a single node significantly higher. This problem can likely be solved by changing the YARN configuration so that only one reduce task may run on a single node.

Table 5 shows that the CTIT cluster's total running time is ten times lower when sorting 1 GB of data. The average map task also took roughly ten times longer on the RPi2 cluster. The runs with more data were a lot slower on the RPi2 cluster because not enough container slots were available in the cluster. The average map took 24 times longer on the RPi2 cluster when sorting 10 GB of data. This higher ratio could be the result of the low write speed to the SD card when more data has to be handled.

Table 6. Pi benchmark for computation of the number π

	Raspberry Pi 2			**CTIT**	
Containers	8	8	14	-	-
Maps	6	12	12	6	12
Total (s)	996	1975	996	40	40
Avg. map (s)	976	981	975	32	32
Avg. shuffle (s)	13	978	13	3	3
Avg. reduce (s)	2	2	2	0	0

The Pi benchmark was executed with a setup of five nodes with eight containers and a setup of eight nodes with 14 containers. The number π was computed in the benchmark with 10^9 samples per map. Increasing the maps or samples for the benchmark makes the estimation of π more accurate. From the Pi benchmarks in Table 6 it became clear that the average shuffle time depends on the availability of the data for the reducers. The Pi benchmark generates very small intermediate data which, if all maps can be allocated, takes only 13 seconds of shuffle time which is mostly overhead time from Hadoop due to hard coded polling intervals. The runs with 8 available containers show the impact of a setup with fewer available slots than there are maps to be run, compared with 6 maps and 12 maps with enough available nodes, the total duration depends on the speed with which individual maps are finished. The results in Table 6 show that the amount of maps does not influence running time for the Pi benchmark if enough container slots are available. Thus we can directly compare the results between the two systems. The CTIT cluster took 40 seconds to complete the benchmark with an average map time of 32 seconds. In comparison the RPi2 cluster took 996 seconds to complete with an average map time of 975 seconds.

This means that for this CPU bound benchmark the processing cores in the CTIT cluster are roughly 30 times faster than the processing cores from the RPi2.

6.2 Video stream benchmarks

The first benchmark streams and tests a video over the RTMP. Apache JMeter is a benchmark tool that is executed on an external machine to measure the number of streams a RPi2 can handle [3]. Apache JMeter allows to measure HTTP capture. As a consequence, RTMP streams can be measured, since these are encapsulated in HTTP. After the RTMP video stream is started, the workload of the stream is analysed over HTTP by accessing a video via a web browser. The RTMP stream has a rate of about 800 kbit/s for a small 230 MB video. The following basic formula defines the theoretical maximum number of users:

$$\text{max users} = \frac{\text{bandwidth}}{\text{bit rate stream}}.$$

The theoretical maximum number of users with this formula is 118 with 100 Mbit bandwidth. The benchmark accessing videos through the web browser allows 25 simultaneously connected users for streaming MPEG-4 (MP4) files over HTTP. In the web browser less then the maximum users can connect due to the buffer and video conversion time.

For *Synchronized Multimedia Integration Language* (SMIL) a special SMIL benchmark is used, that allows testing of different video streaming rates for a single file. It is possible to switch the quality depending on the amount of data that can get over the network, used in for example YouTube. Different video qualities have been created by FFmpeg in the H.264 codec from a 230 MB source video, namely: 720p, 480p, 240p and 120p. During the test with Apache JMeter 100 connections were simulated watching the video. There are two scenarios that use server-side JW Player: the first allows the user to select video quality, the second automatically switches to an appropriate quality based on the maximum achievable bitrate. In the first scenario the user chooses between the 120p, 240p or 480p version of the 230 MB video. When the 480p version is in use, freezing occurs; with the 120p and 240p versions no freezing is observed. As a consequence, freezing can occur when users are allowed to select their own quality; however when quality is automatically selected based on the maximum bitrate, no freezing occurs.

In order to test an automatic adjustable bit stream with quality constraints, a *Video on Demand* (VOD) benchmark is created. First, the converters FFmpeg and nginx need to be started to share a video over RTMP. The media player VLC is opened to indicate if there are any differences between VOD and a RTMP stream. VOD shows no signs of videos freezing. This is because, VOD is equipped to adjust the bit stream depending on the quality the stream and RTMP is not. RTMP only allows to watch the video that is played at that moment, which is similar to normal television.

7 Cluster in server racks

For the RPi2 to be useful in an enterprise environment, it must fit in standardised server racks. Hardware breaks all the time in data centres, so it should be easily accessible and replaceable to keep the data centre manageable. One disadvantage of current RPi2 is the placement of the power connector and Ethernet connector. The connectors are placed perpendicular to each other which makes it harder to place the boards in a confined space. To keep the manageability of the data centre two designs are proposed.

The rack must contain a power supply with sufficient ports and power to handle all RPi2. The casing must contain some fans to generate airflow.

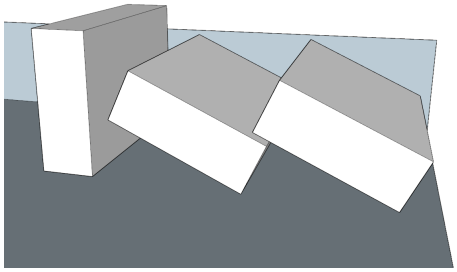

Fig. 7. Vertical and tilted RPi2 in a 1U server

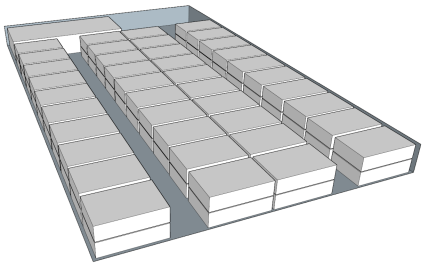

Fig. 8. Proposed RPi2 rack layout

Standard data centre racks contain often 42U of space. As defined by the EIA-310 standard a single U is 44.50mm high [14]. A 1U rack's inside dimensions are defined to be 450mm wide, 44.43mm high and at most 739.775mm deep [15]. The RPi2 is 85.60mm wide, 56mm deep and 21mm high. It has four standard mounting holes for screws or spacers to fit through.

The most efficient way to place the RPi2 in a small contained space is with the power connector facing downwards. So it can be connected to power on the bottom of the rack, and to Ethernet on the side, which would allow the easiest access to a RPi2. Unfortunately, as can be seen in Figure 7, a vertically placed RPi2 is a little higher than a standard U, so a bigger 1.5U rack should be used to make it fit. A variation can be tried by tilting the RPi2 boards so they fit in a 1U rack. The effect of this approach is shown in Figure 7. Because of the low angle, practically no overlap between the RPi2's can exist. This removes the main advantage of this approach.

The most obvious way to place the boards is to stack them in pairs of two and fill up the rack. Stacks can easily be secured on the bottom of the rack server by using spacers. The downside to this approach is the accessibility of the RPi2, as either the top one or both RPi2 have to be removed. 12 RPi2 fit next to each other in the rack, this gives 24 boards for a single row. While keeping space for all cables and connectors, four rows fit in the width of a rack server. With the power supply the estimated amount is 72 RPi2 for a 1U rack, seen in Figure 8.

In order to provide all boards with Ethernet a 2U switch will be needed as a 1U switch can house a maximum of 48 Ethernet ports.

8 Conclusion & Future Work

The contribution of this paper is a fully functional distributed Hadoop and video streaming setup with acceptable performance in the form of a micro data centre consisting of multiple Raspberry Pi 2 Model B (RPi2)'s. A high concurrency and low power setup that fits in a small 1U standardised form factor is proposed. This cheap setup is especially beneficial when lower performance is acceptable compared to expensive performance clusters. In the case of our two applications, acceptable performance is indeed attained, which is shown with the aid of several application specific benchmarks. Moreover, several benchmarks are performed on the cluster to ensure it functions properly inside data centre. A network benchmark confirms an acceptable performance by showing that both applications approach the maximum network bandwidth of about 94 Mbit/s under full load. An amount of 72 RPi2's in a 1U rack is expected to result in a highly concurrent rack with acceptable performance while using only roughly 160 Watt under full load. In comparison to the CTIT cluster that easily consumes kilowatts of power, programs with bigger map/reduce jobs like TeraSort ran only 24 times slower than this cluster. These numbers are promising when realising that the RPi2's have not yet an optimised architecture for support of a gigabit connection over USB and improved SD card reader performance. Before scaling this setup in a data centre environment, an appropriate solution to the large number of cables is still required for manageability purposes. Furthermore, the proposed setup could be used as a cheap micro version of a data centre to simulate existing applications before implementing the applications in an expensive cluster.

Acknowledgements. The authors would like to thank Marijn Jongerden and Boudewijn Haverkort (both from University of Twente) for their constructive feedback.

References

1. 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: Proc. of 5th Int. Conf. on Cloud Computing Technology and Science. vol. 2, pp. 170–175. IEEE (2013)
2. Adhikari, V., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., Zhang, Z.L.: Unreeling netflix: Understanding and improving multi-CDN movie delivery. In: INFOCOM, 2012 Proceedings IEEE. pp. 1620–1628 (2012)
3. Apache Software Foundation: Apache JMeter (2015), `http://jmeter.apache.org/`
4. Arregoces, M., Portolani, M.: Data Center Fundamentals. Cisco Press, Indianapolis (2003)
5. Arutyunyan, R.: NGINX-based Media Streaming Server (2015), `https://github.com/arut/nginx-rtmp-module`

6. Cassandra: Welcome to Apache Cassandra (2015), http://cassandra.apache.org/
7. Cox, S.J., Cox, J.T., Boardman, R.P., Johnston, S.J., Scott, M., OBrien, N.S.: Iridis-pi: a low-cost, compact demonstration cluster. Cluster Computing 17(2), 349–358 (2013)
8. Dan Knight: DietPi for Raspberry Pi's (2014), http://fuzon.co.uk/phpbb/viewtopic.php?f=8&t=6
9. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Communications of the ACM 51(1), 107 (2008)
10. ESnet: iperf/iperf3 (2015), http://fasterdata.es.net/performance-testing/network-troubleshooting-tools/iperf-and-iperf3/
11. FFmpeg: FFmpeg (2015), https://www.ffmpeg.org/
12. Google: Google Datacenters (2015), http://www.google.com/about/datacenters/efficiency/internal/#temperature
13. Huang, S., Huang, J., Liu, Y., Yi, L., Dai, J.: HiBench: A Representative and Comprehensive Hadoop Benchmark Suite. In: Proc. of 26th Int. Conf. on Data Engineering workshops (2010)
14. Innovation First, inc: 19-inch rack (EIA-310) (2007), https://www.server-racks.com/eia-310.html
15. Innovation First, inc: Rack Mounting Depth (2007), https://www.server-racks.com/rack-mount-depth.html
16. Jeff Clark: Raising Data Center Power Density (2013), http://www.datacenterjournal.com/raising-data-center-power-density/
17. Joshua Kiepert: Creating a Raspberry Pi-Based Beowulf Cluster (May 2013), http://coen.boisestate.edu/ece/files/2013/05/Creating.a.Raspberry.Pi-Based.Beowulf.Cluster_v2.pdf
18. JW Player: JW PLayer (2015), http://www.jwplayer.com/
19. Kopytov, A.: SysBench benchmark suite (2015), https://github.com/akopytov/sysbench
20. Leigh, K., Ranganathan, P., Subhlok, J.: General-purpose blade infrastructure for configurable system architectures. Distributed and Parallel Databases 21(2-3), 115–144 (2007)
21. Meisner, D., Gold, B.T., Wenisch, T.F.: Powernap: eliminating server idle power. ACM SIGARCH Computer Architecture News 37(1), 205–216 (2009)
22. Microchip: LAN9514-JZX (2012), http://ww1.microchip.com/downloads/en/DeviceDoc/9514.pdf
23. Nginx: NGINX (2015), http://nginx.com/
24. Raspberry Pi Foundation: Raspberry Pi 2 Model B (2015), https://www.raspberrypi.org/products/raspberry-pi-2-model-b/
25. The Apache software foundation: Welcome to Apache Hadoop! (2015), https://hadoop.apache.org/
26. 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: Proc. of 33rd Int. Conf. on Distributed Computing Systems Workshops. pp. 108–112. IEEE (2013)
27. Uptime Institute: Designing Netflixs Content Delivery Network. Uptime Institute Symposium (2014), https://journal.uptimeinstitute.com/designing-netflixs-content-delivery-network/