An Experimental Evaluation of Datacenter Workloads On Low-Power Embedded Micro Servers

Yiran Zhao, Shen Li, Shaohan Hu, Hongwei Wang
Shuochao Yao, Huajie Shao, Tarek Abdelzaher
Department of Computer Science
University of Illinois at Urbana-Champaign
{zhao97, shenli3, shu17, hwang172, syao9, hshao5, zaher}@illinois.edu

ABSTRACT

This paper presents a comprehensive evaluation of an ultra-low power cluster, built upon the Intel Edison based micro servers. The improved performance and high energy efficiency of micro servers have driven both academia and industry to explore the possibility of replacing conventional brawny servers with a larger swarm of embedded micro servers. Existing attempts mostly focus on mobile-class micro servers, whose capacities are similar to mobile phones. We, on the other hand, target on sensor-class micro servers, which are originally intended for uses in wearable technologies, sensor networks, and Internet-of-Things. Although sensor-class micro servers have much less capacity, they are touted for minimal power consumption ($<$ 1 Watt), which opens new possibilities of achieving higher energy efficiency in datacenter workloads. Our systematic evaluation of the Edison cluster and comparisons to conventional brawny clusters involve careful workload choosing and laborious parameter tuning, which ensures maximum server utilization and thus fair comparisons. Results show that the Edison cluster achieves up to 3.5× improvement on work-done-per-joule for web service applications and data-intensive MapReduce jobs. In terms of scalability, the Edison cluster scales linearly on the throughput of web service workloads, and also shows satisfactory scalability for MapReduce workloads despite coordination overhead.

1. INTRODUCTION

The rising demand for cloud services has been continuously driving the expansion of datacenters, which not only puts excessive pressure on power supply and cooling infrastructure, but also causes inflated energy cost. For industry giants like Google, Microsoft and Yahoo!, up to 50% of the three-year total cost of datacenters is attributed to power consumption [50]. When amortized to monthly total cost of ownership, the energy-related costs can account for up to a third [33]. Therefore, reducing datacenter power consumption has become a hot research topic.

Proceedings of the VLDB Endowment, Vol. 9, No. 9

To reduce datacenter energy cost, power proportionality [47] is one major solution studied and pursued by both academia and industry. Ideally, it allows datacenter power draw to proportionally follow the fluctuating amount of workload, thus saving energy during non-peak hours. However, current high-end servers are not energy-proportional and have narrow power spectrum between idling and full utilization [43], which is far from ideal. Therefore, researchers try to improve energy-proportionality using solutions such as dynamic provisioning and CPU power scaling. The former relies on techniques such as Wake-On-LAN [36] and VM migration [24] to power on/off servers remotely and dynamically. However, the opportunities to go into hibernation state are few, while waking up servers on demand and migrating VMs incur additional costs and unpredictability, which make these techniques less efficient [23]. The latter solution saves energy using Dynamic Voltage/Frequency Scaling (DVFS) [34], which reduces CPU voltage (therefore power) and lowers frequency as the utilization drops. But even if the CPU power consumption is proportional to workload, other components such as memory, disk and motherboard still consume the same energy [47]. Thus the energy-proportionality delivered by DVFS is not satisfactory.

Despite the large number of researches on the aforementioned complex solutions, significant energy savings are rarely reported. The best scenarios only achieve up to 30% energy reduction [26]. However, when running cloud services on energy-efficient embedded devices, the energy saving can exceed 70% in some applications [21, 53]. Compared to conventional high-end servers, embedded micro servers offer three advantages:

1. System components of embedded micro servers are innately better balanced [42]. For modern high-end CPU, the power consumption increases super-linearly with speed, a large part of which is devoted to branch prediction, speculative execution and out-of-order execution [50]. Imbalanced memory, network and I/O bandwidth often leave the CPU under-utilized, causing high-end servers to be less energy efficient.

2. Individual node failure has far less significant impact on micro clusters than on high-end clusters, simply because the number of nodes in micro clusters is larger. Another point in [29] shows that in the event of load increase due to node failure and load redistribution, Xeon cores experience more QoS degradation than small Atom cores if the workload becomes beyond sustainable point.

3. The deployment of micro clusters requires much less sophisticated power and cooling infrastructure. The immense power draw of conventional datacenters entails significant cost of designing and building the power and cooling support. Thanks to the low power nature of embedded devices, much of these investments can be considerably reduced [20, 51].

In this paper, we push the energy efficiency even further by building our cluster with sensor-class Intel Edison devices [17], and conduct more extensive benchmark evaluations. Although the Edison device is typically used for wearable technologies, Internet-of-Things and sensor networks, we show that a larger cluster of such sensor-class micro servers can collectively offer significant processing capacity for datacenter workloads when managed properly, while achieving more work-done-per-joule.

To evaluate our sensor-class Edison cluster and compare with high-end Dell cluster, we run two major categories of datacenter workloads: on-line web services and off-line data analysis. For the first category, we set up the cluster to take on major roles (web servers and cache servers) in a standard web service configuration, and have clients generate HTTP requests to test the throughput and response delay. For the second category, we choose to run well known MapReduce jobs on the widely used Hadoop framework [2]. In addition, we optimize some MapReduce jobs on both platforms to make sure that server capacity is as fully utilized as possible. The scalability of the Edison cluster is also evaluated for both categories of workloads. We show detailed performance comparison of the Edison cluster and Dell cluster with emphasis on the metric of work-done-per-joule. Our meaningful and fair comparison results can be valuable for related system research.

The rest of the paper is organized as follows. In Section 2 we discuss related work. Section 3 gives the overview of our testbed and measurement methodology. Then we benchmark and compare individual Edison server and Dell server to understand the per node performance gap in Section 4. In Section 5 we run various datacenter workloads on the clusters and compare energy efficiency in terms of work-done-per-joule. Section 6 analyzes and compares the total cost of ownership under a simple model. We discuss the lessons that we learn through our experiments in Section 7. Finally, Section 8 concludes this paper.

2. RELATED WORK

Drastic increase in the size of datacenters has attracted many research efforts to reduce power consumption. Generally, related work can be classified into two categories based on the approach of achieving energy efficiency. The first category is to seek energy proportionality with non-energy-proportional servers [53, 32, 31, 27, 35]. The second category is to build more energy-efficient architecture based on low-power CPU [38, 41, 49, 33, 43, 46, 29].

Improving energy-proportionality allows power draw to better match cluster utilization. Most work addresses this problem by intelligently powering down servers or putting them to hibernation state. Covering Set (CS) [35, 32] and All-In Strategy (AIS) [31] are two techniques to optimally shut-down cluster nodes to save energy. CS technique keeps a small set of nodes according to data locality and shut down the rest of the cluster, while AIS technique claims to be more efficient by using all the nodes to complete the job faster and shutting down the entire cluster afterwards. Berkeley Energy Efficient MapReduce [27] divides the cluster into interactive zone and batch zone, where the former takes MapReduce Interactive Analysis (MIA) workloads and the latter is responsible for non-interactive jobs and is often put into low-power state. PowerNap [39] proposes a new server architecture that is able to rapidly change the power state of different components to reduce idle power draw. However, generally speaking, switching power state and migrating workloads inevitably increase overhead and make it harder to guarantee service level agreements.

More recently, the idea of building datacenters based on low-power embedded systems has become popular. Some of these non-traditional server platforms are equipped with low-power processors, such as ARM-based CPU [38, 41, 49], Intel Atom CPU [33, 43, 46, 29, 25] or even embedded CPU [21, 50]. In addition to low-power CPU, more energy-efficiency can be achieved by exploiting low-power flash storage [25, 21] or highly customizable FPGA [40]. In industry, Applied Micro [16, 18, 19] and AMD [15] also target low-power ARM-based servers for datacenter workloads. In academia, several projects have explored using micro servers in order to gain more work-done-per-joule than conventional servers. Based on the CPU and memory capacity of micro servers used in related work, we divide the platform into two categories: 1) mobile-class servers, and 2) sensor-class servers. Shown in Table 1, the first five rows are categorized as mobile-class micro servers, since their capacity is similar to a typical mobile phone. The last two rows in the table represent sensor-class micro servers, in that they have considerably lower specifications on CPU core counts, clock rates, and RAM sizes. Sensor-class micro servers achieve ultra-low power consumption (< 1W), by sacrificing considerable computational power. Within the mobile-class category, [38, 43] run database benchmarks (query processing, OLAP, OLTP) and compare with conventional servers in terms of performance per watt. [38, 25] run MapReduce workloads on their micro servers to compare work-done-per-joule with conventional servers. In [29], mobile cores are used to run Internet-scale web search, and query speed and price efficiency are compared with conventional servers. With the increasing popularity of Raspberry Pi (either earlier or more advanced model), many clusters are built for research or educational purposes [51, 44, 20, 48]. In particular, a video streaming datacenter [51] is built and tested, and a micro datacenter running big data applications [44] is evaluated. However, the mentioned work on Raspberry Pi does not compare the energy-efficiency with conventional clusters. In the sensor-class category, FAWN [21], built upon a cluster of low-power embedded systems with flash storage, proves to be a highly energy-efficient key-value datastore, as demonstrated through a comprehensive evaluation of the system's read/write speed, latency, queries per joule, fault-tolerance, as well as comparisons to conventional database systems. However, the feasibility of running big data applications (MapReduce jobs) on FAWN is not studied.

Our work is the first to carry out comprehensive evaluation of datacenter workloads on a large cluster of sensor-class micro servers. We show, through carefully examined comparisons, that under data-intensive workloads, even the computationally-weak Edison nodes can beat brawny servers in terms of work-done-per-joule.

Table 1: Micro server specifications in related work.

	CPU	Memory
Big.LITTLE [38]	4×600MHz, 4×1.6GHz	2GB
WattDB [43]	2×1.66GHz	2GB
Gordon [25]	2×1.9GHz	2GB
Diamondville [29]	2×1.6GHz	4GB
Raspberry Pi [51]	4×900MHz	1GB
FAWN [21]	1×500MHz	256MB
Edison [17]	2×500MHz	1GB

3. OVERVIEW

Figure 1 shows part of the Edison cluster and the power supply system. The complete Edison cluster is composed of five such boxes, each containing 7 Edison micro servers and 1 switch. Each Edison compute module is stacked on a microSD extension board (red) and a breakout board (blue). The breakout board is connected with a 100Mbps Plugable USB 2.0 Ethernet Adapter and the microSD board provides 8GB extra storage. The power is supplied and measured by Mastech HY1803D DC power supply (7V).

The dimension of one Edison micro server is small, measuring 4.3×1.2×1.2in (including Ethernet adaptor and extension boards). Even if each box holds only 7 Edison micro servers, the whole 35-node cluster can sit comfortably in a 1.3m×0.5m×0.5m cabinet. In a highly compact deployment, assuming sufficient cooling, the typical 1U rack enclosure (39×19×1.75in) can contain 200 Edison micro servers. Therefore, the Edison micro cluster can be potentially more compact than conventional clusters.

To determine the number of high-end Dell servers to compare with the 35-node Edison cluster, Section 3.1 first conducts a back-of-the-envelop calculation, providing a rough estimation of the cluster size ratio to match the performance of the Edison cluster against the Dell cluster. Section 3.2 presents our measurement methodologies and testbed configuration.

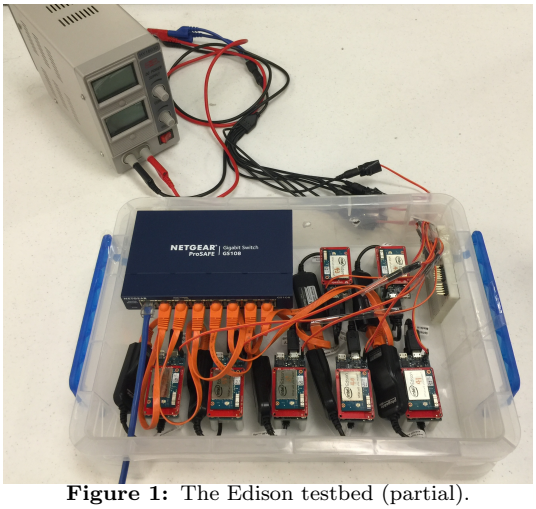

Figure 1: The Edison testbed (partial).

3.1 A Back-of-the-envelope Feasibility Argument

In order to quantitatively estimate the number of Edison servers to replace a high-end server, we compare with a typical conventional server in cost, power, size, and performance. More specifically, the comparison employs Dell PowerEdge R620 server (1U), with a purchasing cost of approximately $2.5K. Its basic configuration includes an Intel Xeon E5-2620 processor of six 2GHz cores (hyper-threaded) and 16GB memory, and comes with a 1 Gigabit Ethernet interface. We have access to a 40-machine testbed featuring the Dell PowerEdge 620R server [1].

From the perspective of resource capacity, we need 16 GB/1 GB = 16 Edison nodes to match server memory, (6 cores * 2GHz)/(2 cores * 0.5GHz) = 12 Edison nodes to match total CPU speed (without hyper-threading), and 1Gbit/0.1Gbit = 10 Edison nodes to match the network interface speed. Shown in Table 2, we estimate that the raw resource capacity of the Dell R620 server can be matched by as few as 16 Edison nodes.

Therefore, given 35 Edison servers, we choose 2 to 3 Dell servers to compare with under various datacenter workloads. Note that the size of the Dell cluster also depends on the workload type and job division among the servers, which will be further explained in evaluation sections.

Table 2: Comparing Edison micro servers to Dell Servers

Resource	Edison	Dell R620	To Replace a Dell
CPU	2×500MHz	6×2GHz	12 Edison servers
RAM	1GB	4×4GB	16 Edison servers
NIC	100Mbps	1Gbps	10 Edison servers
Estimated number of Edison servers: Max (12, 16, 10) = 16			

3.2 Measurement Methodology

Datacenter workloads stress CPU, memory, disk and network I/O, so in Section 4 we first evaluate the performance of individual Edison node and Dell server using various benchmarking tools, and then run several datacenter workloads on both clusters. For individual performance evaluation, we employ benchmarks that are widely used in industry and academia. Specifically, we use Dhrystone [52] to evaluate CPU, and Linux tools such as `dd` [5], `ioping` [6] to test disk throughput and latency, and `iperf3` [7], `ping` [8] to test network throughput and latency. We also use Sysbench [9] to evaluate system components such as CPU and memory.

Then we run two major datacenter workloads on a cluster of up to 35 Edison nodes and compare with results on up to 3 Dell servers. The two major types of workloads are web service applications (Section 5.1) and Hadoop MapReduce jobs (Section 5.2), each consisting of several subtests with different loads and optimizations. For web service applications, we validate the possibility that substituting 3 Dell servers with 35 Edison nodes functioning as web servers and cache servers can preserve the same level of throughput and comparable response delay at peak throughput. For MapReduce workloads, we show that the 35-node Edison cluster achieves higher work-done-per-joule than a 2-node Dell cluster under data-intensive workloads after equally optimized manual tuning on both clusters.

The power consumption of Edison micro servers and Dell servers is measured and presented in Table 3. On the Edison cluster, the power is measured by Mastech HY1803D DC power supply (7V) and individual node power draw is taken as the average of power consumption of 10 nodes. On the Dell cluster, the power consumption is measured by rack mount Power Distribution Unit (PDU) accessed with SNMP protocol.

Table 3: Power consumption of Edison and Dell servers.

Server state	Idle	Busy
1 Edison without Ethernet adaptor	0.36W	0.75W
1 Edison with Ethernet adaptor	1.40W	1.68W
Edison cluster of 35 nodes	49.0W	58.8W
1 Dell server	52W	109W
Dell cluster of 3 nodes	156W	327W

Surprisingly, each USB Ethernet adaptor draws around 1W of power when plugged in each Edison device, which is significantly more than the power consumed by the Edison device itself. However, we do include the power consumption of Ethernet adaptors, even if an integrated Ethernet component would only consume 0.1W [50]. Since including the Ethernet adaptors already over-estimates the total power consumption, we do not further include the power consumption of switches. Also, we do not take into account the power of cooling (fans, air conditioning) for both platforms in all comparisons, which is actually in favor of the Dell servers as they rely on a dedicated power-hungry computer room air conditioning (CRAC), whereas the Edison cluster only uses two low power fans.

The softwares used in the performance test are listed in Table 4. The first five benchmark software systems are widely adopted as standard tools to test computer system components [38], which makes it easier to compare with related work. In web service applications, we use Lighttpd as the web server software due to its high popularity and efficiency [4]. HAProxy is chosen since it is widely used in industry as a reliable, high performance TCP/HTTP load balancer [3]. MySQL, PHP, and Memcached are all renowned tools that have long been industry standard in web service deployment [28]. In MapReduce test, we use Hadoop Yarn [2] which is one of the most popular frameworks for processing of large data sets.

Table 4: Test softwares.

Software	Version on Edison	Version on Dell
Dhrystone [52]	2.1	2.1
dd [5]	8.13	8.4
ioping [6]	0.9.35	0.9.35
iperf3 [7]	3.1	3.1
Sysbench [9]	0.5	0.5
PHP [13]	5.4.41	5.3.3
Lighttpd [4]	1.4.31	1.4.35
Memcached [11]	1.0.8	0.31
Hadoop [2]	2.5.0	2.5.0
MySQL [12]	5.5.44	5.1.73
HAProxy [3]	1.5.8	1.5.2

4. INDIVIDUAL SERVER TEST

This section helps us understand the performance gap between an Edison micro server and a Dell server. Since the types of datacenter workloads used in our paper stress all system components, it is necessary to first evaluate the capacity of each component.

The results in the following sections basically show that the CPU gap between a Dell server and an Edison server is the largest (around 100 times), followed by memory bandwidth (16 times) and network bandwidth (10 times). The storage I/O gap is the smallest, which infer that Edison nodes may be more suitable for data-intensive but less suitable for pure computational workloads.

4.1 CPU test

We measure CPU performance in terms of million instructions per second (MIPS) using Dhrystone. We compile the source code with `gcc` using maximum optimization level `-O3` and specify memory clock speed for Edison (800 MHz) and Dell (1333 MHz) server. Then we start the benchmark on one core (one thread) with the number of runs being 100 million, and divide the result by 1757 to get the Dhrystone MIPS (DMIPS). The Dell server yields 11383 DMIPS, while Edison node has 632.3 DMIPS. We are surprised by this huge discrepancy (1 Edison core only has 5.6% performance of 1 Dell core). So we verify the CPU performance with Sysbench. Sysbench tests CPU by calculating all prime numbers smaller than 20000, using specified number of threads. The test completion time shown in Figure 2 and Figure 3 confirms the result from Dhrystone: when comparing 1 core 1 thread, a Dell server is 15-18 times faster than an Edison server, which means when combining all cores on one machine, a Dell server (hyper-threaded) is 90 to 108 times faster than a single Edison node! This is because the more sophisticated pipeline and cache structures in Xeon CPU achieve much higher instructions per cycle (IPC) [45].

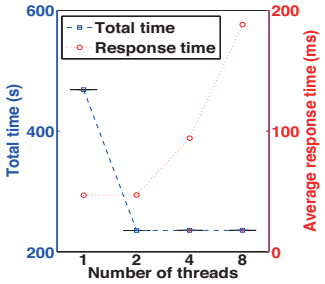

Figure 2: Edison CPU test.

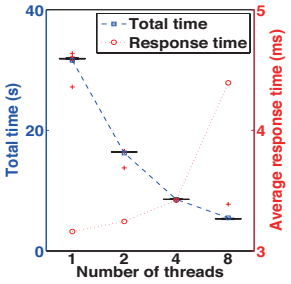

Figure 3: Dell CPU test.

4.2 Memory test

To measure memory bandwidth, we use Sysbench running memory transferring in specified block size with specified number of threads. We iterate over typical block sizes from 4KB to 1MB, each tested with different numbers of threads ranging from 1 to 16. We find that the transfer rates on both the Edison server and Dell server saturate from block size 256KB to 1MB. And the transfer rate stops increasing beyond 2 threads on Edison server and beyond 12 threads on Dell server. Results show that the Dell server has a maximum memory bandwidth of 36 GB/s while an Edison server only has 2.2 GB/s.

4.3 Storage test

The storage on the Dell server is a 1TB SAS hard disk with 15K RPM, while the Edison server uses a 8GB microSD card. Linux tools `dd` and `ioping` are used to test the storage I/O throughput and latency. We measure direct write with `oflag=dsync` option to make sure that every block is committed to disk before making another write request. We also measure buffered write by running without `oflag=dsync` option.

Table 5: Storage I/O test comparison.

	Edison	Dell
Write throughput	4.5 MB/s	24.0 MB/s
Buffered write throughput	9.3 MB/s	83.2 MB/s
Read throughput	19.5 MB/s	86.1 MB/s
Buffered read throughput	737 MB/s	3.1 GB/s
Write latency	18.0 ms	5.04 ms
Read latency	7.0 ms	0.829 ms

Results in Table 5 show that the speed of direct write on Dell is 5.3 times faster than on Edison, and buffered write is 8.9 times faster. For read operations, since OS tends to cache files in memory for faster access, we first test buffered read speed and then flush the cache to test direct read throughput. The buffered read on Dell server is 4.3 times faster than Edison and direct read throughput is

4.4 times faster on Dell. We use `ioping` to measure the read and write latency on both platforms. The read and write latencies on Edison are 8.4 and 3.6 times larger than on the Dell server, respectively.

4.4 Network test

We use `iperf` and `ping` to test network I/O throughput and latency, respectively. On Edison devices, the NIC has bandwidth of 100 Mbps while on Dell servers the bandwidth is 1 Gbps. So we would expect a 10-time gap between them. We transfer 1GB in total over TCP as well as UDP in the following three cases: Dell server to Dell server, Dell server to Edison server, and Edison server to Edison server. The speed of TCP and UDP for the first case is 942 Mbits/s and 948 Mbits/s, respectively. And for both the latter two cases, the speed of TCP and UDP is 93.9 Mbits/s and 94.8 Mbits/s, respectively, which is consistent with the Edison NIC capacity. The network latency between Dell servers is 0.24 ms on average, while pinging between Dell and Edison nodes yields an average of 0.8 ms. And the latency between Edison nodes themselves is around 1.3 ms. This is reasonable since the network bandwidth between Dell servers is 1 Gbps, and that all tested Dell servers are under the same rack. The bandwidth between Dell top-of-rack switch and Edison top-of-rack switch is also 1 Gbps, but between Edison nodes themselves, the bandwidth is limited by the NIC capacity.

5. CLUSTER PERFORMANCE TEST

Datacenter workloads can be categorized into online services and offline data analyses. To evaluate both categories, we choose web service application and MapReduce jobs as representatives respectively. For the web service application, we deploy a typical Linux + Lighttpd + MySQL + PHP (LLMP) stack, where Lighttpd runs as the web server enabled with FastCGI, and PHP as the language for web page that accesses the MySQL database. For MapReduce workloads running on Hadoop Yarn (v2.5.0), we run both data-intensive and computation-intensive workloads with tuned parameters. Scalability evaluations are carried out for all types of workloads by re-running the same tests in different cluster sizes.

5.1 Web service workload

5.1.1 Test setup

We deploy web applications on both the 35-node Edison cluster and 3-node Dell cluster. In each cluster, the servers are divided into web servers (running Lighttpd) and cache servers (running memcached). The web servers in each cluster are about twice as many as cache servers, as web servers would both fetch data from cache servers and assemble data to answer user requests, resulting in twice network traffic compared to cache servers. Web service workloads can exhaust network resources very fast, so in order to fully utilize server capacity, we turn on tcp port reuse option, expand ip local port range, and raise the limit for the number of file descriptors and simultaneous connections.

There are 8 client machines running httperf to generate sustained high-rate HTTP requests, which are evenly distributed to all web servers by another 8 machines running HAProxy [3] as a TCP-layer load-balancer. Using an excessive number of request generators and load-balancers ensures that the benchmarking tool itself never becomes a bottleneck of throughput. Among the parameters of httperf, concurrency means the number of new TCP connections per second, and we increase the concurrency to increase the load. Under each concurrency level, we fine tune the number of calls per connection to allow the httperf-reported concurrency to match the target concurrency level without client-side errors such as unavailable file descriptors. When the concurrency is too large, web servers will be overloaded and return error code 500 (server error). Experiment results with server-side errors are excluded from our evaluation. We apply the same httperf parameters to all clients and coordinate them via remote procedure call so that each test starts and ends almost at the same time on all clients. The test duration under each concurrency level is around 3 minutes, and results show that the run time difference between clients is less than 1s.

Linear scale-up is expected in this type of service since web servers seldom interact with each other. To confirm this behavior, we test and compare between 35-node Edison cluster and 3-node Dell cluster, 18-node Edison cluster and 2-node Dell cluster, and further scale down the Edison cluster to 1/4 and 1/8. The server count configuration and scale factor of both clusters are shown in Table 6.

Table 6: Cluster configuration and scale factor.

Cluster size	Full	1/2	1/4	1/8
# Edison web servers	24	12	6	3
# Edison cache servers	11	6	3	2
# Dell web servers	2	1	N/A	N/A
# Dell cache servers	1	1	N/A	N/A

Both the Edison cluster and the Dell cluster share the same MySQL database running on 2 other Dell R620 servers. This configuration helps us devote as many Edison servers as possible to middle tier services, without overwhelming the database. So in all our tests that involve accessing MySQL database, the requests are randomly directed to the 2 Dell database servers. The data stored in the 2 database servers are imported from the database dumps of the wikipedia website [14] and images crawled from Amazon, Newegg and Flickr websites (totaling 20GB). The database consists of 15 tables, 11 of which contains simple fields (`INT, VARCHAR, VARBINARY`) and the other 4 contain image blobs. The average image size is 30KB.

Upon receiving a request from the client, the web server first randomly chooses a database table and a row number, and then requests the cache server for the contents. If hit, the data is directly retrieved from the cache server without further consulting the database. Otherwise the web server will retrieve the data from one of the database servers. Each test consists of a warm-up stage and a test stage. During the warm up stage, the web servers first retrieve requested data from the database, and then feed the same data to cache servers. After the warm-up, cache misses in the test stage will not lead to insertion into cache servers to ensure consistent cache hit ratio during the test stage. We control the cache hit ratio by adjusting the warm-up time, and the hit ratio is calculated from memcached service statistics.

We test the web server performance under different workloads. The heaviness of the workload is controlled by the percentage of image query in the requests, which also af-

fects average reply sizes. To control the percentage of image requests, we assign different weights to image tables and non-image tables to control their probability to be selected. In our tests, the probabilities of accessing image tables are 0%, 6%, 10%, and 20%, and the average reply sizes are 1.5KB, 3.8KB, 5.8KB and 10KB, respectively.

Apart from the average response delay from `httperf` report, we also want to obtain the response delay distribution. So we set up 30 other Dell servers as clients running Python code to repeatedly generate HTTP requests to random web servers. Each time a request is sent using Python module `urllib2`, a separate thread logs the timestamps of sending the request and receiving the reply without interfering with the requesting thread. The distribution of response delay is obtained from the aggregated log results from the clients.

5.1.2 Web service test results

We first compare the performance of the two clusters under the lightest load (93% cache hit ratio and 0% image). The `httperf` benchmark results for throughput and delay under different cluster sizes are shown in Figure 4 and Figure 7. The numbers in the legend are the numbers of web servers under different cluster sizes, which can be found in Table 6. In terms of throughput, we have 4 observations: 1. Under high concurrency (number of connections per second), the maximum number of requests per second scales linearly with cluster size on both platforms. 2. The peak throughput is almost the same on both Edison cluster and Dell cluster, in both full cluster size and half cluster size. 3. Server error (5xx) occurs sooner on Edison cluster (beyond 1024 concurrency), while Dell cluster can handle more concurrency although the throughput drops (but error also occurs beyond 2048 concurrency). 4. When we further scale down on Edison cluster (to 6 and 3 web servers), the maximum throughput and maximum concurrency level both scale down linearly.

In terms of average response delay shown in Figure 7, we have 3 observations: 1. When the throughput (or concurrency level) is low, the delay on Edison cluster is around 5 times larger than on Dell cluster. 2. When the concurrency level reaches a threshold, the delay on Dell cluster increases significantly, exceeding the delay on Edison cluster. 3. The change of delay is much higher on Dell cluster (increased 200 times) than on Edison cluster (increased 10 times).

The two green lines in Figure 4 are the power consumption of the 3-node Dell cluster and the 35-node Edison cluster under each concurrency level. The power on Edison cluster steadily holds from 56W to 58W (but more than half is consumed by the Ethernet adaptors), while the power of the Dell cluster ranges from 170W to 200W. So at peak throughput, the Edison cluster achieves 3.5 times more energy efficiency than the Dell cluster.

The above results come from the lightest workload in that the proportion of image query is zero and the cache hit ratio is the highest (93%). The following 4 tests evaluate increased workloads on the 35-node Edison cluster and the 3-node Dell cluster. First we decrease the cache hit ratio down from 93% to 77% and 60% while keeping the image proportion to zero percent. Then we increase the image proportion from 0% to 6% and 10% while keeping the 93% cache hit ratio. The rest of the settings are the same as the previous test and the throughput and delay results are shown in Figure 5 and Figure 8, respectively.

The increase in image percentage from 0% to 10% enlarges the average reply size from 1.5KB to 5.8KB. Although the peak throughput at 512 concurrency does not change much, the throughput at 1024 concurrency drops significantly on both clusters. We see this drop as the result of network resource depletion and the inability to create enough threads. Also, the delays on both platforms nearly double even under low concurrency, and show similar trend when concurrency reaches maximum. But in general, the small increase of workload does not cause too much performance penalty in terms of maximum number of requests per second.

Before taking on the heaviest load, we need to determine the maximum image percentage that still ensures fair comparison. The Edison cluster physically locates in a different room from the clients that generate HTTP requests, and the total bandwidth between the clients and the Edison web servers is 1Gbps. However, the 2 Dell web servers and the clients are in the same server room interconnected by ToR switches so that their aggregated bandwidth is 2Gbps. Thus, when the network bandwidth is half utilized on the 24 Edison web servers (around 50Mbps on each node), the aggregated throughput already reaches the 1Gbps capacity. While for the Dell cluster, half utilized bandwidth (0.5Gbps each web server) amounts to just 1Gbps, still half of its total bandwidth. Thus, for fairness, we choose the proper image proportion so that it only utilizes half of individual Edison node's NIC capacity, which turns out to be 20%.

Thus, with 20% image proportion and 93% cache hit ratio, we re-run the `httperf` tests with different scale factors and show the throughput and delay results in Figure 6 and Figure 9. Under peak throughput, on average each Dell web server and each Edison web server shows 45% and 86% CPU usage, 50% and 25% memory usage, 60MB/s and 5MB/s network I/O, respectively. And each Dell cache server and each Edison cache server shows 1.6% and 9% CPU usage, 40% and 54% memory usage, 50MB/s and 4MB/s network I/O, respectively. Although the utilizations of major system components stay under 100%, both clusters are still fully utilized as the throughput is limited by the ability to create new TCP ports and new threads.

Figure 6 and Figure 9 show similar throughput and delay trends, but overall the number of requests per second is only 85% of that under lightest workload. The throughputs on both Edison cluster and Dell cluster drop significantly when the concurrency reaches 1024, and the half-sized Edison cluster can no longer handle that level of concurrency without generating errors. In addition, the overall throughput of Edison cluster changes from slightly better than Dell cluster (Figure 4) to a little worse than Dell cluster. But since the throughput scales linearly with cluster size, the performance gap could be compensated by adding a few more Edison nodes. But still, the Edison cluster achieves 3.5 times more energy efficiency than Dell cluster.

To see the delay distribution histogram under heaviest workload (20% image) and high request rate (around 6000), we run python program on 30 other Dell servers to make the same HTTP requests and log the response delay. The response delay distribution for Edison cluster and Dell cluster is shown in Figure 10 and Figure 11, respectively. We do see that under heavy workload, Edison cluster shows larger average delay. But interestingly, the delay distribution on Dell cluster spikes 1s, 3s, and 7s, which seem to follow a exponential back off pattern. Results from `httperf` report

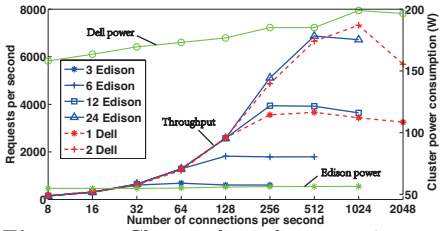

Figure 4: Cluster throughput, no image query.

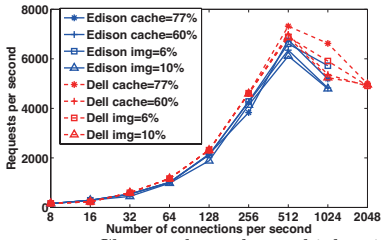

Figure 5: Cluster throughput, higher image percentage and lower cache hit ratio.

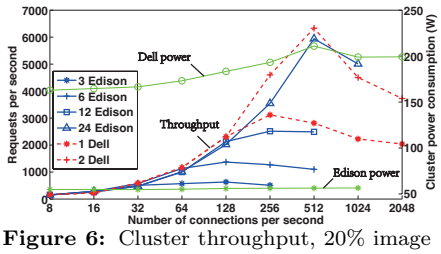

Figure 6: Cluster throughput, 20% image query.

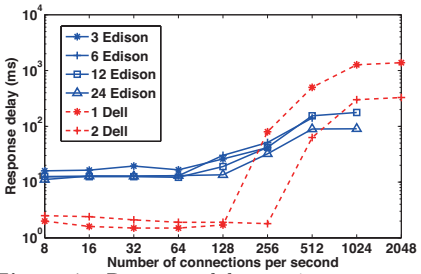

Figure 7: Response delay, no image query.

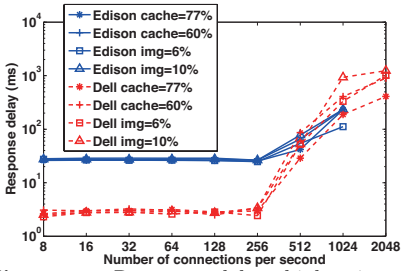

Figure 8: Response delay, higher image portion and lower cache hit ratio.

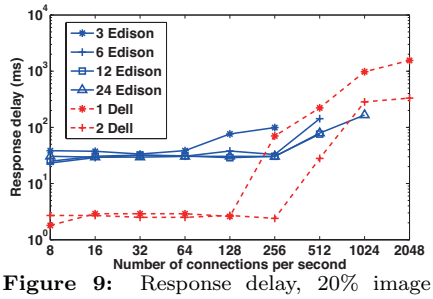

Figure 9: Response delay, 20% image query.

under similar request rate also show high deviation of connection time, some as large as 15 seconds. So we reason that the spikes of delay distribution on Dell cluster is a result of the re-connection delay of the python module `urllib2` when SYNC packets are dropped. Since the number of web servers in Edison cluster is much larger, the re-connection phenomenon is less severe. This would also highlight the advantage that Edison cluster has much more TCP port resources simply because of larger server count.

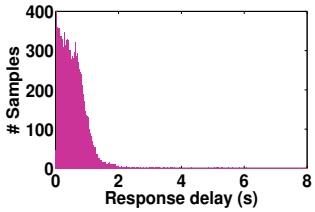

Figure 10: Delay distribution on Edison cluster

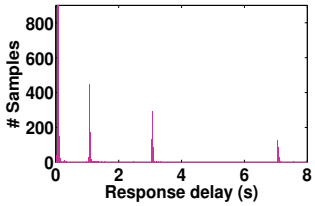

Figure 11: Delay distribution on Dell cluster

To break down the proportion of time spent on fetching from database and from cache, we log the timestamp of the start and end of getting results from MySQL database servers and cache servers. When calculating database query delay, the log entries of cache-hit requests (thus no database access) are ignored. The image query percentage is 20% and cache hit ratio is 93%, and the time delay is the average of aggregated results from all web servers. The delay decomposition is listed in Table 7. Note that the total delay excludes re-connection delay, since the timestamp is logged on web servers.

Table 7: Time delay decomposition in millisecond under different throughput, the first and second number in each tuple is measured on Edison and Dell cluster, respectively.

# Request/s	Database delay	Cache delay	Total
480	(5.44, 1.61)	(4.61, 0.37)	(9.18, 1.43)
960	(5.25, 1.56)	(9.37, 0.38)	(14.79, 1.60)
1920	(5.33, 1.56)	(76.7, 0.39)	(83.4, 1.73)
3840	(8.74, 1.60)	(105.1, 0.46)	(114.7, 1.70)
7680	(10.99, 1.98)	(212.0, 0.74)	(225.1, 2.93)

From Table 7 we observe that for Edison cluster, the time delay of retrieving from cache servers increases significantly faster than the delay of database query does, as the request rate rises from 480 to 7680. The reason is that the cache hit ratio stays high throughout this test so that queries seldom reach the database, but most requests would result in retrieving contents from cache servers, leading to larger cache delay. Another factor is that the network latency within the Edison cluster (1.3ms as mentioned in Section 4.4) is much larger than that within the Dell cluster (0.24ms). The fact that the Edison bandwidth is 10 times less than Dell bandwidth also contributes to the large discrepancy of cache retrieve delay between the two test subjects. From Table 7 we also see that under peak throughput, the total delay observed on the Dell web servers is still small, so we confirm that the large delay spikes in Figure 11 is due to the failure of the client's attempt to connect to web servers.

In summary, for web service workloads, the Edison cluster shows linear scalability and much higher energy efficiency than conventional brawny servers. This is largely due to the massively parallel nature of web service applications, and also because the network and OS resources for creating new connections are more abundant in larger sized micro server clusters.

5.2 MapReduce Workloads

In this section we explore the possibility of achieving higher energy efficiency on Edison cluster when running MapReduce jobs. Initially we deployed Hadoop (Yarn) on all the 35-node Edison cluster, but we soon discovered that an Edison node being a master (namenode for HDFS and resourcemanager for Yarn) would become the bottleneck of the whole system. Sometimes jobs even fail to complete when an Edison node is the master. This is because the namenode and resource-manager are resource-hungry instances, as they keep track of the global status, meta-data and resource allocation. Due to the limited amount of resources, a single Edison node cannot fulfill resource-intensive tasks. Thus we adopt a hybrid solution where the namenode and resourcemanager run on one Dell server (master) and the datanode

and node-manager run on the 35 Edison nodes (slaves). We also seek to understand the scalability of Hadoop on Edison cluster, since [30] points out that the overhead of coordination and data shipping causes "friction loss" that dilutes the benefits of a low-power cluster.

Thus, for the rest of our tests, the configuration on the Edison cluster is 1 Dell master plus 35 Edison slaves, while for the Dell cluster it is one Dell master plus other 2 Dell slaves. Therefore, in the Edison cluster there are 35 nodes running MapReduce jobs while in Dell cluster there are 2. To calculate energy consumption, we exclude the Dell master on both platforms, as the power consumed by the Dell master can be considered as a static offset due to its steady and low resource utilization (1% CPU and 53% memory).

It is confirmed in Section 4 that the aggregated CPU speed on 1 Dell server is around 100 times faster than on 1 Edison server. So the Edison cluster may not excel at computationally intensive workloads. Thus, we first focus on data-intensive but computationally light jobs, including wordcount and logcount [10]. We also optimize server utilization by combining small input files and analyze the performance improvement on both platforms.

First, we give a system capacity overview to help decide how to configure resources for MapReduce jobs. The Edison server has 960MB total physical memory. When it is in idle state, the memory usage is around 260MB, and when running HDFS datanode and Yarn node-manager (without any tasks), the memory usage is around 360MB in total. This means we have around 600MB memory available to run map and reduce tasks. On a Dell server, the total physical memory is 16GB, and when running only HDFS datanode and Yarn node-manager, there is about 4GB memory used. So on each Dell server we have around 12GB available memory to run MapReduce tasks. In Yarn resource configuration file, we allocate 100MB for Application Master on Edison cluster and 500MB on Dell cluster, and set the nameplate available memory resource to run map/reduce tasks to be 600MB on Edison server and 12GB on Dell server. The number of vcore is 2 on Edison cluster and 12 on Dell cluster, thus the container size could be 300MB for Edison and 1GB for Dell. For larger input datasets, the map task input size is often the same as HDFS block size, so we want it to be proportional to the container memory. Thus we set the block size to be 16MB on Edison cluster and 64MB on Dell cluster, unless otherwise specified. The HDFS replication number on Dell cluster is 1, otherwise all map tasks on Dell cluster would be data-local because there are only 2 datanodes. Then we set the replication number to be 2 on Edison cluster, so that on both clusters, the percentage of data-local map tasks is around 95% according to Hadoop log files.

For each MapReduce job on both clusters, we manually tune the parameters to reach the highest utilization and shortest run time. If the job's finish time does not change more than 5% after 3 consecutive fine tuning, its result and log file are adopted for comparison.

5.2.1 Wordcount

Wordcount is relatively computationally light, data-intensive and generating a considerable amount of map output records. There are 200 input files stored in HDFS, totaling 1GB. The original wordcount comes with no combiner class, nor does it merge input files to reduce the number of map containers. So, each container will only process one original input file, resulting in 200 map containers in total. Since each input split is small, we discover that running two or even more containers simultaneously on each virtual core (vcore) sometimes better utilizes CPU and memory resources, especially when the task is not computationally intensive and when there is abundant memory left to run more Java programs.

On Edison cluster, the map task memory is set to 150MB, so that on each server, the 2 vcores run 4 map containers at the same time. Average map output record size is around 10 bytes so we decide to set `io.sort.mb=70` and `io.sort.record.percent=0.6`. The option for Java program memory inside each map container is 140MB (`map.java.opts=-Xmx140m`). The map output is partitioned into 70 reduce tasks so that each Edison vcore gets one reduce container. The reduce task memory is set to 300MB, and the rest of parameters are set accordingly: `io.sort.mb=100`, `reduce.java.opts=-Xmx260m`.

Similarly, on Dell cluster, each server has 12 vcores and can run 24 map containers, so that the map task memory is set to 500MB, with `io.sort.mb=100`, `map.java.opts=-Xmx450m`. So the cluster of 2 Dell workers will run 48 map containers at the same time. The output records of mapping phase is partitioned into 24 reduce tasks so that each Dell vcore will handle one reduce container. The reduce memory task is set to 1GB, with `io.sort.mb=200`, `io.sort.factor=20`, `reduce.java.opts=-Xmx800m`.

We use python module `psutil` (which consumes less than 1% of CPU and memory) to log the resource utilization percentage on both platforms. Results are shown in Figure 12 and Figure 15 together with cluster power consumption and progress percentage of map/reduce phases. The rise of CPU usage (at 45s on Edison and 20s on Dell) is sooner than the rise of memory usage (at 90s on Edison and 35s on Dell) since the calculation of container allocation takes place before the reading of the input files from disk. The dive of memory (at 150s on Edison and 50s on Dell) usage indicates that the map output records are spilled to the disk. Due to limited space, the disk and network I/O usage is omitted.

Observation: 1. Wordcount has a relatively CPU-hungry map phase (splitting up lines into words), especially on Dell cluster, where CPU utilization stays at 100% persistently throughout the map phase. 2. The resource allocation time before the rise of CPU utilization is about 2.3 times longer on Edison cluster than on Dell cluster. 3. The reduce phase starts much sooner on Dell cluster (at 28% of execution time) than on Edison cluster (at 61% of execution time).

Because there are a large number of small input files for this job, the Dell cluster suffers more overhead of container allocation. The total time and total energy spent on Dell cluster are 213s and 40214J, respectively. While on Edison cluster the numbers are 310s and 17670J. Therefore, the Edison cluster spent 45.5% more time, but achieves 2.28 times higher energy efficiency.

Optimized wordcount (wordcount2): Then we seek to optimize the wordcount example (and name it wordcount2) to reduce the container allocation overhead and shuffle workload. We implemented the `CombineFileInputFormat` class that combines the 200 input files into larger files to reduce the number of map containers. The maximum split size is 15MB for Edison cluster and 44MB for Dell cluster so that in mapping phase, each vcore gets only one

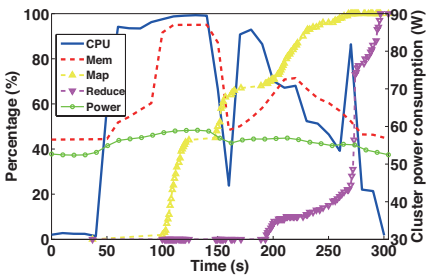

Figure 12: Wordcount on Edison cluster.

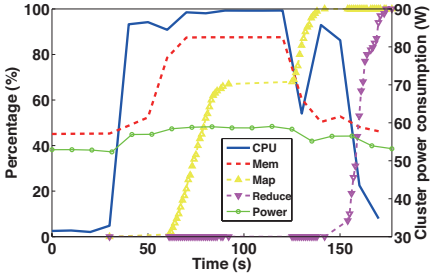

Figure 13: Wordcount2 on Edison cluster.

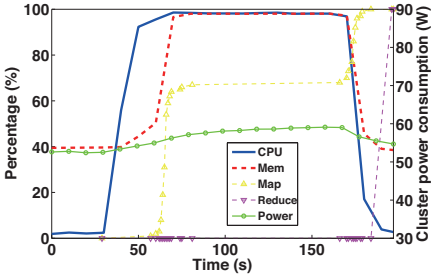

Figure 14: Estimate pi on Edison cluster.

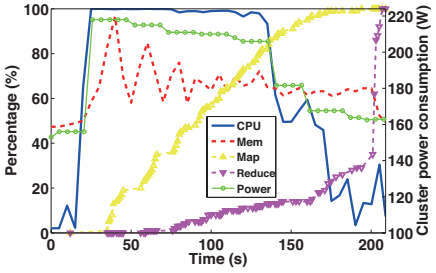

Figure 15: Wordcount on Dell cluster.

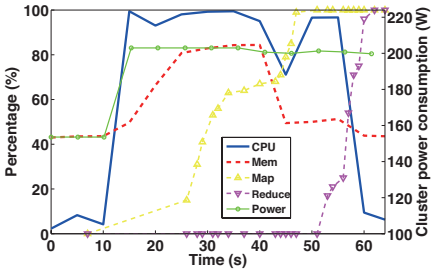

Figure 16: Wordcount2 on Dell cluster.

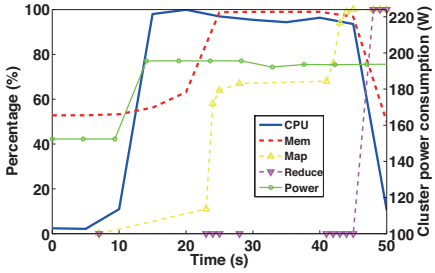

Figure 17: Estimate pi on Dell cluster.

container (much less overhead). In this way, the memory allocated for map task and reduce task is the same (300MB on Edison, 1GB on Dell). In addition, we set the `Combiner` class to be the same as `Reducer` class to reduce the size of spilled records and shuffling traffic. The results of wordcount2 are shown in Figure 13 and Figure 16.

Observation: First of all, both clusters achieve significant reduction in job completion time (41% on Edison and 69% on Dell). The CPU and memory utilization curves on both clusters are surprisingly similar, and the Dell cluster shows more improved memory usage. The map task progress bars on both clusters show a stagnation at around 70%. Combined with high network I/O during the stagnation (omitted), we reason that during that period, the partitioned map output records are being transferred to reducers.

The optimization brings greater benefits to the Dell cluster as it combines the map input into much fewer containers on Dell servers, which dwarfs the energy efficiency advantage of the Edison cluster. The energy spent on Edison cluster and Dell cluster is 10370J and 11695J, respectively, which means Edison cluster achieves only 11.3% more work-done-per-joule.

Discussion: Combining a large number of small input files into a large file for map task seems to dilute the benefits of parallelism and high energy efficiency on the Edison cluster. But first of all, the maximum combined file size is limited by the HDFS block size. So, unless the Dell cluster employs excessively large block sizes, which negatively impacts failure recovery, it is still inevitable to use a large number of mappers. Second, not all jobs can be optimized in this way. If the generation of those small input files is concurrently running along with the MapReduce jobs, it is not optimal to wait for all input files to be ready and combine them together. Some other scenarios may require that input files be processed individually and combining them could lose information about the source of the input. Therefore, unless carefully engineered by developers to allow such optimization, combining input files is not prevalent.

5.2.2 *Logcount*

Logcount [10] job is to extract from each log entry the date and debug level pair as key, and count the number of occurrences of each key. An illustration of the map output is `<'2016-02-01 INFO',1>` and the reduce task is to sum up the number of the same date and debug level. Compared to wordcount, the logcount map task is much lighter and produces fewer output records, and thus the reduce task takes fewer input records. The input files used in this test are 500 log files generated by Yarn and Hadoop, totaling 1GB. The reason to run this test is that we can observe the most "friction loss" or coordination overhead when the cluster size increases (shown in Section 5.3).

Observation: The original logcount program does not combine input files, but does set the `Combiner` class to be the same as `Reducer` class. As expected, the memory usage is only around 60% on both platforms, and the Dell cluster consumes at most 83% of its peak power. Because of the large number of map containers (500), the execution time on the 35-node Edison cluster is 279s, which is even closer to that on the Dell cluster (206s). Finally, in this test the Edison cluster achieves 2.57 times more work-done-per-joule.

Optimized logcount (logcount2): Similar to wordcount2, in this optimized logcount2 test, the small input files are combined so that each vcore only gets one map container. Again, the Dell cluster dramatically reduces the execution time to 59s, but the Edison cluster took 115s to finish the job. However, the Edison cluster still achieves 44.7% more work-done-per-joule.

5.2.3 *Estimate Pi*

The measurement results in Section 4.1 show that it is cumbersome for the Edison cluster to execute CPU-intensive tasks. In this section, we check whether the same claim holds for computationally intensive MapReduce jobs. The total number of samples is 10 billion and the number of map containers is 70 on Edison cluster and 24 on Dell cluster. The number of reducers is 1 by default for both clusters. The results are shown in Figure 14 and Figure 17.

Observation: We observe that both CPU and memory reach full utilization on both clusters. Given the large gap of computation capacity, Edison cluster spends 200s to finish the job while Dell cluster only needs 50s. In consequence, the energy consumption on the Edison cluster is 11445J, which is 2160J more than that on Dell cluster.Therefore, the Edison cluster actually yields 23.3% less energy efficiency.

5.2.4 Terasort

Terasort is composed of three parts: Teragen, Terasort, and Teravalidate. Neither the Edison cluster nor the Dell cluster has enough storage to store 1TB input dataset, so we scale down to 10GB of data. Teragen is a map-only job, which generates input data and stores them in HDFS for Terasort. Since the data generated by Teragen is split to the HDFS block size, to be fair we set the block size to be 64MB on both clusters, so that the total number of input files is 168. TeraSort was configured with 168 map tasks, 24 reduce tasks for Dell cluster, and 70 reduce tasks for Edison cluster. Each vcore is allocated 1 map container and the number of reducers is set to the total number of vcores. Teravalidate validates the results from Terasort, where the mapper number is equal to the reducer number of the Terasort, and the reducer number is one. We only compare the execution time and energy consumption of the Terasort stage.

Observation: From the resource utilization log we find that the Terasort stage is more memory-hungry than CPU-hungry. The average CPU and memory usage is around 60% and 95%, respectively, on both clusters. The Terasort stage on Edison cluster takes 750s, and on Dell cluster it takes 331s. During this period, the Edison cluster consumes 43440J, while Dell cluster consumes 64210J. So the Edison cluster again achieves 32% higher energy efficiency.

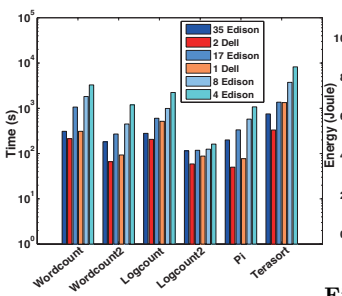

Figure 18: Job finish time.

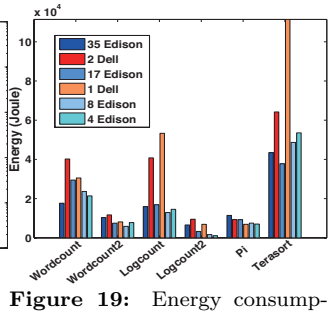

Figure 19: Energy consumption.

5.3 Scalability test

To investigate the execution time and energy consumption when the Edison cluster scales up, we run the same set of tests on both platforms in different cluster sizes. All the MapReduce jobs are tuned to best utilize the resources when the number of servers changes. For example, when running wordcount2 or logcount2 on half-scale Edison cluster, we increase the HDFS block size (to 32MB for example) in order to still allocate just one map container to each vcore. The job finish time and total energy consumption of both clusters are shown in Figure 18 and Figure 19, respectively. A summary is also shown in Table 8, where the bold numbers represent the least energy consumption case.

We can see that except estimating pi, the Edison cluster achieves more work-done-per-joule in all other MapReduce jobs. From wordcount and logcount tests, we observe that the huge parallelism helps the Edison cluster to outperform the Dell cluster more when there are higher container allocation overheads. Also, heavier jobs with more map containers are more likely to yield higher energy efficiency to run on larger cluster sizes (eg. Terasort). But the "friction loss" or coordination overhead begins to dilute the benefit of larger cluster size when container allocation overhead is reduced, so smaller cluster size actually yields higher efficiency (eg. estimating pi). In cases where the job (such as logcount2) is so light-weight that smaller clusters can also finish in short period of time, larger cluster actually consumes more energy.

For every MapReduce job, we calculate the mean speed-up when the cluster size doubles, and then derive the average speed-up of all jobs. The average speed-up is 1.90 on the Edison cluster and 2.07 on the Dell cluster. In some cases the linear speed-up is achieved, and we believe that it is because the parameter tuning happens to find the comfort spot for that particular input size and cluster size. In other cases like logcount2, larger cluster size actually only yields limited speed-up since the job is light-weight and the coordination overhead becomes dominant. But overall, the scalability of Edison cluster is satisfactory even if the speed-up is slightly less than on Dell cluster.

6. TCO ANALYSIS

To compare the total cost of ownership (TCO) of datacenters built upon Edison micro servers and Dell servers, we employ a simple model considering the equipment cost and electricity cost [38]. We assume that there exists a server utility lower bound and upper bound, and that the server is consuming peak power when active, while consuming minimum power when idling. Using the notations in Table 9, we derive the estimated TCO as:

$$C = C_s + C_e = C_s + T_s \cdot C_{eph} \cdot (U \cdot P_p + (1-U) \cdot P_i) \quad (1)$$

Table 9: TCO notations and values.

Notation	Description	Value
$C_{s,Edison}$	Cost of 1 Edison node	\$120
$C_{s,Dell}$	Cost of 1 Dell server	\$2500
C_{eph}	Cost of electricity	\$0.10/kWh
C_s	Total servers cost	To calculate
C_e	Total electricity cost	To calculate
T_s	Server lifetime	3 years
U_h	High Utilization rate	75%
U_l	Low Utilization rate	10%
$P_{p,Dell}$	Peak power of 1 Dell server	109W
$P_{p,Edison}$	Peak power of 1 Edison node	1.68W
$P_{i,Dell}$	Idle power of 1 Dell server	52W
$P_{i,Edison}$	Idle power of 1 Edison node	1.40W

The cost of one Edison server is composed of the device and mini breakout (\$68), plus a USB Ethernet adapter (\$15), a micro SD card and extension board (\$27) and amortized switch and cables cost (\$10), totaling $C_{s,Edison} =$ \$120. Therefore the cost of the 35-node Edison cluster is \$4200, which is lower than the cost of the Dell cluster consisting of either 2 or 3 servers. The electricity price is averaged across United States [22], which is $C_{eph} = \$0.10/kWh$. We expect 3-year lifetime of Dell servers [22], and assume that it is the same for Edison servers.

We calculate the TCO in two application scenarios: web service and big data execution. For web service scenario, we compare 35 Edison nodes to 3 Dell servers according to the configuration in Section 5.1.1. We consider a typical utilization lower bound $U_l = 10\%$ according to [37], and an upper bound $U_h = 75\%$ observed in Google datacenters [22]. For big data execution, we compare 35 Edison nodes to 2 Dell

Table 8: Execution time and energy consumption under different cluster size.

	Edison cluster				Dell cluster	
Cluster size	35	17	8	4	2	1
Wordcount	310s,**17670J**	1065s,29485J	1817s,23673J	3283s,21386J	213s,40214J	310s,**30552J**
Wordcount2	182s,10370J	270s,7475J	450s,**5862J**	1192s,7765J	66s,11695J	93s,**8124J**
Logcount	279s,15903J	601s,16860J	990s,**12898J**	2233s,14546J	206s,**40803J**	516s,53303J
Logcount2	115s,6555J	118s,3267J	125s,1629J	162s,**1055J**	59s,9486J	88s,**6905J**
Pi	200s,11445J	334s,9247J	577s,7517J	1076s,**7009J**	50s,9285J	77s,**6878J**
Terasort	750s,43440J	1364s,**37763J**	3736s,48675J	8220s,53547J	331s,**64210J**	1336s,111422J

servers according to Section 5.2. Since the Edison cluster spends 1.35× to 4× as much time to finish a job as the Dell cluster does, we assume the Edison cluster utilization to be constantly 100%, and for the Dell cluster $U_h = 74\%$ and $U_l = 25\%$. The 3-year TCO comparison is shown in Table 10. It is clear that building a cluster based on ultra-low power Edison servers can save the total cost up to 47%.

Table 10: TCO comparison.

Scenario	Dell cluster	Edison cluster
Web service, low utilization	$7948.7	$4329.5
Web service, high utilization	$8236.8	$4346.1
Big data, low utilization	$5348.2	$4352.4
Big data, high utilization	$5495.0	$4352.4

7. DISCUSSION

The high energy efficiency is the most salient and lucrative property offered by the sensor-class micro server cluster. Nevertheless, restricted by the limited per node capacity, this high energy efficiency does not apply to all workloads. Based on our evaluations, we summarize the advantages (+) and limitations (−) of the sensor-class micro server cluster:

+ Massively parallelizable applications such as web services can easily migrate to and comfortably run on micro servers, enjoying much higher (up to 3.5×) energy efficiency.

+ Data-intensive batch-processing workloads can achieve considerably higher (up to 2.6×) work-done-per-joule on micro servers.

+ The total cost of the micro server cluster is much lower due to much less sophisticated power and cooling infrastructure, as well as lower server purchase cost.

− The measured computational capability gap (≈ 100×) between micro servers and conventional servers surprisingly exceeds their nameplate CPU speed gap (12×) by around one order of magnitude. Therefore, micro servers cannot win the competition when serving computationally-intensive workloads.

− Concomitant with higher energy efficiency is prolonged execution time, which makes the micro server cluster less suitable for interactive and latency-sensitive applications.

− The limited resources in micro servers prevent them from acting as the manager of the data processing framework.

Given the limitations, micro server cluster alone cannot offer a holistic solution for cloud computing. However, the diversity in datacenter workloads exposes myriad chances for micro servers to participate and play a role. Therefore, we believe that a hybrid future datacenter design that orchestrates micro servers and conventional servers would achieve both high performance and low power consumption.

Lastly, during the 10-month deployment of the Edison cluster, we only encountered one breakout board failure (but the Edison SoC stacked on it is still functioning correctly). We believe that failures caused by less reliable peripherals can be reduced through improved system integration.

8. CONCLUSION

In this paper, we present an ultra-low power cluster built upon the sensor-class Edison micro servers, and identify the advantage of such low-power cluster in terms of work-done-per-joule under various datacenter workloads. We show that for web service applications and data-intensive MapReduce jobs, the Edison cluster always achieves higher energy efficiency compared to the conventional high-end cluster. We also show that the micro server cluster scales linearly in web service applications, and also achieves satisfactory scalability in big data execution. However, through our experiments we identify the limitations of the micro server cluster, and conclude that it is not suitable for computationally intensive and latency-sensitive cloud applications. Nevertheless, the diversity of datacenter workloads offers numerous opportunities for micro server clusters to actively take on the tasks that suit them most. Therefore, we envision a future hybrid datacenter design that can harness both the energy-efficiency of micro servers and the computational power of conventional servers.

9. REFERENCES

[1] Dell Cluster testbed. `http://greendatacenters.web.engr.illinois.edu/`.

[2] Hadoop web page. `https://hadoop.apache.org/`.

[3] HAProxy web page. `http://www.haproxy.org/`.

[4] Lighttpd web page. `https://www.lighttpd.net/`.

[5] Linux man page for dd. `http://www.unix.com/man-page/opensolaris/1m/dd/`.

[6] Linux man page for ioping. `http://manpages.ubuntu.com/manpages/saucy/man1/ioping.1.html`.

[7] Linux man page for iperf3. `https://code.google.com/p/iperf/wiki/ManPage`.

[8] Linux man page for ping. `http://linux.die.net/man/8/ping`.

[9] Linux man page for sysbench. `http://www.linuxcertif.com/man/1/sysbench/`.

[10] Logcount example code. `https://github.com/tsirisha/hadoop-mapreduce-examples`.

[11] Memcached web page. `http://memcached.org/`.

[12] MySQL web page. `https://www.mysql.com/`.

[13] PHP web page. `http://www.php.net/`.

[14] Wikimedia raw database downloads. `https://dumps.wikimedia.org/`, 2005.

[15] AMD to Accelerate the ARM Server Ecosystem with the First ARM-based CPU and Development Platform from a Server Processor Vendor. `http://webcitation.org/6PgFAdEFp`, 2014.

[16] Applied Micro, Canonical claim the first ARM 64-bit server production software deployment. `http://webcitation.org/6RLczwpch`, 2014.

[17] Intel Edison Module. http://www.intel.com/content/www/us/en/do-it-yourself/edison.html, 2014.

[18] World's First ARMv8 64-bit Server on a Chip Solution. https://www.apm.com/products/data-center/x-gene-family/x-gene/, 2015.

[19] D. Abdurachmanov, B. Bockelman, P. Elmer, G. Eulisse, R. Knight, and S. Muzaffar. Heterogeneous high throughput scientific computing with apm x-gene and intel xeon phi. In *Journal of Physics: Conference Series*, volume 608, page 012033. IOP Publishing, 2015.

[20] P. Abrahamsson, S. Helmer, N. Phaphoom, L. Nicolodi, N. Preda, L. Miori, M. Angriman, J. Rikkila, X. Wang, K. Hamily, et al. Affordable and energy-efficient cloud computing clusters: The bolzano raspberry pi cloud cluster experiment. In *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, volume 2, pages 170–175. IEEE, 2013.

[21] D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan. Fawn: A fast array of wimpy nodes. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 1–14. ACM, 2009.

[22] L. A. Barroso, J. Clidaras, and U. Hölzle. The datacenter as a computer: An introduction to the design of warehouse-scale machines. *Synthesis lectures on computer architecture*, 8(3):94–95, 2013.

[23] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, (12):33–37, 2007.

[24] R. Boutaba, Q. Zhang, and M. F. Zhani. Virtual machine migration in cloud computing environments: Benefits, challenges, and approaches. *Communication Infrastructures for Cloud Computing. H. Mouftah and B. Kantarci (Eds.). IGI-Global, USA*, pages 383–408, 2013.

[25] A. M. Caulfield, L. M. Grupp, and S. Swanson. Gordon: using flash memory to build fast, power-efficient clusters for data-intensive applications. *ACM Sigplan Notices*, 44(3):217–228, 2009.

[26] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *NSDI*, volume 8, pages 337–350, 2008.

[27] Y. Chen, S. Alspaugh, D. Borthakur, and R. Katz. Energy efficiency for large-scale mapreduce workloads with significant interactive analysis. In *Proceedings of the 7th ACM european conference on Computer Systems*, pages 43–56. ACM, 2012.

[28] R. Ibarra Roca and P. de la Iglesia Couto. Dynamic websites. 2007.

[29] V. Janapa Reddi, B. C. Lee, T. Chilimbi, and K. Vaid. Web search using mobile cores: quantifying and mitigating the price of efficiency. *ACM SIGARCH Computer Architecture News*, 38(3):314–325, 2010.

[30] W. Lang, S. Harizopoulos, J. M. Patel, M. A. Shah, and D. Tsirogiannis. Towards energy-efficient database cluster design. *Proceedings of the VLDB Endowment*, 5(11):1684–1695, 2012.

[31] W. Lang and J. M. Patel. Energy management for mapreduce clusters. *Proceedings of the VLDB Endowment*, 3(1-2):129–139, 2010.

[32] W. Lang, J. M. Patel, and J. F. Naughton. On energy management, load balancing and replication. *ACM SIGMOD Record*, 38(4):35–42, 2010.

[33] W. Lang, J. M. Patel, and S. Shankar. Wimpy node clusters: What about non-wimpy workloads? In *Proceedings of the Sixth International Workshop on Data Management on New Hardware*, pages 47–55. ACM, 2010.

[34] E. Le Sueur and G. Heiser. Dynamic voltage and frequency scaling: The laws of diminishing returns. In *Proceedings of the 2010 international conference on Power aware computing and systems*, pages 1–8. USENIX Association, 2010.

[35] J. Leverich and C. Kozyrakis. On the energy (in) efficiency of hadoop clusters. *ACM SIGOPS Operating Systems Review*, 44(1):61–65, 2010.

[36] P. Lieberman. White paper: Wake on lan technology, 2006.

[37] H. Liu. A measurement study of server utilization in public clouds. In *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, pages 435–442. IEEE, 2011.

[38] D. Loghin, B. M. Tudor, H. Zhang, B. C. Ooi, and Y. M. Teo. A performance study of big data on small nodes. *Proceedings of the VLDB Endowment*, 8(7):762–773, 2015.

[39] D. Meisner, B. T. Gold, and T. F. Wenisch. Powernap: eliminating server idle power. In *ACM Sigplan Notices*, volume 44, pages 205–216. ACM, 2009.

[40] R. Mueller, J. Teubner, and G. Alonso. Data processing on fpgas. *Proceedings of the VLDB Endowment*, 2(1):910–921, 2009.

[41] N. Rajovic, L. Vilanova, C. Villavieja, N. Puzovic, and A. Ramirez. The low power architecture approach towards exascale computing. *Journal of Computational Science*, 4(6):439–443, 2013.

[42] S. Rivoire, M. A. Shah, P. Ranganathan, and C. Kozyrakis. Joulesort: a balanced energy-efficiency benchmark. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 365–376. ACM, 2007.

[43] D. Schall and T. Härder. Energy and performance-can a wimpy-node cluster challenge a brawny server? *arXiv preprint arXiv:1407.0386*, 2014.

[44] N. Schot. Feasibility of raspberry pi 2 based micro data centers in big data applications. 2015.

[45] E. Sprangle and D. Carmean. Increasing processor performance by implementing deeper pipelines. In *Computer Architecture, 2002. Proceedings. 29th Annual International Symposium on*, pages 25–34. IEEE, 2002.

[46] A. S. Szalay, G. C. Bell, H. H. Huang, A. Terzis, and A. White. Low-power amdahl-balanced blades for data intensive computing. *ACM SIGOPS Operating Systems Review*, 44(1):71–75, 2010.

[47] N. Tolia, Z. Wang, M. Marwah, C. Bash, P. Ranganathan, and X. Zhu. Delivering energy proportionality with non energy-proportional systems-optimizing the ensemble. *HotPower*, 8:2–2, 2008.

[48] F. P. Tso, D. R. White, S. Jouet, J. Singer, and D. P. Pezaros. The glasgow raspberry pi cloud: A scale model for cloud computing infrastructures. In *Distributed Computing Systems Workshops (ICDCSW), 2013 IEEE 33rd International Conference on*, pages 108–112. IEEE, 2013.

[49] B. M. Tudor and Y. M. Teo. On understanding the energy consumption of arm-based multicore servers. In *ACM SIGMETRICS Performance Evaluation Review*, volume 41, pages 267–278. ACM, 2013.

[50] V. Vasudevan, D. Andersen, M. Kaminsky, L. Tan, J. Franklin, and I. Moraru. Energy-efficient cluster computing with fawn: Workloads and implications. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, pages 195–204. ACM, 2010.

[51] P. Velthuis. Small data center using raspberry pi 2 for video streaming. 2015.

[52] R. P. Weicker. Dhrystone: a synthetic systems programming benchmark. *Communications of the ACM*, 27(10):1013–1030, 1984.

[53] D. Wong and M. Annavaram. Knightshift: Scaling the energy proportionality wall through server-level heterogeneity. In *Microarchitecture (MICRO), 2012 45th Annual IEEE/ACM International Symposium on*, pages 119–130. IEEE, 2012.