THESIS:
Testing Cassandra's Scalability On Raspberry Pi

THESIS

Daniel P. Richardson, Capt, USAF

Set document designator number using \docdesignator

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

Set document designator number using \docdesignator

THESIS:
TESTING CASSANDRA'S SCALABILITY ON RASPBERRY PI

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Engineering

Daniel P. Richardson, B.S.E.E.

Capt, USAF

February 2, 2017

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Set document designator number using \docdesignator

THESIS:

TESTING CASSANDRA'S SCALABILITY ON RASPBERRY PI

THESIS

Daniel P. Richardson, B.S.E.E.
Capt, USAF

Committee Membership:

Dr. John Pecarina, PhD
Chair

Dr. Alan Lin, PhD
Member

Dr. Kenneth Hopkinson, PhD
Member

Table of Contents

Page

List of Figures

Figure Page

Figure | Page

Figure Page

Figure Page

List of Tables

Acknowledgements

Daniel P. Richardson

Set document designator number using \docdesignator

Abstract

This paper explores distributed NoSQL database Cassandras performance limitations in an IoT using limited hardware, namely the Raspberry Pi 2. Our aim is to use Cassandras reliable and efficient data distribution to enable distributed exploits on real-time streaming data. At the time of this writing, the proposition of Cassandra on IoT-like hardware carries some development risk for a would-be application developer. This work not only demonstrates that actual operation of Cassandra is possible on Raspberry Pi, but also varies the conditions of operation to serve the expectation management of the variations inherent in new, creative, and cutting-edge applications. This work uses the YCSB not only for its popularity but to catalyze infusion of this research with existing and future research that will fully characterize IoT in the realm of controlled, flexible and resilient data storage. To represent IoT, we vary the memory on a virtual machine among 1GB, 2GB, and 4GB as well as implement Cassandra on the Raspberry Pi platform. Four (4) different YCSB workloads were executed on five (5) different network types: virtual machine equipped with 1GB of RAM on an internal virtual network, virtual machine equipped with 2GB of RAM on an internal virtual network, virtual machine equipped with 4GB of RAM on an internal virtual network, Raspberry Pi's with 1GB of RAM on an Ethernet LAN, and Raspberry Pi's with 1GB of RAM on an 802.11a/b/g/n wireless LAN. Node cluster sizes ranged from 1 to 6. From this, between at least five (5) but no more than seven (7) effects were evaluated: (1) comparison between virtual machine equipped with 1GB of RAM on an internal virtual network and a comparable configuration in another paper, (2) comparison among the effects of varying RAM within the virtual machine environment, (3) effect of varying cluster size given limited hardware, (4)

comparison between the Raspberry Pi node and a network from another paper, (5) comparison between the limited hardware (Raspberry Pi) and the 1GB RAM virtual machine, (6) the effect of cluster size on a 802.11a/b/g/n wireless given limited hardware (Raspberry Pi) nodes, and/or (7) comparison between wireless 802.11a/b/g/n network and a similarly configured wired Ethernet network. This work demonstrates the feasibility and expected performance drops when porting a distributed database like Cassandra from powerful, stationary nodes to less powerful, but more flexible nodes.

THESIS:

TESTING CASSANDRA'S SCALABILITY ON RASPBERRY PI

I. Introduction

1.1 Background and Motivation

There is a trend that information technology and electronics generally become cheaper, powerful, and physically smaller as time progresses, which has led to the ensuing ubiquity of the IoT. Open source software and limited hardware like the Raspberry Pi embody this trend, both in terms of of lowering requisite eclectic expertise as well as price. Databases, key to IoT, are no exception, and it is of interest to explore the trade space involved in porting distributed database technology among such limited hardware. At the time of this writing, distributed databases like Cassandra are normally associated with large capacity nodes.

1.1.1 Potential Challenges.

Using Cassandra on nodes like the Raspberry Pi series in IoT is not without its challenges. First, while software developers may have the luxury of assuming increasingly powerful processing units and limitless memory, an IoT application cannot take such resources for granted. The Raspberry Pi 2 Model B, which will be used in this thesis, has 1GB of RAM available [1]. Storage for a single node depends on the SD card, whose disk storage can vary from 8 GB to 256 GB and whose Input/Output (I/O) data rates can from 2 MB/s to 30 MB/s at the time of this writing. These amounts pale to the 768GB memory and 74.4 TB storage for, say, the Lenovo

Thinkserver RD650 [2] or some other server on the market. The actual performance of software is something you would want to predict prior to making any kind of hardware purchase.

Second, using a node like the Raspberry Pi in IoT can imply a requirement for a finite (primary cell) or periodic (secondary cell) power source. Larger nodes, of course, consume their fair share of power, which comes with its own set of constraints. The prospect of deploying a node in new and unusual places is part of the motivation of porting software to low-power nodes like the Raspberry Pi series.

The prospect of nodes implying less power consumption, less weight, smaller dimensions, and/or maybe even a better price compared to an alternative all do their part to chip away barriers that creative minds would otherwise face in deploying applications for the betterment of mankind.

1.1.2 Potential Benefits.

Giving up computing power for less in power consumption, weight, dimensions, and price also has potential positive implication for innovation management.

Potential Benefits Particular to Supply Chain Management.

In supply chain management, especially in government, there is perpetual interest in items that qualify as Commercial-Off-The-Shelf (COTS). This interest, naturally, ties directly to cost and economies of scale. The more purposes something can serve, the greater demand is likely to be in the free market. This not only save on the actual unit price, but also engineering and administrative costs that may be incurred by triggering the full acquisition process of an application specific integrated circuit. The defense acquisition process is notorious for being overly burdensome, and limited hardware like the Raspberry Pi can represent the potential for rapid or general schedule acquisition.

There is another supply chain benefit as well. Despite security measures, it can be difficult to completely conceal a supply chain from an interested and skilled third party. The more application-specific a device is, the more insight it may give an unsavory adversary knowledge of mission parameters, whether it is physically captured, or if technical requirements leave a leaky paper trail.

Potential Benefits Particular to Wireless.

There is no shortage of options for networking different hosts together, but the ability to exchange a wired networking medium for a wireless networking medium enables increased mobility and flexibility in placement. Wired networking options require cables that may incur installation costs and risk damage when exposed to many environments. Allowing for wireless technology may be a critical enabler for some applications that have yet to be seen. However, along with these potential benefits comes with the risk of decreased or degraded performance. It is helpful to know in advance how similar endeavors have fared.

1.2 Problem Statement

This paper seeks to gain insight into how modern distributed databases operate in an IoT environment and what actions or configurations may be required or recommended to be in place to ensure feasible operation.

1.3 Research Goals and Hypothesis

1.3.1 Characterization of IoT and the IoT device.

Echoing [?], there is no "standard identification" of IoT or an IoT device, although invoking the phrase can identify some archetypes, such as home automation systems or facilities management systems. All of these incorporate transducing some process over time, such as water pressure, steam pressure, temperature, etc, and may provide indication, such as a thermometer or dashboard, and/or actuation, such as actuation of a furnace or pump. This concept and its relationship to limited hardware is further addressed in the Chapter II.

1.3.2 Relationship between Raspberry Pi and IoT?.

The Raspberry Pi is able to receive digital data, audio signals, and video signals, and thus represents some of the computation and storage that could be attached or otherwise networked to one of these transducers. This question will be addressed in Chapter II through examination of current published work as well as commercial specifications.

1.3.3 Other Representatives of Limited Hardware Platforms?.

The Raspberry Pi has no shortage of competitors. These will be briefly examined in Chapter II. In addition, Chapter IV will evaluate a virtual machine set up to emulate limited hardware.

1.3.4 Relationship between distributed databases and IoT.

Section 2.5.1 explains the current use and the benefits of porting said operation to a thicker client.

1.3.5 Networking Considerations in IoT.

While IoT may imply the internet protocol (IP) at the network layer, there is no shortage of options at the data link and physical layers. Chapter II will discuss some networking considerations and explain some of the background behind varying between data link layers.

1.3.6 Feasibility of Distributed Database on Limited Hardware.

First, there is the question of whether a given distributed database will work on a given hardware platform or not. Any one-off results at a nonzero measure of performance indicate feasibility, but replicating operation and slightly varying the conditions gradually increases the confidence that such feasibility can be extended to yet untested environments. With the distributed database being represented by Cassandra, and limited hardware being represented by virtual machines as well as the Raspberry Pi, the experiments in this work aim toward refining the answer to this question.

1.3.7 Distributed Database Selection.

Section 2.1.1 explains the reasoning behind selecting Cassandra as the representative of a distributed database.

1.3.8 Benchmark Tool Selection.

Section 2.3.2 addresses the reasoning behind the selection of YCSB.

1.3.9 Does the amount of RAM affect Cassandra's performance?.

Chapter IV describes the empirical method used to determine an effect with respect to varying memory.

1.3.10 Platform Testing, Scalability Testing - What are the implications of using limited hardware?.

Varying hardare platforms implies varying a variety of factors closely related to performance: Central Processing Unit (CPU), RAM, networking interfaces, and hard disk interfaces. Chapter IV describes the empirical method used to measure the effect of porting Cassandra between a virtual machine the Raspberry Pi, as well as measuring the effect of cluster size. A common selling point for distributed databases is an ability to accept additional nodes for storage, as opposed to say, more storage in-situ.

1.3.11 Link Layer Testing - What effects on performance result from varying networking schemes?.

Chapter IV describes the empirical method used to measure the effect of switching between Ethernet links and 802.11a/b/g/n wireless links, as well as the scalability implied given 802.11a/b/g/n links.

1.3.12 Sensitivity Testing - How does variation in the configuration of a distributed database affect performance?.

Different applications may imply different configurations associated with distributed databases, including compression strategies and replication factors. Chapter III does some investigating into how these can affect performance. Depending, some configurations may be elected to compensate for or otherwise diminish the negative performance affects of network configuration.

1.4 General Approach and Research Activity Overview

The general approach to this will be to implement a scientific methodology for understanding the effect of inherent aspects of IoT networks on factors that limit performance of distributed databases. We are particularly interested in the effects of low memory and processor speed, limited bandwidth and scalability on IoT networked devices. In general, this study follows a template that includes varying configuration and environment settings, performing stress testing, measuring results, and interpreting the results to form a conclusion.

This paper will apply the YCSB benchmarking tool to gauge performance changes over variation in the following: keyspace configuration, network configuration, platform choice, and node scaleup.

1.5 Assumptions and Limitations

Assumptions and limitations are detailed in Chapter IV.

1.6 Expected Contributions

Aggregating lower-cost, lightweight hardware spawns a lingering question of possibilities and performance due to lower barriers to proliferation. With distributed database Cassandra representative of application, and the Raspberry Pi a representative of low-cost hardware, we explore the performance of a distributed database over Raspberry Pi networked clusters. This paper asserts the following contributions:

1.6.1 Methodology.

First, this paper develops a methodology to leverage existing tools [3] to evaluate a NoSQL distributed database in IoT. This will expand on such work as [4] and

[3] developing a test methodology to explore the limits of Cassandra, and how its performance is affected by the number of nodes, nature of hardware, and links.

1.6.2 Performance Implications of Limited Hardware.

Second, this paper reports results that suggest and characterize a differential between a simulated environment for IoT using virtual machines and an internal network and a physical one using Raspberry Pi's and an Ethernet network.

1.6.3 Performance Implications of Link Layer Variance.

Third, this paper reports results that suggest a predictable performance cost associated with exchanging an 802.11a/b/g/n wireless links with those of a wired Ethernet network.

1.7 Organization

Chapter II paints the relevant parts of the landscape that have emerged as a result of much interest in distributed databases, limited hardware, and IoT. We also present the gaps of the current literature to describe the technical goals for the current work.

Chapter IV presents a methodology to examine and evaluate the performance implications of varying hardware, network, and cluster size.

Chapter **??** describes the results and evaluation of the methodology presented in Chapter IV.

In Chapter VI, this study is brought to a conclusion and future work is discussed.

II. Background and Related Works

2.1 Representative Technologies

2.1.1 Cassandra.

A distributed database offers a trade space among consistency, availability, and partition tolerance. Mostly this means that a system of nodes can still operate as expected even if there is a loss of one or two nodes. There is no shortage of distributed databases to choose from, but Cassandra is widely used and is known to have a high write throughput [?]. Cassandra specifies the latest versions of both Java 8 and Python 2.7. In turn, Java can run on Windows, Mac OS X, Linux, and Solaris [5].

Cassandra is a widely used distributed Not Only Structured Query Language (NoSQL) database with many use cases [6] and boasts a high write throughput. Not only has Cassandra been reportedly been used in practice [7], but has been, using the Yahoo Cloud Services Benchmark [3], formally evaluated in scholarly literature against other databases such as MongoDB and proposed as the NoSQL database of choice in the Internet of Things and distributed sensor networks [8].

If possible, it is important to get realistic expectation from available specifications whether or not the database of interest, Cassandra, would be supported by the node of interest, in this case Raspberry Pi. At the time of this writing, Cassandra specifies it is supported by the latest versions of both Java 8 and Python 2.7 [5]. In turn, Java can run on Windows, Mac OS X, Linux, and Solaris. Raspbian, a Linux-based operating system, can be run on Raspberry Pi. In addition, there have been credible claims of Cassandra being used on Raspberry Pi [9, 10], but to the author's knowledge, no white paper with the details exists.

This author's interest in Cassandra lies in the fact that Cassandra is a distributed database used in practice for cloud computing. Although it describes itself as a

NoSQL database, the interface allows for Structured Query Language (SQL) commands and has a Python Application Program Interface (API). Cassandra allows for configuration of distributed systems parameters, such as replication factor, but detailed knowledge of distributed systems protocols is not critical for operation.

The aim of this paper is to examine Cassandra's performance coupled with a simplified Wi-Fi collection and analysis application, where the nature of link nodes may be less reliable than wired Ethernet.

From an experimental standpoint, the distributed nature of a Cassandra "keyspace" lies in four parameters [11]: cluster size (the number of nodes), replication factor (configured in software), write level (configured in software), and read level (configured in software). As alluded to in section **??**, these factors will be held constant for this experiment.

2.1.2 Raspberry Pi 2.

The Raspberry Pi 2 (Model B) [1] is a low-cost computer designed sold from the United Kingdom. It can be described as a motherboard for a about the size of a 3x5 index card and has been available since February 2015. This experiment is interested in the Raspberry Pi 2 as a representative of the low-cost hardware domain, which implies low cost, low power consumption, and low in terms of size and weight.

This author's interest in the Raspberry Pi 2 is that its ARM Cortex-A7 processor and 1GB RAM [1] makes it a key representative of the low-cost hardware domain and IoT. The Raspberry Pi 2 has cost as low as 35 USD [12]. It is lightweight and has limited power consumption [1]. Constraints on size, weight, power, and cost can all be barriers to entry for applications seeking computing nodes.

Expanding this experiment, to say the BeagleBone black [13], is in touch with the spirit of this experiment but outside the scope of this paper and is reserved for future work.

2.2 Small Cluster Computing

Considering the educational purpose of Raspberry Pis, it is no surprise to find academic interest in Raspberry Pi clusters. For instance, [17] evaluates a cluster of 8 Raspberry Pi nodes using SysBench.

Existing Raspberry Pi clusters, built to serve as a "practical balance" [18], such as in [19] and [18], suggest the value of Raspberry Pi nodes compared to large, traditional servers both in terms of power construction and actual purchasing price. In [20], Cassandra is used to store videos for a video streaming application on after a "a lot of configuration", albeit the configuration parameters were unspecified. However, [20] shows a high index of suspicion that Cassandra can be used in a small cluster environment.

2.3 Benchmarking Distributed Databases

2.3.1 Benchmarking Tools, Stress Testing Tools.

Benchmarks are the common parlance for a way to test a computing system's capabilities. There has been a lot of interest in testing distributed databases, databases that cover multiple nodes.

cassandra-stress.

In this experiment, the "cassandra-stress" [21] tool is used. Documented use includes [10], which, could be used as an anchor for methodology. This tool is developed along with, and is optimized for Cassandra. However, it falls short if one desires to compare two different databases for the same task.

Name	CPU	RAM	DISK	Price
Banana Pi M3 [14]	Octa-core 1.8GHz CPU	2 GB RAM	8 GB eMMC flash storage	73.00
CHIP [15]	R8 1GHz	512 MB	4 GB	9.00
VoCore	360 MHz MIPS CPU	32 MB	8 MB Flash	20.00
Arduino INDUSTRIAL 101 []	Atheros AR9331 Processor	64 MB	16 MB Flash	40.00
NanoPi 2 Fire []	Samsung S5P4418 quad-core ARM Cortex-A9, 1.4 GHz	unk	1 GB MicroSD card	22.99
NanoPC-T3 []	Samsung S5P6818 octa-core ARM Cortex-A53 up to 1.4 GHz	1-2GB of RAM	8GB of flash storage	60.00
Intel Edison with Kit for Arduino	Dual-core, dual-threaded Intel Atom CPU with a 32-bit Intel Quark microcontroller	1GB of RAM	4GB of flash storage	92.00
cloudBit []	Freescale i.MX23 ARM926EJ-S processor	64MB of RAM	4GB SD Card	59.95
Parallella [16]	16-core Epiphany RISC SOC	unk	unk	unk
Zynq SOC	(FPGA + ARM A9)	1GB SDRAM	Micro-SD storage	99
PixiePro	Freescale i.MX6Q Soc Quad Core ARM Cortex-A9 up to 1GHz	2GB of RAM	SD Card	129.95
Raspberry Pi	900 MHz quad-core ARM Cortex-A7 CPU	1GB RAM	MicroSD Card	35.90

Table 1. Raspberry Pi Alternatives, Depending on Application

2.3.2 YCSB.

Paper [?], presents the YCSB, highlighting "scaleup" and "elastic speedup" as parameters for benchmarking. It provides a survey of five databases: PNUTS, BigTable, HBase, Cassandra, and Sharded MySQL. As might be expected, Cassandra has the ability to be tuned based on the application, data distribution, and workload type. In [?], they claimed to "[tune] each system as best [they] know how." In contrast, this paper will attempt to identify any tuning parameters that have been modified from the default. It is also worth noting that the version of Cassandra has evolved from year 2010, the time [?] was published.

The YCSB provides six pre-defined workloads and also allows one to determine a custom workload.

2.3.3 Works Evaluating Cassandra.

Cassandra was shown in [8] to be favorable to write-heavy workloads compared to another database in the domain. Notable about this paper is that the paper scales the node's RAM down to 2GB, compared to higher powered machines in other papers such as the Cooper paper [?] or as specified on the website [5]. Although it is not explicitly mentioned as an interest in the paper, this shows a transition of using Cassandra for lower powered machines. One thing that is not clear in [?] is how cache

Workload	Description	Breakdown
A	Update heavy workload	50/50 reads and writes
B	Read mostly workload	95/5 reads/write
C	Read only	100% read
D	Read latest workload	95/5 reads/writes
E	Short ranges	95 scans / 5 inserts
F	Read-modify-write	50 read / 50 read-modify-write

Table 2. This table describes the predefined workloads available from the YCSB

effects are accounted for. If unaccounted for, a cache effect may result in the initial run resulting in longer execution times than subsequent runs, all other factors being constant. The key cache is set at 100 MB and the row cache at 0. In contrast, this paper clears the data from (or truncates) the table of interest. [?] mention that each 7200 rpm with no stated limits on hard drive space. Moving into the realm of in-situ storage, this paper takes a significant deviation in limiting the hard disk space to 8 or 16 GB.

Benchmarking on Limited Hardware.

Restricting the database schema to time series data, [?] claims to have achieved about 4 million inserts on a relational database on a Raspberry Pi 2 B+, one of the same models used in this work.

2.4 Networking Considerations

Although this paper takes an empirical, top-down approach to evaluation, some networking concepts may be useful to parse out, and predict what they mean to this work. This work only varies parameters at the link level. All terminology follows from [?].

2.4.1 Physical Layer Considerations.

This section will address delay in a local area network implied by the physical layer. Explanations are paraphrased from [?]. Variation in the physical layer may imply variation in end-to-end delay of a message composed of the following: Queuing Delay, Propogation Delay, Transmission Delay, and Processing Delay. Propogation delay is the most predictable, and reflects on the physical properties of the transmission medium. Transmission delay is dependent upon the message length and the transmission rate. In these experiments, the Maximum Transmission Unit (MTU) is set to 1500 on the router, so the message length would not change between Ethernet and 802.11a/b/g/n. However, the transmission rates do differ between Ethernet and 802.11a/b/g/n. Queuing delay, the outgoing delay, and processing delay, the incoming delay are not as predictable.

2.4.2 Link Layer Considerations.

Ethernet Protocol Considerations.

Ethernet is the dominant technology in modern wired LANs. Ethernet uses carrer sense multiple access with collision detection (CSMA/CD) and often uses switches.

802.11 Protocol Considerations.

The 802.11 protocol avoids collisions one of two ways. For shorter data frames, after sensing a channel idle for a specified interval called the Distributed Inter-frame Space (DIFS), the source sends the data frame. Then, it waits for an Acknowledge (ACK). For longer data frames, instead of sending the data frame right away, the source might send an Request-To-Send (RTS). This threshold can be set on the router, and in the case of these experiments it is set high, such that the RTS is never employed.

A key difference between the 802.11 protocol, or really any wireless protocol, is can be referred to as the 'hidden terminal problem' [?]. If nodes are spread out, it is possible for one node to miss interference caused by another node. A transmission source waits for an acknowledgement or retransmits. Contrasted with CSMA/CD, which does not require an acknowledgement from the receiver, this has the potential to cause greater latency.

2.5 Potential Uses

2.5.1 Application Space.

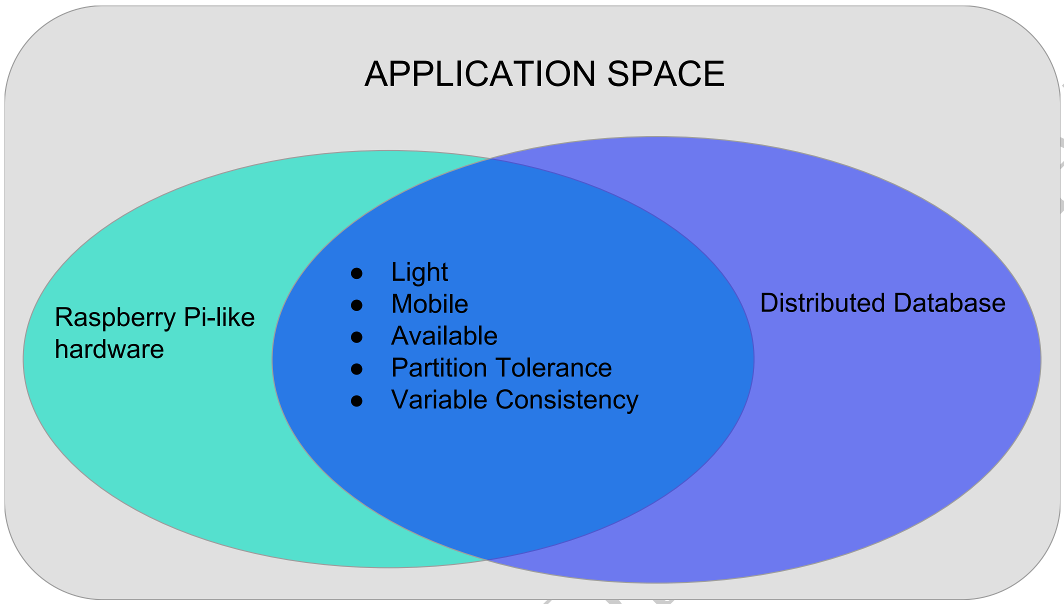

Figure 1. Venn Diagram of Application Space. The application space of interest takes advantage of some combination of lightness in weight, mobility, availability, partition tolerance, and variable consistency.

At the time of this writing, back-end database operating on powerful, but stationary nodes dominate documented use cases of Cassandra, as well as other distributed databases. This paper explores the idea of moving toward a more in-situ database, where the sensing nodes may also contain serve as mechanisms for storage and do not require a pipeline to another datacenter, a pipeline that from an operational perspective, represents a single point of failure, or from a security perspective, a threat to privacy.

The next few sections map this general concept to some particular applications.

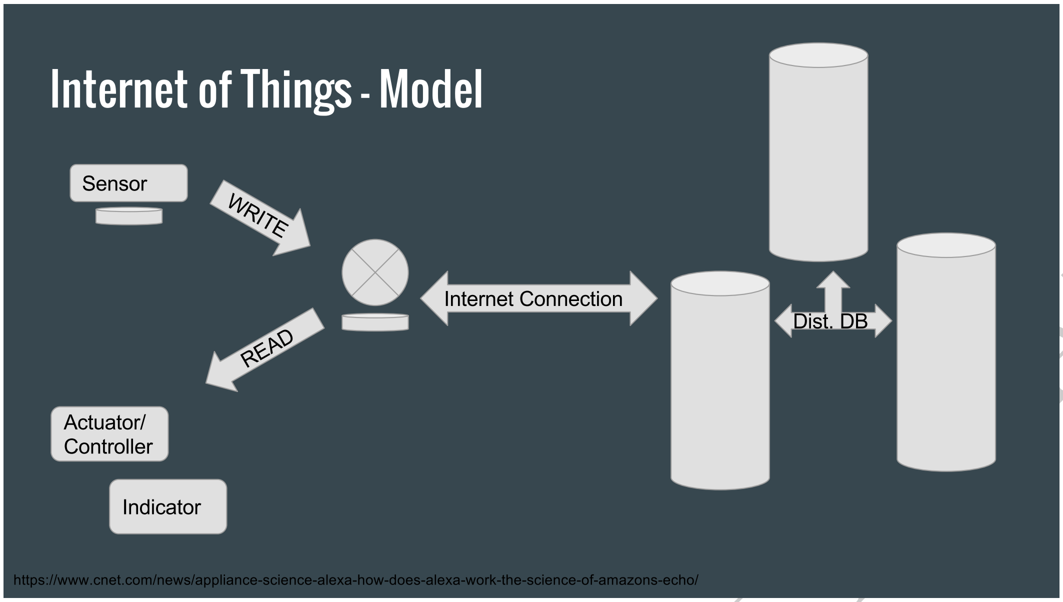

Figure 2. IoT This represents what for some might be a nominal "IoT" application. Some measurement of interest, such as temperature is sensed, that raw data is stored and sent afar, and if desired, the user may also query for some kind of feedback or indication.

2.5.2 WiFi Collection, Mapping and Analysis.

WiFi Sniffing, Collection.

WiFi (802.11) sniffing, or war-driving, has been explored by a number of enthusiasts, both in and out of the academic realm. These techniques for WiFi sniffing, at least on the front-end, is well-documented and thus represents a low barrier to entry for initial operations, often just requiring a specific WiFi chipset, such as the ALFA card, and open-source software. One example that has been well-documented goes by "Snoopy" [22]. Snoopy provides a framework for collecting WiFi data and observing with another handy piece of software, Maltego. Of course, a multitude of other software exists. For example, one can sniff wireless traffic by using software Aircrack-ng [23] or Airsnort [24]. However, although sensing the data is easy, the subject of storage has not been fully explored with respect to the selling points mentioned above. Investigating the limits of storage operations is key to unlocking realistic aspirations

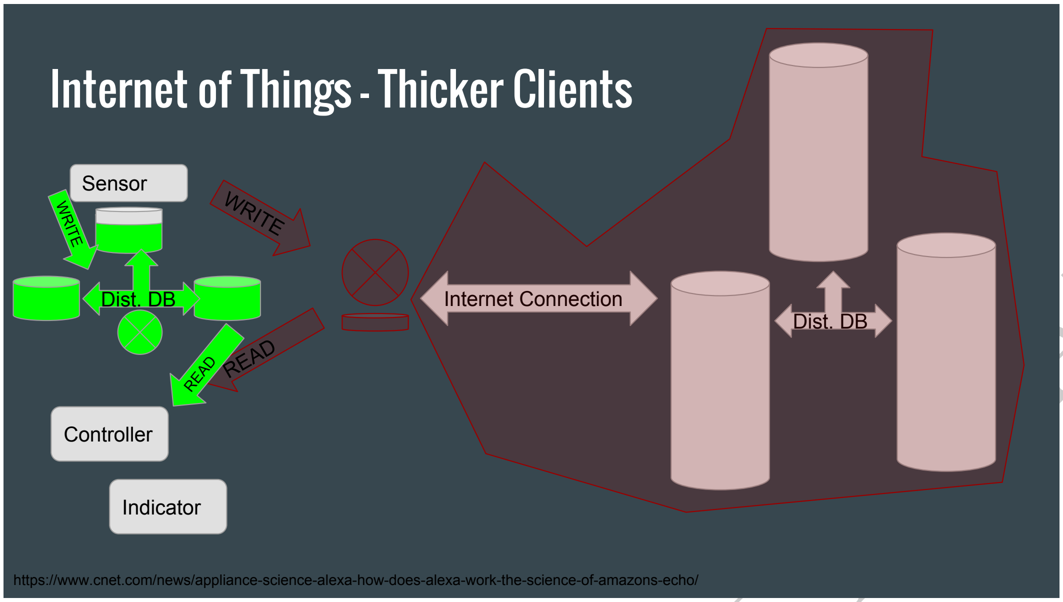

Figure 3. IoT Application with Thicker Clients...Depending on the application, using a distributed database on thicker clients in-situ may suitably replace a remote database operating in a remote location. From an operational standpoint, the pipeline may be unreliable, resulting in a loss of data. From a security standpoint, this pipeline may be subject to undesired third-party monitoring.

and application development.

WiFi Mapping.

It may worth noting one off-shoot of the WiFi sniffing mentioned above: WiFi mapping. There is much interest in WiFi mapping: it implies extra assurance to existing technologies that may be overly taken for granted. There has been much work done with respect to WiFi mapping. Argos [25] describes a similar system of a distributed system, but there is no mention of Cassandra or any distributed database, which may serve as an improvement on such a sensor network. Wigle.net [26] is an aggregate map that distributes a smart-phone application to collect GPS coordinate-Access Point pairs, but relies on a central database, and has an unreliable user base and irregular sampling and update frequencies (Relying on the public will often do that, unfortunately.). Heat Mapper [27] is partially free and commercial software that can generate a heat map for a small room or office. Wi2Me [28] performs this mapping as well, with an emphasis on performance and data throughput. It uses an instance of SQLite to store the traces on the individual's smart-phone, but again, none of these make use of a distributed database like Cassandra as part of the sensor network. Once again, this is a realm that may benefit integration with distributed databases, expanding the range possible node types, or both. However, there are limits that have to be anticipated. This paper aims to add a piece to that expectation management.

WiFi/Wireless Crowd Detection.

There is also considerable interest in crowd detection and related data gathering. Privacy implications notwithstanding, this kind of data can give a marketing-type a sense if certain areas are popular or versus other areas, or if certain paths are well-traveled or not. The way WiFi broadcasts Service Set Identifier (SSID)s in plain text, crowds can even be characterized as to whether individuals connect to similar access points or not. The data can indicate whether you have a crowd of people from a certain country, a crowd of people who know each other or maybe just a crowd of strangers. It can add assurance to more primitive forms of tracking, such as person-by-person registration. Informally, some have even reported to make art exploiting this mechanism [29], and not surprisingly, there are numerous claims that members of the public are routinely tracked via commercial entities via WiFi [30]. There have been more academic efforts to track crowds as well, notably a university campus and a music festival in [31] and an airport in [32]. In [31], data travels through Global System for Mobile Communications / Groupe SpecialMobile (GSM) to a central node, but does not use any kind of distributed database, like Cassandra. There are commercial entities that claim to track crowds and report data, namely "Bluemark" [33]. Here as well, a central server is utilized to collect the data. Users then reportedly log into the web to view the metrics. According to their marketing literature, they do use Raspberry Pi, but not for data storage. Paper [34] used this product line for their tests. Although WiFi utilizes Media Access Control (MAC) addresses supposedly unique to each device, research has found that this leaves more to be desired for many applications, namely crowd-tracking. For instance, some devices have been reported to change their MAC addresses [34]. There exist a number of papers that explore characterizing mobile devices and their users at the individual level [35, 36, 37, 38, 39, 40, 41] and propose techniques to better prepare data for analysis [34]. What is clear in the subtext of all these papers, is that these types of application research can only benefit from a wider choice in nodes, and a wider choice storage options as well as a reasonable amount of foresight into their performance. In all of these, however, distributing the stored data among the nodes, like with Cassandra is either not used or not mentioned. Many utilize a central server that represents a single point of vulnerability.

2.5.3 CBIR and others and summary statement on application space.

The motivation for WiFi mapping can also be extended to other types of mapping, such as Content-based image retrieval (CBIR). Paving the way for feasible, desirable in-situ data storage increases the portability and development of these and many more applications.

The experiments that follow read and write random data in the context of a generic schema, but the experiments were done with the following in mind: A schema supporting any of these applications could be substituted in instead, and ideally with predictable results.

III. Cassandra Pilot Tests

This section will demonstrate some of the variance that is possible by varying configuration of a cassandra keyspace itself. These configurations may differ from application to application.

Cassandra comes with its own stress-testing tool, denoted 'cassandra-stress' [21] in the tarball installation. By all appearances, it is designed to give an operator an idea of how Cassandra performs over time on a given hardware setup. But, it also allows an operator to vary a limited set of configuration parameters with each command line execution. Naturally, this stress testing tool does not allow one to compare Cassandra to a competitor, hence the YCSB in the main section of this paper, but Cassandra-stress may also serve as a check on the YCSB.

3.1 Experimental Setup

The set-up follows the guidelines established in the main methodology. The set of experiments described in this section explored how Cassandra performs. Here a cluster of 3 Raspberry Pi 2 nodes was used. The tarball version of Cassandra version 3.9 was downloaded and installed. The stressor application cassandra-stress was executed from a laptop also connected to the LAN.

3.2 Variance in Nature of the Links with Compression Algorithms

The plot in 5 shows the effect of varying compression strategies for a given configuration on a pure write load as load-tested through the cassandra-stress module. In the left, wireless 802.11 links were used. On the right, wired Ethernet links were used. A one-way analysis-of-variance (ANOVA) may be able to test whether there is a significant differential between either of these two means, but from visual inspec-

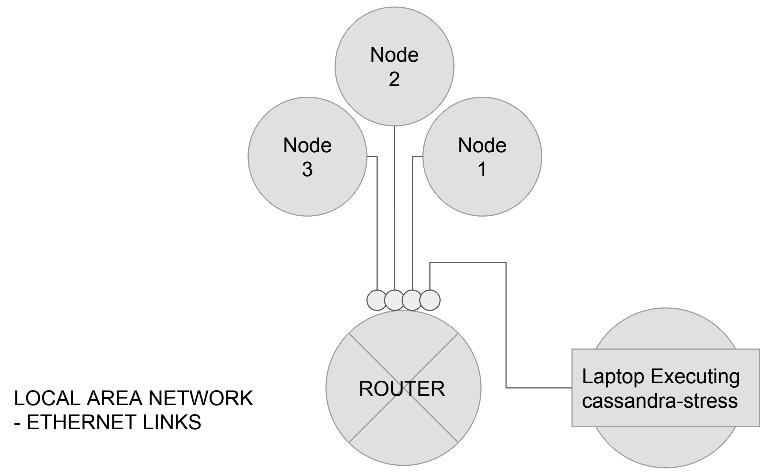

Figure 4. Experimental Setup for cassandra-stress Tests

tion, one could say that for a given configuration and a write-heavy load, varying the compression strategy does not have a large effect on performance as far as writes per second.

To represent a read-heavy load, a pure read load was put through cassandra-stress, varying both compression strategies and wireless versus wired link nature. These results are graphed in Figure 6 In the wired domain (right), a hierarchy seems to emerge. The LZ4 compression algorithm renders the most reads per second, followed by the Snappy compression algorithm, and finally the Deflation compression algorithm. This correlates with the expectations put forth by the manual, as the selection in compression algorithms represents a trade-off between speed (operations per second) and compression effectiveness (storage in bytes) for the user. Such a hierarchy does not seem to emerge from the wireless links, but there is no known explanation for this at this time. Increased variation is not unexpected, perhaps due to industrial, scientific, and medical (ISM) band interference, but one would expect the means to follow a similar pattern. A series or regressions or multi-factor Analysis

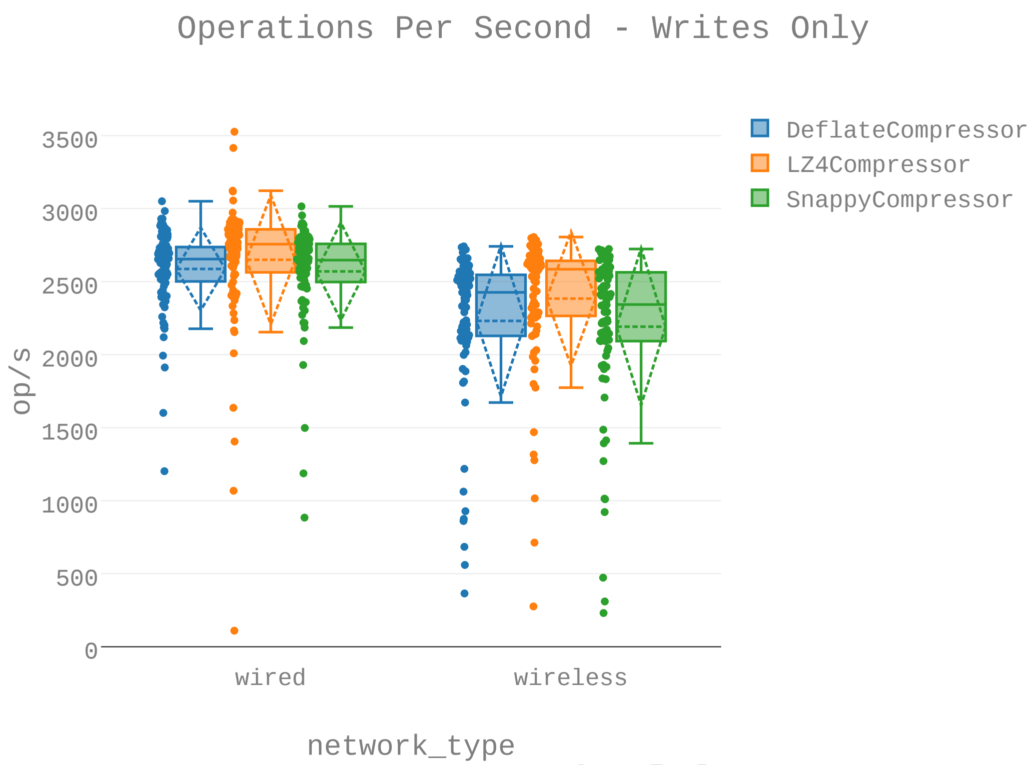

Figure 5. Varying Compression Methods: Writes ... Results are grouped by wired and wireless LAN configurations. From top to bottom, the solid horizontal lines of each box represent the maximum, 75 percentile, median, 25 percentile, and minimum. The dashed horizontal line represents the mean. The oblique dashed lines represent the standard deviation. The plotted points to the left of each box represent the actual data points.

of Variance (ANOVA) test could be utilized to see if the graph is deceiving and the means truly are significantly different, but is not merited at this time.

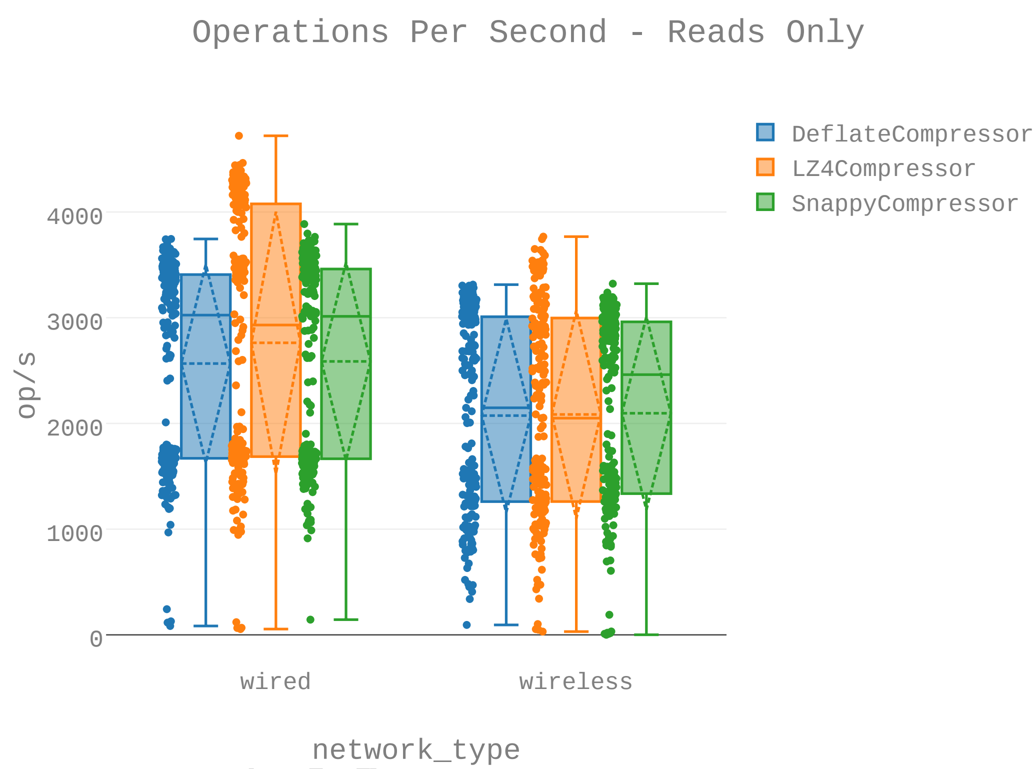

Figure 6. Varying Compression Methods: Reads ... Results are grouped by wired and wireless LAN configurations. From top to bottom, the solid horizontal lines of each box represent the maximum, 75 percentile, median, 25 percentile, and minimum. The dashed horizontal line represents the mean. The oblique dashed lines represent the standard deviation. The plotted points to the left of each box represent the actual data points.

IV. Methodology

4.1 Overview of Similar Experiment

Although this work has a slightly different aim than [4]'s stated purpose, this paper aims to follow [4]'s methodology closely enough as to anchor its results to a cross-section of similar work that has been done. This work assumes that an in-situ storage application in the realm of IoT implies a small database, in this case represented by 1 million records, as opposed to 10 million or 100 million or more. Although [4] seems to imply there is feasibility for large database with many, many nodes given the right balance, this work focuses more on the initial impact to performance of introducing limited hardware in order to lighten costs or actual physical weight for an application that would see this as a benefit.

Using standard workloads A, C, and E from the popular YCSB, the authors of [4] examined and evaluated Cassandras scalability over database [sizes] and cluster sizes [4]. The authors found that this trend, depicted in Figures 7, 8, and 9 did not necessarily hold true across database sizes, that in fact for larger database sizes of 10 million and 100 million records, 3 node clusters performed better than both a single node cluster and a 6-node cluster. The authors concluded that for sufficiently small databases, which is the likely case for IoT, more nodes imply more time to execute, which overwhelms any advantageous parallelism that may ensue with increasing nodes.

Because there are so many variables that can be at work, this work aims to anchor its results by replicating part of Abramovas study. The extent of the details of the network in [4] is detailed below:

"The characteristics of nodes used are, as follows: Node 1 Dual Core (3.4 GHz), 2GB RAM and disk with 7200 rpm; Node 2 Dual Core (3.4 GHz), 2GB RAM and disk with 7200 rpm; Node 3 Dual Core (3.4 GHz), 2GB RAM and disk with 7200 rpm; Node 4 Dual Core (3.0 GHz), 2GB RAM and disk with 7200 rpm; Node 5 Dual Core (3.0 GHz), 2GB RAM and disk with 7200 rpm; Node 6 Virtual Machine with one Core (3.4 GHz), 2GB RAM and disk with 7200 rpm." [4]

The results of Workloads A, C, and E in [4] are depicted in Figures 7, 8, and 9 respectively, all depicting a positive correlation between execution time and the number of nodes. The actual values are depicted in Table 3.

4.2 Objective of This Set of Experiments

The objective of this experiment is to characterize varying configurations for Cassandra. This characterization will be in service of assessing the utility of Cassandra on the Raspberry Pi 2, which will in turn be an indicator toward the greater population of both distributed databases, archetype Cassandra, and hardware of archetype Raspberry Pi.

We aim to recreate results in [4] and then extend these experiments for a better characterization for IoT. In doing so, we answer the following research questions:

- Research Question 1: How much is the execution time for a given number of op-

Nodes	Workload	[OVERALL] RunTime(ms)
1	A	58430
3	A	65650
6	A	87310
1	C	88000
3	C	90210
6	C	118090
1	E	223180
3	E	330820
6	E	404660

Table 3. Results from [4] for 1 million record size database

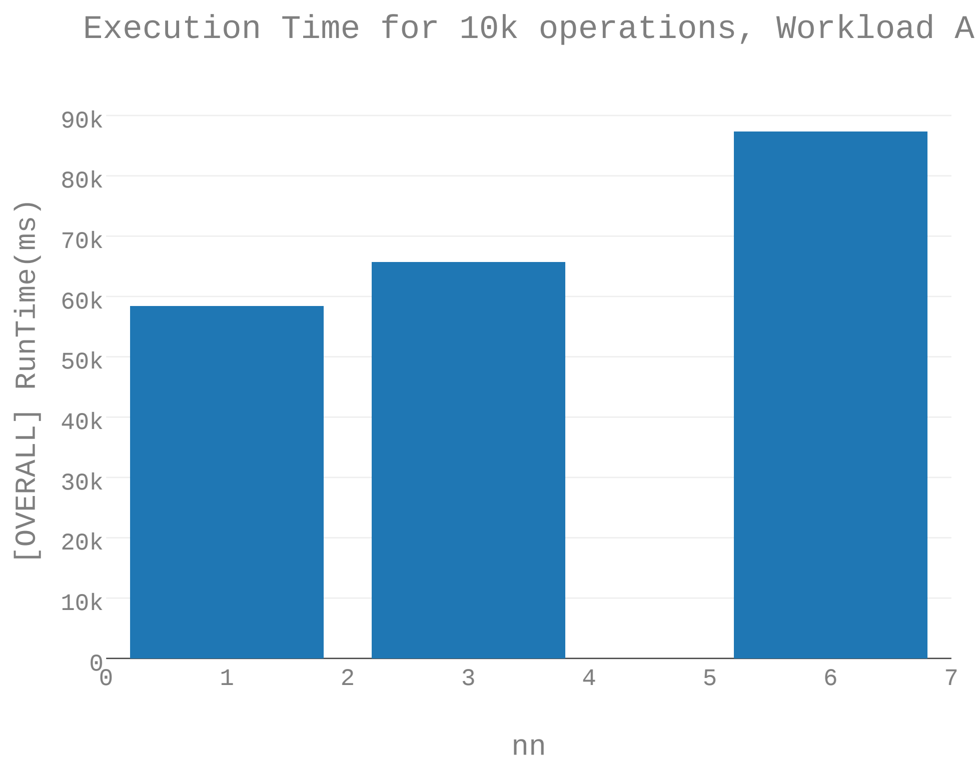

Figure 7. Cross Section of Results Imported from [4]. This figure shows the execution time reported for 10,000 operations of Workload A over three given configurations: a network with only one (1) node, a network with 3 nodes, and a network with 6 nodes.

erations extended in the system under test using workload A (reads/updates)?

- Research Question 2: How much is the execution time for a given number of operations extended in the system under test using workload C (reads)?

- Research Question 3: How much is the execution time for a given number of operations extended in the system under test using workload E (insert/scans)?

- Research Question 4: And finally, how much is the execution time for a given number of operations extended in the system under test using custom workload I, the anticipated representative of in-situ distributed database IoT application?

To answer each of these questions, we pose the following questions:

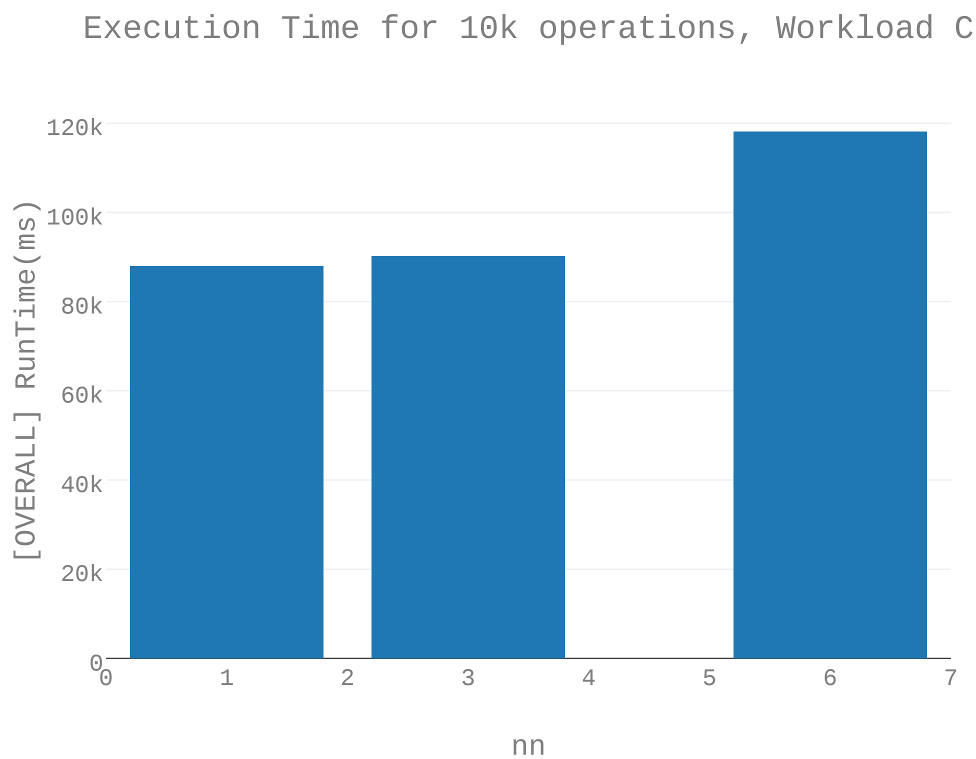

Figure 8. Cross Section of Results Imported from [4]. This figure shows the execution time reported for 10,000 operations of Workload C over three given configurations: a network with only one (1) node, a network with 3 nodes, and a network with 6 nodes.

- How do the results from virtual machine compare to the corresponding values from [4]?
- How do the results compare among the various RAM levels assigned to virtual machines?
- How do the results from the limited hardware on an Ethernet LAN scale as the cluster size increases?
- How do the results from the limited hardware on an Ethernet LAN compare to the corresponding values from [4]?
- How do the results from the limited hardware on an Ethernet LAN compare to the corresponding virtual machine results?

Figure 9. Cross Section of Results Imported from [4]. This figure shows the execution time reported for 10,000 operations of Workload E over three given configurations: a network with only one (1) node, a network with 3 nodes, and a network with 6 nodes.

- How do the results from the limited hardware on a wireless LAN scale as the cluster size increases?
- How do the results from the limited hardware on an Ethernet LAN compare to that of the limited hardware on a wireless LAN?

4.3 Expectations

The results of these experiments are anticipated to be within reasonable bounds of previous work [4]. Increasing the cluster size will result in an increase in execution time. Wireless links will cost significantly in performance compared to their wired counterparts.

4.4 System Boundaries

Virtual machine nodes were contained in a single laptop. The wired LAN experiments were performed with all nodes and the associated router within about 3 meters of one another on a dedicated LAN. Likewise, the wireless LAN experiments were performed with all nodes and the router within a radius of 3 meters on a dedicated and secured LAN. The experiments were done in a residential, suburban area. The network was periodically monitored for unexpected hosts.

4.5 Experimental Limitations, Nuisance Factors, Known/Suspected Interactions

The amount of interference on the ISM band could not be controlled. It is left to the assumptions that any variation in interference was negligible between any two trials.

The laptop running the YCSB may have been running other minor programs or other processes to a limited extent. No experiments were done to determine if this would have a significant effect, and this was not strictly controlled.

4.6 Coordination

4.6.1 Ethernet LAN.

No coordination was needed. This network was physically isolated and no additional traffic was expected to interfere with it.

4.6.2 Wireless LAN.

Because the scale and timing of this experiment was limited, there was no formal coordination needed. However, because a larger experiment could possibly fill up one (or more) frequency channels, one must be courteous of the environment. An extended test would not be appropriate in an uncontrolled environment.

4.7 Treatments, Independent Variables

The independent variables of interest are the node type and link type. Node type is characterized by memory, or RAM, processor speed, and I/O rates, and are represented by virtual nodes and the Raspberry Pi.

The link types are the internal nodes on the virtual machine network, Ethernet links, and wireless 802.11 links.

There are many other factors at work, but other factors, like workload, are only varied to give appropriate context to the variance in node type and link type.

4.8 Factors

4.8.1 Hard Disk Storage: SD Card.

The manufacturer and model for each SD card was kept constant for each node. The details can be found in Table 4.

4.8.2 Database Size.

This work assumes that an in-situ storage application in the realm of IoT implies a small database, in this case represented by 1 million records, as opposed to 10 million

Specification	Value
Capacity	16 GB
Read Speed	up to 90 MB/s
Write Speed	up to 40 MB/s
Video Speed	C10 U3

Table 4. Specifications for SD Cards [?]

or 100 million or more. Although Abramovas paper seems to imply there is feasibility for large database with many, many nodes given the right balance, this work focuses more on the initial impact to performance of introducing less-capable hardware in order to lighten costs or actual physical weight for an application that would see this as a benefit.

4.8.3 Number of Operations Per Trial.

The results in [4] report an execution time after 10,000 operations, and the decision was made to keep this constant rather than vary the representative number of operations. No explicit justification is given in [4] for the number 10,000, but it can be inferred that is a compromise between under-sampling and an undue burden in time spent sampling.

As a brief exploration into the possbility of under-sampling, one can take a look at what it might look like to vary the sample size in a given trial, and how the execution time would vary over the sequence of trials. The graph above shows this for two different configurations: 1GB and 4GB RAM on a virtual machine. As expected, the execution time does reflect initial cache warm-up time, but then, more or less, reflects some sort of steady state, albeit oscillating performance. It was also not made explicit whether or not the 10,000 operations represented one of many trials of 10,000 or represented a single and only trial. For the purposes of exploring the data, this author chose to run 30 trials of 10,000 operations each.

The oscillating behavior seen in trials 5 through 120 of Figure 10 is distracting and risks inaccurate comparisons among configurations. Although it is beyond the scope of this paper to determine the exact cause of this oscillation, one might consider that activities required for the operation of the database, such as compaction, the gossip protocol, and other operations compete with the reads and will contribute

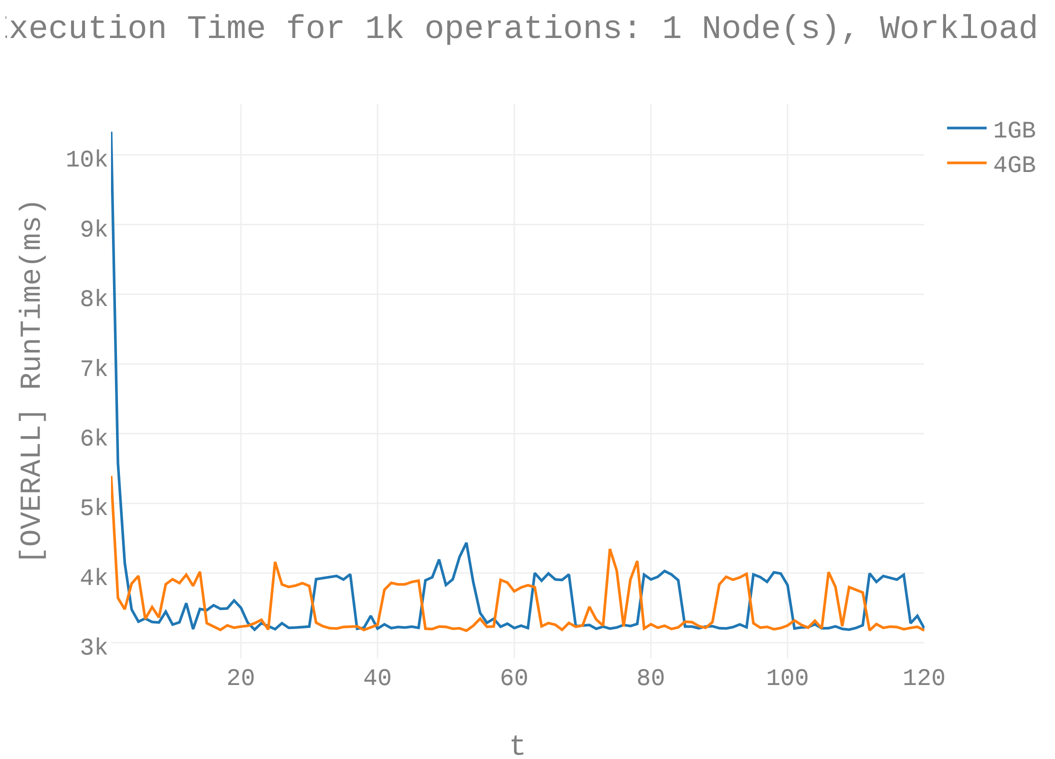

Figure 10. Two sample runs, adjusting the trials such that only 1000 operations of Workload A to a given trial. Once the cache is warmed up, it can be seen that for both series, there is a periodicity present with respect to the progress of trials. In an experiment concerned with steady state performance and overall trends, such oscillation may be an undue distraction.

to continuous variation over time, and may affect performance measurements. The reason for choosing 10,000 operations is not explicitly reported in [4], but one may infer the reason is to integrate this variation to make better comparisons among configurations.

Contrast the 1k operation trials with the 10k operation trials depicted in 11. Here, at least with the naked eye, there is no oscillation, and most certainly not to the extent seen in the 1k trial case. Whatever effected the oscillating behavior in the previous graph is integrated into the 10k trial case and is no longer a distraction to steady-state analysis, as each 10k operation trial soaks up, or rather integrates, the performance of quantity 10 1k operation trials. Of course, more operations take longer to run, and

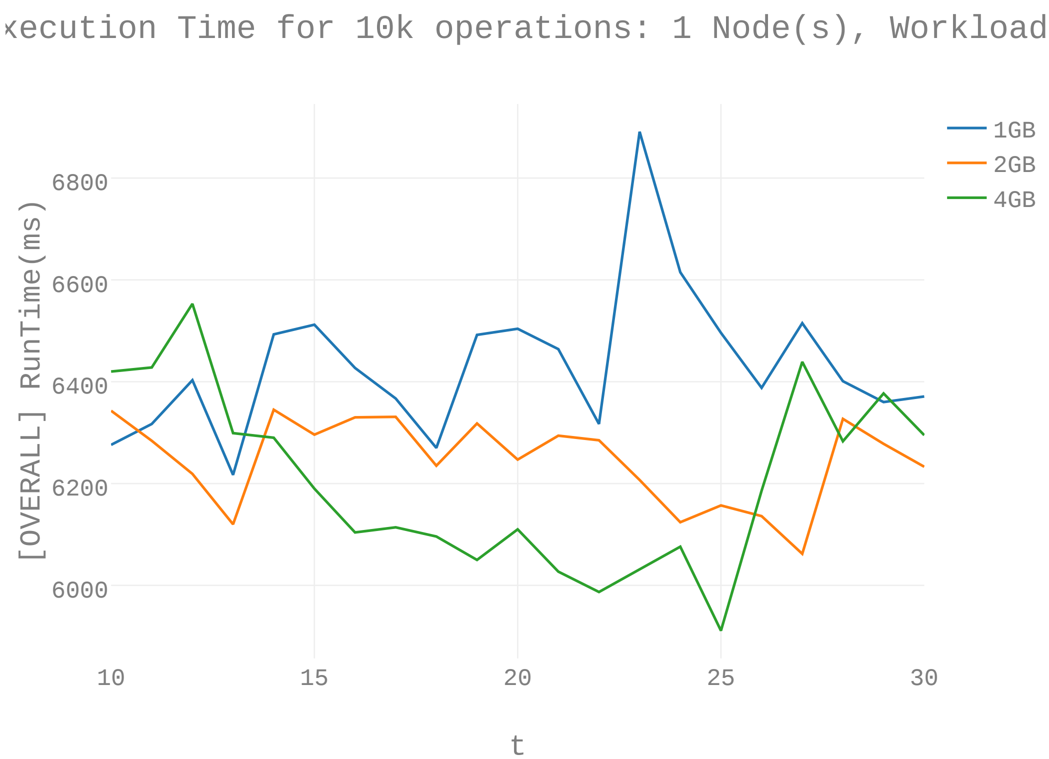

Figure 11. A series of trials for 1GB, 2GB, and 4GB RAM virtual machines. The inset demonstrates that any pattern that could significantly distinguish one trial from another, such as the oscillation seen in Figure 10 is integrated such that, after the cache warm-up period, the relative position of the trial does not predict the relative outcome.

thus the cost in testing time starts to curtail the value of integrating more trials.

4.8.4 Workloads.

Just as in [4], standard workloads A, C, and E were put to the test, while B, D, and F are de-prioritized. In addition, this paper introduces a custom workload, denoted I, to represent an IoT application, many inserts and few reads, a 99 percent to 1 percent ratio.

Standard workloads A, C, and E, from the YCSB are summarized in Table 5.

Workload	Read	Update	Scan
A	0.5	0.5	0.0
C	1.0	0.0	0.0
E	0.0	0.0	0.95

Table 5. Standard YCSB workloads used in this methodology. Workload A consists of 50 percent reads and 50 percent updates. Workload C consists of 100 percent reads. Workload E consists of 100 percent scans.

Workload	Read	Update	Scan	Insert
A	0.50	0.50	0.00	0.00
C	1.00	0.00	0.00	0.00
E	0.00	0.00	0.95	0.05
I	0.01	0.00	0.00	0.99

Table 6. Standard YCSB workloads used in this methodology. Workload A consists of 50 percent reads and 50 percent updates. Workload C consists of 100 percent reads. Workload E consists of 100 percent scans. In addition, a custom workload I is summarized, which consists of 99 percent writes and 1 percent reads to represent IoT.

4.8.5 Configuration of Cassandra.

Unless otherwise specified, the configuration of Cassandra, the keyspace, and the table within the keyspace are all held constant to their default values. All three of these things can be configured hierarchically, and their configuration can affect the performance of Cassandra on a given load. A skillful adjustment of these parameters can result in an optimized performance of Cassandra.

This paper aims to highlight the differences in varying hardware, and thus the exact configuration, despite that it might raise the absolute level of performance, is not expected to elevate the relative level of performance one would see shifting between, say a virtual node on a laptop and a Raspberry Pi module.

There are a few settings that had to be taken into account. Denoted *commitlog_total_space_in_mb*, this will accumulate to an undesired level. This was set to 512 MB in the Cassandra configuration file, cassandra.yaml. (Changing this to 256 MB for the 512 RAM case did not correct the error mentioned above.) Also, to prevent the accumulation of space, the setting *auto_snapshot* to false in configuration file cassandra.yaml. Having *auto_snapshot* set to true automatically backs up, or saves a snapshot of, the data on Cassandra. For a limited-capacity node, this would quickly lead to a crash.

The configuration files can be found among the appendices.

4.8.6 Threads in the YCSB.

The number of threads was kept constant at 1, although by increasing the number of threads one could achieve greater throughput. However, since it was desired to compare different calculations, the default was retained for all configurations.

It may be worth noting that for loads, this number was increased for practical reasons. However, these preparatory loads were not measured in these experiments, only the defined workloads A, C, E, and custom workload I.

4.9 Assumptions

Naturally, in order to perform the experiment and evaluate the results, some assumptions had to be made. Investigation into any of these assumptions may be an avenue for future work.

1. The benchmark represents the application, which assumes a simple schema. In other words, Cassandra's performance is not particularly sensitive to the schema.

2. There is no active attacker or intrusion into the local area networks. Both the local area networks are isolated.

3. There are no errors with the custom benchmark that would skew the results. Any error is due to the fact that the system's limits have been reached.

4. There are no bugs in the benchmark that would skew the results.

5. Effects on the network due to distance are negligible.

6. This experiment assumes that nodes are homogeneous. The basis for this assumption is that all nodes have been specified to the same model of Raspberry Pi 2. The same make and model for the SD Cards have been used. The image upon the SD Cards has been copied and only adjusted to account for specific, differentiated IP addresses.

7. This experiment assumes an uninterrupted power supply. Power is not measured nor accounted for in the model. As long as the power has been turned on, it stays on, and fluctuations in voltage or any kind of imperfections in the power supply are negligible with respect to Cassandra's performance.

8. Although the ISM band is unregulated, this experiment assumes invariant interference from other emitters. The experiment assumes an urban to suburban environment. In other words, congestion that overwhelms Cassandra's performance can be assumed to be rare with respect to the population, and is ignored for the purposes of the experiment.

4.10 Experimental Setup

In this experiment, we measure the total run time of a fixed number of operations of Workload A for various memory sizes. The choice of memory size is due to the expectations of IoT devices. The Raspberry Pi 2 and 3 have 1GB of memory and is our representative technology for IoT. The 4GB configuration can be considered be more representative of a low-end desktop, laptops, or virtual machine. The 2GB configuration is an intermediate stage that show an intermediate performance level and naturally, may represent the aim of future of IoT nodes. In addition, [4] used 2GB machines, which may help this work to be compared against existing work.

The experiments performed can be summarized in Table 7.

4.10.1 Virtual Node Setup.

For this work, virtual nodes were created to match these characteristics to a reasonable extent. Facing minimal propagation delay due to being connected on a nodal network, experiments with the virtual machines seek to place an upper limit on potential IoT performance expectations. Replicating the exact network in [4] is not absolutely necessarily, notwithstanding the details and materials available to do so are unavailable.

On a 64-bit 31.1GiB RAM laptop with Intel Core i7-4910MQ 2.90GHz 8-core central processing unit, six (6) identical virtual machines were created in software VirtualBox. Each machine consisted of Ubuntu 64 bit machine was was allocated 8 GB of hard drive disk space. The full details for these machines, reported by means of the lshw Linux command, are located among the appendices. Also in VirtualBox, a host-only network entitled vboxnet0 was instantiated, to which all six machines were connected.

The YCSB was installed and run on the host laptop. PyCharm drove a terminal process, which in turn drove the YCSB software.

Communication (nm)	Platform (nt)	Assigned RAM
Nodal	Virtual Machine	1 GB
Nodal	Virtual Machine	2 GB
Nodal	Virtual Machine	4 GB
Ethernet LAN	Raspberry Pi	1 GB
802.11 LAN	Raspberry Pi	1 GB

Table 7. This table summarizes each network topology that was explored in each research question. The RAM was varied on the Virtual Machines.

To further paint the picture of the network setup, the IP addresses are included in Table 9.

4.10.2 Ethernet LAN Setup.

The wiring of the Ethernet LAN is depicted in Figure 12. Note that for the Ethernet set up, the wireless capability for the home router was switched OFF.

All nodes, including the laptop used to execute the YCSB, were on the same network. Their IP addresses are listed in Table 8.

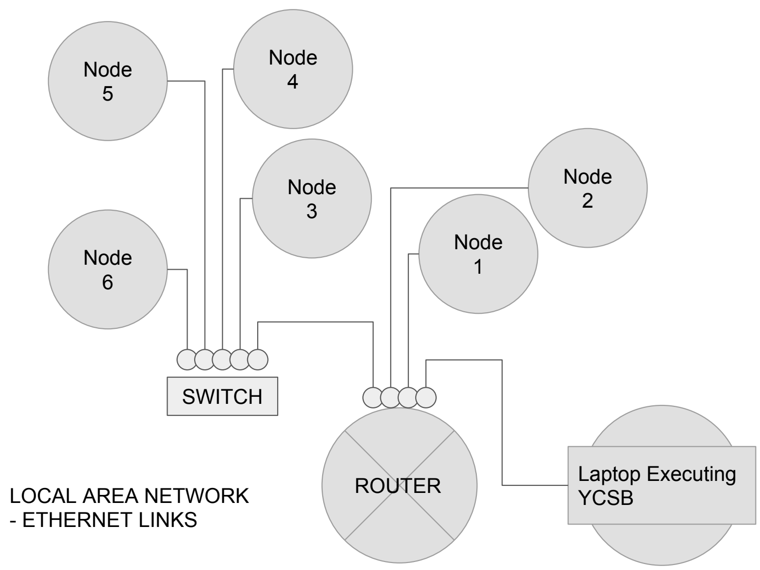

Figure 12. Topology and Wiring for Ethernet Setup

Node	Host Name	IP Address	Model
Node 1	raspberrypi0	192.168.1.100	2B
Node 2	raspberrypi1	192.168.1.101	2B
Node 3	raspberrypi2	192.168.1.102	2B
Node 4	raspberrypi3	192.168.1.103	2B
Node 5	raspberrypi4	192.168.1.104	2B
Node 6	raspberrypi9	192.168.1.109	3
Laptop	daniel-ThinkPad-W541	192.168.1.200	-

Table 8. This table describes the IP addresses in order to paint a further detailed understanding of network set up. The netmask is 255.255.255.0

Node	Host Name	IP Address
Node 1	c0	192.168.56.100
Node 2	c1	192.168.56.101
Node 3	c2	192.168.56.102
Node 4	c3	192.168.56.103
Node 5	c4	192.168.56.104
Node 6	c5	192.168.56.105
Laptop	daniel-ThinkPad-W541	192.168.56.200

Table 9. This table describes the IP addresses in order to paint a further detailed understanding of network set up. The netmask is 255.255.255.0

4.10.3 802.11 LAN Setup.

Table 10 describes the local area network as the wireless the. Note that for this setup, the Ethernet cables were physically unplugged.

To enable wireless links on the Raspberry Pi 2's, the Wi-Pi Universal Serial Bus (USB) module was utilized, with transmission speed capabilities listed at "11b 1/2/5.5/11Mbps, 11g 6/9/12/18/24/36/48/54Mbps, 11n up to 150Mbps" [?]. The Raspberry Pi 3 model comes with its own built-in WiFi capabilities.

Node	Host Name	IP Address
Node 1	raspberrypi0	192.168.1.130
Node 2	raspberrypi1	192.168.1.131
Node 3	raspberrypi2	192.168.1.132
Node 4	raspberrypi3	192.168.1.133
Node 5	raspberrypi4	192.168.1.134
Node 6	raspberrypi9	192.168.1.139
Laptop	daniel-ThinkPad-W541	192.168.1.200

Table 10. This table describes the IP addresses in order to paint a further detailed understanding of network set up. The netmask is 255.255.255.0

4.11 Execution

The YCSB, installed on the host laptop, is also run from the host laptop. The YCSB can be run from the terminal, but for convenience, a Python script was developed to drive a series terminal processes.

4.12 Analysis

For each experiment, the trials for each configuration will be reported as a summary of execution times for 10,000 operations. All execution times will be reported in milliseconds.

4.12.1 Response Variables.

The YCSB reports a number of measured values, including operations per second and latency distribution (minimum, mean, 50 percentile or median, 75 percentile, 99 percentile, maximum). The total execution time in milliseconds was chosen in order to keep the measured values true to the limits of the experiment. Although this paper aims to analyze results to reflect the steady state, this paper does not deny the possibility that variance in operations per second or variance in latency may result from variance in the number of operations per trial.

4.13 Cache Warm-Up Period and the Logic of Using the Median from This Point Forward

Also, one can note in both Figure 10 and Figure 11, that in the first five trials or so, one can observe the cache warm-up period. This steep decline is expected due to the effect of the key cache, which is at its default setting: Cassandra sets the key cache to the either 5% of the heap, or 100 MB, whichever is less. In [?], the key cache is reported to be at 100 MB. The steady state operation is dependent on the keys requested, so for a workload like the YCSB, one would not expect a lot of variation.

Truncating the head of the trials, trials 1 though 9, the cache effect is no longer depicted, rendering an expectation of steady-state performance after cache warm-up. This is necessary to meet the assumptions of any ANOVA test.

Taking the issues of oscillating behavior and cache warm-up period into account, we can remove them to find a stable viewing window into the behavior of the Cassandra database, as depicted in Figure 11. Figure 11 shows the desired observation of the behavior in question for 1GB, 2GB ad 4GB memory sizes. We use this data to recreate Abramovas work and extend it for other memory sizes.

This is also the logic behind using the median to summarize the execution times, which should not differ significantly whether the cache warm-up trials are included or not.

Another important observation here, is that once the cache effect is cropped out, there is no obvious correlation between trials and the performance measurement. This further supports that 10,000 operations, minus cache effect, does represent a steady state that is likely to extend beyond 10,000 operations.

V. Results and Evaluation

5.1 Results for Workload A

5.1.1 Comparing Existing Work: Virtual Machine vs the Reference Value.

Initial Observations.

The result medians are displayed in Figure 14. As expected, the virtual machine results imply much, much less execution time compared to the reference value, presumably accounting for diminished network latency. Such network latency seems to have an increasing effect on the reference as the cluster size goes up, but further analysis would be required to even make this claim.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload A performed on the virtual machines are in Tables 12, 11, and 13.

Analysis.

This section will take a more in-depth look at the data.

5.1.2 1GB RAM vs 2GB RAM vs 4GB RAM.

Initial Observations.

This section discusses testing and performance for memory sizes: 1GB, 2GB, and 4GB. The results are displayed in Figure 15. While it appears the varying the amount of memory has some effect on the results, theredoes not seem to be a predictable

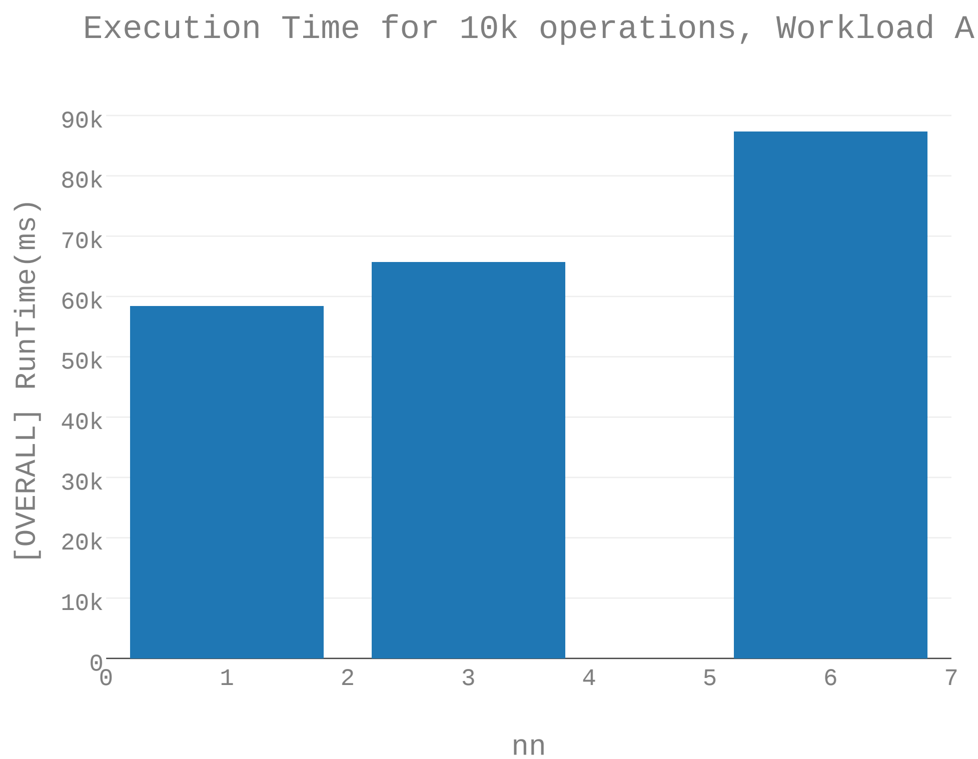

Figure 13. This scatterplot compares the values from efAbramova2014TestingCassandra to the median result of Workload A executed on the virtual machine.

pattern across nodes. An ANOVA test will determine if there is an effect, and a linear regression will further testfor an effect.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload A performed on the virtual machines are in Tables 12, 11, and 13.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	1.0	3.0	6.0	Overall
25%	6.2e+03	1e+04	1.4e+04	6.3e+03
50%	6.3e+03	1e+04	1.4e+04	1e+04
75%	6.3e+03	1.1e+04	1.4e+04	1.4e+04
count	21	21	21	63
max	6.3e+03	1.1e+04	1.5e+04	1.5e+04
mean	6.2e+03	1.1e+04	1.4e+04	1e+04
min	6.1e+03	1e+04	1.4e+04	6.1e+03
range	2.8e+02	6.8e+02	6.8e+02	8.6e+03
std	84	2.2e+02	1.7e+02	3.3e+03

Table 11. Summary Statistics for Workload A performed on a 2GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

cluster_size	1.0	3.0	6.0	Overall
25%	6.4e+03	1e+04	1.4e+04	6.5e+03
50%	6.4e+03	1e+04	1.5e+04	1e+04
75%	6.5e+03	1e+04	1.5e+04	1.4e+04
count	21	21	21	63
max	6.9e+03	1.1e+04	1.5e+04	1.5e+04
mean	6.4e+03	1e+04	1.5e+04	1e+04
min	6.2e+03	9.6e+03	1.4e+04	6.2e+03
range	6.7e+02	9e+02	9.5e+02	9e+03
std	1.4e+02	2.3e+02	2.7e+02	3.4e+03

Table 12. Summary Statistics for Workload A performed on a 1GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

cluster_size	1.0	3.0	6.0	Overall
25%	6.1e+03	1e+04	1.5e+04	6.3e+03
50%	6.2e+03	1.1e+04	1.5e+04	1.1e+04
75%	6.3e+03	1.1e+04	1.5e+04	1.5e+04
count	21	21	21	63
max	6.6e+03	1.1e+04	1.5e+04	1.5e+04
mean	6.2e+03	1.1e+04	1.5e+04	1.1e+04
min	5.9e+03	1e+04	1.5e+04	5.9e+03
range	6.4e+02	1.3e+03	7.7e+02	9.4e+03
std	1.7e+02	3.4e+02	2.2e+02	3.6e+03

Table 13. Summary Statistics for Workload A performed on a 4GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

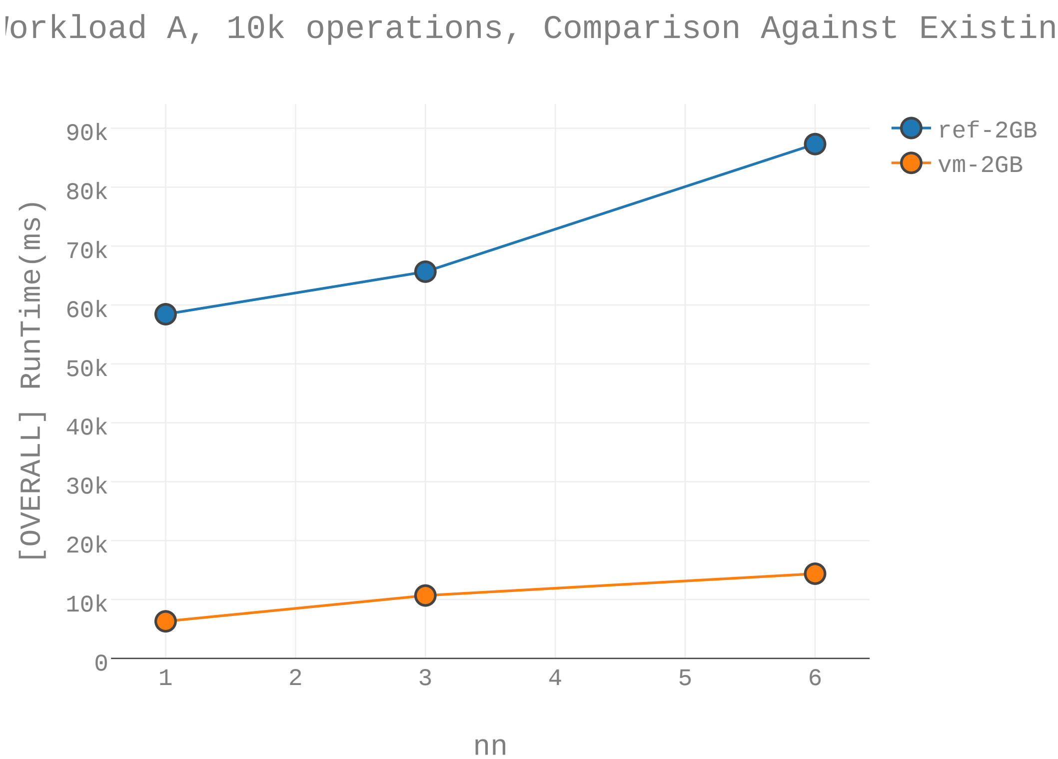

Figure 14. This compares the 2GB virtual machine with the corresponding value from ??.

cluster_size	slope	intercept	r_value	p_value	std_err
1	-69	6.5e+03	-0.51	2.1e-05	15
3	1.2e+02	1e+04	0.46	0.00016	30
6	1.5e+02	1.4e+04	0.51	1.7e-05	32

Table 14. Linear Regression over amount of RAM

5.1.3 Implementation on Raspberry Pi.

Initial Observations.

The results are displayed in Figure 16.

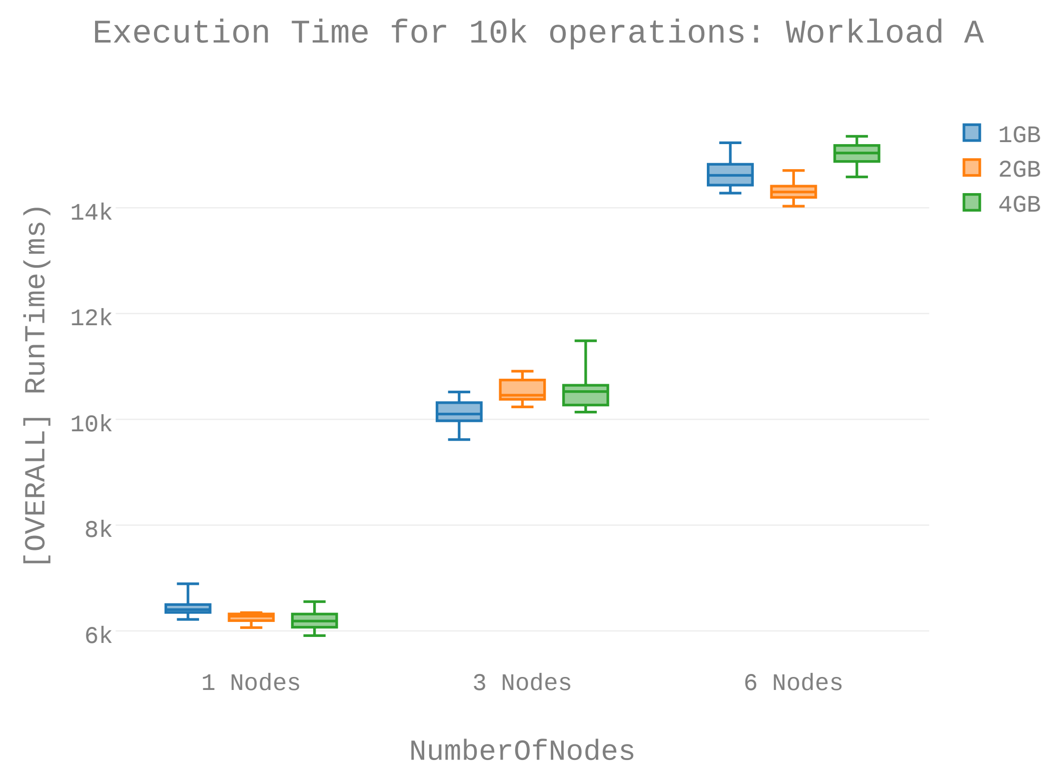

Figure 15. Execution time for virtual machines with 1GB, 2GB, and 4GB of RAM. The first 9 trials have been removed in order to filter out the trials representing cache effect and thus represents the steady state.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload A performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 15.

Analysis.

This section will take a more in-depth look at the data.

slope	intercept	r_value	p_value	std_err
6e+03	7.3e+04	0.42	1.1e-06	1.2e+03

Table 16. Linear Regression over Cluster Size, Workload a

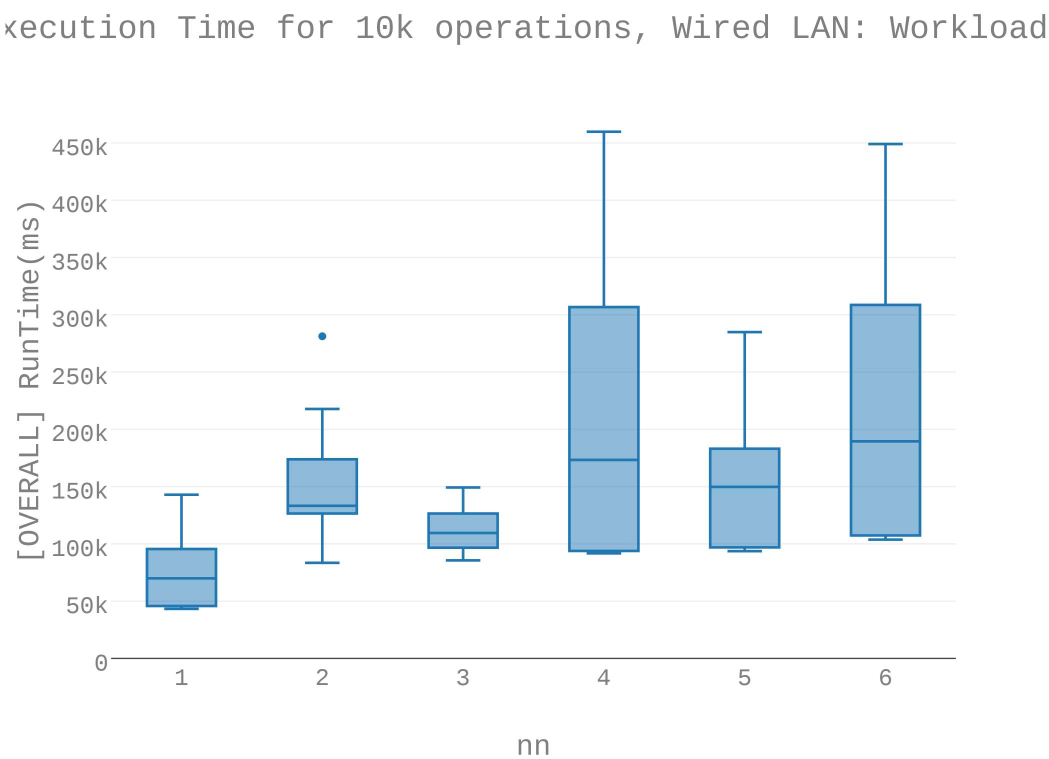

Figure 16. Results of limited hardware, the Raspberry Pi, on an Ethernet LAN. Execution time is plotted over cluster size.

5.1.4 Raspberry Pi vs Reference Value.

Initial Observations.

The results are displayed in Figure 17. The performance of the limited hardware, the Raspberry Pi, seems comparable with the results from the more appropriate in the other paper. In addition, there is no reason to suspect a significant differential in the overall pattern with respect to scalability: performance over cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload A performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 15.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	4.5e+04	1.3e+05	9.6e+04	9.4e+04	9.6e+04	1.1e+05	9.4e+04
50%	4.6e+04	1.3e+05	9.7e+04	9.4e+04	9.6e+04	1.1e+05	9.7e+04
75%	4.6e+04	1.3e+05	9.8e+04	9.5e+04	9.8e+04	1.1e+05	1.1e+05
count	21	21	21	21	21	21	1.3e+02
max	4.8e+04	1.3e+05	1e+05	9.6e+04	9.9e+04	1.1e+05	1.3e+05
mean	4.6e+04	1.3e+05	9.7e+04	9.4e+04	9.6e+04	1.1e+05	9.5e+04
min	4.3e+04	1.2e+05	9.4e+04	9.3e+04	9.4e+04	1e+05	4.3e+04
range	4.9e+03	1.5e+04	6.8e+03	3.6e+03	4.9e+03	7.3e+03	8.8e+04
std	1.1e+03	3.5e+03	1.6e+03	9.2e+02	1.5e+03	1.8e+03	2.5e+04

Table 15. Summary Statistics for Workload A performed on a 1GB limited hardware, Raspberry Pi node over a(n)Ethernet network. Except for count, all values are in milliseconds.

Analysis.

This section will take a more in-depth look at the data.

5.1.5 Raspberry Pi vs Virtual Machine.

Initial Observations.

The results are displayed in Figure 17. As expected, there is a significant differential between the limited hardware, the Raspberry Pi configuration and the virtual machine. However, it is not clear if this is linear across cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload A performed on the virtual machines are in Tables 12, 11, and 13. The summary statistics for Workload A performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 15.

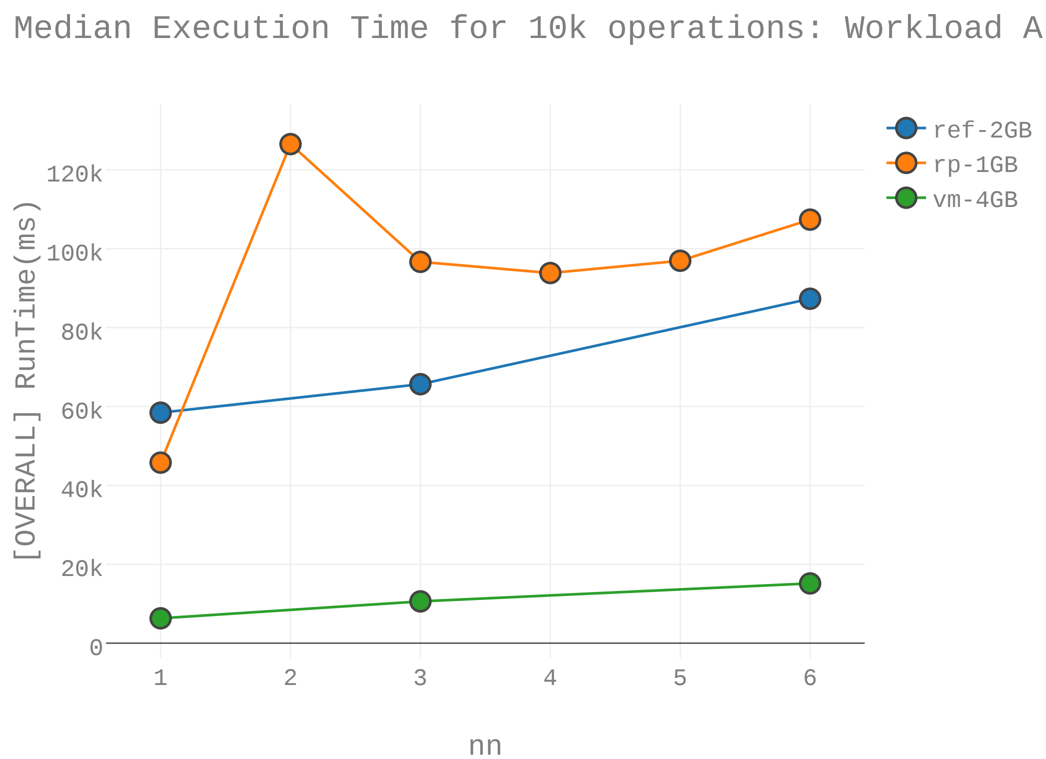

Figure 17. Comparison among the Raspberry Pi nodes (rp-1GB), the results reported in ??, and the virtual nodes with 1GB of RAM.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	slope	intercept	r_value	p_value	std_err
1	3.9e+04	6.4e+03	1	1.3e-57	2.5e+02
2	nan	nan	0	1	inf
3	8.7e+04	1e+04	1	8.6e-65	3.6e+02
4	nan	nan	0	1	inf
5	nan	nan	0	1	inf
6	9.2e+04	1.5e+04	1	2.3e-64	3.9e+02
OVERALL	8.4e+04	1e+04	0.89	4.5e-66	3.1e+03

Table 17. Linear Regression over the effect of limited hardware, Workload A

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.16	NaN	0.11	NaN	NaN	0.15	0.35
ratio_min_to_max	0.13	NaN	0.096	NaN	NaN	0.13	0.047
ratio_of_the_means	0.14	NaN	0.1	NaN	NaN	0.14	0.11
ratio_of_the_medians	0.14	NaN	0.1	NaN	NaN	0.14	0.1
ratio_of_the_stddevs	0.13	NaN	0.14	NaN	NaN	0.15	0.14

Table 18. Speedup over the effect of limited hardware, Workload A

5.1.6 Wireless Links Only.

Initial Observations.

The results are displayed in Figure 18. Although there is a general trend of increased execution time over cluster size, the oscillation that occurs between odd and even node cluster sizes is hard to miss. This may be result of the collision avoidance strategy, but further experiments would be needed to determine a more specific explanation.

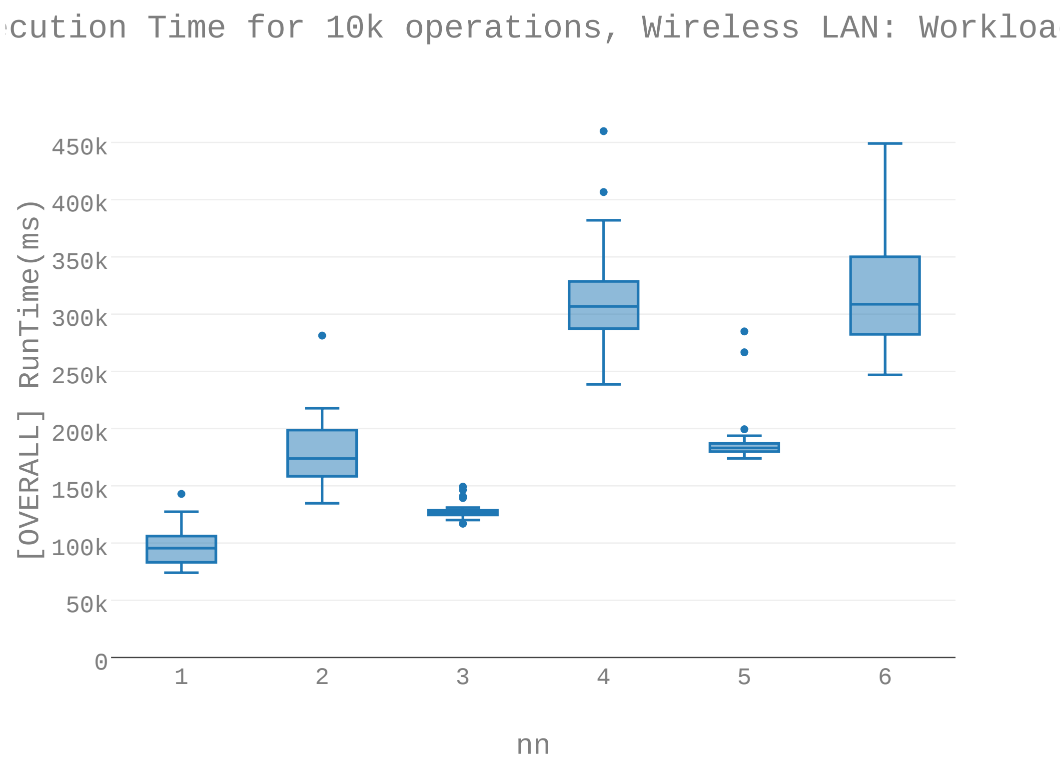

Figure 18. Results of wireless testing. There seems to be a steady climb in execution time as the cluster size increases. Any oscillation cannot be explained with current analysis and would require additional experimentation.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload A performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 19.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	8.2e+04	1.7e+05	1.3e+05	3e+05	1.8e+05	2.7e+05	1.3e+05
50%	9e+04	1.8e+05	1.3e+05	3.2e+05	1.8e+05	3e+05	1.8e+05
75%	1e+05	2e+05	1.3e+05	3.3e+05	1.8e+05	3.1e+05	2.8e+05
count	21	21	21	21	21	21	1.3e+02
max	1.2e+05	2.8e+05	1.5e+05	4.6e+05	2.8e+05	3.6e+05	4.6e+05
mean	9.1e+04	1.9e+05	1.3e+05	3.2e+05	1.9e+05	3e+05	2e+05
min	7.4e+04	1.5e+05	1.2e+05	2.6e+05	1.7e+05	2.5e+05	7.4e+04
range	4.6e+04	1.3e+05	2.6e+04	1.9e+05	1.1e+05	1.1e+05	3.9e+05
std	1.3e+04	2.9e+04	7.4e+03	4.5e+04	2.9e+04	3.2e+04	8.7e+04

Table 19. Summary Statistics for Workload A performed on a 1GB limited hardware, Raspberry Pi node over a(n)802.11a/b/g/n network. Except for count, all values are in milliseconds.

slope	intercept	r_value	p_value	std_err
3.5e+04	8e+04	0.69	7.5e-19	3.3e+03

Table 20. Linear Regression over Cluster Size, Workload a

5.1.7 Wireless Links vs Wired Links.

Initial Observations.

The results are displayed in Figures 19 and 20. For 1, 2, and 3-node clusters, despite the expected disparity in execution time, the wired and wireless trends seem to follow each other. However, from 3 nodes up through 6 nodes, the execution times starts to diverge, suggesting that the wireless has a increasing effect as the number of nodes increases.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload A performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 15. The summary statistics

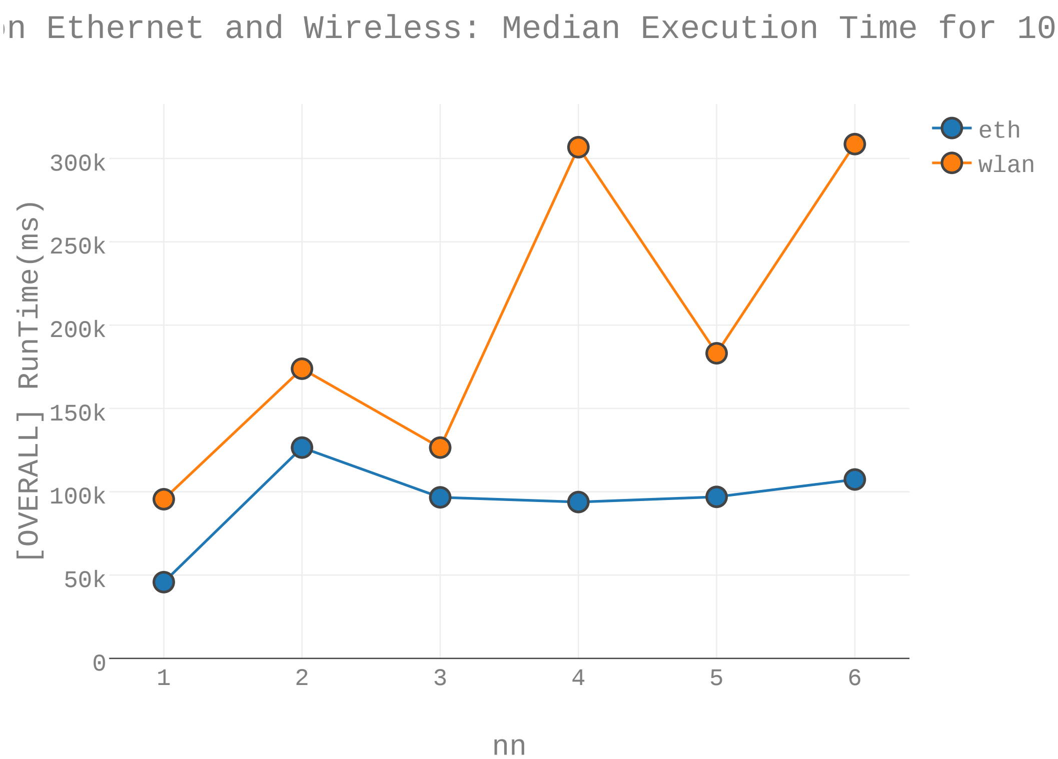

Figure 19. Comparison between links: Ethernet wired versus 802.11 wireless. As the cluster sizes increase, these show a tendency to diverge.

for Workload A performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 19.

Analysis.

This section will take a more in-depth look at the data.

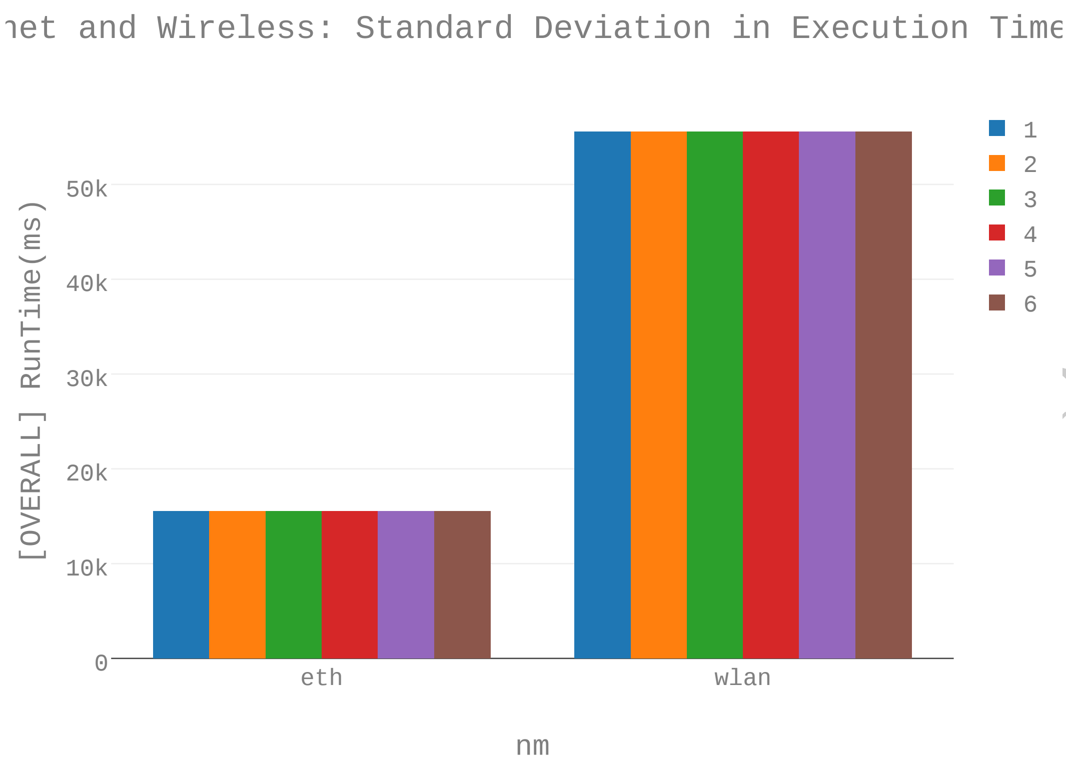

Figure 20. Standard deviation of execution time in milliseconds. There is a significant increase when going from the wired to the wireless configuration.

cluster_size	slope	intercept	r_value	p_value	std_err
1	4.6e+04	4.6e+04	0.93	3.9e-19	2.8e+03
2	6.1e+04	1.3e+05	0.84	3.7e-12	6.3e+03
3	3.3e+04	9.7e+04	0.95	2.9e-22	1.7e+03
4	2.3e+05	9.4e+04	0.96	9.4e-25	9.8e+03
5	9.4e+04	9.6e+04	0.92	4.6e-18	6.3e+03
6	1.9e+05	1.1e+05	0.97	1.4e-27	6.9e+03
OVERALL	1.1e+05	9.5e+04	0.65	3.4e-31	8.1e+03

Table 21. Linear Regression over the effect of 802.11 links, Workload A

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.65	0.87	0.82	0.36	0.57	0.45	1.8
ratio_min_to_max	0.36	0.42	0.63	0.2	0.33	0.29	0.094
ratio_of_the_means	0.5	0.67	0.75	0.29	0.51	0.36	0.47
ratio_of_the_medians	0.51	0.72	0.76	0.3	0.53	0.35	0.53
ratio_of_the_stddevs	0.087	0.12	0.22	0.02	0.052	0.057	0.28

Table 22. Speedup over the effect of 802.11 links, Workload A

5.2 Results for Workload C

5.2.1 Comparing Existing Work: Virtual Machine vs the Reference Value.

Initial Observations.

The result medians are displayed in Figure 22. As expected, the virtual machine results imply much, much less execution time compared to the reference value, presumably accounting for diminished network latency. Such network latency seems to have an increasing effect on the reference as the cluster size goes up, but further analysis would be required to even make this claim.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload C performed on the virtual machines are in Tables 24, 23, and 25.

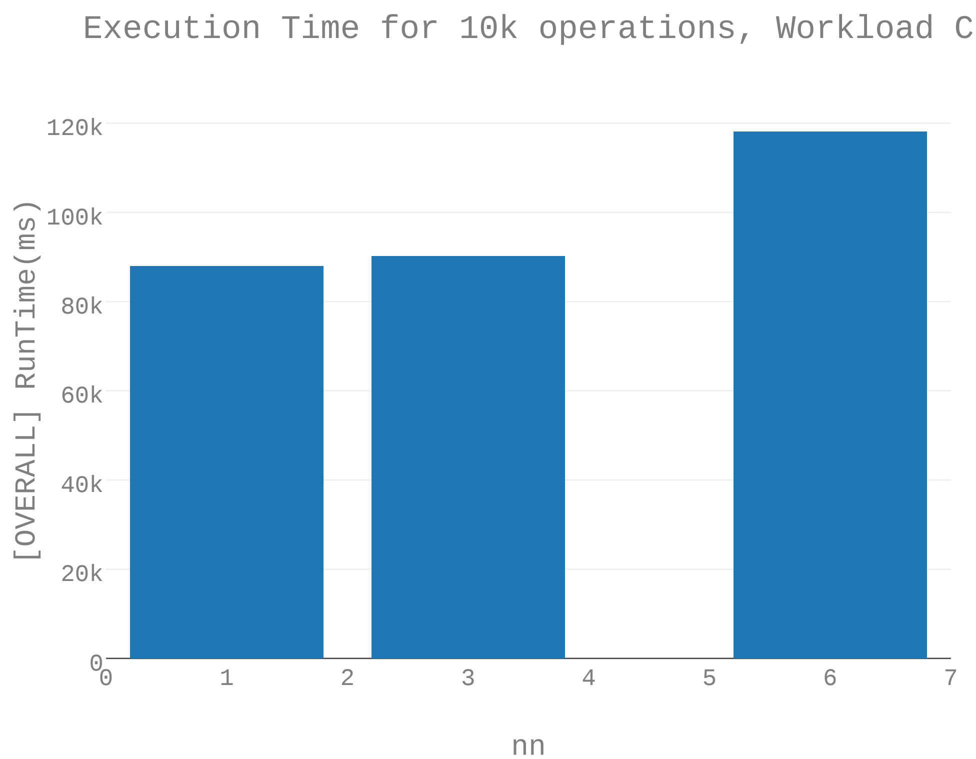

Figure 21. **This scatterplot compares the values from efAbramova2014TestingCassandra to the median result of Workload C executed on the virtual machine.**

Analysis.

This section will take a more in-depth look at the data.

5.2.2 1GB RAM vs 2GB RAM vs 4GB RAM.

Initial Observations.

This section discusses testing and performance for memory sizes: 1GB, 2GB, and 4GB. The results are displayed in Figure 23. While it appears the varying the amount of memory has some effect on the results, theredoes not seem to be a predictable pattern across nodes. An ANOVA test will determine if there is an effect, and a linear regression will further testfor an effect.

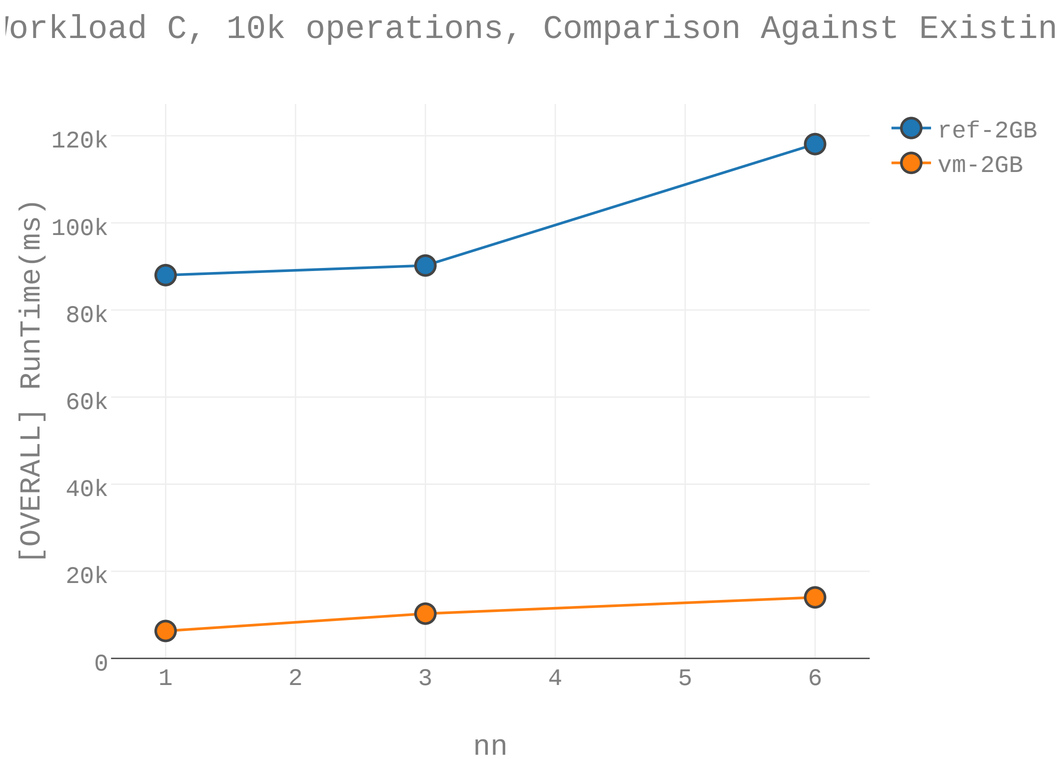

Figure 22. This compares the 2GB virtual machine with the corresponding value from ??.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload C performed on the virtual machines are in Tables 24, 23, and 25.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	1.0	3.0	6.0	Overall
25%	6.2e+03	1e+04	1.4e+04	6.3e+03
50%	6.2e+03	1e+04	1.4e+04	1e+04
75%	6.3e+03	1e+04	1.4e+04	1.4e+04
count	21	21	21	63
max	6.9e+03	1e+04	1.4e+04	1.4e+04
mean	6.3e+03	1e+04	1.4e+04	1e+04
min	6e+03	9.8e+03	1.3e+04	6e+03
range	9.8e+02	6e+02	9.8e+02	8.5e+03
std	2.2e+02	1.8e+02	2.6e+02	3.2e+03

Table 23. Summary Statistics for Workload C performed on a 2GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

cluster_size	1.0	3.0	6.0	Overall
25%	6.3e+03	9.7e+03	1.4e+04	6.5e+03
50%	6.4e+03	9.8e+03	1.4e+04	9.8e+03
75%	6.5e+03	9.9e+03	1.5e+04	1.4e+04
count	21	21	21	63
max	6.7e+03	1.1e+04	1.5e+04	1.5e+04
mean	6.4e+03	9.9e+03	1.4e+04	1e+04
min	6.1e+03	9.5e+03	1.4e+04	6.1e+03
range	6.3e+02	1e+03	9.8e+02	8.9e+03
std	1.7e+02	2.6e+02	3e+02	3.3e+03

Table 24. Summary Statistics for Workload C performed on a 1GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

cluster_size	1.0	3.0	6.0	Overall
25%	6e+03	9.9e+03	1.4e+04	6.4e+03
50%	6.1e+03	1e+04	1.5e+04	1e+04
75%	6.4e+03	1e+04	1.5e+04	1.4e+04
count	21	21	21	63
max	6.6e+03	1.2e+04	1.5e+04	1.5e+04
mean	6.2e+03	1e+04	1.5e+04	1e+04
min	5.9e+03	9.8e+03	1.4e+04	5.9e+03
range	7.4e+02	2.4e+03	6.8e+02	9.1e+03
std	2.3e+02	5e+02	1.8e+02	3.5e+03

Table 25. Summary Statistics for Workload C performed on a 4GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

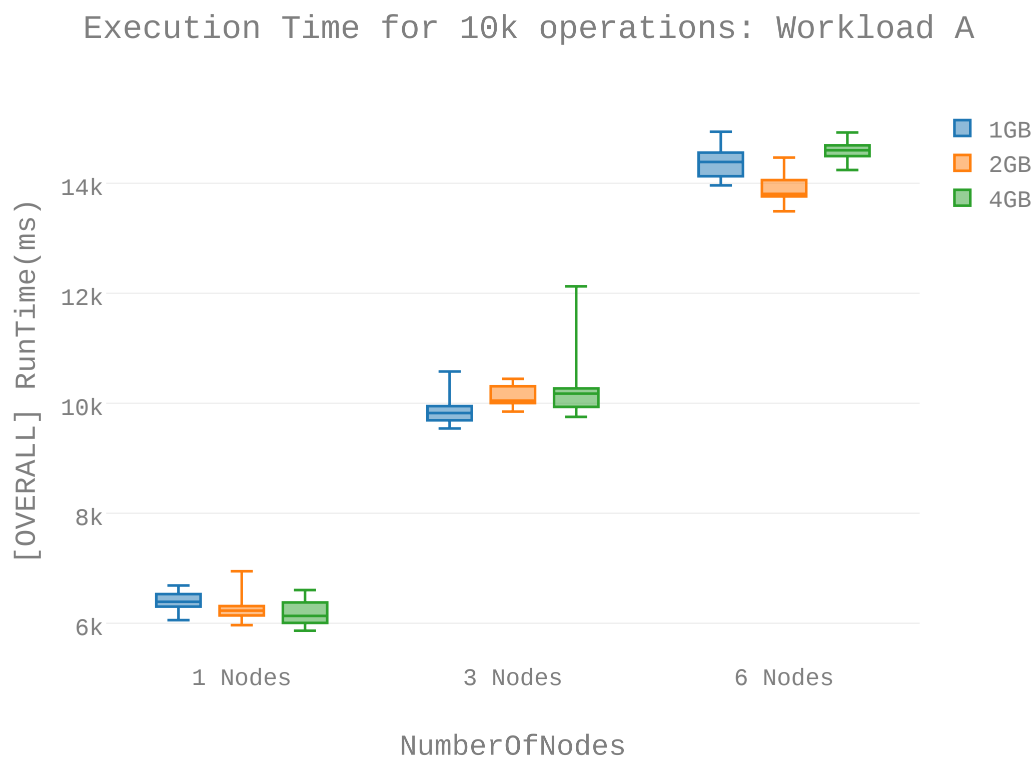

Figure 23. Execution time for virtual machines with 1GB, 2GB, and 4GB of RAM. The first 9 trials have been removed in order to filter out the trials representing cache effect and thus represents the steady state.

cluster_size	slope	intercept	r_value	p_value	std_err
1	-71	6.4e+03	-0.39	0.0014	21
3	1e+02	9.8e+03	0.35	0.0046	35
6	1.1e+02	1.4e+04	0.36	0.0035	36

Table 26. Linear Regression over amount of RAM

5.2.3 Implementation on Raspberry Pi.

Initial Observations.

The results are displayed in Figure 24.

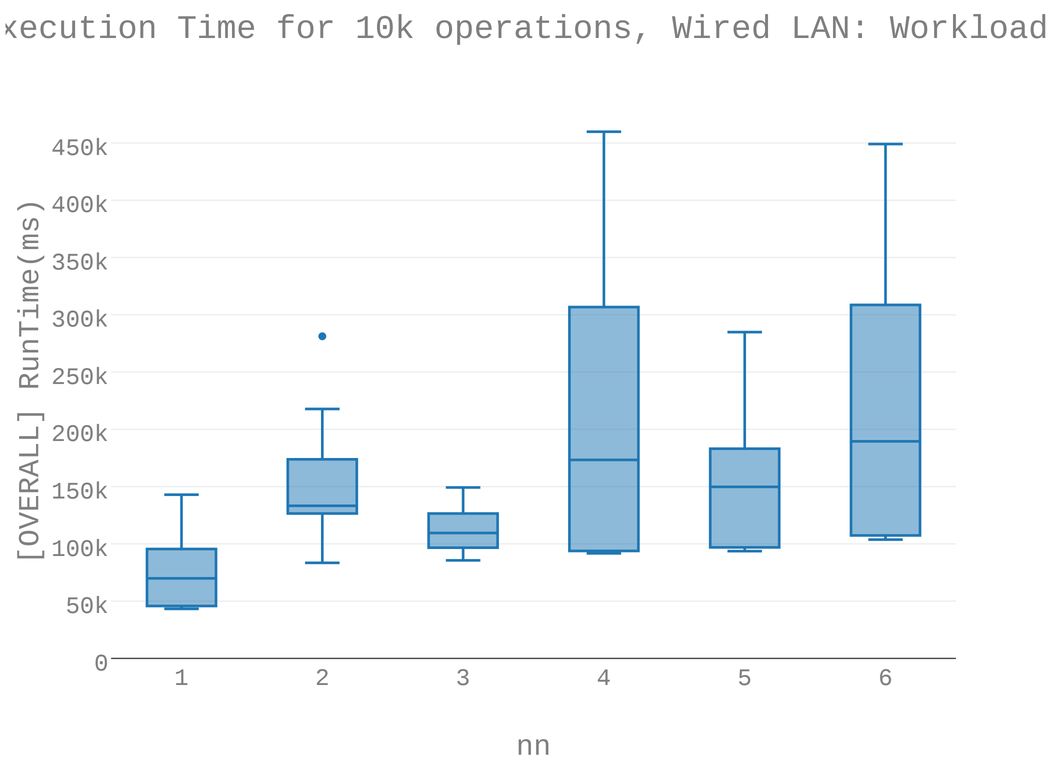

Figure 24. Results of limited hardware, the Raspberry Pi, on an Ethernet LAN. Execution time is plotted over cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload C performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 27.

Analysis.

This section will take a more in-depth look at the data.

slope	intercept	r_value	p_value	std_err
1.4e+03	1e+05	0.06	0.5	2e+03

Table 28. Linear Regression over Cluster Size, Workload c

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	4.7e+04	1.8e+05	1.1e+05	1e+05	9.7e+04	1.1e+05	9.8e+04
50%	4.9e+04	1.8e+05	1.1e+05	1e+05	9.8e+04	1.1e+05	1e+05
75%	5e+04	1.8e+05	1.1e+05	1e+05	1e+05	1.1e+05	1.1e+05
count	21	21	21	21	21	21	1.3e+02
max	5e+04	1.9e+05	1.3e+05	1e+05	1e+05	1.1e+05	1.9e+05
mean	4.8e+04	1.8e+05	1.1e+05	1e+05	9.9e+04	1.1e+05	1.1e+05
min	4.5e+04	1.6e+05	1.1e+05	1e+05	9.6e+04	1e+05	4.5e+04
range	5.4e+03	3.2e+04	1.7e+04	4e+03	7e+03	8.1e+03	1.5e+05
std	1.8e+03	6.7e+03	3.2e+03	1.2e+03	2e+03	2.1e+03	3.9e+04

Table 27. Summary Statistics for Workload C performed on a 1GB limited hardware, Raspberry Pi node over a(n)Ethernet network. Except for count, all values are in milliseconds.

5.2.4 Raspberry Pi vs Reference Value.

Initial Observations.

The results are displayed in Figure 25. The performance of the limited hardware, the Raspberry Pi, seems comparable with the results from the more appropriate in the other paper. In addition, there is no reason to suspect a significant differential in the overall pattern with respect to scalability: performance over cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload C performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 27.

Analysis.

This section will take a more in-depth look at the data.

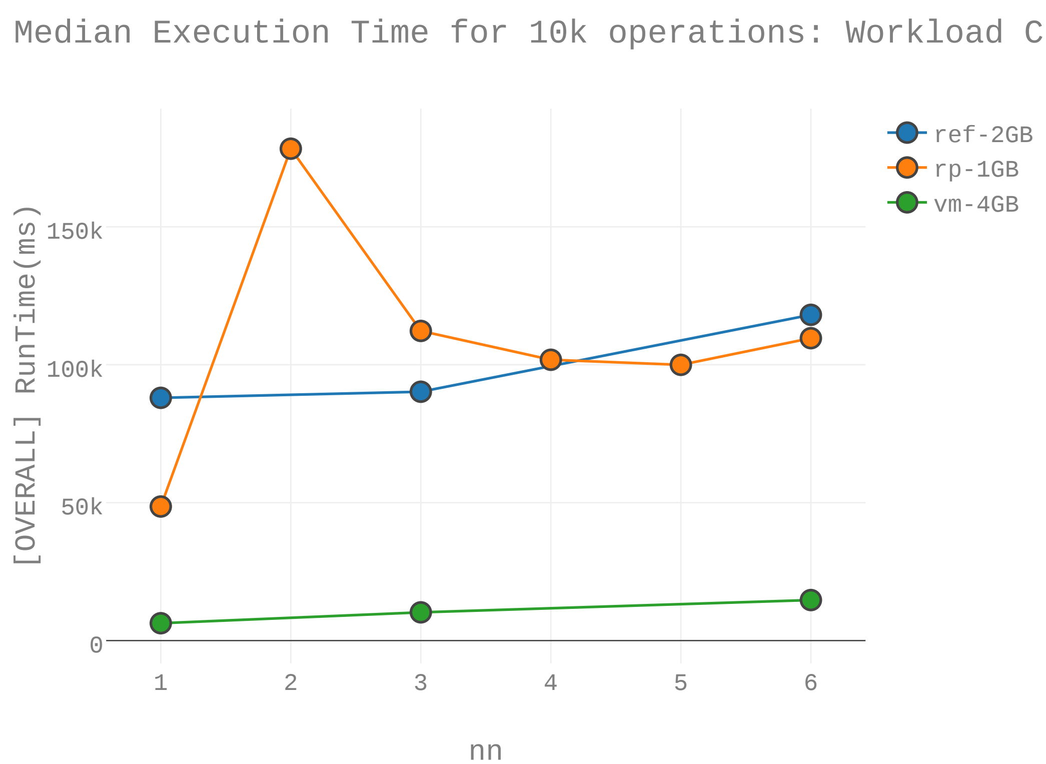

Figure 25. Comparison among the Raspberry Pi nodes (rp-1GB), the results reported in ??, and the virtual nodes with 1GB of RAM.

5.2.5 Raspberry Pi vs Virtual Machine.

Initial Observations.

The results are displayed in Figure 25. As expected, there is a significant differential between the limited hardware, the Raspberry Pi configuration and the virtual machine. However, it is not clear if this is linear across cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload C performed on the virtual machines are in Tables 24, 23, and 25. The summary statistics for Workload C performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 27.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	slope	intercept	r_value	p_value	std_err
1	4.2e+04	6.4e+03	1	1.1e-50	3.9e+02
2	nan	nan	0	1	inf
3	1e+05	9.9e+03	1	2.4e-56	7e+02
4	nan	nan	0	1	inf
5	nan	nan	0	1	inf
6	9.4e+04	1.4e+04	1	1.2e-61	4.7e+02
OVERALL	9.8e+04	1e+04	0.83	1.6e-48	4.9e+03

Table 29. Linear Regression over the effect of limited hardware, Workload C

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.15	NaN	0.097	NaN	NaN	0.14	0.33
ratio_min_to_max	0.12	NaN	0.076	NaN	NaN	0.12	0.032
ratio_of_the_means	0.13	NaN	0.087	NaN	NaN	0.13	0.094
ratio_of_the_medians	0.13	NaN	0.087	NaN	NaN	0.13	0.094
ratio_of_the_stddevs	0.092	NaN	0.079	NaN	NaN	0.14	0.085

Table 30. Speedup over the effect of limited hardware, Workload C

5.2.6 Wireless Links Only.

Initial Observations.

The results are displayed in Figure 26. Although there is a general trend of increased execution time over cluster size, the oscillation that occurs between odd and even node cluster sizes is hard to miss. This may be result of the collision avoidance strategy, but further experiments would be needed to determine a more specific explanation.

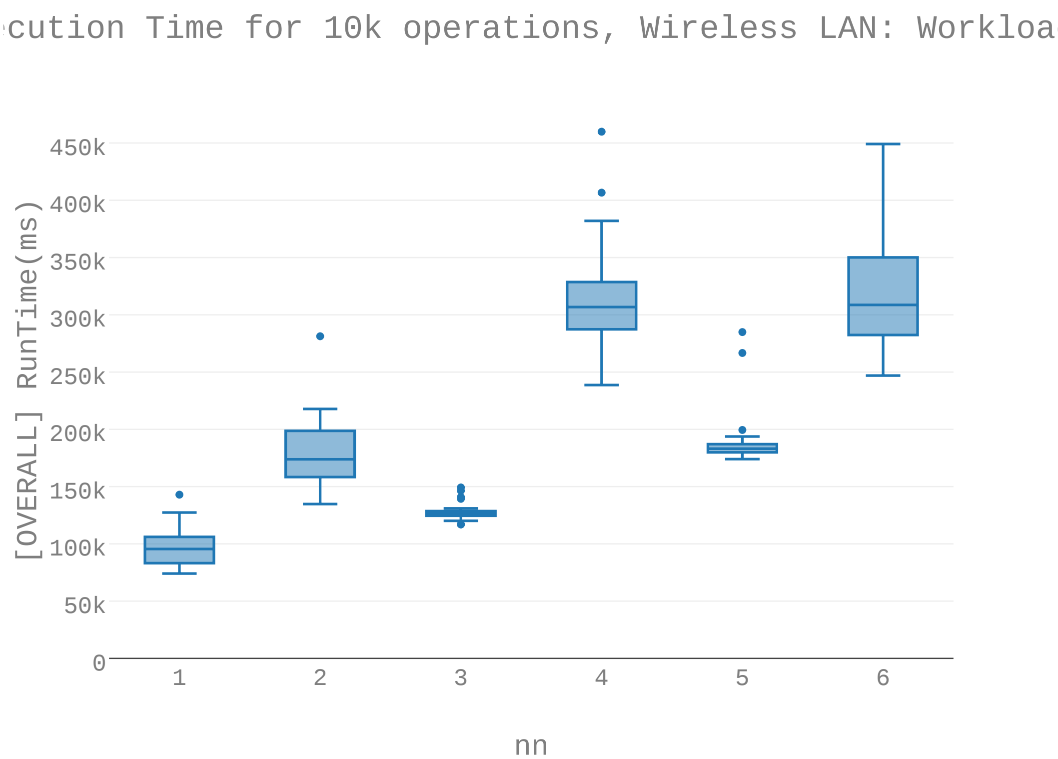

Figure 26. Results of wireless testing. There seems to be a steady climb in execution time as the cluster size increases. Any oscillation cannot be explained with current analysis and would require additional experimentation.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload C performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 31.

Analysis.

This section will take a more in-depth look at the data.

slope	intercept	r_value	p_value	std_err
3.1e+04	8e+04	0.75	6.7e-24	2.5e+03

Table 32. Linear Regression over Cluster Size, Workload c

5.2.7 Wireless Links vs Wired Links.

Initial Observations.

The results are displayed in Figures 27 and 28. For 1, 2, and 3-node clusters, despite the expected disparity in execution time, the wired and wireless trends seem to follow each other. However, from 3 nodes up through 6 nodes, the execution times starts to diverge, suggesting that the wireless has a increasing effect as the number of nodes increases.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload C performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 27. The summary statistics for Workload C performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 31.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	6.4e+04	1.8e+05	1.4e+05	2.7e+05	1.8e+05	2.5e+05	1.4e+05
50%	7e+04	1.9e+05	1.4e+05	2.7e+05	1.9e+05	2.7e+05	1.9e+05
75%	8.3e+04	2e+05	1.4e+05	2.8e+05	1.9e+05	2.9e+05	2.6e+05
count	21	21	21	21	21	21	1.3e+02
max	9.4e+04	2.2e+05	1.9e+05	3.1e+05	2.4e+05	3.1e+05	3.1e+05
mean	7.3e+04	1.9e+05	1.4e+05	2.8e+05	1.9e+05	2.7e+05	1.9e+05
min	5.6e+04	1.7e+05	1.4e+05	2.3e+05	1.8e+05	2.2e+05	5.6e+04
range	3.8e+04	4.8e+04	5.5e+04	8.3e+04	6.2e+04	8.3e+04	2.6e+05
std	1.2e+04	1.5e+04	1.1e+04	1.9e+04	1.6e+04	2.7e+04	7.2e+04

Table 31. Summary Statistics for Workload C performed on a 1GB limited hardware, Raspberry Pi node over a(n)802.11a/b/g/n network. Except for count, all values are in milliseconds.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	slope	intercept	r_value	p_value	std_err
1	2.5e+04	4.8e+04	0.83	1.3e-11	2.6e+03
2	1.1e+04	1.8e+05	0.43	0.0049	3.7e+03
3	3e+04	1.1e+05	0.88	2.2e-14	2.6e+03
4	1.7e+05	1e+05	0.99	2.5e-34	4.2e+03
5	9.2e+04	9.9e+04	0.97	2.7e-26	3.6e+03
6	1.6e+05	1.1e+05	0.97	7.4e-27	6e+03
OVERALL	8.1e+04	1.1e+05	0.58	9e-24	7.3e+03

Table 33. Linear Regression over the effect of 802.11 links, Workload C

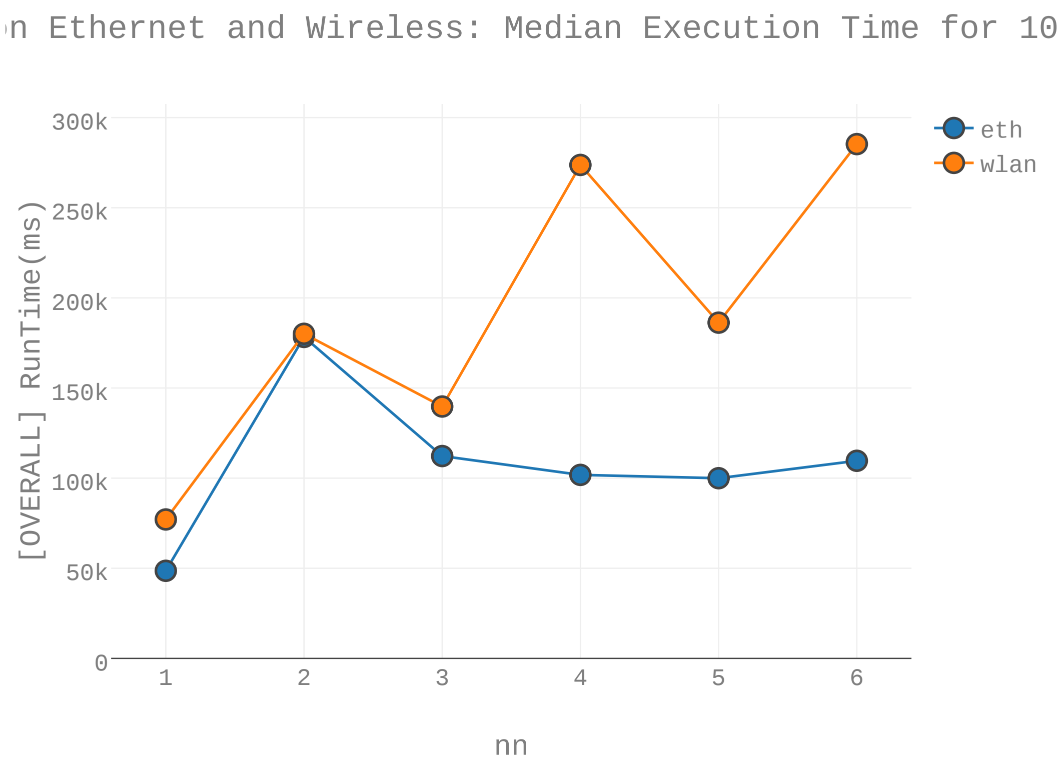

Figure 27. Comparison between links: Ethernet wired versus 802.11 wireless. As the cluster sizes increase, these show a tendency to diverge.

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.91	1.1	0.92	0.46	0.57	0.5	3.4
ratio_min_to_max	0.48	0.73	0.57	0.32	0.4	0.34	0.14
ratio_of_the_means	0.66	0.94	0.79	0.37	0.52	0.41	0.57
ratio_of_the_medians	0.69	0.94	0.8	0.38	0.53	0.4	0.56
ratio_of_the_stddevs	0.15	0.43	0.28	0.061	0.12	0.078	0.54

Table 34. Speedup over the effect of 802.11 links, Workload C

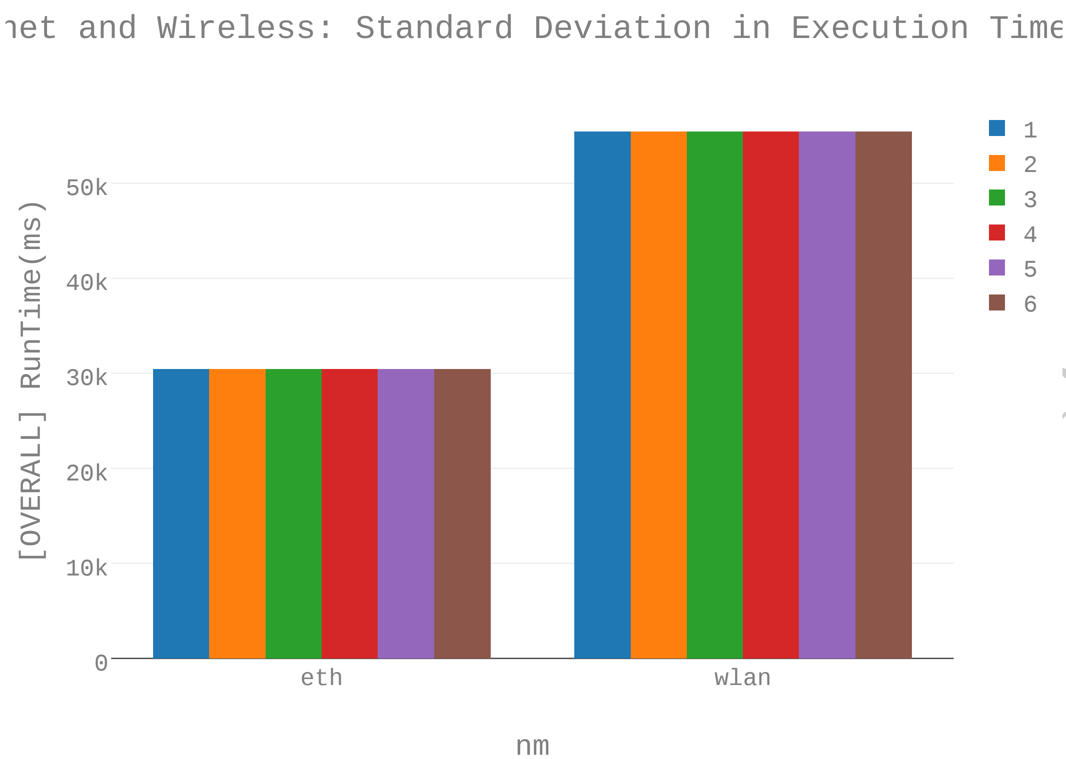

Figure 28. Standard deviation of execution time in milliseconds. There is a significant increase when going from the wired to the wireless configuration.

5.3 Results for Workload E

5.3.1 Comparing Existing Work: Virtual Machine vs the Reference Value.

Initial Observations.

The result medians are displayed in Figure 30. As expected, the virtual machine results imply much, much less execution time compared to the reference value, presumably accounting for diminished network latency. For the reference value, it seems a higher cluster size seems to imply a cost, but a cost that decreases as the cluster size increases. However, further analysis would be required to see if this is not just the product of normal variation. Because of the high contrast, it difficult to reach any initial predictions for virtual machine. The flat curve may just be an illusion, a minimization of the differences due to the scale of the graph.

Figure 29. This scatterplot compares the values from efAbramova2014TestingCassandra to the median result of Workload E executed on the virtual machine.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload E performed on the virtual machines are in Tables 36, 35, and 37.

Analysis.

This section will take a more in-depth look at the data.

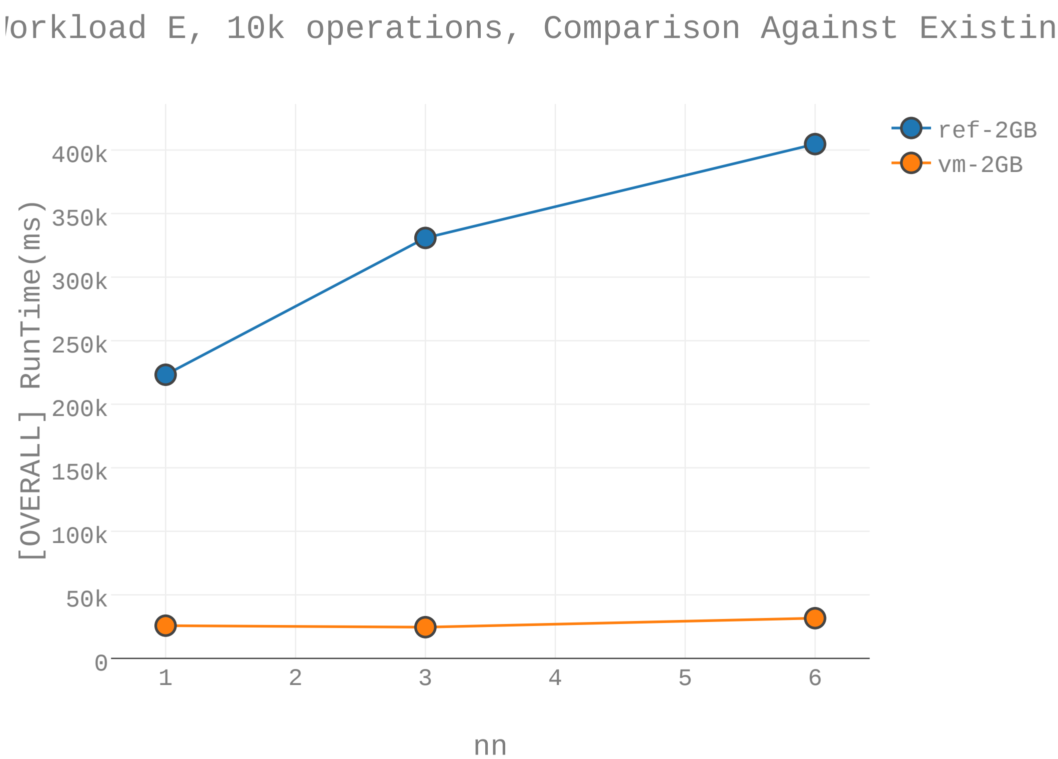

Figure 30. This compares the 2GB virtual machine with the corresponding value from ??.

5.3.2 1GB RAM vs 2GB RAM vs 4GB RAM.

Initial Observations.

The results are displayed in Figure 31. First of all, the performance for the a one-node cluster differs significantly between 1GB RAM andall other cases. While this could be some kind of implementation error, it is best not to draw any conclusions fromthis. It would be best to run these tests again, possibly switching the order to eliminate the possible effect of interference or a mix-up of data. That one case aside, the amount of RAM seems to have had an effect on the results, but there doesnot seem to be any predictive relationship that holds true across cluster sizes.

cluster_size	1.0	3.0	6.0	Overall
25%	2.3e+04	2.4e+04	3.1e+04	2.4e+04
50%	2.5e+04	2.4e+04	3.1e+04	2.6e+04
75%	2.6e+04	2.5e+04	3.2e+04	3.1e+04
count	21	21	21	63
max	2.9e+04	2.6e+04	3.3e+04	3.3e+04
mean	2.5e+04	2.5e+04	3.1e+04	2.7e+04
min	2.1e+04	2.3e+04	3.1e+04	2.1e+04
range	8e+03	3e+03	2e+03	1.1e+04
std	2.2e+03	7.5e+02	5.3e+02	3.5e+03

Table 35. Summary Statistics for Workload E performed on a 2GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload E performed on the virtual machines are in Tables 36, 35, and 37.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	slope	intercept	r_value	p_value	std_err
1	-1.1e+05	3.9e+05	-0.81	5e-21	8.3e+03
3	32	2.4e+04	0.037	0.77	1.1e+02
6	4.7e+02	3.1e+04	0.57	1.4e-06	88

Table 38. Linear Regression over amount of RAM

5.3.3 Implementation on Raspberry Pi.

Initial Observations.

The results are displayed in Figure 32.

cluster_size	1.0	3.0	6.0	Overall
25%	3.2e+05	2.3e+04	3.2e+04	3e+04
50%	3.5e+05	2.3e+04	3.2e+04	1.7e+05
75%	3.5e+05	2.4e+04	3.3e+04	3.4e+05
count	42	21	21	84
max	3.7e+05	2.6e+04	3.4e+04	3.7e+05
mean	3.4e+05	2.3e+04	3.2e+04	1.8e+05
min	3.1e+05	2.2e+04	3.2e+04	2.2e+04
range	5.9e+04	4.1e+03	2.6e+03	3.5e+05
std	1.8e+04	1e+03	6.1e+02	1.6e+05

Table 36. Summary Statistics for Workload E performed on a 1GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

cluster_size	1.0	3.0	6.0	Overall
25%	2.4e+04	2.3e+04	3.3e+04	2.4e+04
50%	2.5e+04	2.4e+04	3.4e+04	2.5e+04
75%	2.7e+04	2.4e+04	3.4e+04	3.3e+04
count	21	21	21	63
max	2.8e+04	2.8e+04	3.5e+04	3.5e+04
mean	2.5e+04	2.4e+04	3.4e+04	2.8e+04
min	1.9e+04	2.3e+04	3.3e+04	1.9e+04
range	8.4e+03	5.6e+03	1.9e+03	1.5e+04
std	2.1e+03	1.2e+03	6.1e+02	4.6e+03

Table 37. Summary Statistics for Workload E performed on a 4GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

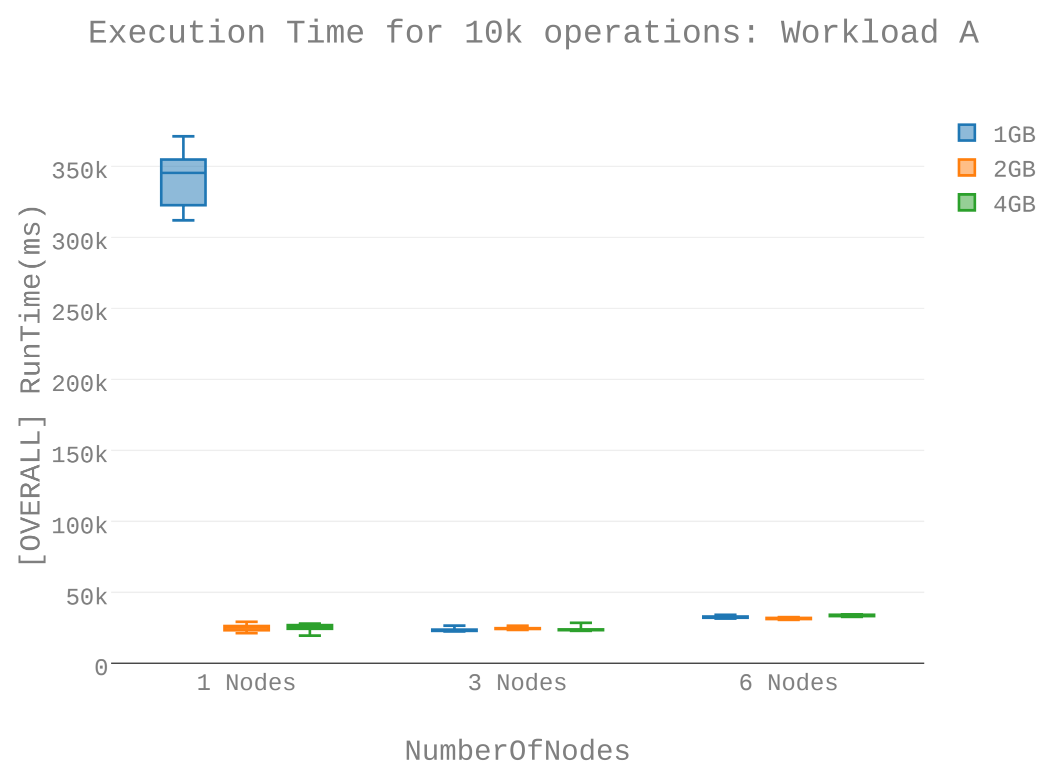

Figure 31. Execution time for virtual machines with 1GB, 2GB, and 4GB of RAM. The first 9 trials have been removed in order to filter out the trials representing cache effect and thus represents the steady state.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload E performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 39.

Analysis.

This section will take a more in-depth look at the data.

slope	intercept	r_value	p_value	std_err
7.2e+03	5.1e+05	0.84	2.8e-35	4.1e+02

Table 40. Linear Regression over Cluster Size, Workload e

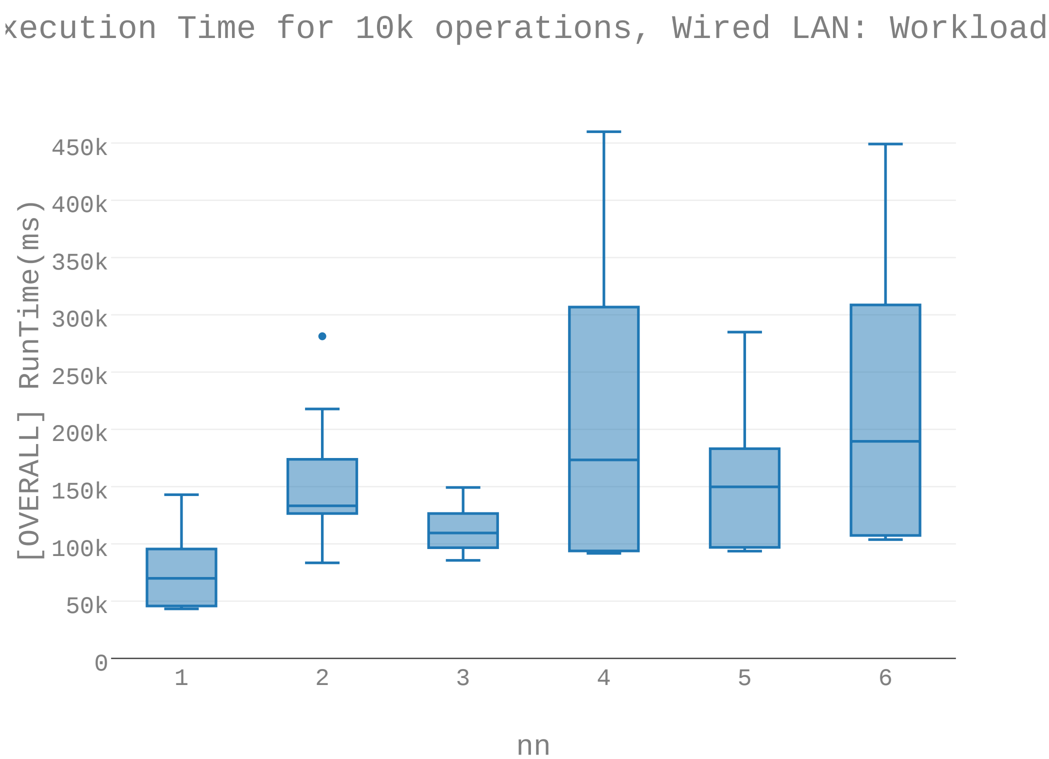

Figure 32. Results of limited hardware, the Raspberry Pi, on an Ethernet LAN. Execution time is plotted over cluster size.

5.3.4 Raspberry Pi vs Reference Value.

Initial Observations.

The results are displayed in Figure 33. These medians seem to indicate, that for this workload, any effect of the Raspberry Pi is exacerbated with this particular workload.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload E performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 39.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	5.2e+05	5.2e+05	5.1e+05	5.3e+05	5.4e+05	5.6e+05	5.2e+05
50%	5.2e+05	5.2e+05	5.2e+05	5.3e+05	5.4e+05	5.6e+05	5.3e+05
75%	5.2e+05	5.3e+05	5.2e+05	5.4e+05	5.4e+05	5.6e+05	5.4e+05
count	21	21	21	21	21	21	1.3e+02
max	5.2e+05	5.3e+05	5.3e+05	5.4e+05	5.4e+05	5.7e+05	5.7e+05
mean	5.2e+05	5.2e+05	5.2e+05	5.3e+05	5.4e+05	5.6e+05	5.3e+05
min	5.2e+05	5.2e+05	5.1e+05	5.2e+05	5.3e+05	5.6e+05	5.1e+05
range	8.3e+03	1.3e+04	2e+04	1.8e+04	9.9e+03	1e+04	5.3e+04
std	2.1e+03	3.5e+03	5.5e+03	6e+03	2.8e+03	3e+03	1.5e+04

Table 39. Summary Statistics for Workload E performed on a 1GB limited hardware, Raspberry Pi node over a(n)Ethernet network. Except for count, all values are in milliseconds.

Analysis.

This section will take a more in-depth look at the data.

5.3.5 Raspberry Pi vs Virtual Machine.

Initial Observations.

The results are displayed in Figure 33. As expected, there is a significant differential between the limited hardware, the Raspberry Pi configuration and the virtual machine. However, it is not clear if this is linear across cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload E performed on the virtual machines are in Tables 36, 35, and 37. The summary statistics for Workload E performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 39.

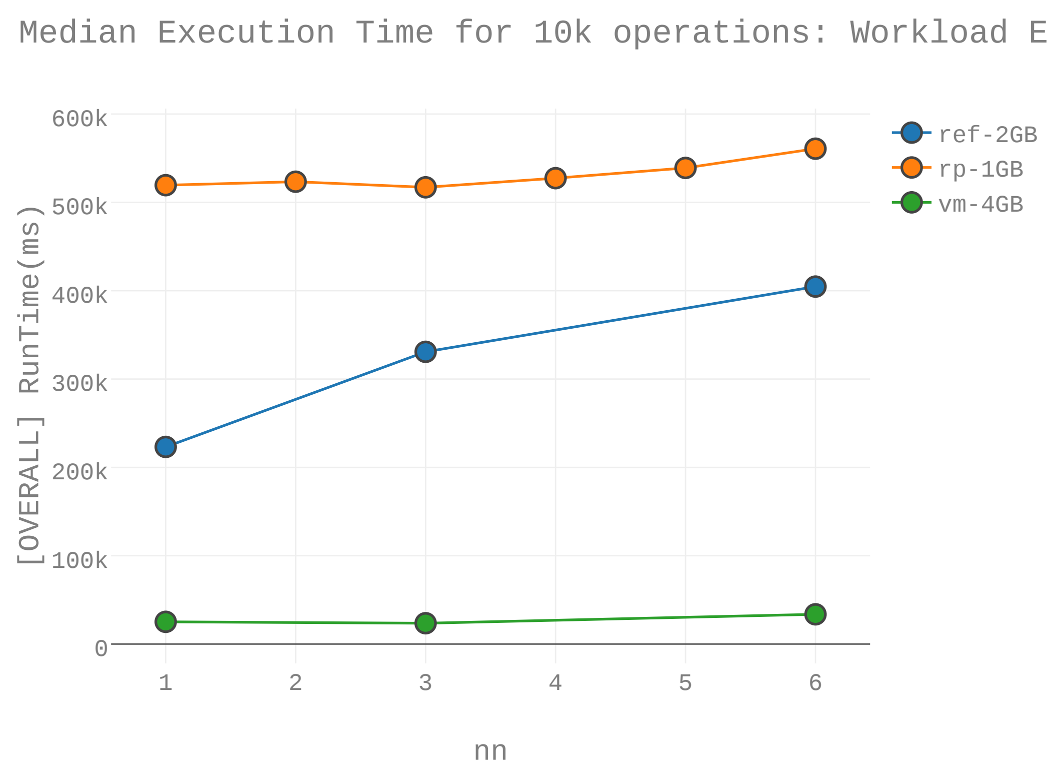

Figure 33. Comparison among the Raspberry Pi nodes (rp-1GB), the results reported in ??, and the virtual nodes with 1GB of RAM.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	slope	intercept	r_value	p_value	std_err
1	1.8e+05	3.4e+05	0.99	2e-49	3.9e+03
2	nan	nan	0	1	inf
3	5e+05	2.3e+04	1	8.9e-74	1.2e+03
4	nan	nan	0	1	inf
5	nan	nan	0	1	inf
6	5.3e+05	3.2e+04	1	1.3e-85	6.6e+02
OVERALL	3.5e+05	1.8e+05	0.86	1.3e-63	1.4e+04

Table 41. Linear Regression over the effect of limited hardware, Workload E

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.72	NaN	0.052	NaN	NaN	0.061	0.72
ratio_min_to_max	0.6	NaN	0.042	NaN	NaN	0.056	0.04
ratio_of_the_means	0.65	NaN	0.045	NaN	NaN	0.058	0.35
ratio_of_the_medians	0.66	NaN	0.045	NaN	NaN	0.058	0.33
ratio_of_the_stddevs	8.4	NaN	0.18	NaN	NaN	0.21	11

Table 42. Speedup over the effect of limited hardware, Workload E

5.3.6 Wireless Links Only.

Initial Observations.

The results are displayed in Figure 34. Although there is a general trend of increased execution time over cluster size, the oscillation that occurs between odd and even node cluster sizes is hard to miss. This may be result of the collision avoidance strategy, but further experiments would be needed to determine a more specific explanation.

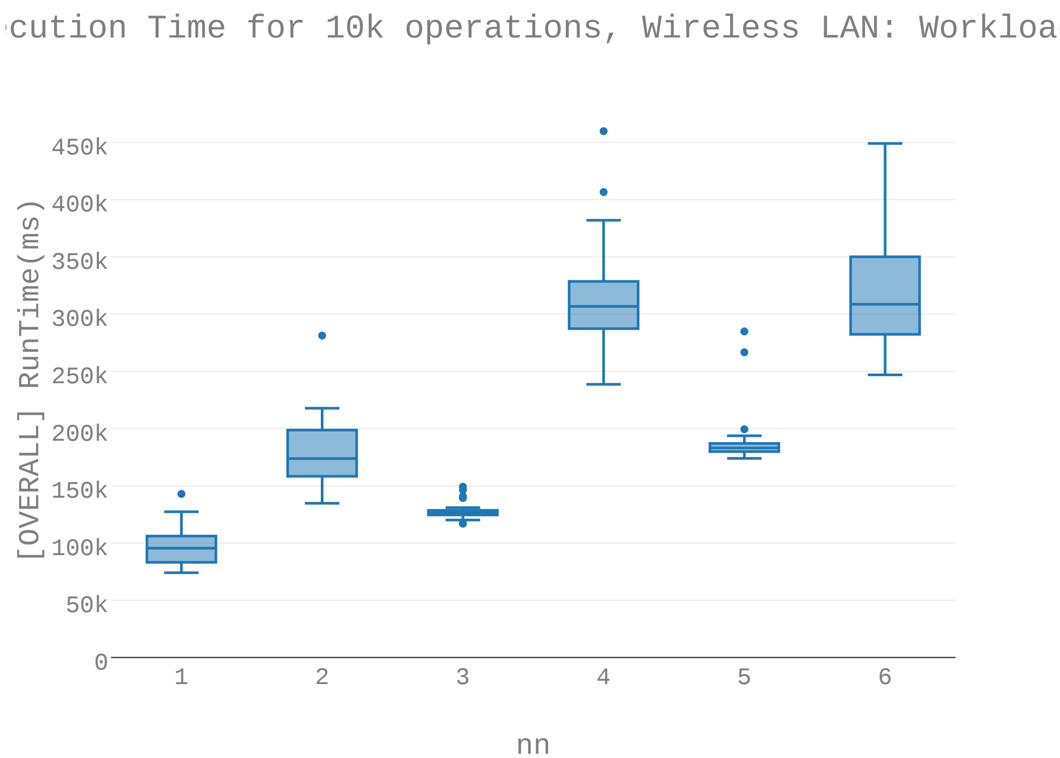

Figure 34. Results of wireless testing. There seems to be a steady climb in execution time as the cluster size increases. Any oscillation cannot be explained with current analysis and would require additional experimentation.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload E performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 43.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	6.4e+05	1.2e+06	1.3e+06	1.8e+06	1.4e+06	2.6e+02	6.8e+05
50%	6.5e+05	1.2e+06	1.3e+06	1.8e+06	1.5e+06	2.8e+02	1.3e+06
75%	6.6e+05	1.3e+06	1.3e+06	1.9e+06	1.5e+06	1.8e+06	1.6e+06
count	21	21	21	21	21	21	1.3e+02
max	7.2e+05	1.9e+06	1.5e+06	2.5e+06	1.6e+06	2e+06	2.5e+06
mean	6.5e+05	1.3e+06	1.3e+06	1.9e+06	1.5e+06	6.9e+05	1.2e+06
min	6.3e+05	1.1e+06	1.2e+06	1.7e+06	1.4e+06	2.6e+02	2.6e+02
range	9.9e+04	8e+05	2.6e+05	7.6e+05	2.4e+05	2e+06	2.5e+06
std	2.4e+04	2.1e+05	6.4e+04	1.7e+05	6.8e+04	9e+05	5.8e+05

Table 43. Summary Statistics for Workload E performed on a 1GB limited hardware, Raspberry Pi node over a(n)802.11a/b/g/n network. Except for count, all values are in milliseconds.

slope	intercept	r_value	p_value	std_err
4.1e+04	1.1e+06	0.12	0.18	3e+04

Table 44. Linear Regression over Cluster Size, Workload e

5.3.7 Wireless Links vs Wired Links.

Initial Observations.

The results are displayed in Figures 35 and 36. For 1, 2, and 3-node clusters, despite the expected disparity in execution time, the wired and wireless trends seem to follow each other. However, from 3 nodes up through 6 nodes, the execution times starts to diverge, suggesting that the wireless has a increasing effect as the number of nodes increases.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data.

The summary statistics for Workload E performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 39. The summary statistics

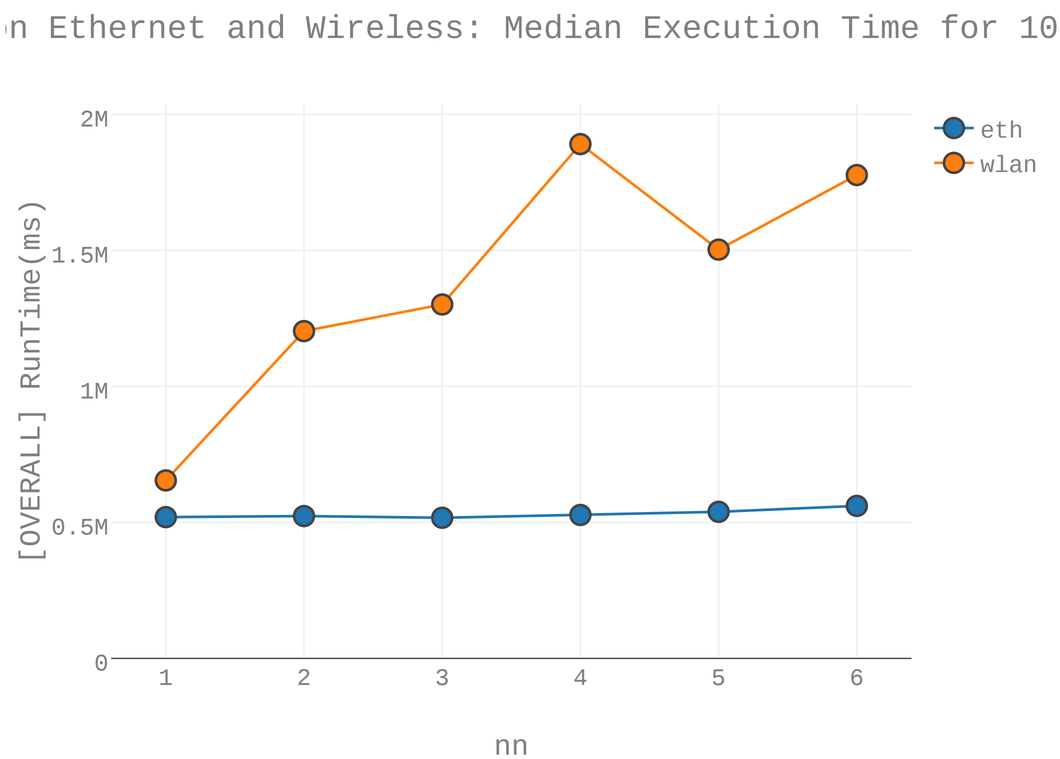

Figure 35. Comparison between links: Ethernet wired versus 802.11 wireless. As the cluster sizes increase, these show a tendency to diverge.

for Workload E performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 43.

Analysis.

This section will take a more in-depth look at the data.

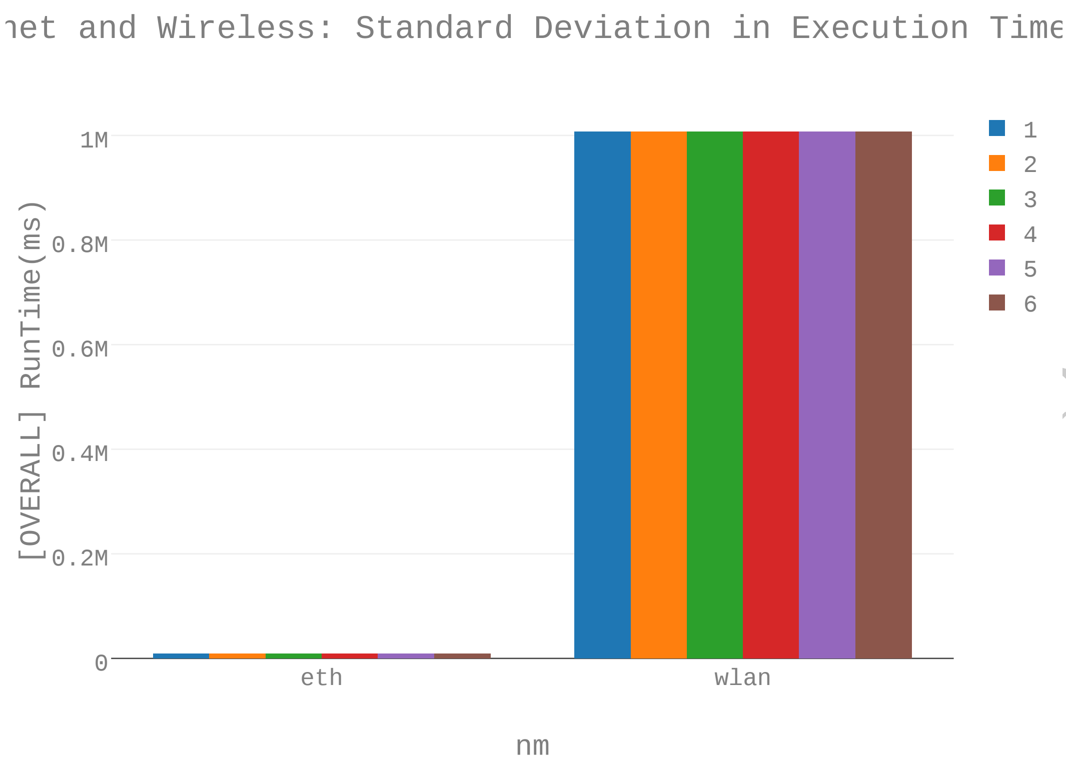

Figure 36. Standard deviation of execution time in milliseconds. There is a significant increase when going from the wired to the wireless configuration.

cluster_size	slope	intercept	r_value	p_value	std_err
1	1.3e+05	5.2e+05	0.97	5e-26	5.4e+03
2	7.5e+05	5.2e+05	0.93	1.8e-19	4.5e+04
3	7.8e+05	5.2e+05	0.99	1.5e-39	1.4e+04
4	1.4e+06	5.3e+05	0.98	5.8e-32	3.8e+04
5	9.5e+05	5.4e+05	1	5.2e-42	1.5e+04
6	1.3e+05	5.6e+05	0.1	0.51	2e+05
OVERALL	6.8e+05	5.3e+05	0.64	9e-31	5.2e+04

Table 45. Linear Regression over the effect of 802.11 links, Workload E

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.84	0.48	0.44	0.32	0.4	2.2e+03	2.2e+03
ratio_min_to_max	0.71	0.27	0.35	0.21	0.33	0.27	0.21
ratio_of_the_means	0.8	0.41	0.4	0.28	0.36	0.81	0.44
ratio_of_the_medians	0.8	0.43	0.4	0.29	0.36	2e+03	0.4
ratio_of_the_stddevs	0.085	0.017	0.087	0.034	0.041	0.0033	0.025

Table 46. Speedup over the effect of 802.11 links, Workload E

5.4 Results for Workload I

5.4.1 1GB RAM vs 2GB RAM vs 4GB RAM.

Initial Observations.

The results are displayed in Figure 37. Here there is a visible trend of 4GB RAM implying better performance results than both 1GB RAM and 2GB RAM. However, the relationship between 1GB RAM and 2GB RAM does not seem to be predictable across cluster node sizes.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload I performed on the virtual machines are in Tables 47, **??**, and 48.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	9.9e+03	1e+04	1.3e+04	1.4e+04	1.9e+04	1.4e+04	1.3e+04
50%	1e+04	1.4e+04	1.6e+04	1.9e+04	2.4e+04	1.5e+04	1.6e+04
75%	1.5e+04	2.2e+04	2e+04	2.2e+04	2.7e+04	1.6e+04	2e+04
count	21	21	21	21	21	21	1.3e+02
max	2e+04	2.3e+04	2.6e+04	4e+04	3.5e+04	1.6e+04	4e+04
mean	1.2e+04	1.5e+04	1.7e+04	1.9e+04	2.4e+04	1.5e+04	1.7e+04
min	8.9e+03	9.9e+03	1.2e+04	1.3e+04	1.7e+04	1.4e+04	8.9e+03
range	1.1e+04	1.3e+04	1.4e+04	2.7e+04	1.9e+04	1.7e+03	3.1e+04
std	3.7e+03	5.3e+03	4.4e+03	6.3e+03	5.7e+03	6.8e+02	6e+03

Table 47. Summary Statistics for Workload I performed on a 1GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	6.5e+03	8.8e+03	1e+04	1.1e+04	1.3e+04	1.4e+04	9.6e+03
50%	7e+03	8.9e+03	1e+04	1.1e+04	1.3e+04	1.4e+04	1.1e+04
75%	1.3e+04	8.9e+03	1.1e+04	1.2e+04	1.4e+04	1.4e+04	1.4e+04
count	21	21	21	21	21	21	1.3e+02
max	2.4e+04	9.4e+03	1.2e+04	1.2e+04	1.4e+04	1.4e+04	2.4e+04
mean	1.1e+04	8.9e+03	1e+04	1.1e+04	1.3e+04	1.4e+04	1.2e+04
min	6.3e+03	8.7e+03	1e+04	1.1e+04	1.3e+04	1.4e+04	6.3e+03
range	1.8e+04	7.4e+02	1.5e+03	1.4e+03	1.8e+03	4.7e+02	1.8e+04
std	5.7e+03	1.5e+02	3.6e+02	3.6e+02	5.5e+02	1.4e+02	2.9e+03

Table 48. Summary Statistics for Workload I performed on a 4GB virtual machine node over a(n)nodal network. Except for count, all values are in milliseconds.

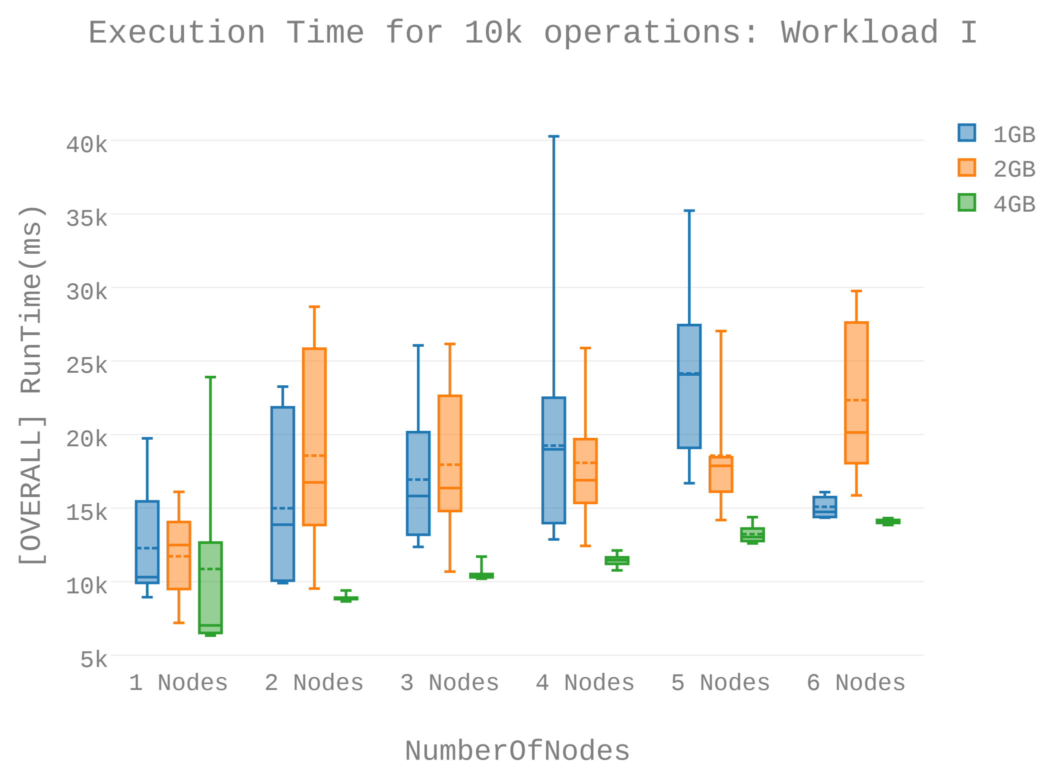

Figure 37. Execution time for virtual machines with 1GB, 2GB, and 4GB of RAM. The first 9 trials have been removed in order to filter out the trials representing cache effect and thus represents the steady state.

cluster_size	slope	intercept	r_value	p_value	std_err
1	-4.7e+02	1.3e+04	-0.14	0.28	4.2e+02
2	-2.4e+03	2e+04	-0.49	4e-05	5.5e+02
3	-2.4e+03	2.1e+04	-0.6	2.2e-07	4.1e+02
4	-2.7e+03	2.3e+04	-0.62	6.7e-08	4.4e+02
5	-3.5e+03	2.7e+04	-0.74	6.3e-12	4.1e+02
6	-8.8e+02	1.9e+04	-0.24	0.062	4.6e+02

Table 49. Linear Regression over amount of RAM

5.4.2 Implementation on Raspberry Pi.

Initial Observations.

The results are displayed in Figure 38.

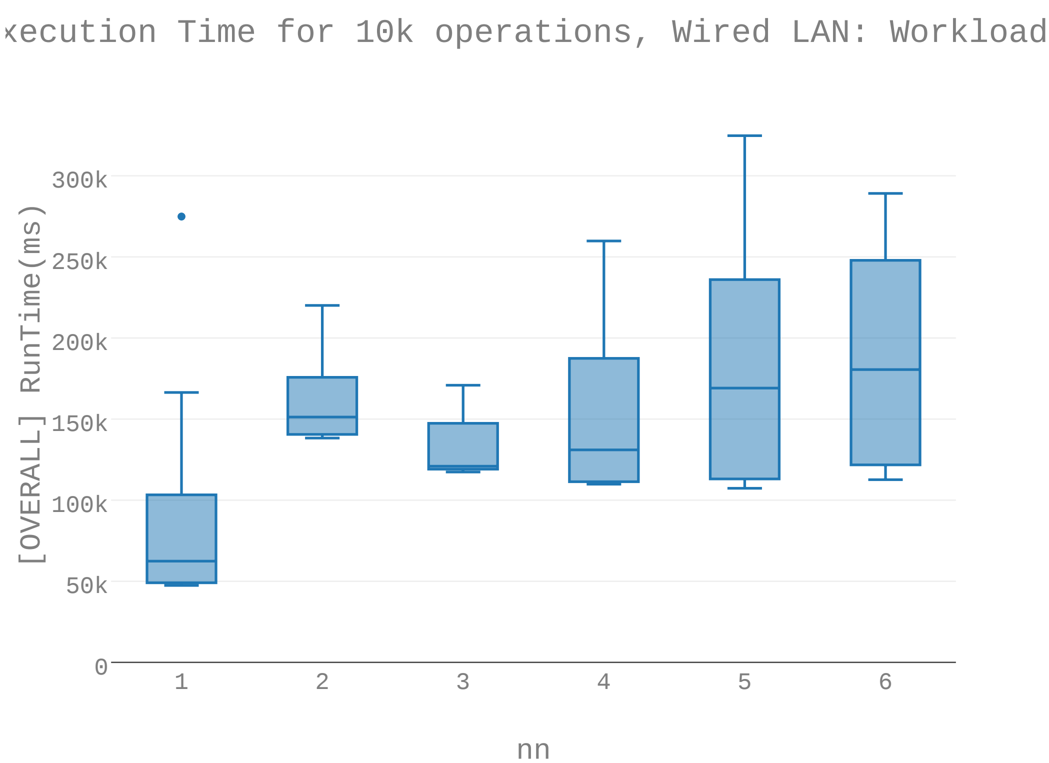

Figure 38. Results of limited hardware, the Raspberry Pi, on an Ethernet LAN. Execution time is plotted over cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload I performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 50.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	4.9e+04	1.4e+05	1.2e+05	1.1e+05	1.1e+05	1.2e+05	1.1e+05
50%	4.9e+04	1.4e+05	1.2e+05	1.1e+05	1.1e+05	1.2e+05	1.2e+05
75%	5e+04	1.4e+05	1.2e+05	1.1e+05	1.2e+05	1.2e+05	1.2e+05
count	21	21	21	21	21	21	1.3e+02
max	5.2e+04	1.4e+05	1.2e+05	1.1e+05	1.2e+05	1.3e+05	1.4e+05
mean	4.9e+04	1.4e+05	1.2e+05	1.1e+05	1.1e+05	1.2e+05	1.1e+05
min	4.7e+04	1.4e+05	1.2e+05	1.1e+05	1.1e+05	1.2e+05	4.7e+04
range	4.8e+03	4.8e+03	3.4e+03	3.1e+03	7.8e+03	8e+03	9.6e+04
std	9.2e+02	1.1e+03	9.9e+02	7.7e+02	2.2e+03	2.1e+03	2.9e+04

Table 50. Summary Statistics for Workload I performed on a 1GB limited hardware, Raspberry Pi node over a(n)Ethernet network. Except for count, all values are in milliseconds.

slope	intercept	r_value	p_value	std_err
7.8e+03	8.2e+04	0.47	3e-08	1.3e+03

Table 51. Linear Regression over Cluster Size, Workload i

5.4.3 Raspberry Pi vs Virtual Machine.

Initial Observations.

The results are displayed in Figure **??**. As expected, there is a significant differential between the limited hardware, the Raspberry Pi configuration and the virtual machine. However, it is not clear if this is linear across cluster size.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload I performed on the virtual machines are in Tables 47, **??**, and 48. The summary statistics for Workload I performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 50.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	slope	intercept	r_value	p_value	std_err
1	3.7e+04	1.2e+04	0.99	1.1e-35	8.3e+02
2	1.3e+05	1.5e+04	1	1.2e-50	1.2e+03
3	1e+05	1.7e+04	1	2.1e-50	9.8e+02
4	9.2e+04	1.9e+04	1	1.4e-42	1.4e+03
5	8.9e+04	2.4e+04	1	9.1e-43	1.3e+03
6	1.1e+05	1.5e+04	1	4.8e-63	4.9e+02
OVERALL	9.2e+04	1.7e+04	0.91	1.7e-99	2.6e+03

Table 52. Linear Regression over the effect of limited hardware, Workload I

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.42	0.17	0.22	0.37	0.32	0.14	0.85
ratio_min_to_max	0.17	0.069	0.1	0.11	0.14	0.11	0.062
ratio_of_the_means	0.25	0.11	0.14	0.17	0.21	0.12	0.16
ratio_of_the_medians	0.21	0.099	0.13	0.17	0.21	0.12	0.13
ratio_of_the_stddevs	4	4.6	4.4	8.2	2.5	0.32	0.21

Table 53. Speedup over the effect of limited hardware, Workload I

5.4.4 Wireless Links Only.

Initial Observations.

The results are displayed in Figure 39. There does not seem to be the oscillation that there was in the other workloads, which seems to suggest that somehow the reads and scans that dominate the other workloads are somehow correlated with the oscillation previously observed. Furthermore, this would seem to suggest that Workload I or any similar workload could be used as a control in such an experiment to further investigate the source of the oscillation observed in other, more read-heavy workloads. As far as the general trend, the results depicted here seem to suggest that at from 3-node clusters on, there seems to be an diminishing increase in execution time as the node cluster increases.

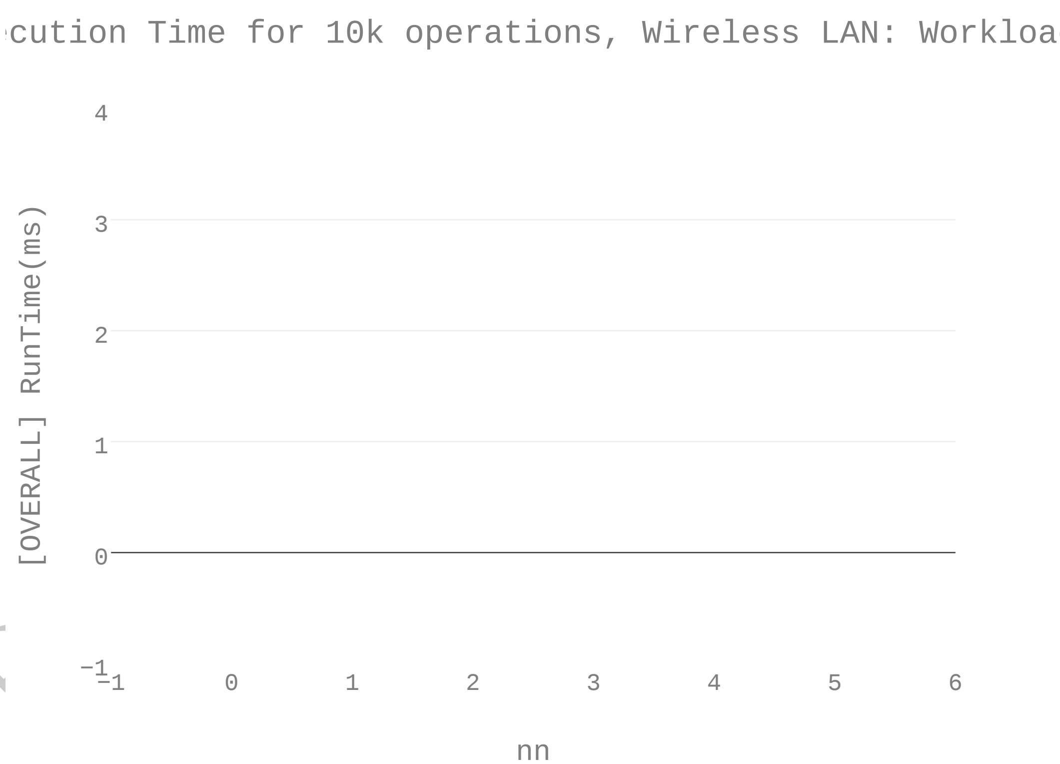

Figure 39. Results of wireless testing. There seems to be a steady climb in execution time as the cluster size increases. Any oscillation cannot be explained with current analysis and would require additional experimentation.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload I performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 54.

Analysis.

This section will take a more in-depth look at the data.

slope	intercept	r_value	p_value	std_err
2.7e+04	8.7e+04	0.83	3.4e-33	1.7e+03

Table 55. Linear Regression over Cluster Size, Workload i

5.4.5 Wireless Links vs Wired Links.

Initial Observations.

The results are displayed in Figures 40 and 41. For 1, 2, and 3-node clusters, despite the expected disparity in execution time, the wired and wireless trends seem to follow each other. However, from 3 nodes up through 6 nodes, the execution times starts to diverge, suggesting that the wireless has a increasing effect as the number of nodes increases.

Ordinal Statistics.

This section will describe some of the summary statistics that describe the data. The summary statistics for Workload I performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 50. The summary statistics for Workload I performed on the limited hardware, Raspberry Pi, on the Ethernet local area network are in Table 54.

cluster_size	1.0	2.0	3.0	4.0	5.0	6.0	Overall
25%	7.6e+04	1.7e+05	1.3e+05	1.7e+05	2.3e+05	2.4e+05	1.5e+05
50%	8.8e+04	1.8e+05	1.4e+05	1.9e+05	2.4e+05	2.5e+05	1.8e+05
75%	1.2e+05	1.8e+05	1.5e+05	1.9e+05	2.4e+05	2.6e+05	2.4e+05
count	21	21	21	21	21	21	1.3e+02
max	2.7e+05	1.9e+05	1.6e+05	2.6e+05	3.2e+05	2.9e+05	3.2e+05
mean	1.1e+05	1.8e+05	1.4e+05	1.9e+05	2.4e+05	2.5e+05	1.8e+05
min	6.9e+04	1.7e+05	1.2e+05	1.5e+05	2.2e+05	2.4e+05	6.9e+04
range	2.1e+05	2.4e+04	4.3e+04	1.1e+05	1e+05	5.4e+04	2.6e+05
std	4.7e+04	6.1e+03	1.4e+04	2.3e+04	2.1e+04	1.3e+04	5.7e+04

Table 54. Summary Statistics for Workload I performed on a 1GB limited hardware, Raspberry Pi node over a(n)802.11a/b/g/n network. Except for count, all values are in milliseconds.

Analysis.

This section will take a more in-depth look at the data.

cluster_size	slope	intercept	r_value	p_value	std_err
1	5.6e+04	4.9e+04	0.65	2.6e-06	1e+04
2	3.7e+04	1.4e+05	0.97	2e-27	1.3e+03
3	2.1e+04	1.2e+05	0.74	1.8e-08	3e+03
4	7.4e+04	1.1e+05	0.92	9.9e-18	5.1e+03
5	1.2e+05	1.1e+05	0.97	3.6e-27	4.7e+03
6	1.3e+05	1.2e+05	0.99	1.6e-35	2.9e+03
OVERALL	7.4e+04	1.1e+05	0.64	3.6e-30	5.6e+03

Table 56. Linear Regression over the effect of 802.11 links, Workload I

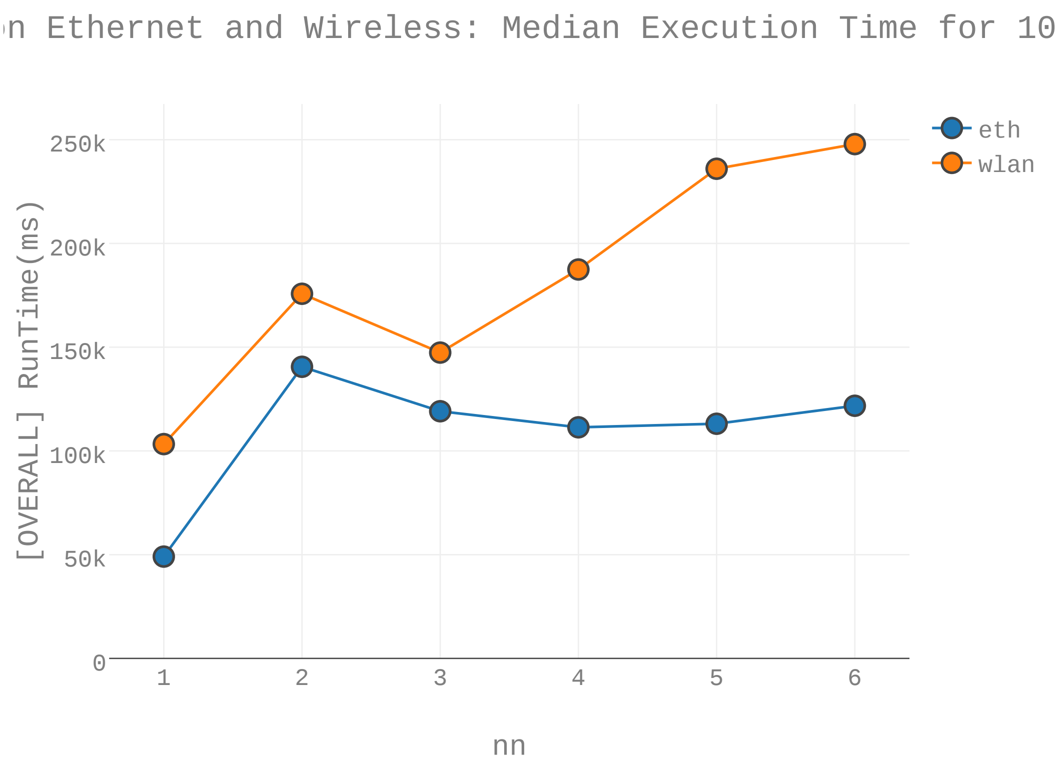

Figure 40. Comparison between links: Ethernet wired versus 802.11 wireless. As the cluster sizes increase, these show a tendency to diverge.

	0	1	2	3	4	5	6
cluster_size	1	2	3	4	5	6	OVERALL
ratio_max_to_min	0.75	0.85	1	0.76	0.52	0.53	2.1
ratio_min_to_max	0.17	0.72	0.72	0.43	0.34	0.41	0.15
ratio_of_the_means	0.47	0.79	0.85	0.6	0.48	0.48	0.6
ratio_of_the_medians	0.56	0.79	0.84	0.6	0.48	0.49	0.66
ratio_of_the_stddevs	0.02	0.19	0.071	0.033	0.1	0.16	0.51

Table 57. Speedup over the effect of 802.11 links, Workload I

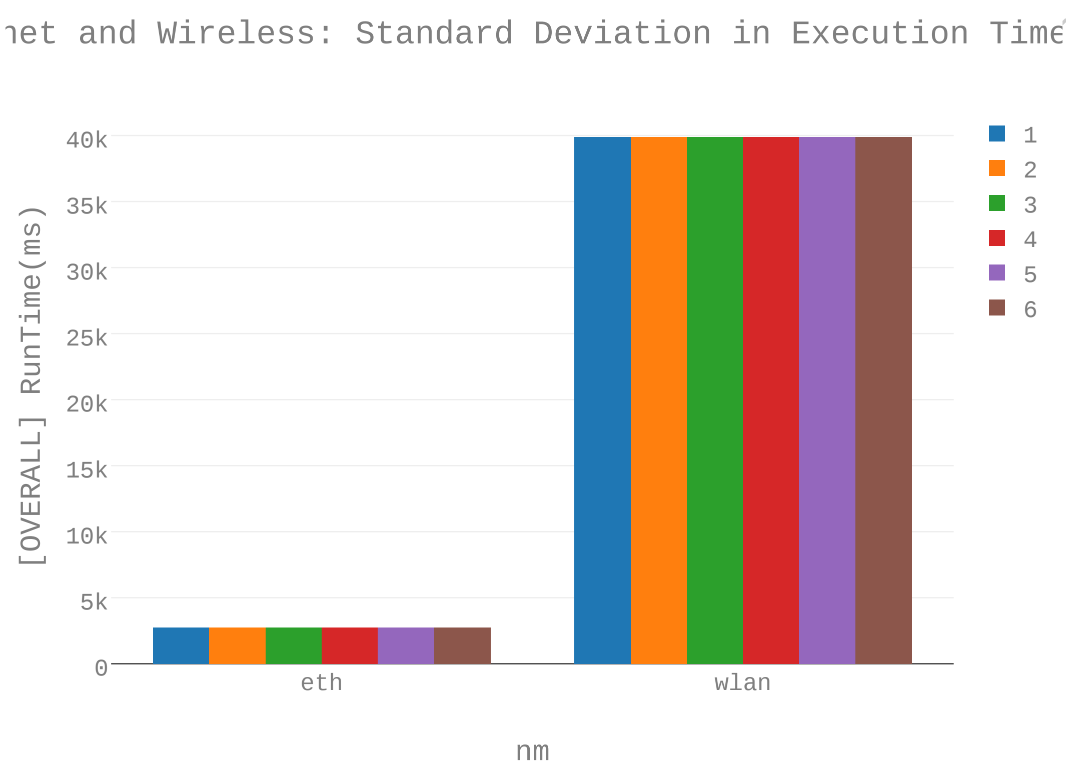

Figure 41. Standard deviation of execution time in milliseconds. There is a significant increase when going from the wired to the wireless configuration.

VI. Conclusion and Future Work

6.1 Conclusion

One of the selling points of a distributed system, and thus a distributed database, is the option to increase the likelihood of survival of data by spreading nodes geographically. While servers can and are distributed geographically as well, the light weight of the Raspberry Pi and like devices represents a mobility and flexibility that, in certain contexts, gives these nodes potential competitive advantage over a heavy server-type node. An open question for an application considering the variance in potential nodes is how the less capable CPU may affect performance.

This work imported some results from [?] to form a set of reference points, reference points that for all practical purposes represent a likely candidate platform and network upon which one would nominally use Cassandra. The fact that the machines in [?] were restricted to 2GB was a bonus for this work, as these nodes hinted at getting slightly closer to the lucrative domain of interest: in-situ IoT storage. Using these references and some controlled variation within this work, the tests done here can start to paint a picture of what one can expect porting a distributed database like Cassandra to a platform like the Raspberry Pi 2 or 3. To summarize some of the data collected, this work was able to make the following determinations with respect to the research questions 1 through 4 posed earlier:

6.1.1 Research Question 1.

For 21 trials, each consisting of 10,000 operations of workload A (50 percent reads and 50 percent updates) and performed over 1, 3, and 6-node configurations, the greatest absolute deviation from the reference to the Raspberry Pi configuration was found to be 34851.0 milliseconds, or about 35 second(s). Second, for 10,000 operations of this workload and over 1, 2, 3, 4, 5, and 6-node configurations, the greatest absolute deviation from any given trial in the wired configuration to an analogous trial using wireless configuration was found to be 367262.0 milliseconds, or about 6.1 minute(s). Third, using this workload and the one-way ANOVA test, it was determined with 95 percent confidence that varying the memory of a device does not necessarily imply a differential in performance.

6.1.2 Research Question 2.

For 21 trials, each consisting of 10,000 operations of workload C (100 percent reads) and performed over 1, 3, and 6-node configurations, the greatest absolute deviation from the reference to the Raspberry Pi configuration was found to be 43025.0 milliseconds, or about 43 second(s). Second, for 10,000 operations of this workload and over 1, 2, 3, 4, 5, and 6-node configurations, the greatest absolute deviation from any given trial in the wired configuration to an analogous trial using wireless configuration was found to be 211759.0 milliseconds, or about 3.5 minute(s). Third, using this workload and the one-way ANOVA test, it was determined with 95 percent confidence that varying the memory of a device does not necessarily imply a differential in performance.

6.1.3 Research Question 3.

For 21 trials, each consisting of 10,000 operations of workload E (100 percent scans) and performed over 1, 3, and 6-node configurations, the greatest absolute deviation from the reference to the Raspberry Pi configuration was found to be 301019.0 milliseconds, or about 5 minute(s). Second, for 10,000 operations of this workload and over 1, 2, 3, 4, 5, and 6-node configurations, the greatest absolute deviation from any given trial in the wired configuration to an analogous trial using wireless configuration was found to be 1933941.0 milliseconds, or about 32 minute(s). Third, using this workload and the one-way ANOVA test, it was determined with 95 percent confidence that varying the memory of a device does not necessarily imply a differential in performance.

6.1.4 Research Question 4.

For 21 trials, each consisting of 10,000 operations of workload I (99 percent inserts, 1 percent reads) and performed over 1, 3, and 6-node configurations, the greatest absolute deviation from the virtual machine at 1GB to the Raspberry Pi configuration was found to be nan milliseconds, or about nan week(s). Second, for 10,000 operations of this workload and over 1, 2, 3, 4, 5, and 6-node configurations, the greatest absolute deviation from any given trial in the wired configuration to an analogous trial using wireless configuration was found to be nan milliseconds, or about nan week(s). Varying the RAM was omitted for Research Question 4.

6.2 Future Work

As was alluded to in the introduction, this research seeks to support a number of possible future endeavors.

6.2.1 Generalized Model.

Overview/Introduction/Motivation.

Let's say you have a distributed sensor network (thermocouples, radars, pressure sensors or something), and you are already sold on the idea of a distributed database. It doesn't really have to be Cassandra, but let's say you were already searching for Cassandra research when you came across this report. The only question is, what kind of nodes do you choose? There is a lot of hardware out there. The application in your head narrows this down a bit, and let's further suppose you know you want something that doesn't take a whole lot of space, bringing your attention to the Raspberry Pi series and like devices.

Let's further suppose that after curling up with this document, you've decided the results are enough for you to purchase some Raspberry Pi 2's to handle your database. Problem solved: Your research phase has ended with this sentence. But, wait. Further suppose that upon logging onto Amazon, you find the Raspberry Pi 2's are out of stock. Or further suppose both that the stock has depleted, and the devices no longer get manufactured. Now you have to choose something else. You might agree on the benefit of having some way to predict performance based on hardware parameters. That method of prediction would be a mathematical model.

Hold on, you might say, before we get into all this math, what about simulation? Simulation has its place, naturally, but first of all, the simulator you want may not exist. But, even if a Google search brings up several options, you have to ask yourself a few questions. How reliable is it? And can you make the same assumptions about timing? To what ends are you required to, not to mention willing, to go to verify its utility? Simulation is certainly not ignored by this paper, however, it is really is just a discrete mathematical model with a lot of computations that comes with its own risks and costs. The software may be free or affordable, but one must ask, is it really worth your time to learn and to execute, or does an alternative exist? Sometimes, depending on what you really, truly want to know, you can trade up a small amount of precision or certainty for a dis-proportionally greater amount of your time back.

This generalized mathematical model aims for just that. The next few paragraphs will propose a relatively simple, generalized model. It will explain how the results in earlier sections of the report sow the seeds this model, and how future work could refine and develop this model into something more predictive.

The Model.

This section will describe the model in question: how to define a set of hardware nodes in terms of parameters as well as relationships among them.

Despite the complexities of hardware, this proposition asserts that a node can be abstracted to RAM, rated I/O speeds, and the processor speed. The experiments done in this report do not vary these parameters sufficiently to do an empirical model, but they provide a data point as well as a framework for developing other experiments to further refine this model. This work has already suggested that the amount of memory may not be critical or useful in predicting hardware performance, leaving I/O speeds and processor speed. This work grouped them all as one.

There has been a lot of work in trying to characterize networks, but really for a model like this, this proposition suggests the network can be characterized by the mean ping team between nodes.

Experiments that would Refine This Model.

Naturally, trying other databases, like MongoDB, in Cassandra's place is one way to achieve further confidence. In part, the motivation behind using the YCSB was its portability for testing a multitude of distributed database applications. Although the workloads for Cassandra were varied, and different databases are typically optimized and designed for particular workloads, a different database or databases would increase qualitative confidence that the model can be extended to a database not explicitly tested.

In addition, the absolute values in the results of this work do not necessarily represent Cassandra at its best. In the interest of replication, default values were used much more often than not. Cassandra boasts a multitude of parameters with which one can vary in attempt to optimize performance, almost to the point where you are almost guaranteed to be running it sub-optimally without employing a Cassandra expert.

6.2.2 Wifi Collection, Mapping and Crowd Detection.

Overview, application purpose

Data Schema.

Implementation WiFiPi prototype.

Discussion.

Bibliography

1. "Raspberry Pi 2 Model B." [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/

2. "LENOVO THINKSERVER RD650." [Online]. Available: http://cc.cnetcontent.com/inlinecontent/production/13/13e03a308886/cnet{_}4d4c{_}doc.pdf

3. "Yahoo Cloud Serving Benchmark." [Online]. Available: https://research.yahoo.com/news/yahoo-cloud-serving-benchmark

4. V. Abramova, J. Bernardino, and P. Furtado, "Testing Cloud Benchmark Scalability with Cassandra," *2014 IEEE World Congress on Services*, pp. 434–441, 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6903301

5. "CassandraHardware - Cassandra Wiki." [Online]. Available: https://wiki.apache.org/cassandra/CassandraHardware

6. A. Lakshman and P. Malik, "Cassandra - A Decentralized Structured Storage System," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 2, p. 35, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1773912.1773922

7. "Apache Cassandra Use Cases." [Online]. Available: http://www.planetcassandra.org/apache-cassandra-use-cases/

8. V. Abramova and J. Bernardino, "NoSQL databases: MongoDB vs cassandra," *Proceedings of the International C* Conference on Computer Science and Software Engineering, ACM 2013*, pp. 14–22, 2013.

9. B. Van Ryswyk, "Multi-Datacenter Cassandra on 32 Raspberry Pi's." [Online]. Available: http://www.datastax.com/dev/blog/32-node-raspberry-pi-cassandra-cluster

10. J. Sercel, "Cassandra on RaspberryPi 2 Medium." [Online]. Available: https://medium.com/@johnsercel/cassandra-on-raspberrypi-2-a84602953b23{#}.2mywozbod

11. "Cassandra Parameters for Dummies." [Online]. Available: http://www.ecyrd.com/cassandracalculator/

12. "Raspberry Pi 2 on sale now at $35 - Raspberry Pi." [Online]. Available: https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale/

13. "BeagleBoard.org - black." [Online]. Available: https://beagleboard.org/black

14. "Banana Pi M3." [Online]. Available: http://www.banana-pi.org/m3.html

15. "CHIP." [Online]. Available: https://nextthing.co/pages/chip

16. "Parallella." [Online]. Available: https://www.parallella.org/

17. C. Baun, "Mobile clusters of single board computers: an option for providing resources to student projects and researchers." *SpringerPlus*, vol. 5, no. 1, p. 360, 2016. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84961794768{&}partnerID=tZOtx3y1

18. 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," *Proceedings - International Conference on Distributed Computing Systems*, pp. 108–112, 2013.

19. J. Kiepert, "Creating a Raspberry Pi-Based Beowulf Cluster," p. 16, 2013.

20. P. J. E. Velthuis, "Small Data Center using Raspberry Pi 2 for Video Streaming," *23th Twente Student Conference on IT*, 2015.

21. Datastax, "The cassandra-stress tool." [Online]. Available: https://docs.datastax.com/en/cassandra/2.1/cassandra/tools/toolsCStress{_}t.html

22. "SensePost — Snoopy: a distributed tracking and profiling framework." [Online]. Available: https://www.sensepost.com/blog/2012/snoopy-a-distributed-tracking-and-profiling-framework/

23. "Aircrack-ng." [Online]. Available: https://www.aircrack-ng.org/

24. "AirSnort Homepage." [Online]. Available: http://faculty.ccri.edu/jbernardini/JB-Website/ETEK1500/LinuxTools/AirSnort{%}20Homepage.htm

25. I. Rose and M. Welsh, "Mapping the urban wireless landscape with Argos," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems - SenSys '10*. New York, New York, USA: ACM Press, 11 2010, p. 323. [Online]. Available: http://dl.acm.org/citation.cfm?id=1869983.1870015

26. "WiGLE: Wireless Network Mapping." [Online]. Available: https://wigle.net/

27. "HeatMapper — Free Wi-Fi coverage mapping software for homes and small offices." [Online]. Available: http://www.ekahau.com/wifidesign/ekahau-heatmapper

28. G. Castignani, A. Lampropulos, A. Blanc, and N. Montavont, "Wi2Me: A mobile sensing platform for wireless heterogeneous networks," *Proceedings - 32nd IEEE International Conference on Distributed Computing Systems Workshops, ICDCSW 2012*, pp. 108–113, 2012.

29. E. Keeble, "Casual Encounters (2013)." [Online]. Available: http://edwardkeeble.com/portfolio/casual-encounters/

30. S. Haigh, "Tracking people via Wifi (even when not connected)." [Online]. Available: https://www.crc.id.au/tracking-people-via-wifi-even-when-not-connected/

31. B. Bonne, A. Barzan, P. Quax, and W. Lamotte, "WiFiPi: Involuntary tracking of visitors at mass events," *2013 IEEE 14th International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2013*, 2013.

32. L. Schauer, M. Werner, and P. Marcus, "Estimating Crowd Densities and Pedestrian Flows Using Wi-Fi and Bluetooth," *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pp. 171–177, 2014.

33. R. Schiphorst, "Blue Mark Innovations." [Online]. Available: https://bluemark.io/aboutus/

34. C. Chilipirea, A.-c. Petre, and C. Dobre, "Presumably simple : monitoring crowds using WiFi," no. 1.

35. J. Pang, B. Greenstein, R. Gummadi, S. Srinivasan, and D. Wetherall, "802. 11 User Fingerprinting," *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*, vol. 9, pp. 99–110, 2007.

36. M. Chernyshev, C. Valli, and P. Hannay, "Service Set Identifier Geolocation for Forensic Purposes: Opportunities and Challenges," *International Conference on System Sciences*, 2016.

37. M. Cunche, M. A. Kaafar, and R. Boreli, "Linking wireless devices using information contained in Wi-Fi probe requests," *Pervasive and*

Mobile Computing, vol. 11, pp. 56–69, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.pmcj.2013.04.001

38. S. Du, J. Hua, Y. Gao, and S. Zhong, "EV-Linker: Mapping eavesdropped Wi-Fi packets to individuals via electronic and visual signal matching," *Journal of Computer and System Sciences*, vol. 82, no. 1, pp. 156–172, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022000015000677

39. M. Cunche and R. Boreli, "I know who you will meet this evening! Linking wireless devices using Wi-Fi probe requests," in *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 6 2012, pp. 1–9. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6263700

40. A. D. Luzio, A. Mei, and J. Stefa, "Mind Your Probes : De-Anonymization of Large Crowds Through Smartphone WiFi Probe Requests," in *IEEE INFOCOM 2016*, 2016.

41. A. B. M. Musa and J. Eriksson, "Tracking Unmodified Smartphones Using Wi-Fi Monitors," in *SynSys*, 2012.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
3–24–2017	Master's Thesis	Aug 2015 — Mar 2017

4. TITLE AND SUBTITLE

AFIT/ENG THESIS:
TESTING CASSANDRA'S SCALABILITY ON RASPBERRY PI

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Daniel P. Richardson

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of Electrical and Computer Engineering
2950 Hobson Way
WPAFB OH 45433-7765
DSN 271-0690, COMM 937-255-3636
Email: ???

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

—-

15. SUBJECT TERMS

LaTeX,Thesis

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. John Pecarina, PhD, AFIT/ENG
					19b. TELEPHONE NUMBER *(include area code)*
U	U	U	U	cxxvi	(937) 255-3636, x4555; daniel.richardson@afit.edu

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18