Application Power Signature Analysis

Chung-Hsing Hsu*, Jacob Combs†, Jolie Nazor†, Fabian Santiago†, Rachelle Thysell†, Suzanne Rivoire†
and Stephen W. Poole*

*Computer Science and Mathematics Division
Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
Email: {hsuc,spoole}@ornl.gov
†Department of Computer Science
Sonoma State University, Rohnert Park, CA 94928, USA
Email: {combsj,nazor,santiagf,thysell}@seawolf.sonoma.edu
rivoire@sonoma.edu

***Abstract*—The high-performance computing (HPC) community has been greatly concerned about energy efficiency. To address this concern, it is essential to understand and characterize the electrical loads of HPC applications. In this work, we study whether HPC applications can be distinguished by their power-consumption patterns using quantitative measures in an automatic manner. Using a collection of 88 power traces from 4 different systems, we find that basic statistical measures do a surprisingly good job of summarizing applications' distinctive power behavior. Moreover, this study opens up a new area of research in power-aware HPC that has a multitude of potential applications.**

***Keywords*-high performance computing; power signature; clustering**

I. Introduction

In high-performance computing (HPC), energy efficiency has become a major concern [1]. Many of the current leaders on the TOP500 list [2] consume multiple megawatts, costing agencies like the U.S. Department of Energy one million dollars per megawatt per year. In the commercial sector, Google's data centers consume 300 megawatts, according to the New York Times (Sep. 22, 2012). Higher energy efficiency can help reduce the operating costs and carbon dioxide emissions of these facilities.

Understanding the power consumption behavior of workloads and facilities is a critical step to improving energy efficiency. If workload-specific power consumption patterns do exist, they would be valuable in a variety of contexts. For example, pattern-driven job and resource management schemes (such as [3], [4]) can leverage application power signatures to enhance their quality of service. The idea of systematically identifying such patterns is new in HPC. Most previous studies use visual inspection for a handful of applications [5]–[10], but this work treats the problem more thoroughly and efficiently.

Our central goal is to determine whether applications exhibit distinctive power consumption behavior, which we refer to as *application power signatures*, and to what extent this behavior is independent of input data, runtime configuration, and hardware platform. In particular, we want to find out whether HPC applications can be distinguished by their power traces using quantitative measures in an automatic manner. Automation is important because it makes the integration with systems software easier. This paper presents an initial effort at this task.

The contributions of the paper are listed as follows.

- *Formulating the problem as a clustering problem.* Given a collection of power traces, we seek to identify a feature-based clustering algorithm that can cleanly distinguish all (or most of) the power traces of one application from the power traces of other applications.
- *Establishing a baseline for the solution space.* We explored 63 different feature spaces using hierarchical clustering. Based on the 88 power traces we collected from 4 different machines, we identified the best feature space as a base solution to the clustering problem.
- *Showing that the quality of the baseline is surprisingly good.* We found that summarizing traces using basic statistical measures yields clusters that are grouped first by application, then by system, and then by application class.
- *Opening up a new area of research.* We have only explored a small portion of the solution space. In addition, how to define the goodness of a clustering result quantitatively is still a challenge. More broadly, are there more clever ways to formulate the problem?

The rest of the paper is organized as follows. In Section II, we describe the study of power signatures in HPC and in other fields. Then we present our analysis approach in Section III and an instantiation of this approach in Section IV. In Section V, we discuss the preliminary results. We conclude the paper and outline future work in Section VI.

II. Related Work

As previously mentioned, the study of power signatures is new in HPC. The term *power signature* has been mentioned

in the literature (e.g., [5]–[10]), but most references only use the term qualitatively. The only exception, to the best of our knowledge, is a use of mathematical models to capture the power signatures of server power-management algorithms based on published benchmark results [10].

In other fields, power signatures have been studied for some time. One such field is cryptography research. In 1999, Kocher et al. [11] presented an analysis of side-channel attacks on embedded devices. The attacks are based on collecting and analyzing the power traces of a device. In the simplest form, an attacker visually interprets the traces to identify the encryption algorithm used by the device. A more advanced attack can uncover secret keys through statistical analysis. Thus, understanding the power signature of an encryption algorithm is a key focus in the field.

From another perspective, knowing the power signatures of trusted software can help maintain system integrity. For example, malware can be detected by comparing the generated power signature with those in a database of trusted signatures [12], [13]. Interestingly, the growing popularity of energy-proportional computing in server design [14] has also caused some concerns about privacy and security. Server energy-proportional computing, in which power consumption scales closely with workload, can expose properties of the workload based on its power consumption. This is argued to be both harmful to privacy and beneficial for malware detection [15].

The smart-grid research community is also interested in the study of power signatures. For example, they want to know how to determine the operating schedule of home appliances in a residential home from measurements made at the electric utility service entry [16]. They call it *non-intrusive load monitoring* (NILM). While NILM can significantly reduce sensor installation and maintenance costs, it requires the knowledge of power signatures for the electrical loads. As a result, there has been a substantial number of power-signature studies done in this research area; for example, [17]–[19] to name a few.

What we have discussed so far are mostly the uses of power signatures. Given the variety of uses across several fields, it can be expected that many methods have been developed to study power signatures.

In terms of general methodology, our analysis method is related to clustering of time-series data [20], [21]. In general, there are three types of clustering methods: those working directly with the raw data, those working indirectly with features extracted from the raw data, and those working indirectly with models built from the raw data [20]. Our analysis method falls under the second approach: feature-based clustering [22].

In feature-based clustering, the majority of feature extraction methods are generic in nature, but the extracted features are usually context-dependent. Thus, a set of features that works well in one context might not be relevant to another [20]. In the context of HPC power signatures, we found out that basic summary statistics perform surprisingly well. It is unknown whether the more complex methods developed for other fields would remain effective in our context.

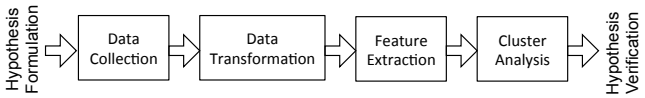

Figure 1. The workflow of our analysis approach.

III. The Approach

Our approach to analysis starts with hypothesis formulation, followed by data collection, feature extraction, and cluster analysis, and finally ends with hypothesis verification. Figure 1 depicts the workflow.

A. Hypothesis Formulation

Our hypothesis is that different runs of the same application will exhibit similar power consumption patterns, even when the input data changes. Ideally, these patterns should allow these runs to be distinguished from runs of a different application. However, this latter property may not hold in reality; we may only be able to describe the power signature of a class of similarly-behaving applications. For the former property, we suspect that it is more likely to hold for runs taken from the same machine than across different (brands of) machines.

B. Data Collection

We measure and record the power consumption of various application runs as *power traces*. A power trace is a set of timestamped power measurements (in watts) collected during an application run with a particular data input on a specific machine.

As a basis for our analysis, we have amassed a collection of power traces: measurements of power consumption sampled once per second during an application's execution. We have collected power traces for 10 benchmarks on 4 different machines. For most applications, we have collected traces using multiple configurations and/or input data sets on each machine.

C. Data Transformation

Data transformation generally refers to converting power traces into a different representation to facilitate feature extraction. This often occurs in geometric approaches for feature extraction. For statistical approaches, this type of data transformation is not needed. However, some statistical approaches may perform data processing such as sampling, noise reduction or data normalization.

D. Feature Extraction

Feature extraction casts power traces into a feature space as points in the space. There are many feature spaces that we can choose. For example, the feature space can be based on the time domain or the frequency domain [15]. Frequency-domain approaches are easier to carry out in general. They can easily handle power traces of different lengths and can ignore alignment issues that could arise in time-domain approaches. However, they may have less predictive power due to the loss of the temporal information.

E. Cluster Analysis

The goal of cluster analysis is to group points in a feature space into clusters. This can be done manually or automatically. Direct visualization of a feature space is a manual method. However, it becomes challenging when the space is high-dimensional. There also exist many automatic clustering techniques. The popular approaches include density-based clustering, centroid-based clustering, and hierarchical clustering [23]. These techniques are generally parameterized, and tuning the parameter values can be nontrivial. In our experiments, we focus on hierarchical clustering. Hierarchical clustering requires minimal tuning and yields results that provide more insight than a simple listing of a fixed number of clusters.

F. Hypothesis Verification

We effectively formulate application power signature analysis as a clustering problem. To verify the hypothesis, we check if the runs of an application are in the same cluster or not. In addition, the description of the cluster can be used to describe the power signature of the application.

IV. Experiments

In this section we detail an instantiation of the approach. We will describe a database of power traces and the choice of feature spaces and clustering technique.

A. Power Traces

We have collected 88 power traces from 10 benchmarks and 4 machines. We ran each benchmark with at least two different data inputs. The benchmarks are:

1) Nsort (nsort) [24].
2) STREAM (stream) [25].
3) Prime95 (p95) [26].
4) Linpack (linpack-cblas) [27].
5) TILT (sb-tilt) [28].
6) FFT1D (sb-fft1d) [28].
7) FFT2D (sb-fft2d) [28].
8) DGEMM (sb-dgemm) [28].
9) GUPS (sb-gups) [28].
10) Graph500 (graph500) [29].

Note that benchmark names with the sb prefix indicate that they come from SystemBurn [28]. SystemBurn is a software tool engineered to allow a system user to methodically create a maximal system load on large scale systems for the purposes of testing and validation.

Table I
Four platforms used in data collection.

Machine	RR	OC	LC	RF
CPU	AMD Athlon X2 4800+ 2-core	Intel Core i5-750 4-core	Intel Core i5-750 4-core	Intel Core i7-3770 4-core
GHz	1.0-2.5	1.2-2.67	1.2-2.67	1.6-3.4
RAM	4GB	8GB	8GB	8GB
GPU	NVIDIA GeForce 9800 GT	NVIDIA GeForce GTX 285	NVIDIA GeForce GTX 650 Ti	NVIDIA GeForce GTX 670
Power	115-196W	120-226W	73-211W	74-312W
Traces	14	14	28	32

Table I shows the configuration of the 4 machines. These machines all run the Linux operating system. The `ondemand` frequency scaling governor [30] was turned on when we collected power traces. For these workloads, this essentially means that the operating system will peg the CPU at its lowest frequency when idle and its highest frequency when busy. Intel Turbo Boost was turned off during data collection.

To monitor power consumption, we used a Watts Up? PRO power meter. This power meter reports instantaneous power consumption every second. We think that this sampling rate is sufficient to capture the steady-state behavior of long-running HPC applications. The accuracy of power measurement is 1.5% plus 3 counts of the reported value.

B. Features

The features of interest are basic summary statistics. These features describe the histogram derived from a power trace. We consider 6 features, listed as follows.

1) Maximum (Max).
2) Minimum (Min).
3) Mean.
4) Median (Med).
5) Range.
6) Standard deviation (Std).

The first four features describe the location of the histogram, and the last two describe its scale.

C. Feature Spaces

We explore all possible combinations of the 6 features. Each combination forms a feature space. For example, Mean is a feature space, and MeanStd is another. There are a total of 63 possible combinations. In fact, we consider more than 63 feature spaces. As we will describe below, data normalization increases the count by a factor of 10.

D. Data Normalization

Features may have different value ranges. For comparability, feature values are often normalized before cluster

analysis. We consider two types of data normalization. Given a matrix of feature vectors, where each row corresponds to a power trace and each column a feature, one type of normalization is to standardize for each row, and the other is for each column.

Specifically, the row-based normalization divides each feature vector by the idle power of the machine in order to reduce the influence of the machine. The column-based normalization scales feature values so that each feature has approximately equal weight in distance calculations during cluster analysis. The following lists all these data normalization techniques.

1) Normalize row r_i by machine M:

$$r_i = r_i/\text{idle}(M).$$

2) Normalize column c_i to have Mean=0 and Std=1:

$$c_i = (c_i - \text{Mean}(c_i))/\text{Std}(c_i).$$

3) Normalize column c_i to have Mean=0:

$$c_i = c_i - \text{Mean}(c_i).$$

4) Normalize column c_i by root-mean-square (RMS):

$$c_i = c_i/\text{RMS}(c_i).$$

5) Normalize column c_i to the range [0,1]:

$$c_i = (c_i - \text{Min}(c_i))/\text{Range}(c_i).$$

E. Clustering Technique

In this work we only consider hierarchical clustering. Although there are other techniques available, hierarchical clustering has a nice visual representation of the result. The clustering result can be represented as a tree-structured graph called a *dendrogram*. The dendrogram provides more insight than simply showing the final clusters. It shows *how* each cluster is formed. This additional information is extremely helpful when we have to visually evaluate the goodness of a clustering result.

We use the hierarchical clustering algorithm **hclust** [31]. The idea of **hclust** is to start with single-element clusters and iteratively join the two closest clusters together until the entire dataset is grouped in one large cluster. The algorithm requires a definition of the distance between two clusters. In this study we use the so-called complete-linkage distance, i.e., the largest distance between pairs of points from each cluster. This choice of distance tends to result in compact clusters of points that are all relatively close to each other.

All clustering approaches requires a notion of distance between a pair of points in the feature space. In this paper we focus on the most common distance: the Euclidean distance.

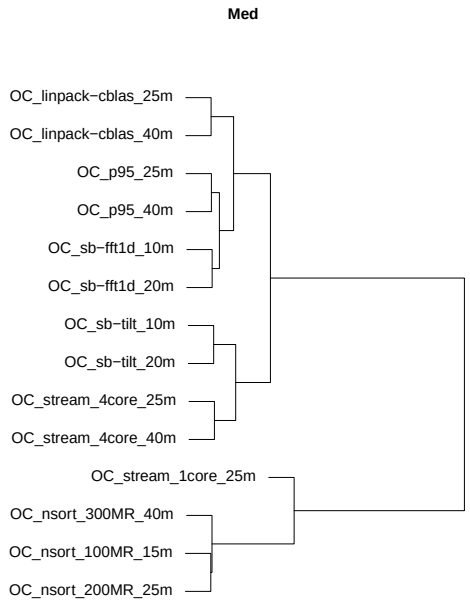

Figure 2. The results of applying **hclust** to feature space Med with power traces collected from machine OC.

F. Implementation Notes

All data normalization, feature extraction, and cluster analysis are performed in R, a free software environment for statistical computing and graphics. We use the most recent version 3.0.2. Hypothesis verification is done manually, by visually checking the clustering results.

V. Preliminary Results

We begin our evaluation of feature spaces with the 32 traces collected on machines OC and RR. Restricting our initial exploration to this subset serves two purposes. First, since we are evaluating clusterings by inspection, using a smaller dataset makes the results more tractable. Second, it allows us to check the generality of our results by applying them to the newer machines, LC and RF.

Using **hclust**, we compare the clustering results from all 63 feature spaces, beginning with the one-dimensional spaces and then adding more features. Then, we discuss the impact of data normalization. Overall, the main finding is that application-level power consumption patterns can be distinguished based on a simple feature vector. Specifically, the combination of a trace's minimum and median power provides reasonably good clustering results.

A. One-Dimensional Feature Space

On machine OC, feature space Med is considered the best. It has correctly grouped the runs of an application in the same cluster, as shown in Figure 2. Feature space Max is considered the second best.

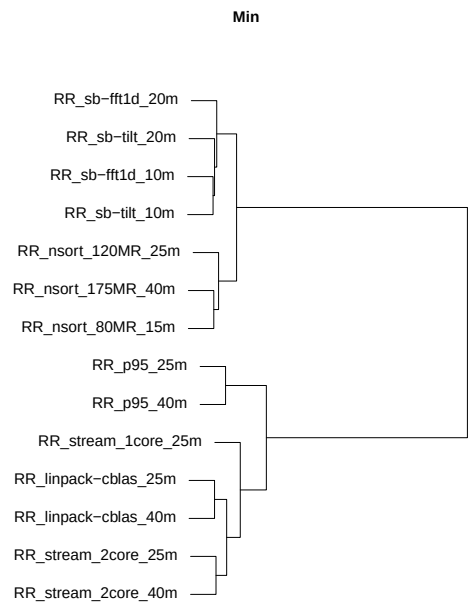

Figure 3. The results of applying **hclust** to feature space Min with power traces collected from machine RR.

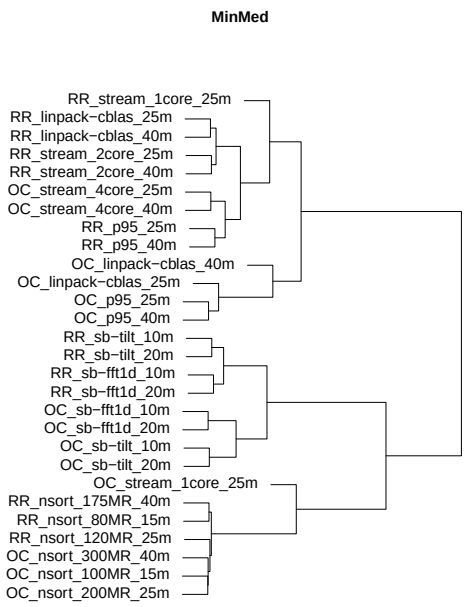

Figure 4. The results of applying **hclust** to feature space MinMed with power traces collected from machines RR and OC.

On machine RR, the best are feature spaces Max and Min. However, the quality of their clustering results is not as good as Med on OC. Figure 3 shows the result from feature space Min, in which the runs of benchmarks fft1d and tilt are not correctly separated.

The results suggest that a single feature is insufficient for distinguishing application-level power-consumption patterns. Higher-dimensional feature spaces are needed. Furthermore, features describing the histogram location are essential for getting a good clustering result. Med, Max, and Min are all such features.

B. Two-Dimensional Feature Space

For two-dimensional feature space, MinMed does the best for machine RR. The best feature spaces for machine OC are MaxStd, MaxMed, MedStd, and MeanStd. However, MinMed is near-optimal, too. On the other hand, the four best feature spaces for OC do poorly for RR. These results suggest the consistent effectiveness of feature space MinMed. Most likely MinMed will do well across machines, and it does. Figure 4 shows the clustering result of the power traces from the two machines.

The figure shows that the runs of an application *on the same machine* are grouped together. We also see three application classes. One is the class of linpack-cblas, stream with all CPU cores, and p95. Another one is the class of tilt and fft1d. The third one is the class of nsort.

To verify whether the application classes are valid or not, we compare the histograms of all the power traces visually. We observe that linpack-cblas, stream, p95, and nsort exhibit similar, unimodal patterns. The difference is in their location. The histograms of linpack-cblas and p95 are located close to each other. The histograms of nsort is located further to the left (i.e., smaller wattage). Depending on the number of CPU cores used, stream is either close to p95 or to nsort.

We also observe that benchmarks fft1d and tilt exhibit a similar bimodal pattern. In addition, the histograms look alike if they are from the same application.

Note that the runs of stream with 1 CPU core are not in the same cluster as stream with all cores. This suggests that, for application power signature analysis, the runs of an application on a machine with two different configurations can be as dissimilar as runs on two different machines.

Figure 5 shows the histogram of a sb-fft1d run on OC. The histogram is generated with the bin width of one watt. We can see that the values of median and maximum are similar, all around 202 watts. In contrast, the value of mean is about 187 watts.

It is not always the case that median is close to maximum. Figure 6 shows the histogram of a p95 run on OC. In this figure we see that median is actually close to mean and distant from maximum. Both figures suggest that feature spaces that can pinpoint the peaks of a histogram are generally more effective. This explains why MinMed does well in our case.

In statistics, there is a measure called *mode* that pinpoints the peaks of a histogram. A weakness of the use of mode is that its value is highly dependent on the bin width of the histogram. Our exploration shows that mode is the worst for the power traces of OC.

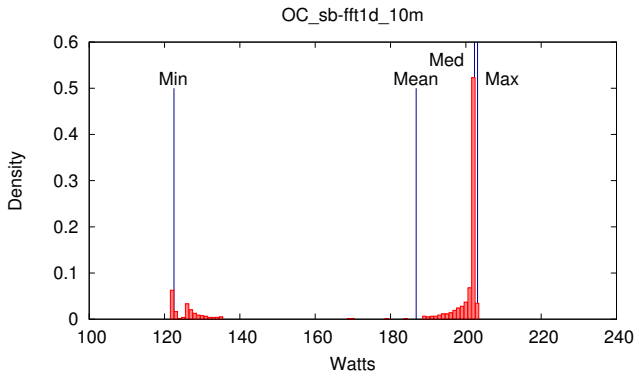

Figure 5. The location of Min, Mean, Med, and Max.

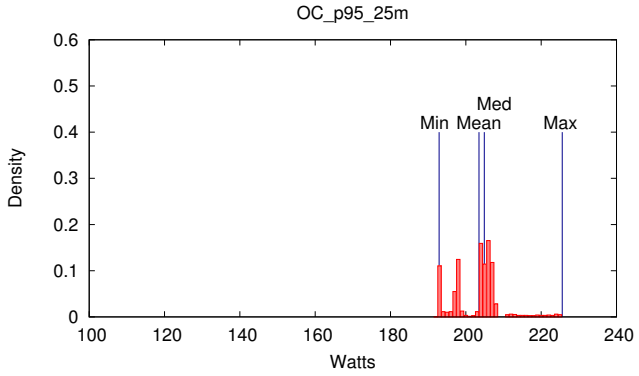

Figure 6. The location of Min, Mean, Med, and Max.

Note also that both Min and Med are describing the histogram location. Combining a location feature with a scale feature turns out to be not as effective as combining two location features. It remains to be seen whether this observation holds in general.

C. Higher-Dimensional Feature Space

For higher-dimensional feature spaces, we found out that none performs as well as MinMed. We also observe a converging trend in higher-dimensional feature spaces. Specifically, four- and higher-dimensional feature spaces all generate sub-optimal results. This seems to suggest the case of over-fitting. In other words, more features do not necessarily lead to a better result.

D. Data Normalization

Exploring all possible combinations of the data normalization techniques, we found out that column-based normalization has little effect. In contrast, normalization by machine has a detrimental effect. This suggests that data normalization is not needed.

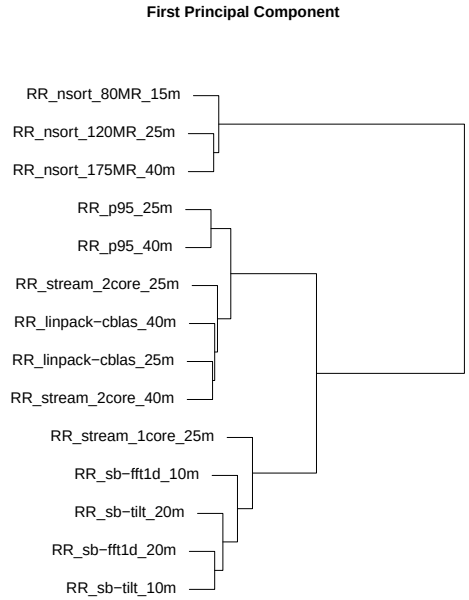

Figure 7. The results of applying **hclust** to the feature space based on the first principal component with power traces collected from machine RR.

E. Dimensionality Reduction

So far we have explored feature spaces constructed by a subset of features. Each feature in the subset spans one dimension of the feature space. We can also combine multiple features into a new feature. This approach reduces the dimensionality of the feature space. The combination makes sense when the features are strongly correlated.

We found out that Max, Med, and Mean are strongly correlated. Features Range and Std are also strongly correlated. Feature Min is not strongly correlated with any other feature. These correlations are expected.

Through principal component analysis, we found out that the first principal component accounts for 99% of variance. In this component, Max, Med, and Mean receive the highest weight, followed by Min. Features Range and Std receive low weights, indicating that they are less important. Unfortunately, clustering power traces based on the first principal component does not lead to the desired result. Figure 7 shows the clustering result.

F. Further Validation

A fundamental weakness of the approach is that we try to identify a clustering algorithm that would make our hypothesis valid. We do not validate the hypothesis directly. In other words, we are doing a modeling work. This leads to the question of the generality of the derived model.

To check if feature space MinMed works well on other machines, we analyzed the power traces collected from two new machines LC and RF. To strengthen the validation, new

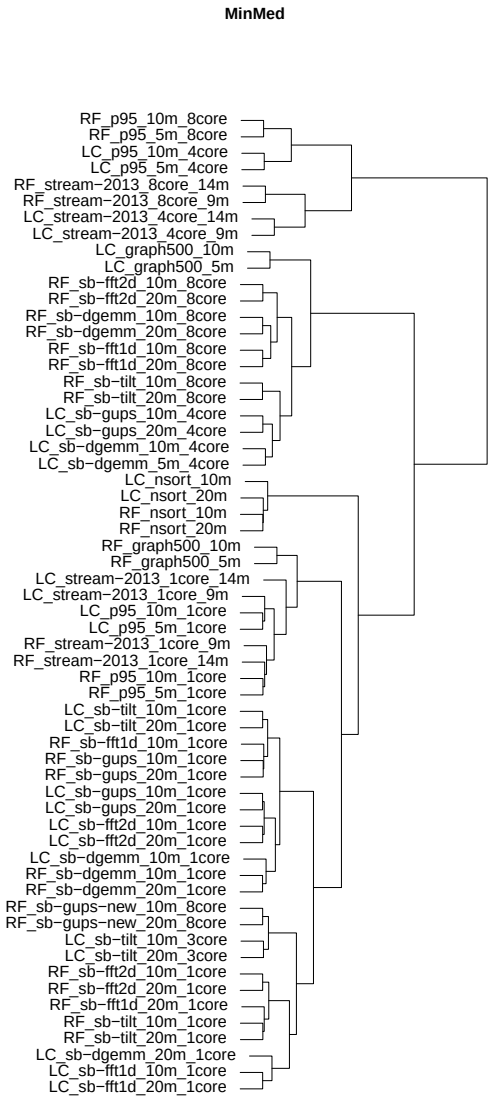

Figure 8. The results of applying **hclust** to feature space MinMed with power traces collected from machines RF and LC.

benchmarks were added. Figure 8 shows the resulting clustering, which demonstrates that MinMed does a reasonably good job.

G. Cross-Platform Signatures

With respect to cross-platform signatures, MinMed becomes less effective when considering power traces from multiple machines. On the other hand, some application classes remain the same across machines. For example, stream and p95 are always in the same class for the 4 machines we tested. In other words, the histogram shapes of some applications are platform-independent. This is good news, as the power-management policies tailored for these applications are likely to be more portable across platforms.

VI. Conclusions and Future Work

We have presented a study of application-level power signatures, formulated as a clustering problem. Using hierarchical clustering, we studied 63 feature spaces for a collection of 88 power traces from 10 applications on 4 different systems. The experimental results show that simple statistical summaries of each trace, particularly the combination of minimum and median, yield clusters that are grouped first by application, then by system, and then by application class. We discussed why this combination of features works the best, which is related to the histogram shapes of the power traces.

We plan to expand the research in the following directions.

- We want to collect power traces from more workloads, especially those considered as more complex such as SPEC CPU benchmarks. We are also interested in workloads that also stress accelerators such as the GPU.
- We would like to explore other summary statistics, distance functions, clustering methods, and data normalization techniques. We are particularly interested in statistical measures for time series data.
- Time-domain approaches are yet to be explored. The combination of frequency-domain and time-domain approaches is also possible.
- We will need to develop automated methods to evaluate the goodness of a clustering result or the similarity of two different clustering results.
- We desire to understand external factors that can affect clustering, such as the accuracy and the sampling rate of power measurements.

Acknowledgment

This work is supported by the United States Department of Defense and used resources of the Extreme Scale Systems Center located at the Oak Ridge National Laboratory. The submitted manuscript has been authored by a contractor of the U.S. Government under Contract No. DE-AC05-00OR22725. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

Matthew Hardwick and Lowell Olson contributed to the collection of traces, which was partially supported by an SSU Undergraduate Research Grant.

References

[1] S. Hemmert, "Green HPC: From nice to necessity," *Computing in Science and Engineering*, vol. 12, no. 6, pp. 8–10, November/December 2010.

[2] TOP500 Supercomputing Sites, http://www.top500.org.

[3] Z. Gong and X. Gu, "PAC: Pattern-driven application consolidation for efficient cloud computing," in *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, Aug. 2010.

[4] R. Zhang, R. Routray, D. Eyers, D. Chambliss, P. Sarkar, D. Willcocks, and P. Pietzuch, "IO Tetris: Deep storage consideration for the cloud via fine-grained workload analysis," in *International Conference on Cloud Computing*, Jul. 2011.

[5] S. Kamil, J. Shalf, and E. Strohmaier, "Power efficiency in high performance computing," in *Workshop on High-Performance, Power-Aware Computing*, Apr. 2008.

[6] S. Song, R. Ge, X. Feng, and K. Cameron, "Energy profiling and analysis of the hpc challenge benchmarks," *The International Journal of High Performance Computing Applications*, vol. 23, no. 3, pp. 265–276, Aug. 2009.

[7] J. L. III, K. Pedretti, S. Kelly, J. Vandyke, K. Ferreira, C. Vaughan, and M. Swan, "Topics on measuring real power usage on high performance computing platforms," in *IEEE International Conference on Cluster Computing*, Sep. 2009.

[8] B. Subramaniam, W. Saunders, T. Scogland, and W. Feng, "Trends in energy-efficient computing: A perspective from the Green500," in *International Green Computing Conference*, Jun. 2013.

[9] K. Kasichayanula, D. Terpstra, P. Luszczek, S. Tomov, S. Moore, and G. Peterson, "Power aware computing on GPUs," in *Symposium on Application Accelerators in High-Performance Computing*, Jul. 2012.

[10] C. Hsu and S. Poole, "Power signature analysis of the SPECpower_ssj2008 benchmark," in *International Symposium on Performance Analysis of Systems and Software*, Apr. 2011.

[11] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *International Cryptology Conference*, Aug. 1999.

[12] H. Kim, J. Smith, and K. Shin, "Detecting energy-greedy anomalies and mobile malware variants," in *International Conference on Mobile Systems, Applications, and Services*, Jun. 2008.

[13] C. Gonzalez and J. Reed, "Power fingerprinting in SDR and CR integrity assessment," in *Military Communication Conference*, Oct. 2009.

[14] L. Barroso and U. Hölzle, "The case for energy-proportional computing," *IEEE Computer*, vol. 40, no. 12, pp. 33–37, Dec. 2007.

[15] S. Clark, B. Ransford, and K. Fu, "Potentia est Scientia: Security and privacy implications of energy-proportional computing," in *USENIX Workshop on Hot Topics in Security*, Aug. 2012.

[16] M. Zeifman and K. Roth, "Nonintrusive appliance load monitoring: Review and outlook," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 1, pp. 76–84, Feb. 2011.

[17] C. Laughman, K. Lee, R. Cox, S. Shaw, S. Leeb, L. Norford, and P. Armstrong, "Power signature analysis," *IEEE Power and Energy Magazine*, vol. 1, no. 2, pp. 56–63, March/April 2003.

[18] J. Liang, S. Ng, G. Kendall, and J. Cheng, "Load signature study–part I: Basic concept, structure, and methodology," *IEEE Transactions on Power Delivery*, vol. 25, no. 2, pp. 551–560, Apr. 2010.

[19] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey," *Sensors*, vol. 12, no. 12, pp. 16 838–16 866, Dec. 2012.

[20] T. Liao, "Clustering of time series data – a survey," *Pattern Recognition*, vol. 38, no. 11, pp. 1857–1874, Nov. 2005.

[21] D. Kotsakos, G. Trajcevski, D. Gunopulos, and C. Aggarwal, "Time series data clustering," in *Data Clustering: Algorithms and Applications*, C. Aggarwal and C. Reddy, Eds. Chapman and Hall/CRC, Aug. 2013.

[22] X. Wang, K. Smith, and R. Hyndman, "Characteristic-based clustering for time series data," *Data Mining and Knowledge Discovery*, vol. 13, no. 3, pp. 335–364, Nov. 2006.

[23] R. Xu and D. Wunsch, *Clustering*. Wiley-IEEE Press, Oct. 2008.

[24] The Nsort Benchmark, http://www.ordinal.com.

[25] J. McCalpin, "STREAM: Substainable memory bandwidth in high performance computers," http://www.cs.virginia.edu/stream/.

[26] Great Internet Mersenne Prime Search, http://www.mersenne.org/freesoft/.

[27] LINPACK, http://www.netlib.org/linpack/.

[28] J. Lothian, J. Kuehn, M. Baker, J. Schrock, and S. Poole, "SystemBurn: Principles of design and operation release 3.1," Computer Science and Mathematics Division, Oak Ridge Natioal Laboratory, Technical Report TM-2013-234, Jun. 2013.

[29] The Graph 500 List, http://www.graph500.org.

[30] V. Pallipadi and A. Starikovskiy, "The Ondemand governor: Past, present, and future," in *Ottawa Linux Symposium*, Jul. 2006.

[31] D. Defays, "An efficient algorithm for a complete link method," *The Computer Journal*, vol. 20, no. 4, pp. 364–366, Nov. 1977.