Investigating a Raspberry Pi Cluster for Detecting Anomalies in the Smart Grid

Kasey Candelario†, Chris Booth†, Aaron St. Leger*, Suzanne J. Matthews*
Department of Electrical Engineering & Computer Science
United States Military Academy, West Point, NY 10996
*Corresponding Authors, Email: [aaron.stleger, suzanne.matthews]@usma.edu
†Email: [kasey.r.candelario.mil, christopher.j.booth2.mil]@mail.mil

***Abstract*—Smart Grid Technology is an integral part of ensuring the security of the power grid. To provide situational awareness to grid operators, a smart grid system must be able to detect alarm events (such as sudden voltage fluctuations or drops in current) in close to real-time. In this paper, we propose the use of a low energy Raspberry Pi cluster to detect anomalies in the Smart Grid. We build a prototype cluster and test our approach on a real data set of approximately 1 million measurements derived from 8 PMUS from a 1000:1 scale emulation of a working power grid. Our results show that a cluster of 12 Raspberry Pis is capable of achieving better performance than a more power-hungry multicore server at lower cost and a significant reduction in power consumption.**

Keywords: Smart Grid, Raspberry Pi, Anomaly Detection, Multicore

I. Introduction

Smart grid technology is being developed and implemented in power grids to improve reliability, efficiency and resiliency of grid operation. Numerous online and offline smart grid applications are currently being researched and developed [1]. Online applications can improve the response of grid operators, through manual intervention or automated control actions, by providing a more detailed view of the power grid in real-time or near real-time. Examples of online applications include Wide area Monitoring Systems (WAMS) and Wide Area Control Systems (WACS) that leverage real-time synchrophasor data from Phasor Measurement Units (PMU).

Synchrophasors are time synchronized phasor measurements of voltages and currents in a power grid. Specifically, each measurement consists of a magnitude, phase, and a time stamp based on a GPS synchronized timing source. WAMS and WACS can use these measurements for rapid state measurement, system monitoring, and control. Measurements rates up to 60Hz are supported by current PMU technology. As deployment of PMUs increases the number of measurements and quantity of data increases substantially from traditional SCADA measurements which have far fewer measurements and a slower update rate (on the order of seconds to minutes depending on the sensor). A key research area in smart grids is how to handle this increase in data effectively and provide meaningful information to grid operators and controllers. Prior work has leveraged parallel computing as a solution to rapidly process raw PMU data in near real-time [2]. The primary smart grid application of this work was a model free methodology of processing raw PMU data for anomaly detection. Anomalous conditions were specifically defined apriori for this process. Raw PMU data was scanned and any detected anamolies flagged and reported in near real-time. This information can provide rapid feedback to grid operators and provide information to other downstream WAMS and WACS.

The work presented in [2] investigated two types of anomalies within raw PMU data: constraint and temporal anomalies. Constraint anomaly detection monitors PMU data at specific time slices and evaluates if voltage magnitude, current magnitude, and frequency is within acceptable predetermined limits. Temporal anomaly detection monitors PMU data across a specified time window and detects if any unacceptable temporal variation occurs. For example, a rapid decrease in voltage that could be indicative an event within the power grid. Experimental results showed that the proposed anomaly detection approach could detect both temporal and constrain anomalies from a dataset consisting of 18 million measurements in under three seconds on an eight core workstation. This is representative of near real-time performance from 4,500 PMUs installed in a power grid.

In this paper, we investigate the applicability of a Raspberry Pi microcluster for anomaly detection and compare to a multicore workstation. A "microcluster" is a small, compact cluster of single board computers. For simplicity, we only look at the performance of the cluster on constraint anomalies. We compare the performance of our cluster to a quad-core multicore server, using a real dataset consisting of approximately 1 measurements and 8 PMUs, derived from a 1000:1 scale emulation of a working power grid. PMU data is aggregated and time aligned by a phasor data concentrator and then sent to the cluster for anomaly detection. Our results show that 8 nodes of our Raspberry Pi microcluster can detect anomalies at the same rate as our multicore server. This lends credence to the idea that a microcluster can be used for anomaly detection in the power grid.

The rest of the paper is organized as follows. Section II provides an overview of related work regarding parallel computing for smart grid applications. Section III outlines our methods in this work to include experimental data, anomaly detection algorithm, and implementation. This is followed by results and conclusion in sections IV and V respectively.

978-1-5386-2534-7/17/$31.00 ©2017 IEEE

TABLE I
RASPBERRY PI 2 CLUSTER COST BREAKDOWN.

Component	Cost p/Unit	Total
Raspberry Pi 2 nodes (28)	$35.00	$980.00
5V (20A) power supply (2)	$39.95	$79.90
8GB MicroSD card (28)	$5.98	$167.44
Case Components (28)	$5.00	$140.00
Ethernet Cables (28)	$2.00	$56.00
Router (1)	$20.00	$20.00
16-port 10/100 switch (2)	$48.47	$96.94
Powered USB hub (1)	$22.93	$22.93
Cooling fan (2)	$15.00	$30.00
1 TB USB SSD (1)	$50.00	$50.00
TOTAL		**$1,643.21**

Fig. 1. Picture of Raspberry Pi 2 Cluster

II. RELATED WORK

Several researchers [2]–[5] have explored using parallel computing for applications related to the smart grid. For anomaly detection, researchers have typically used Hadoop [6], a cluster computing framework that leverages MapReduce. While Hadoop clusters have been successfully used by the Tennesee Valley Authority (TVA) for historical analysis of anomalous events [7], it has been less successful for anomaly detection at a smaller scale [8]–[10]. This raises the question if a cluster is needed to do anomaly detection in a localized setting.

In 2017, Matthews and St. Leger presented two multicore MapReduce algorithms to detect anomalies on a real dataset of 18 million measurements [2]. The authors used the Phoenix++ MapReduce framework to implement their algorithms on multicore architectures, and found that they could detect anomalies in under three seconds on a quad-core system. Their work showed that a large cluster was not needed to achieve anomaly detection at near real-time speed.

In a separate paper, Matthews *et al.* studied the efficacy of single board computer (SBC) clusters for compute-intensive cyber applications [11]. A single board computer is a computer which is printed entirely on a single circuit board; a Beowulf cluster can be created from a set of these devices. In their paper, Matthews *et al.* experimentally demonstrate how two inexpensive SBC clusters can outperform a high-end laptop for a password cracking application. They argue that SBCs clusters may be a power-efficient alternative for many cyber applications, and their use for the rapid detection of alarm events in ICS/SCADA systems [11].

Our paper presents a prototype that demonstrates the utility of a Raspberry Pi microcluster for anomaly detection in the Smart Grid. To the best of our knowledge, we are the first group of researchers to use a Raspberry Pi cluster for this application. To test our approach, we compare the performance of our algorithm on a 28-node Raspberry Pi cluster and compare it to its performance on a multicore server.

In addition of building on the work of other researchers [2], [11], our work explores the notion of "energy-proportional computing" [12], which argues that CPU utilization on systems should be close to 100% to maximize their energy efficiency. Raspberry Pi cores are relatively weak compared to those found in typical multicore workstations. However, we predict that many Raspberry Pis will outperform a multicore workstation at the task of anomaly detection while achieving greater power efficiency and lower cost.

III. METHODS

Table I gives an overview of the cluster we used for analysis. Our cluster is composed of 28 Raspberry Pi 2 nodes and costs roughly $1,650.00 to build. Using a Kill A Watt meter [13], we determined that the cluster consumes approximately 120 Watts at peak CPU loading. The cases components were 3-D printed using open-sourced Raspberry Pi case materials [14]. A picture of our assembled cluster is shown in Figure 1. Each node can communicate with others using the Message Passing Interface (MPI) [15]. A 1 TB solid state hard drive is mounted for the network file system. Each Raspberry Pi has a 900 MHz ARM v7 processor and 1 GB of RAM. Each Pi consumes roughly 4.25 Watts of power peak, and runs Raspbian Linux.

We compare our performance to a Dell 9010 lab workstation running Red Hat Enterprise Linux with 32 GB of availalbe RAM. The CPU consists of a quad-core Intel i7-3770 processor @ 3.4 GHz. Due to hyper-threading, the workstation has 8 virtual cores. The system cost approximately $1,200.00 dollars when it was purchased in 2014, and consumes roughly 80 Watts of power.

While the multicore workstation is a beefy (yet lower cost) system, our research goal was to determine if the 28-node Raspberry Pi cluster (or a *subset* thereof) could offer the same anomaly detection performance achievable on our multicore lab workstation at a fraction of the total cost.

A. Experimental Data & Implementation

For this paper, we concentrated solely on *constraint* anomalies. That is, anomalies that occur outside a given window of allowed variation on the power grid [2]. Experimental data and our definitions of constraint windows was gathered from a 1000 : 1 scale emulated three-phase distribution power grid located at West Point [16]. The system is implemented with 7 buses, 9 transmission lines, and contains real measurement devices, such as Phasor Measurement Units (PMUs), and real voltage/current tranducers.

978-1-5386-2534-7/17/$31.00 ©2017 IEEE

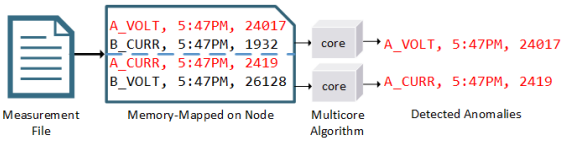

Fig. 2. Overview of Anomaly Detection (1 Node)

The rated line voltage and line currents of the system are 46 V and 2-Amperes respectively; constraint windows were derived directly from these specifications. For example, during normal operating procedures, voltage should not vary more than $\pm 5\%$ of the expected value (26.3 kV). Likewise, current should never exceed 2000 mA.

We use the emulated power grid to gather approximately ≈ 1 million real measurements (318 MB) of PMU data. While the MapReduce approach described by Matthews and St. Leger leveraged a dataset of 18 million measurements, we were limited to a smaller dataset for our experiments due to memory restrictions on the Raspberry Pi.

Initially, we attempted to port the Phoenix++ MapReduce algorithm described by Matthews and St. Leger to the Raspberry Pi. However, we discovered a fundamental incompatibility between Phoenix++ MapReduce framework and the ARM architecture of the Raspberry Pi, forcing us to abandon Phoenix++. Since re-implementing MapReduce was beyond the scope of this project, we opted to implement a simpler algorithm using POSIX threads (Pthreads) on our Pi cluster. We also measure the performance of our Pthreads implementation on the multicore server and compare the results.

B. Algorithm

Figure 2 gives an overview of our anomaly detection approach, on a single node system with 2 cores. The multicore workstation follows a very similar workflow to that depicted in Figure 2, except our workstation has 4 physical cores. For our Pi cluster, each node receives its own measurement file. For simplicity, we do not depict the "constraint" file which each node takes as input. The constraint file lists , for each measurement type, the window of allowed variation for that measurement. We note that while in Figure 2 individual measurements are are shown as a text descriptor (e.g. `B_CURR`), in reality they are each a hash. Measurements that should be identified as anomalous are highlighted in red.

Each node begins by loading the constraint file into a hashtable which it stores in shared memory. Each node then reads its assigned measurement file and memory maps it to make it available to all the cores on the node. We then spawn threads equivalent to the number of cores on the system, and assign each core an equally-sized segment of the memory-mapped file, according to thread id. In Figure 2 our system has two cores, so we spawn two threads, with each thread assigned two measurements.

Next, each thread linearly scan its assigned segment for anomalies, using the shared hashtable to determine the allowed constraint window for the particular measurement. For example, the first core compares the `A_VOLT` measurement with the associate range of values for voltages, and finds that it falls outside the allowable range. Thus, this measurement is outputted as an anomaly to the user. When it compares the `B_CURR` measurement to the allowable window however, it finds that it is OK. Thus, the measurement is silently ignored. In parallel, the second core compares the `A_CURR` measurement to the allowable window for current, and discovers that it is too high. Therefore, this measurement too is emitted to the user as anomalous.

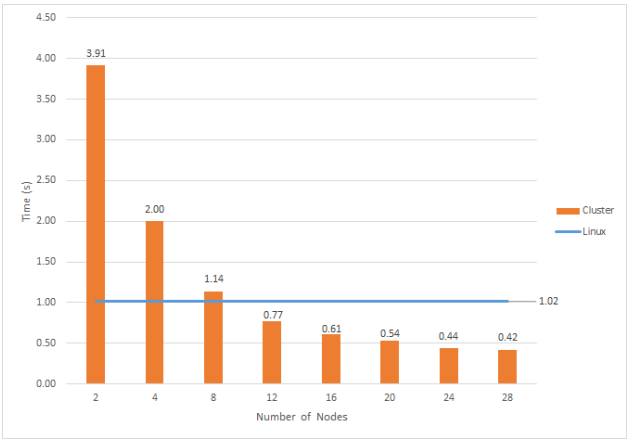

Fig. 3. Cluster Performance Compared to Multicore

The algorithm follows the classic single program, multiple data (SPMD) model for threaded applications. Let M be the number of measurements, n be the number of nodes, and c be the number of cores on each node. For the multicore workstation, the amortized run-time of our algorithm is simply $O(\frac{M}{c})$. For a cluster, the measurement file may first need to be split into n subfiles each containing $\frac{M}{n}$ measurements, which requires approximately $O(M)$ time. Thus, the time to process each file for anomalies should take approximately $O(\frac{M}{nc})$ time.

IV. Results

We first ran our Pthreads implementation on the multicore workstation and a single rapsberry pi. The algorithm was run three times, and the average time (in seconds) was recorded. One a single core of the workstation, the algorithm took 2.03 seconds to process our dataset. On four cores, the algorithm took 1.02 seconds. In contrast, the Pthreads implementation took 25.48 seconds on a single core of the Raspberry Pi, and 8.56 seconds on four cores of the Raspberry Pi. These results are completely unsurprising, as our multicore lab workstation is much more powerful than a single Raspberry Pi.

Figure 3 show the results of experimentation with Raspberry Pi cluster. The horizontal line represents the multicore workstation's time on four cores, 1.02 seconds. The x-axis of the plot depicts the number of nodes in the cluster we ran the contraint algorithm on, while the y-axis depicts the average time (in seconds) it took to process our dataset. Each bar represents the average of three runs when using all four cores of the Raspberry Pis.

978-1-5386-2534-7/17/$31.00 ©2017 IEEE

When using two Raspberry Pis, the run-time of our constraint algorithm drops to 3.91 seconds on average. Increase the number of Raspberry Pis to four reduces the run-time further to 2 seconds. At this point, the Pi cluster performs on par with the workstation running the algorithm on 1 core. Increasing the number of cores to 8 further reduces the run-time to 1.14 seconds, which is close to the time it took the workstation to process our dataset on 4 cores. From 12 cores onward, the Raspberry Pi cluster outperforms the workstation, requiring 0.77 seconds on 12 cores and 0.42 seconds on 28 cores.

Most striking about our results is the cluster's ability to outperform the multicore workstation using only 12 nodes of the Raspberry Pi cluster. We estimate that an equivalent 12-node cluster would cost approximately $\$750.00$ and consume approximately 50 watts of power, roughly the same as a laptop computer. An 8-node cluster would be even more efficient, costing approximately $\$550.00$, and consuming approximately 34 watts of power as compared to the multicore workstation which consumes 80 watts for similar performance.

V. Conclusions

In this paper, we discuss the use of a Raspberry Pi cluster for detecting anomalies in the smart grid. Our goal was to determine if a Raspberry Pi cluster could outperform a multicore server for the task of detecting contraint anomalies. We implement a novel multi-threaded algorithm and use a real dataset of 1 million measurements from a $1000:1$ scale emulated power grid. We test our approach on a multicore workstation and subsets of a 28-node Raspberry Pi cluster. Our results suggest that 12 Raspberry Pi nodes are able to outperform our multicore workstation, and 8 nodes offer on-par performance.

Our results are meaningful for several reasons. First, we show that a Raspberry Pi cluster can be used for the application of anomaly detection in a smart grid. Second, we show that 12 nodes of our Raspberry Pi cluster is capable of analyzing data at the same speeds as our quad-core workstation. The latter is especially significant as the 12-node Pi cluster consumes roughly the same power as a laptop computer. These results suggest that Raspberry Pi clusters may be a more power-efficient and cost-effective way to conduct anomaly detection on local power grids.

There are many avenues for future research. First, we did not measure the latency to transfer data between our power grid and our Raspberry Pi nodes, or our workstation. It is very possible that the amount of time it takes to transmit 318 MB of data over the network may overshadow any benefit received in fast performance. In the future, we will study this latency in greater depth. Secondly, we also plan on implementing an algorithm to detect temporal anomalies [2]. Lastly, we plan to explore the Raspberry Pi 3 for future cluster experiments, as the updated version is likely to offer better speedups.

Acknowledgments

Funding for this project was provided by the U.S. Army Armament Research, Development and Engineering Center (ARDEC). This work was completed as part of a year-long undergraduate capstone project that Kasey Candelario and Chris Booth worked together on. Special thanks to Mr. Robert McKay of the Engineer Support Staff for his assistance in assembling the Raspberry Pi cluster, and Mr. John Dwyer and Mr. Jim Beck of the Computer Support Group for providing cost and power estimates for the Dell workstation used in this study. The opinions in this work are solely of the authors and do not necessarily reflect those of the U.S. Military Academy, the U.S. Army, or the Department of Defense.

References

[1] NERC, "Real-time application of synchrophasors for improving realiability," *Tech. Rep. 1*, 2010.

[2] S. J. Matthews and A. St. Leger, "Leveraging mapreduce and synchrophasors for real-time anomaly detection in the smart grid," *IEEE Transactions on Emerging Topics in Computing*, vol. PP, no. 99, pp. 1–1, 2017.

[3] J. C. H. Peng, A. Meads, and N. K. C. Nair, "Parallel computing for smart power oscillation monitoring using synchrophasor measurements," in *TENCON 2010 - 2010 IEEE Region 10 Conference*, Nov 2010, pp. 657–662.

[4] M. Giuntoli, P. Pelacchi, and D. Poli, "Parallel computing of sequential montecarlo techniques for reliable operation of smart grids," in *EUROCON 2015 - International Conference on Computer as a Tool (EUROCON), IEEE*, Sept 2015, pp. 1–6.

[5] S. Jin, Z. Huang, Y. Chen, D. Chavarría-Miranda, J. Feo, and P. C. Wong, "A novel application of parallel betweenness centrality to power grid contingency analysis," in *2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*. IEEE, 2010, pp. 1–7.

[6] T. White, *Hadoop: The definitive guide*. " O'Reilly Media, Inc.", 2012.

[7] P. Trachian, "Machine learning and windowed subsecond event detection on pmu data via hadoop and the openpdc," in *IEEE PES General Meeting*. IEEE, 2010, pp. 1–5.

[8] M. Edwards, A. Rambani, Y. Zhu, and M. Musavi, "Design of hadoop-based framework for analytics of large synchrophasor datasets," *Procedia Computer Science*, vol. 12, pp. 254–258, 2012.

[9] F. Bach, H. K. Çakmak, H. Maass, and U. Kuehnapfel, "Power grid time series data analysis with pig on a hadoop cluster compared to multi core systems," in *2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. IEEE, 2013, pp. 208–212.

[10] M. Khan, P. M. Ashton, M. Li, G. A. Taylor, I. Pisica, and J. Liu, "Parallel detrended fluctuation analysis for fast event detection on massive pmu data," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 360–368, 2015.

[11] S. J. Matthews, R. W. Blaine, and A. F. Brantly, "Evaluating single board computer clusters for cyber operations," in *2016 International Conference on Cyber Conflict (CyCon U.S.)*, Oct 2016, pp. 1–8.

[12] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, 2007.

[13] P3 International. (2005) Kill a watt - electricity usage monitor [p3]. [Online]. Available: http://www.p3international.com/products/p4400.html

[14] S. J. Matthews and W. Blackmon. (2015) Raspberry pi case and cluster files. [Online]. Available: https://www.thingiverse.com/thing:892959

[15] W. Gropp, E. Lusk, N. Doss, and A. Skjellum, "A high-performance, portable implementation of the mpi message passing interface standard," *Parallel computing*, vol. 22, no. 6, pp. 789–828, 1996.

[16] A. St. Leger, J. Spruce, T. Banwell, and M. Collins, "Smart grid testbed for wide-area monitoring and control systems," in *2016 IEEE/PES Transmission and Distribution Conference and Exposition (T D)*, May 2016, pp. 1–5.