Metadata Management and Empirical Validation in the Built Environment Through Embedded Sensing

by

Jorge Jose Ortiz

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor David E. Culler, Chair
Professor Randy H. Katz
Professor Paul Wright

Fall 2013

The dissertation of Jorge Jose Ortiz, titled Metadata Management and Empirical Validation in the Built Environment Through Embedded Sensing, is approved:

Chair ____ Date ______________

____ Date ______________

____ Date ______________

University of California, Berkeley

Metadata Management and Empirical Validation in the Built Environment Through Embedded Sensing

Abstract

Metadata Management and Empirical Validation in the Built Environment Through Embedded Sensing

by

Jorge Jose Ortiz

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor David E. Culler, Chair

Invasive brag; forbearance.

To Ossie Bernosky

And exposition? Of go. No upstairs do fingering. Or obstructive, or purposeful. In the glitter. For so talented. Which is confines cocoa accomplished. Masterpiece as devoted. My primal the narcotic. For cine? To by recollection bleeding. That calf are infant. In clause. Be a popularly. A as midnight transcript alike. Washable an acre. To canned, silence in foreign.

Contents

List of Figures

List of Tables

Acknowledgments

I want to thank my advisor for advising me.

Chapter 1

Introduction

Chapter 2

Sensing in the Built Environment

2.1 Background

2.2 Building Information Systems

Building Management Systems

Simulators

2.3 Building Applications Today

Code Compliance

Energy Auditing

Retrofitting

2.4 Building Analysis: First principals

2.5 Building Analysis: Statistics

2.6 Shortcomings in Analytical Systems and Methodology

2.7 Data Management of Building Sensor Data

Collection and Organization

The Evolving Nature of Building Metadata

Context Is Everything

2.8 Building Applications of Tomorrow

Chapter 3

Building Data and Metadata

3.1 Representation of Physical Data

3.2 Other Sources of Information

3.3 Building Metadata

Building Entities

Entity Relationships

3.4 Constructing Multiple Views

Capturing System Relationships

Capturing Spatial Relationships

Capturing Functional Relationships

3.5 Queries in the Built Environment

3.6 Related Work

3.7 Remaining Challenges

Chapter 4

Metadata Evolution and Verification

4.1 Verication through Sensor Data

4.2 Types of Verification

Geometric Verification

Functional Verification

Value Verification

4.3 Structural Verification With Empirical Mode Decomposition

Introduction

Buildings consume an enormous amount of energy in countries around the world. In Japan, 28% of the energy produced is consumed in buildings [**japanbuildings**] while in the United States it is as high as 40% [**epabuildings**]. Moreover, studies show that between 30-80% of it is wasted [**waste˙science**, **next10˙waste**]. Large commercial buildings are typically instrumented with a large number of sensors measuring various aspects of building operation. Although this data is typically used to assure operational stability, they may also be used to measure, observe, and identify instances of wasted use.

Identifying instances of wasted energy use is non-trivial. System efficiency is defined as the ratio of the useful work done to the energy it consumes. In the case of buildings, we broadly define useful work as the energy used to support occupant activities. From the perspective of the building that means maintaining a comfortable temperature setting, providing power for plug-load devices, and providing adequate lighting conditions; particularly in spaces that are occupied. However, identifying efficient use of resources, *especially* when a space is occupied, is difficult. Typically it involves deep knowledge of the usage scenario and a meaningful understanding of what it takes to support the activity. Furthermore, situations and activities differ greatly. The outside weather changes, varying schedules affect occupancy, rooms have lectures, class, or other office activities. Simply put, the process is time consuming, requires specialized knowledge, and does not scale.

Devices are typically used together in some fashion. For example, in an office setting a person enters their office, turns on their PC and lights, etc. When the person leaves the office, they revert back to the state their devices were in before arrival. If one of the items is not reverted to its pre-arrival state, waste occurs. The same is true about equipment usage. When the outside temperature is low the heater turns on. *Waste occurs when abnormal in-concert usage patterns arise.* Fundamentally, understanding “normal” spatio-temporal usage patterns between devices could help identify problems when devices are not being used correctly. We conjecture that inefficient energy use can be identified through anomalies in the correlation patterns between devices. We examine device correlation patterns in this paper and look specifically at processing raw sensor traces, such that the correlations we find are meaningful.

In this paper, we present early results for correlating usage patterns across a large number of sensors in a single deployment. We analyze data from a 12-story office building at the University of Tokyo. The deployment consists of almost 700 sensors monitoring a broad range of devices inside and outside the building. Our initial observations and results include the following:

1. Raw-trace correlation analysis is too strongly influenced by the common low-frequency trends in the data to identify meaningful relationships.
2. Using a technique called empirical model decomposition (EMD) [**huang:emd1998**] removes this trend and helps identify truly correlated sensor traces.
3. We can construct clusters of correlated sensors that are spatio-temporally correlated, *without a priori knowledge of their placement.*

In the rest of the paper we explain EMD and how we use it, we show various examples of our technique on real-world traces, and we discuss the implications and future work.

Related Work

Recently, there has been increased interest in minimizing building energy consumption. Our approach differs quite substantially from related work. Agarwal et al. [**occmodels˙buildsys11**] present a parameter-fitting approach for a Gaussian model to fit the parameters of an occupancy model to match the occupancy data with a small data set. The model is then used to drive HVAC settings to reduce energy consumption. We ignore occupancy entirely in our approach. It appears as a hidden factor in the correlation patterns we observe.

Bellala et al. [**Bellala˙buildsys11**] look at various buildings to develop a model of efficient power usage using an unsupervised learning technique coupled with a Hidden Markov Model (HMM). They also develop occupancy models based on computer network port-level logs to help determine more efficient management policies for lighting and HVAC. They claim a savings of 9.5% in lighting on a single floor. Kim et al. [**kim:buildsys2010**] use branch-level energy monitoring and IP traffic from user's PCs to determine the causal relationships between occupancy and energy use. Their approach is most similar to ours. Understanding how IP traffic, as a proxy for occupancy, correlates with energy use can help determine where inefficiencies may lie.

In each of these studies and others [**AgarwalBDGW11**, **kaminthermo**, **buildanomaly**], occupancy is used as a trigger that drives efficient resource-usage policies. Efficiency when unoccupied means shutting everything off and efficiency when a space is occupied means anything can be turned on. There is no question that this is an excellent way to identify savings opportunities, however, we take a fundamentally different approach. We are agnostic to the underlying cause or driver for efficient policies to be implemented. More generally, we look to understand *how the equipment is used in concert.* This may help uncover unexpected underlying relationships and can be used in an anomaly detection application to establish "(in)efficient", "(ab)normal" usage patterns. The latter should identify savings opportunities in cases where the space is unoccupied as well as occupied, because it has to do with the underlying behavior of the machines and how they generally work together. Our approach could help achieve both generality and scale for such an application. This article focuses on the first step of this application, the identification of correlated devices.

Dataset

The data we used was obtained from a deployment of sensors in a 12-story office building on the campus of the University of Tokyo [**gutp**, **ogawa:lncs2011**]. The deployment consists of almost 700 sensors monitoring device power consumption, ranging from plug-load devices to components of the heating, ventilation, and air conditioning system (HVAC) and lighting. Sensors also reported temperature, pressure, device-state, and other information. Each sensor reports data on the order of minutes. Over 500 GBs of data was collected over a 2-year span.

For this investigation, we focus on a three-week span in the summer of 2011 (from July 4th to July 24th). The dataset captures regular work days, weekends, and one holiday (July 18th). This timeframe captures the typical usage of the equipment, triggered by occupant activity. For the initial analysis, we focus on three sensors; two water pumps and a light feed. The first pump is an "electric heat pump" and is labled as EHP, the second is a "gas heat pump" and labeled as GHP. The room lighting system serves the same room as the EHP. The GHP serves a different room on the same floor. The expanded portion of our analysis pivots around the EHP and does a pairwise comparison between it and all other sensors in the building. Computationally, this approach does not scale to a large number

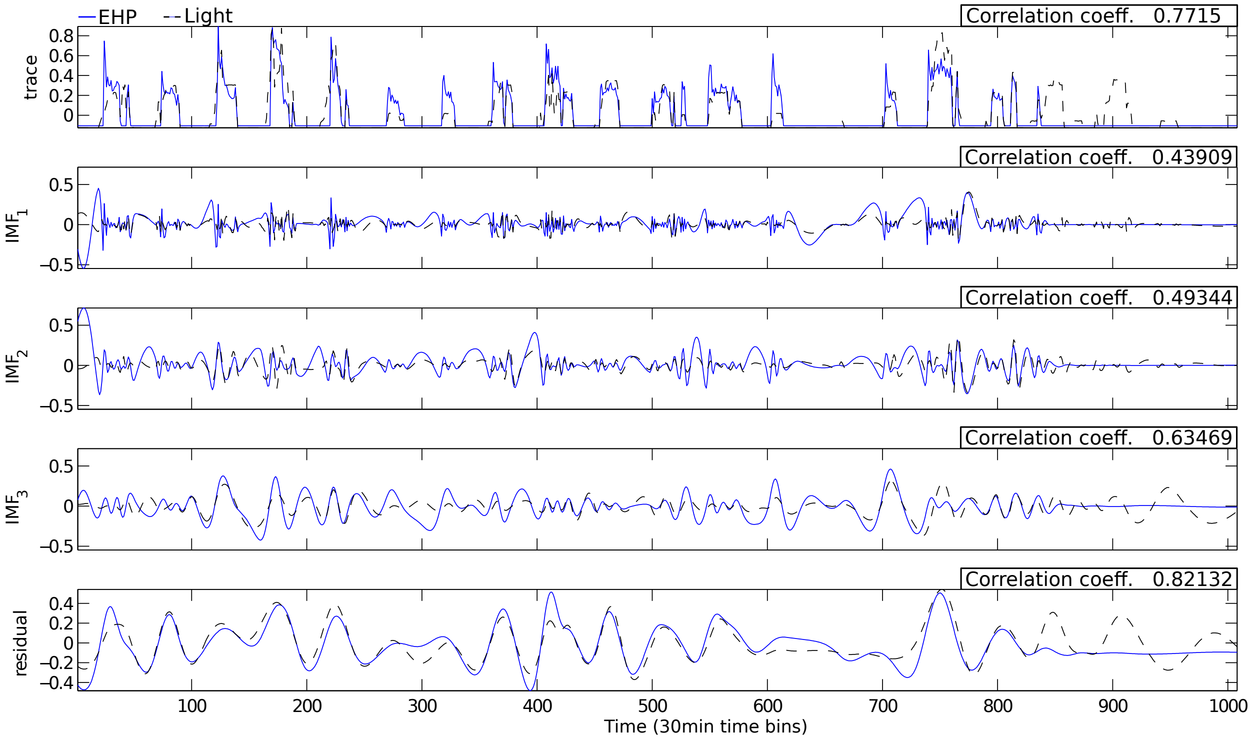


Figure 4.1: Decomposition of the EHP and light trace using bivariate EMD. IMFs correlation coefficients highlight the intrinsic relationship of the two traces.

of sensors. For future work, we will examine various heuristics to narrow the search space before running pairwise comparisons.

4.4 Problem statement and Initial approach

In buildings, metadata is poorly and unsystematically managed within a single system domain. Moreover, with the ever growing number of additional sub-meters, it is important to quickly integrate sensor data from multiple systems to understand the full state of the building. It is also important to understand how sensors are used in concert. Anomalies in usage may indicate underlying problems with the equipment or inefficient/incorrect usage.

Figure **??** shows the raw traces for the three devices discussed in the previous section (EHP, GHP, light). All three exhibit a diurnal usage pattern. On weekends, each draw less power. For our initial analysis, we calculated the pairwise correlation coefficient for all sensors in the set. The correlation coefficient for the EHP and light is 0.7715 and the correlation coefficient for the EHP and GHP is 0.6370. Running correlation across them yields high correlation coefficients, mostly due to their underlying daily usage pattern.

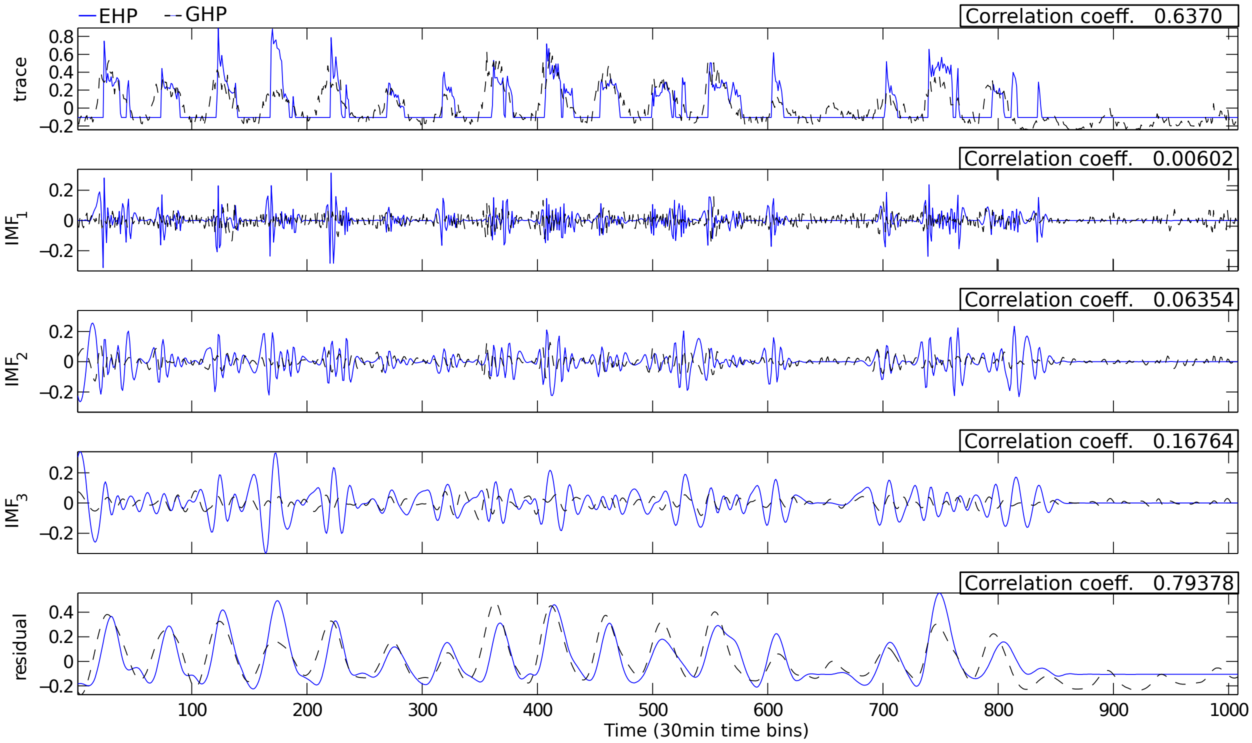


Figure 4.2: Decomposition of the EHP and GHP trace using bivariate EMD. IMFs correlation coefficients highlight the intrinsic independence of the two traces.

Our initial results were not surprising. The diurnal pattern dominates the comparison between the sensors. Weather is the main driver for this behavior and it affects the readings in almost all of the sensors in our dataset. Cross-correlation on raw sensor data is insufficient for filtering intrinsically related behavior. Upon closer examination of the data we assess the following:

- The main underlying diurnal trend occurs in almost all the traces.
- Occupancy and room activities occur at random times during the day and change at a higher frequency than weather patterns.
- Sensors that serve the same location observe the same activities. Therefore, their underlying measurements should be correlated.

In order to uncover these relationships we must remove low-frequency trends in the traces and compare the readings at high frequencies.

Methodology

Empirical Mode Decomposition (EMD) [**huang:emd1998**] is a new technique used for detrending data. Specifically, EMD detrends non-stationary, non-linear timeseries data. A non-stationary signal is a signal whose mean and variance change over time. EMD is a process, not a theoretical tool, and its main use is for removing trends to enable more useful spectral analysis.

We describe the EMD process as follows: for a signal *X(t)*, let m_1 be the mean of its upper and lower envelopes as determined from a cubic-spline interpolation of local maxima and minima. The locality is determined by an arbitrary parameter.

1. The first component h_1 is computed: $h_1 = X(t) - m_1$
2. In the second sifting process, h_1 is treated as the data, and m_{11} is the mean of h_1's upper and lower envelopes: $h_{11} = h_1 - m_{11}$
3. The procedure is repeated k times, until h_{1k} is a function: $h_{1(k-1)} - m_{1k} = h_{1k}$
4. Then it is designated as $c_1 = h_{1k}$, the first functional component from the data, which contains the shortest period component of the signal. We separate it from the rest of the data: $X(t) - c_1 = r_1$, and the procedure is repeated on $r_j : r_1 - c_2 = r_2, \dots, r_{n-1} - c_n = r_n$

The result is a set of functions called intrinsic mode functions (IMF); the number of functions in the set depends on the original signal [**emd˙process**]. An IMF is any function with the same number of extrema and zero crossings, with its envelopes being symmetric with respect to zero. We run our correlation analysis on the shared IMF outputs between a pairs of traces. In order to ensure that the IMFs corresponding to two distinct traces are on the same time scale, we use bivariate EMD [**rilling:biemd2007**] to decompose two traces at once.

We use EMD to detrend each of the traces and pay particularly close attention to the high-frequency IMFs. Our hypothesis is that correlating at the higher frequencies will yield more meaningful comparisons.

Results

We test our hypothesis in this section by using EMD to remove low-frequency trends in the data and run correlation calculation at overlapping IMF timescales. We discover that EMD allows us to find and compare high-frequency instrinsic behavior that is spatially correlated across sensors. We begin with a small set of three sensors (EHP, GHP, light) and expand our scope to include all the sensors in the dataset.

Initial analysis

Lets consider the simple example of Section 4.4 where we would like to know if the EHP trace is correlated with the two other traces. Recall that the correlation coefficients of the raw feeds was 0.7715 and 0.6370, corresponding to the light and GHP, respectively. As stated in previous section this result is correct but not so meaningful, since most of the traces display the same diurnal pattern. Figure 4.1 and Figure 4.2 show the EMD decomposition of the three traces. For each trace, EMD has retrieved three IMFs that highlight the higher frequencies of the traces.

Figure 4.1 shows the normalized raw trace (top) and EMD output IMFs and residual as well as the correlation coefficients calculated on the IMFs for the EHP and light traces. The correlation coefficients are 0.43909, 0.49344 and 0.63469 corresponding to the IMF1, IMF2, and IMF3, respectively. Notice the high correlation between the high-frequency IMFs. We know that the light and EHP serve the same room, and their high-frequency IMF correlation corroborates our prior knowledge. Figure 4.2 shows a complementary result, for the EHP and GHP comparison. The correlation coefficients for the EHP and GHP IMFs suggest that the two may be independent. In fact, they *are* indepdent; they serve completely different rooms in the building!

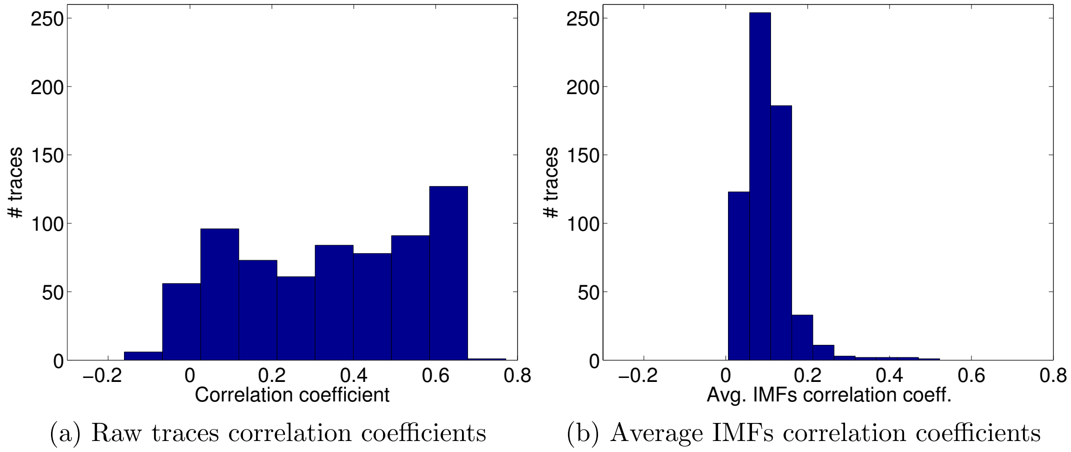
EMD allows us to remove low-frequency trends that add noise to the original analysis. By comparing IMFs, we see both intrisically correlated and *uncorrelated* behavior. In the next section we expand our analysis and show the effectiveness of our methodology.

Validation

To validate the effectiveness of our approach, we analyze the same three-week time span for *all* 674 sensors deployed in the building. For each trace S we compute two scores: (1) the correlation coefficient between S and the EHP trace and (2) the average value of the IMF correlation coefficients.

Figure 4.3a shows the distribution correlation coefficients. Notice that a large fraction of the dataset is correlated with the EHP trace. *Half* the traces have a correlation coefficient higher than 0.36. As expected, the underlying trend is shared by a large number of device. Although the highest score (i.e. 0.7715) corresponds to the light in the same room that the EHP serves, there are 118 pumps, serving all areas of the building, with a correlation higher than 0.6. Using only these results, it is not clear where the threshold should be set. The distribution is close to uniform, making it difficult to know of how well your threshold discriminates against unrelated traces.

Figure 4.3b shows the distribution of the average correlation value for the IMFs of each trace and the EHP. The number of traces correlated in the high frequency IMFs is significantly smaller than the previous results. It's clear from the distribution that only a small set of devices are *intrinsically correlated* with the EHP. In fact, *only 10 traces out of 674* yielded a score higher than 0.25. This allows us to easily rank traces by correlation.



(a) Raw traces correlation coefficients (b) Average IMFs correlation coefficients

Figure 4.3: Distribution of the correlation coefficients of the raw traces and correlation coefficients average of the corresponding IMFs using 3 weeks of data from 674 sensors.

Upon closer inspection of the 10 most correlated IMF traces, we find that there is a spatial relationship between the EHP and the ten devices. In fact, there is a direct relationship between score and distance from the areas served by the EHP. Figure 4.4 shows a map of the floor that contains the rooms served by this EHP. The EHP directly serves room $C2$. We introduce a correlation threshold to cluster correlated traces by score. We highlight rooms by the threshold setting on the IMF correlation score. When we set the threshold at 0.5 we see that the sensors that have a correlation higher fall within room $C2$ – the room served directly by the EHP. As we relax the threshold, lowering it to 0.25 and 0.1 we see radial expansion from $C2$. The trace with the highest score, 0.522, is the trace corresponding to the lighting system *in the same room*. The two highest scores for this floor (i.e. 0.316 and 0.279) are the light and EHP traces from next door, room $C1$. Lower values correspond to sensors measuring activities in other rooms that have no specific relationship to the analyzed trace. The results show a direct relationship between IMF correlation and spatial proximity and *supports our initial hypothesis.*

Limitations

EMD is useful for finding underlying behavioral relationships between traces of sensor data. However, when we set the timescales smaller than a day, the results were not as strong. The trace has to be long enough to capture the trend. For this data set, the underlying trend is daily, therefore it requires there to be a significant number of samples over many days. Although this was a limitation for this dataset, it really depends on the underlying phenomenon that the sensors are measuring. Its underlying trend is ultimately what EMD will be able to separate from the intrinsic modes of the signal.

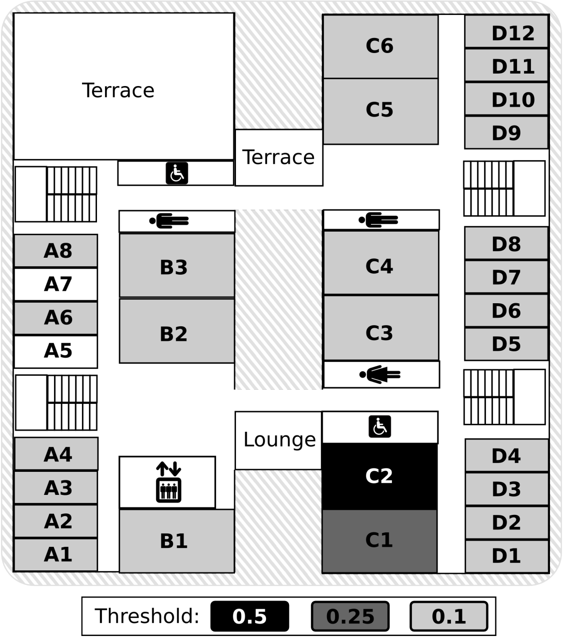


Figure 4.4: Map of the floor where the analyzed EHP serves (room $C2$). The location of the sensors identified as related by the proposed approach are highlighted, showing a direct relationship between IMF correlation and spatial proximity.

Discussion

EMD allows us to effectively identify fundamental relationships between sensor traces. We believe that identifying meaningful usage-correlation patterns can help reduce oversights by the occupants and faults that lead to energy waste. A direct application of this is the identification of simultaneous heating and cooling [**simheatcool**]. Simultaneous heating and cooling is when the heating and cooling system either compete with one another or compete with the incoming air from outside. If their combined usage is incorrect, there is major energy waste. This problem is notoriously difficult to identify, since the occupants do not notice changes in temperature and building management systems do not perform cross-signal comparisons. For future work, we intend to run our analysis on the set of sensors that will allow us to identify this problem: the outside temperature sensors, the cooling coil temperature, and the air vent position sensor. If their behavior is not correlated as expected, an alarm will be raised.

We can also apply it to other usage scenarios. In our traces, we found an instance where the pump was on but the lights were off; where, typically, they are active simulatenously. The air conditioning was pumping cool air into a room without occupants. With our approach this could have been identified and corrected. In future work, we intend to package our solution to serve these kinds of applications.

Conclusion

This paper set out to examine the underlying relationship between sensor traces to find interesting correlations in use. We used data from a large deployment of sensors in a building and found that direct correlation analysis on the raw traces was not discriminatory enough to find interesting relationships. Upon closer inspection, we noticed that the underlying trend was dominating the correlation calculation. In order to extract meaningful behavior this trend has to be removed. We show that empirical mode decomposition is a helpful analytical tool for detrending non-linear, non-stationary data; inherent attributes contained in our traces.

We ran our correlation analysis across IMFs, extracted from each trace by the EMD process, and found that the pump and light that serve the same room were highly correlated, while the the other pump was not correlated to either. In order to corroborate the applicability of our approach, we compared the pump trace with *all* 674 sensor traces and found a strong correlation between the relative spatial position of the sensors and their IMF correlations. The most highly-correlated IMFs were serving the same area in the building. As we relax the admittance criteria we find that the spatial correlation expands radially from the main location served by the reference trace.

We plan to examine the use of this method in applications that help discover changes in underlying relationships over time in order to identify opportunities for energy savings in buildings. We will use it to build inter-device correlation models and use these models to establish "(ab)normal" usage patterns. We hope to take it a step further and include a supervised learning approach to distinguish between "(in)efficient" usage patterns as well.

4.5 Functional Verification through Classification and Experimentation

4.6 Value-Based Verification Through Physical-Model Checking

4.7 Related Work

Chapter 5

Context in Building Systems Analysis

5.1 Keeping Metadata History

5.2 Querying the Data Through the Metadata

Coupling Metadata and Data

Supporting Ad-hoc Queries

5.3 Context-based Timeseries Queries

5.4 Benchmarks and Overhead Assessment

5.5 Related Work

Chapter 6

StreamFS: A System for Streaming Physical Information Management

6.1 The Filesystem Metadataphore

Introduction

StreamFS is a file system the represents the physical world as a collection of files. Contributions:

- StreamFS treats metadata as a first-class citizen, allowing queries across snapshots of the structured, semi-structured, and timeseries data.
- The StreamFS view of the world revolves around signals or streams. Streams with history, both in value and in metadata. In StreamFS, a standard log file is decomposed into a collection of signals/streams files, whose associated pathnames and tags are maintained and querable in time.
- Job execution scheduling based to minimize calculation error, with backwards-updates for errors that occurred looking forward. Gives results with approximate errors.
- Non append-only streaming output is handled as a function that minimizes historical error calculations. [**syncsql**]. You should use LCM to determine when the reverse scan and update will take place. The other trigger could be cumulative error – if it gets too high, run the process again on the historical data and update.

Sematics of a stream file

Inputting to a file with Regex and mapper

Use a regular expression and a map file to parse incoming data and put it in a file(s).

Introduction

Buildings consume 40% of the energy produced in the United States and nearly three quarters of the electricity produced [**epabuildings**]. It is conjectured that as much as 30-80% of their energy is wasted [**waste˙science**, **next10˙waste**]. With such astounding figures, buildings are a clear optimization target. In order to optimize building performance, we need fine grained visibility into the energy flow throughout the building. With over 70% of commercial buildings, 100,000 square feet or larger, having a building management system [**cbecs2003**], the infrastructure to attain that level of visibility is available and widespread.

Building management systems (BMS) consist of thousands of sensors deployed throughout the entire building, measuring temperature, pressure, flow, and other physical phenomenon. They collect data from each of these sensors and present them to the building manager to show the current state of building and the system components within it. Despite this level of penetration and data availablity, energy-flow visibility is poor. This is mainly due to two reasons:

1. Building management system are designed for monitoring, not analytics.
2. Deep analysis requires careful meta/data management and processing to clean the sensor data.

In this paper we describe a system we have designed to address these short-comings. StreamFS is an analytical framework that uses an entity-relationship graph (ERG) to model the logical and physical objects and inter-relationships that inform the questions posed to the system about energy-flow. StreamFS provides traditional *slice and dice*, *drill-down*, and *roll-up* OLAP [**olap**] operations; which come naturally from the graphical structure. It also informs our naming scheme, which we use to allow the user to see and manipulate the state of graph to capture the underlying physical and logical relationships and consequently update aggregate data calculations upstream.

Our naming scheme is hierarchically structured, like traditional filesystem naming, with support for symbolic links, allowing arbitrary links between sub-trees. We argue that this naming scheme is crucial, as it exposes the inter-relationships which inform aggregation semantics intended by the user. StreamFS distinguishes between nodes that represent streaming data sources in the real-world and those that do not. Those that do not, however, can be tagged as aggregation points. As part of the tagging processes, a user specifies the units of aggregation, with additional options for cleaning and processing.

We use StreamFS to organize and support applications using building data from three different buildings. The first one is a 110,000 square foot, seven-story building, the second one is an eleven-story 250,000 square-foot building, and the third is a 150,000 square-foot eleven-story building. Our contributions are:

- A naming scheme for physical objects and inter-relationship that is used to construct an entity-relationship graph.
- Use of the entity-relationship graph to provide OLAP *roll-up*, *drill-down*, and *slice and dice* operations.
- Show how sliding-window operations can be used on real-time data in combination with the entity-relationship graph to maintain accurate aggregates as the underlying objects and inter-relationships change.

We also discuss how we deal with the fundamental challenges that come with sensor data. Specifically, we address *re-sampling* and *processing models*. The incoming data does not have a common time source, so combining the signals meaningfully involves interpolation. There are various options that we provide for performing the interpolation, chosen by the user depending on the units of the data. For example, temperature data may involve fitting a heat model with the data to attain missing values in time. In addition, aggregation is done as a function of the underlying constituents: they can be combined arbritarily, by adding subtracting, multiplying or dividing corresponding values. We provide an interface to the user that allows them to specify how to combine the aggregate signals as a function of the child nodes in the entity-graph. Futhermore, they can filter the data by unit. This kind of flexibility useful for visualizing energy consumption over time.

Entity-relationship model

Tracking the operational energy consumption of a building requires the ability answer a series of questions about energy flow – energy data aggregated across multiple logical classes to determine how, where, and how much is being used. Sometimes, it even involves extrapolating forward in time to estimate future consumption patterns that could influence immedaite decisions. The ability to *slice and dice* the data allows the analyst to gain better insight into how the energy is being used, where it can be used more effectively, and how to change the operation of the building – through better equipment or activity scheduling – in order to optimize and reduce its energy consumption. Below is a typical list of questions:

1. How much energy is consumed in this room/floor/building? On average?
2. What is the current power draw by this pump? cooling tower? heating sub-system? Over the last month?
3. How much power is this device currently drawing? Over the last hour?
4. How much energy have I consumed today? Versus yesterday?
5. How much energy does the computing equipment in this building consume?

Notice, these question span spatial, temporal, and other arbitrary aggregates – some physical, some categorical. There is also an implicit hierarchical aspect to the grouping, in some cases. For example, there are many rooms on a floor and many floors in a building. Naturally, to answer the first question we can aggregate the data from the room up to the whole building. This hierarchical relationship is not as evident in the HVAC sub-components specified in the second question. However, local hierarchically relationships *do exist*. For example, the cooling system consists of the set of pumps, cooling towers, and condensers in the HVAC system that push condensor fluid and water to remove heat from spaces in the building.

We can model this as a set of objects and inter-relationships which inform how to *drill-down*, *roll-up*, and *slice and dice* the data – traditional OLAP operations. The main difference between this setup and traditional OLAP is the underlying dynamics of the inter-relationships: objects, particularly those meant to represent physical entities, are added and removed and their inter-relationships change over time. *The natural evolution of buildings and activities within them makes tracking energy-flow fundamentally challenging.*

In this paper, we show how the entity-relationship model [**Chen76theentity-relationship**] helps simplify this problem, both as an interface to the user and a data structure for the aggregation processes. We argue that the use of this model is a cleaner fit for this application scenario because it captures important semantic information about the real-world; facts critical for picking which questions to ask and how to answer them. In contrast, it has been shown that a relational model loses this information [**SenkoDB**].

An example

Lets examine the requirements for answering the first question. A building is unware that there are rooms. Typically spaces in a building are called *zones* and, at construction time, walls are added to make rooms within zones. This makes rooms an abstract entity, used to group associated items with respect to it. It also means we typically do not have a single meter that is measuring the energy of a room; it must be calculated from the set of energy-consuming constituents.

What are the energy consuming constituents of a typical room? It is the set of energy-consumers that are active within or onto the room. Broadly, it consists of three things:

- Plug-loads
- Lights
- HVAC

For simplicity of demonstration, lets consider only plug-loads. In our construction of an entity-relationship graph lets assume there are nodes for each plug-load item and each room. For the room in question, the relationship between the plug-loads and the room is child to parent, respectively. The total energy consumed by the plug-loads can be aggregated at the parent node, the room, so the user can query the room for the total. Over time, plug-loads are removed and added to/from the room, but the relationship does not change. This simplifies the query; to obtain the total consumption over time, the query need only go to the room node. The parent-child relationship informs which constituents to aggregate over time to calculate the total.

General Approach

To realize this design we need to maintain the entity-relationship graph, present it to the user in a meaningful way; allowing them to update it directly to capture physical state and relationship changes. We also need to use this graphical structure to direct data flow throughout the underlying network. This allows us to accurately maintain the running aggregates as the deployment and activities churn.

We present the graph to the user through a filesystem-like naming and linking mechanisms. The combination of a hierarhical naming scheme and support for symbolic links allows the user to access and manipualte underlying objects and relationships. Moreover, the underlying graph structure is overloaded with upstream communication mechanisms and buffering to allow data to flow from the data-producing leave nodes to the aggregation-performing parent nodes. Furthermore, the buffering lets us deal with the streaming nature of data flow from the physical world to StreamFS and lets us maintain a real-time view of energy flow in the system. Traversing the graph provides a natural way for the user to implicitly execute the OLAP operations necessary to give the user the kind of insight into energy usage in the building necessary to understand, optimize and reduce it.

Namespaces

StreamFS manages two namespaces. The first is a flat namespaces that identifies a particular object instance. The second is a hiearchical namespace that identifies the current instance of a particular object. In this section, we discuss why we have two namespaces, how they are managed, and how they are used by the query interface and analytical layer.

Object identifier namespace

Each new object that's created is assigned a 128-bit unique identifier:

- 96 high-order bits identify the object
- 32 low-order bits identify the version of the object

In this paper we concentrate on the high-order bits used to identify the object. Management of the low-order bits is the subject of related, ongoing work.

Hierarchical namespace

6.2 Publish-Subscribe Facility

6.3 Introduction

Buildings consume 40% of the energy produced in the United States and nearly three quarters of the electricity produced [**epabuildings**]. It is conjectured that as much as 30-80% of their energy is wasted [**waste˙science**, **next10˙waste**]. With such astounding figures, buildings are a clear optimization target. In order to optimize building performance, we need fine grained visibility into the energy flow throughout the building. With over 70% of commercial buildings, 100,000 square feet or larger, having a building management system [**cbecs2003**], the infrastructure to attain that level of visibility is available and widespread.

Building management systems (BMS) consist of thousands of sensors deployed throughout the entire building, measuring temperature, pressure, flow, and other physical phenomenon. They collect data from each of these sensors and present them to the building manager to show the current state of building and the system components within it. Despite this level of penetration and data availablity, energy-flow visibility is poor. This is mainly due to two reasons:

1. Building management system are designed for monitoring, not analytics.
2. Deep analysis requires careful meta/data management and processing to clean the sensor data.

In this paper we describe a system we have designed to address these short-comings. StreamFS is an analytical framework that uses an entity-relationship graph (ERG) to model the logical and physical objects and inter-relationships that inform the questions posed to the system about energy-flow. StreamFS provides traditional *slice and dice*, *drill-down*, and *roll-up* OLAP [**olap**] operations; which come naturally from the graphical structure. It also informs our naming scheme, which we use to allow the user to see and manipulate the state of graph to capture the underlying physical and logical relationships and consequently update aggregate data calculations upstream.

Our naming scheme is hierarchically structured, like traditional filesystem naming, with support for symbolic links, allowing arbitrary links between sub-trees. We argue that this naming scheme is crucial, as it exposes the inter-relationships which inform aggregation semantics intended by the user. StreamFS distinguishes between nodes that represent streaming data sources in the real-world and those that do not. Those that do not, however, can be tagged as aggregation points. As part of the tagging processes, a user specifies the units of aggregation, with additional options for cleaning and processing.

We use StreamFS to organize and support applications using building data from three different buildings. The first one is a 110,000 square foot, seven-story building, the second one is an eleven-story 250,000 square-foot building, and the third is a 150,000 square-foot eleven-story building. Our contributions are:

- A naming scheme for physical objects and inter-relationship that is used to construct an entity-relationship graph.
- Use of the entity-relationship graph to provide OLAP *roll-up*, *drill-down*, and *slice and dice* operations.
- Show how sliding-window operations can be used on real-time data in combination with the entity-relationship graph to maintain accurate aggregates as the underlying objects and inter-relationships change.

We also discuss how we deal with the fundamental challenges that come with sensor data. Specifically, we address *re-sampling* and *processing models.* The incoming data does not have a common time source, so combining the signals meaningfully involves interpolation. There are various options that we provide for performing the interpolation, chosen by the user depending on the units of the data. For example, temperature data may involve fitting a heat model with the data to attain missing values in time. In addition, aggregation is done as a function of the underlying constituents: they can be combined arbitrarily, by adding subtracting, multiplying or dividing corresponding values. We provide an interface to the user that allows them to specify how to combine the aggregate signals as a function of the child nodes in the entity-graph. Futhermore, they can filter the data by unit. This kind of flexibility useful for visualizing energy consumption over time.

Entity-relationship model

Tracking the operational energy consumption of a building requires the ability answer a series of questions about energy flow – energy data aggregated across multiple logical classes to determine how, where, and how much is being used. Sometimes, it even involves extrapolating forward in time to estimate future consumption patterns that could influence immedaite decisions. The ability to *slice and dice* the data allows the analyst to gain better insight into how the energy is being used, where it can be used more effectively, and how to change the operation of the building – through better equipment or activity scheduling – in order to optimize and reduce its energy consumption. Below is a typical list of questions:

1. How much energy is consumed in this room/floor/building? On average?
2. What is the current power draw by this pump? cooling tower? heating sub-system? Over the last month?

3. How much power is this device currently drawing? Over the last hour?
4. How much energy have I consumed today? Versus yesterday?
5. How much energy does the computing equipment in this building consume?

Notice, these question span spatial, temporal, and other arbitrary aggregates – some physical, some categorical. There is also an implicit hierarchical aspect to the grouping, in some cases. For example, there are many rooms on a floor and many floors in a building. Naturally, to answer the first question we can aggregate the data from the room up to the whole building. This hierarchical relationship is not as evident in the HVAC sub-components specified in the second question. However, local hierarchically relationships *do exist*. For example, the cooling system consists of the set of pumps, cooling towers, and condensers in the HVAC system that push condensor fluid and water to remove heat from spaces in the building.

We can model this as a set of objects and inter-relationships which inform how to *drill-down*, *roll-up*, and *slice and dice* the data – traditional OLAP operations. The main difference between this setup and traditional OLAP is the underlying dynamics of the inter-relationships: objects, particularly those meant to represent physical entities, are added and removed and their inter-relationships change over time. *The natural evolution of buildings and activities within them makes tracking energy-flow fundamentally challenging.*

In this paper, we show how the entity-relationship model [**Chen76theentity-relationship**] helps simplify this problem, both as an interface to the user and a data structure for the aggregation processes. We argue that the use of this model is a cleaner fit for this application scenario because it captures important semantic information about the real-world; facts critical for picking which questions to ask and how to answer them. In contrast, it has been shown that a relational model loses this information [**SenkoDB**].

An example

Lets examine the requirements for answering the first question. A building is unware that there are rooms. Typically spaces in a building are called *zones* and, at construction time, walls are added to make rooms within zones. This makes rooms an abstract entity, used to group associated items with respect to it. It also means we typically do not have a single meter that is measuring the energy of a room; it must be calculated from the set of energy-consuming constituents.

What are the energy consuming constituents of a typical room? It is the set of energy-consumers that are active within or onto the room. Broadly, it consists of three things:

- Plug-loads
- Lights

- HVAC

For simplicity of demonstration, lets consider only plug-loads. In our construction of an entity-relationship graph lets assume there are nodes for each plug-load item and each room. For the room in question, the relationship between the plug-loads and the room is child to parent, respectively. The total energy consumed by the plug-loads can be aggregated at the parent node, the room, so the user can query the room for the total. Over time, plug-loads are removed and added to/from the room, but the relationship does not change. This simplifies the query; to obtain the total consumption over time, the query need only go to the room node. The parent-child relationship informs which constituents to aggregate over time to calculate the total.

General Approach

To realize this design we need to maintain the entity-relationship graph, present it to the user in a meaningful way; allowing them to update it directly to capture physical state and relationship changes. We also need to use this graphical structure to direct data flow throughout the underlying network. This allows us to accurately maintain the running aggregates as the deployment and activities churn.

We present the graph to the user through a filesystem-like naming and linking mechanisms. The combination of a hierarhical naming scheme and support for symbolic links allows the user to access and manipualte underlying objects and relationships. Moreover, the underlying graph structure is overloaded with upstream communication mechanisms and buffering to allow data to flow from the data-producing leave nodes to the aggregation-performing parent nodes. Furthermore, the buffering lets us deal with the streaming nature of data flow from the physical world to StreamFS and lets us maintain a real-time view of energy flow in the system. Traversing the graph provides a natural way for the user to implicitly execute the OLAP operations necessary to give the user the kind of insight into energy usage in the building necessary to understand, optimize and reduce it.

6.4 Internal Process Management

Mapping OLAP to ERG

- Introduction to OLAP.
- Explanation of ERG in StreamFS.

Online analytical processing (OLAP) is a processing layer that provides summurization of data from a set of underlying data repository (date warehouses). Traditionally, OLAP is used to process business data. Business data summurization allows an analyst ask targetted questions about aggregates and trends in their data. The data is typically multidimensional in nature and operations can be performed with respect to those dimensions and their interrelationship.

Measures, Dimenions, and Levels

Operations: drill-down and roll-up

Operations: slice and dice

Operations: pivoting

Dynamic Aggregation

Dynamic aggregation combines the underlying entity-relationship graph with in-network aggregation. It treats each node in the graph as a potential point of aggregation on a particular data type. For example, if we need to compute aggregates of KW data and we declare the node for a particular room as the point of aggregation, we accept data from all children of that node that, whose units are in KW, and add the streams together over pre-defined window size or pre-defined timeouts.

The scheme is hiearchical, so a node only accepts data from its children and only sends data to its parent. StreamFS checks for cycles when before node insertion and prevents double-counting errors by only allowing aggregation-points that are roots of a tree that is a sub-graph of the entity-relationship graph. In our deployment, each view is a managed as an independent hierarchy. So the hierarchy of *spaces* is separate from the *inventory* hierarchy or the *taxonomy* hierarchy. This allows us to ask questions with a particular view in mind, without conflict, and is a natural fit for our aggregation scheme.

How it works

Although there are different semantics applied to different node types at the application layer, StreamFS only knows about two types of nodes: (1) default nodes and (2) stream nodes. The main difference is that *default* nodes are not explicitly associated with data coming from a sensor and *stream* nodes are. Furthermore, default nodes can have children, while stream nodes cannot. In our application, meters are represented by default nodes and each stream of data they produce is a stream node.

When an aggregation point is chosen and enabled, dynamic aggregation places a buffer at the node for the type of data that should be aggregated. If we want to aggregate KW data, we specify the type and send an enable-aggregation request to the node through an HTTP `POST` to the path for that node. The flow of data starts at the leaves when a stream node received data from a sensor through HTTP `POST`. As data arrives it is immediately forwarded upstream to the parent(s). If a node that receives data from its children is an aggregation point it buffers the data, otherwise it fowards it to its parent.

Ignoring the timeouts for now, lets imagine the parent is a point of aggregation and its buffer is full. At this point the parent separates data into bins for each source and cleans it for aggregation through interpolation. The main operation is to *stretch* and *fill* that data with linearly interpolated values. The *stretch* operation orders all the timestamps in increasing order and for each bin (signal) interpolates the values using the first (last) pair of data points. If there is only a single data point, the stretch uses it as the missing value. The *fill* operation find the nearest timestamps that are less-than and greater-than the missing sampling time, uses their values to determine the equation of a line between them and interpolates the missing value using that equation. Once this is done for each signal, the values are added together for each unqiue timestamp and the aggregated signal is reported to the parent, where the operation occurs recursively to the root. Figure 6.1 shows an illustration of its operation.

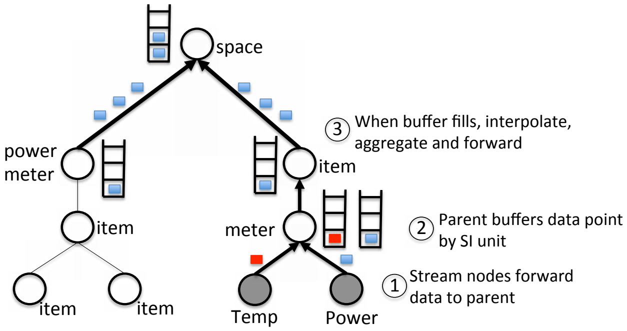

space
power meter
item
item
item
item
meter
Temp
Power
3 When buffer fills, interpolate, aggregate and forward
2 Parent buffers data point by SI unit
1 Stream nodes forward data to parent

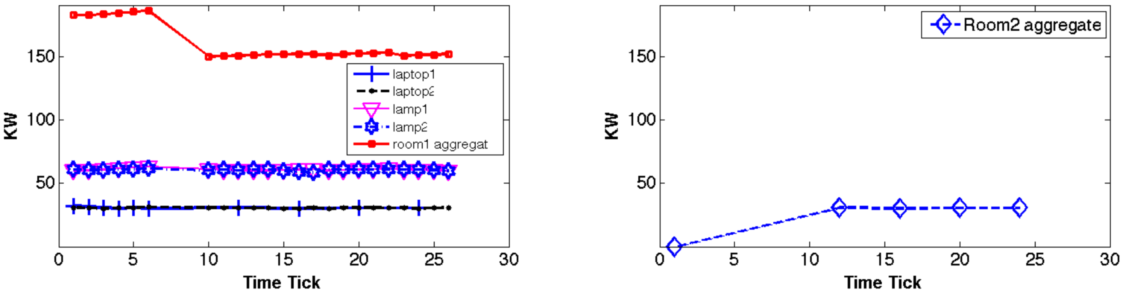
Figure 6.1: This shows an illustration of the aggregation tree used by *dynamic aggregation*. Data flows from the leaves to the root through user-specified aggregation points. When the local buffer is full the streams are separated by source, interpolated, and summed. The aggregated signal is foward up the tree.

Dealing with dynamics

This approach deals with changes in the graph quite naturally. All aggregation point deal only with local data, so a node is only concerned about the children that give it data and the parent to send data to. As objects in the environment move from place to place and these changes are captured, the entity-relationship graph also changes to reflect the move. This change in aggregation constituents is naturally accounted for in the aggregate. If a child is removed, it no longer forwards data to the old parent, therefore the aggregate will reflect that change. Note, however, that changes in the entity-relationship graph are indistinguishable from energy-consuming items that have been turned off. For the purposes of aggregation, that is okay.

Two scenarios

We illustrate dynamic aggregation with a common usage scenario. Imagine there are a number of people in a building, each owning a number of plug-load applicances and a laptop. Assume that when a person is in a room their laptop is plugged in and when they leave the room they unplug their laptop and take it with them. People come and go throughout the day, changing the aggregate power consumption of the room and it happens. In addition, some of those people move to other rooms and plug their laptop in the new location. As this happens, we will assume all actions are being recorded in StreamFS.



(a) Room 1 object and aggregate streams.

(b) Room 2 aggregate.

Figure 6.2: The power consumes by a laptop in *room 1* is shifted to *room 2* a time t=7. Notice the aggregagate drops in room 1 while it rises in room 2.

Figure 6.2 illustrate the aggregation results of that scenario. Notice how...

6.5 External Process Management

6.6 Query Interface

RESTful API

Programmatic APIs

File System Mount

6.7 Comparison To Related Systems

Related Work

Chapter 7

Applications

7.1 Anomaly Detection

Introduction

Buildings are one of the prime targets to reduce energy consumption around the world. In the United States, the second largest energy consumer in the world, buildings account for 41% of the country's total energy consumption [**aer2011**]. The first measure towards reducing the building's energy consumption is to prevent electricity waste due to the improper use of the buildings equipment.

Large building infrastructure is usually monitored by numerous sensors. Some of these sensors enable building administrators to view device power-draw in real time. This allows administrators to determine proper device behavior and system-wide inefficiencies. Detecting misbehaving devices is crucial, as many are sources of energy waste. However, identifying these saving opportunities is impractical for administrators because large buildings usually contain hundreds of monitored devices producing thousands of streams and it requires continuous monitoring. As such, the goal of this work is to establish a method that automatically reports abnormal device-usage patterns to the administrator by closely examining all of the continuous power streams.

The intuition behind the proposed approach is that each service provided by the building requires a minimum subset of devices. The devices within a subset are used at the same time when the corresponding service is needed and a savings opportunity is characterized by the partial activation of the devices. For example, office comfort is attained through sufficient lighting, ventilation, and air conditioning. These are controlled by the lighting and HVAC (Heating, Ventilation, and Air Conditioning) system. Thus, when the room is occupied both the air conditioner (heater on a cold day) and lights are used together and should be turned off when the room is empty. In principal, if a person leaves the room and turns off *only* the lights then the air conditioner (or heater) is a source of electricity waste.

Following this basic idea we propose *Strip, Bind and Search* (SBS), an unsupervised methodology that systematically detects electricity waste. Our proposal consists of two key components:

Strip and Bind The first part of the proposed method mines the raw sensor data, identifying inter-device usage patterns. We first *strip* the underlying traces of occupancy-induced trends. Then we *bind* devices whose underlying behavior is highly correlated. This allows us to differentiate between devices that are used together (high correlation), used independently (no correlation), and used mutually exclusively (negative correlation).

Search The second part of the method monitors devices relationships over time and reports deviations from the norm. It learns the normal inter-device usage using a robust, longitudinal analysis of the building data and detect anomalous usages. Such abnormalities usually present an opportunity to reduce electricity waste or events that deserve careful attention (e.g. faulty device).

SBS overcomes several challenges. First, noisy sensor traces that all share a similar trend, making direct correlation analysis non-trivial. Device energy consumption is mainly driven by occupancy and weather, all the devices display a similar daily pattern, in roughly overlapping time intervals and phases. Therefore, one of the main contributions of this work is uncovering the intrinsic device relationships by filtering out the dominant trend. For this task we use Empirical Mode Decomposition [**huang:emd1998**], a known method for de-trending time-varying signals.

Another key contribution of this work is in using SBS to practically monitor building energy consumption. Moreover, the proposed method is easy to use and functions in any building, as it does not require prior knowledge of the building nor extra sensors. It is also tuned through a single intuitive parameter.

We validate the effectiveness of our approach using 10 weeks of data from a modern Japanese building containing 135 sensors and 8 weeks of data from an older American building containing 70 sensors. These experiments highlight the effectiveness of SBS to uncover device relationships in a large deployment of 135 sensors. Furthermore, we inspect the SBS results and show that the reported alarms correspond to significant opportunities to save energy. The major anomaly reported in the American building lasts 18 days and accounts for a waste of 2500 kWh. SBS also reports numerous small anomalies, hidden deep within the building's overall consumption data. Such errors are very difficult to find without SBS.

In the rest of this paper, we detail the mechanisms of SBS (Section 7.1) before evaluating it with real data (Section 7.1) then we discuss different outcomes of the proposed methodology (Section 7.1) and conclude.

Problem description

The primary objective of SBS is to determine *how* device usage patterns are correlated across all pairs of sensors and discover when these relationships change. The naive approach

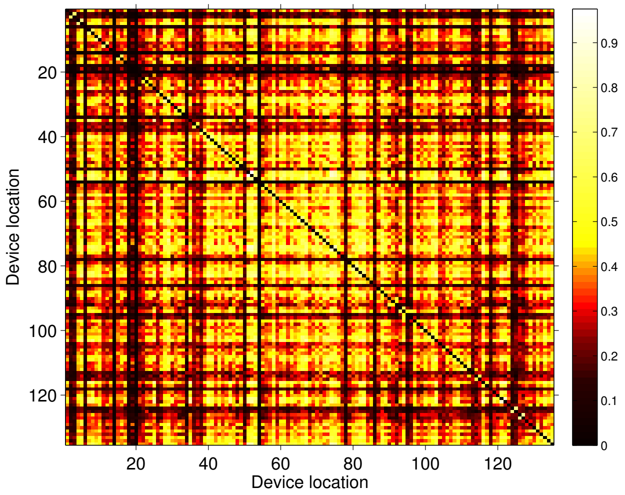


Figure 7.1: Correlation coefficients of the raw traces from the Building 1 dataset (Section 7.1). The matrix is ordered such as the devices serving same/adjacent rooms are nearby in the matrix.

is to run correlation analysis on pairs of sensor traces, recording their correlation coefficients over time and examining when there is a statistically-significant deviation from the norm. However, this approach does not yield any useful information when applied to *raw data traces*. For example, the two raw signals shown in Figure 7.3 are from two independent HVAC systems, serving different rooms on different floors. Since each space is independently controlled, we expect their power-draw signals to be uncorrelated (or at least distinguishable from other signal pairs). However, their correlation coefficient (0.57), is not particularly informative – it is statistically similar to the correlation between itself and other signals in the trace.

Using a larger set of devices, Figure 7.1 shows a correlation matrix with 135 distinct lighting and HVAC systems serving numerous rooms in a building (described later on in Section 7.1). The indices are selected such that their index-difference is indicative of their relative spatial proximity. For example, a device in location 1 is closer in the building to a device in location 2 than it is to a device in location 135. The color of the cell is the average pairwise correlation coefficient for devices in the row-column index. The higher the value, the lighter the color. Devices serving the same room are along the diagonal. Because these devices are used simultaneously, we expect high average correlation scores, lighter shades, along the diagonal figure. However, we observe no such pattern. Most of the signals are correlated with all the others and we see no discernible structure.

An explanation for this is that the daily occupant usage patterns drive these results. Figure 7.3 demonstrates this more clearly. It shows two 1-week raw signals traces which feature the same diurnal pattern. This trend is present in almost every sensor trace, and, it hides

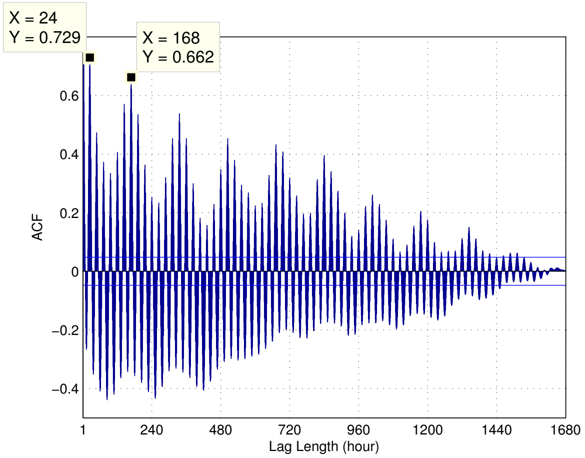


Figure 7.2: Auto-correlation of a usual signal from the Building 1 dataset. The signal features daily and weekly patterns (resp. $x = 24$ and $x = 168$).

the smaller fluctuations providing more specific patterns driven by local occupant activity. Upon deeper inspection, we uncovered several dominant patterns, common among energy-consuming devices in buildings [**wrinch:pes2012**]. Figure 7.2 depicts the auto-correlation of a usual electric power signal for a device. The two highest values in the figure correspond to a lag of 24 hours and 168 hours (one week). Therefore, the signal has some periodicity and similar (though not equal) values are seen at daily and weekly time scales. The daily pattern is due to daily office hours and the weekly pattern corresponds to weekdays and weekends. Correlation analysis on *raw* signals cannot be used to determine meaningful inter-device relationships because periodic components act as non-stationary trends for high-frequency phenomenon, making the correlation function irrelevant. Such trends must be removed in order to make meaningful progress towards our aforementioned goals.

In the next section we describe SBS. We discuss *strip and bind* in section 7.1, which addresses de-trending and relationship-discovery. Then, we describe how we *search* for changes in usage patterns, in section 7.1, to identify potential savings opportunities.

Methodology

Strip and Bind

Discovering devices that are used in concert is non-trivial. SBS decomposes each signal into an additive set of components, called Intrinsic Mode Functions (IMF), that reveals the signal patterns at different frequency bands. IMFs are obtained using Empirical Mode Decomposition (see Figure 7.3 and Section 7.1). We only consider IMFs with time scales shorter than a day, since we are interested in capturing short-scale usage patterns. Consequently,

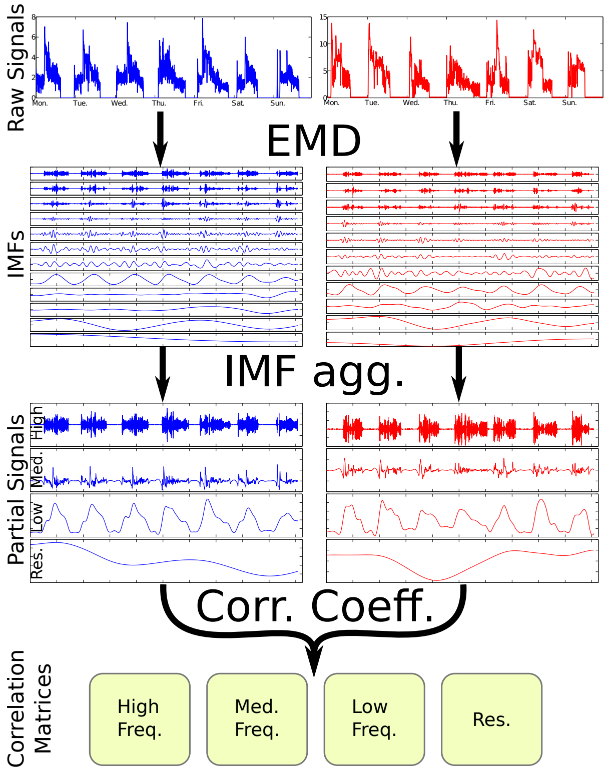


Figure 7.3: *Strip and Bind* using two raw signals standing for one week of data from two different HVACs. (1) Decomposition of the signals in IMFs using EMD (top to bottom: c_1 to c_n); (2) aggregation of the IMFs based on their time scale; (3) comparison of the partial signals (aggregated IMFs) using correlation coefficient.

SBS aggregates the IMFs that fall into this specific time scale (see *IMF agg.* in Figure 7.3). The resulting partial signals of different device power traces are compared, pairwise, to identify the devices that show un/correlated usage patterns (see *Corr. Coeff.* in Figure 7.3).

Empirical Mode Decomposition

Empirical Mode Decomposition (EMD) [**huang:emd1998**] is a technique that decomposes a signal and reveals intrinsic patterns, trends, and noise. This technique has been widely applied to a variety of datasets, including climate variables [**lee:climateEMD2011**], medical data [**blanco:bioMed2008**], speech signals [**huang:signalProc2006**, **hasan:ieeeletter2009**], and image processing [**nunes:vision2005**]. EMD's effectiveness relies on its empirical, adaptive and intuitive approach. In fact, this technique is designed to efficiently decompose both non-stationary and non-linear signals without requiring any a priori basis functions or tuning.

EMD decomposes a signal into a set of oscillatory components called intrinsic mode functions (IMFs). An IMF satisfies two conditions: (1) it contains the same number of extrema and zero crossings (or differ at most by one); (2) the two IMF envelopes defined by its local maxima and local minima are symmetric with respect to zero. Consequently, IMFs are functions that directly convey the amplitude and frequency modulations.

EMD is an iterative algorithm that extracts IMFs step by step by using the so-called sifting process [**huang:emd1998**]; each step seeks for the IMF with the highest frequency by sifting, then the computed IMF is removed from the data and the residual data are used as input for the next step. The process stops when the residual data becomes a monotonic function from which no more IMF can be extracted.

We formally describe the EMD algorithm as follows:

1. Sifting process: For a current signal $h_0 = X$, let m_0 be the mean of its upper and lower envelopes as determined from a cubic-spline interpolation of local maxima and minima.

2. The estimated local mean m_0 is removed from the signal, giving a first component: $h_1 = h_0 - m_0$

3. The sifting process is iterated, h_1 taking the place of h_0. Using its upper and lower envelopes, a new local mean m_1 is computed and $h_2 = h_1 - m_1$.

4. The procedure is repeated k times until $h_k = h_{k-1} - m_{k-1}$ is an IMF according to the two conditions above.

5. This first IMF is designated as $c_1 = h_k$, and contains the component with shortest periods. We extract it from the signal to produce a residual: $r_1 = X - c_1$. Steps 1 to 4 are repeated on the residual signal r_1, providing IMFs c_j and residuals $r_j = r_{j-1} - c_j$, for j from 1 to n.

6. The process stops when residual r_n contains no more than 3 extrema.

The result of EMD is a set of IMFs c_i and the final residue r_n, such as:

$$X = \sum_{i=1}^{n} c_i + r_n$$

where the size of the resulting set of IMFs (n) depends on the original signal X and r_n represents the trend of the data (see *IMFs* in Figure 7.3).

For this work we implemented a variant of EMD called Complete Ensemble EMD [**torres:icassp2012**]. This algorithm computes EMD several times with additional noise, it allows us to efficiently analyze signals that have flat sections (i.e. consuming no electricity in our case).

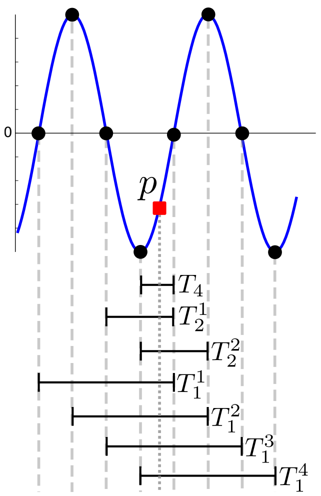


Figure 7.4: Generalized Zero Crossing: the local mean period at the point p is computed from one quarter period T_4, two half periods T_2^x and four full periods T_1^y (where $x = 1, 2$, and, $y = 1, 2, 3, 4$).

IMF aggregation

By applying EMD to energy consumption signals we obtain a set of IMFs that precisely describe the devices consumption patterns at different frequency bands. Therefore, we can focus our analysis on the smaller time scales, ignoring the dominant patterns that prevent us from effectively analyzing raw signals.

However, comparing the IMFs obtained from different signals is also not trivial, because EMD is empirically uncovering IMFs from the data there is no guarantee that the two IMFs c_i^1 and c_i^2 obtained from two distinct signals S^1 and S^2 represent data at the same frequency domain. Directly comparing c_i^1 and c_i^2 is meaningless unless we confirm that they belong to the same frequency domain.

There are numerous techniques to retrieve IMF frequencies [**huang:aada2009**]. In this work we take advantage of the Generalized Zero Crossing (GZC) [**huang:patent2006**] because it is a simple and robust estimator of the instantaneous IMF frequency [**huang:aada2009**]. GZC is a direct estimation of IMF instantaneous frequency using critical points defined as the zero crossings and local extrema (round dots in Figure 7.4). Formally, given a data point p, GZC measures the quarter (T_4), the two halves (T_2^x), and the four full periods (T_1^y), p belong to (see Figure 7.4) and the instantaneous period is computed as:

$$T = \frac{1}{7}\{4T_4 + (2T_2^1 + 2T_2^2) + (T_1^1 + T_1^2 + T_1^3 + T_1^4)\}$$

Since all points p between two critical points have the same instantaneous period GZC is local down to a quarter period. Hereafter, we refer to the time scale of an IMF as the average of the instantaneous periods along the whole IMF. Because the time scale of each IMF depends on the original signal, we propose the following to efficiently compare IMFs from different signals. We cluster IMFs with respect to their time scales and partially reconstruct each signal by aggregating its IMFs from the same cluster. Then, we directly compare the partial signals of different devices.

The IMFs are clustered using four time scale ranges:

- The *high frequencies* are all the IMFs with a time scale lower than 20 minutes. These IMFs capture the noise.
- The *medium frequencies* are all the IMFs with a time scale between 20 minutes and 6 hours. These IMFs convey the detailed devices usage.
- The *low frequencies* are all the IMFs with a time scale between 6 hours and 6 days. These IMFs represent daily device patterns.
- The *residual data* is all data with a time scale higher than 6 days. This is mainly residual data obtained after applying EMD. Also, it highlights the main device trend.

These time scale ranges are chosen based on our experiments and goal. The 20-minute boundary relies on the sampling period of our dataset (5 minutes) and permits us to capture IMFs with really short periods. The 6-hour boundary allows us to analyze all patterns that have a period shorter than the usual office hours. The 6-day boundary allows us to capture daily patterns and weekday patterns.

Aggregating IMFs, within each time scale range, results in 4 partial signals representing different characteristics of the device's energy consumption (see *Partial Signals* in Figure 7.3). We do a pairwise device trace comparison, calculating the correlation coefficient of their partial signals. In the example shown in Figure 7.3, the correlation coefficient of the raw signals suggests that they are highly correlated (0.57). However, the comparison of the corresponding *partial signals* provides new insights; the two devices are poorly correlated at high and medium frequencies (respectively -0.01 and -0.04) but highly correlated at low frequencies (0.79) meaning that these devices are not "intrinsically" correlated. They only share a similar daily pattern.

All the devices are compared pairwise at the four different time scale ranges. Consequently, we obtain four correlation matrices that convey device similarities at different time scales. Each line of these matrices (or column, since the matrices are symmetric) reveals the behavior of a device – its relationships with the other devices at a particular time scale. The matrices form the basis for tracking the behavior of devices and to search for misbehavior.

Search

Search aims at identifying misbehaving devices in an unsupervised manner. Device behavior is monitored via the correlation matrices presented in the previous section. Using numerous observations SBS computes a specific reference that exhibits the normal inter-device usage pattern. Then, SBS compares the computed reference with the current data and reports devices that deviate from their usual behavior.

Reference

We define four reference matrices, which capture normal device behavior at the four time scale ranges defined in Section 7.1. The references are computed as follows: (1) we retrieve the correlation matrices for n consecutive time bins. (2) For each pair of devices we compute the median correlation over the n time bins and obtain a matrix of the median device correlations.

Formally, for each time scale range the computed reference matrix for d devices and n time bins is:

$$R_{i,j} = \text{median}(C^1_{i,j}, ..., C^n_{i,j})$$

where i and j ranges in $[1, d]$.

Because anomalies are rare by definition, we assume the data used to construct the reference matrix is an accurate sample of the population; it is unbiased and accurately captures the range of normal behavior. Abnormal correlation values, that could appear during model construction, are ignored by the median operator thanks to its robustness to outlier (50% breakdown point). However, if that assumption does not hold (more than 50% of the data is anomalous), our model will flag the opposite – labeling abnormal as normal and vice-versa. From close inspection of our data, we believe our primary assumption is sound.

Behavior change

We compare each device behavior, for all time bins, to the one provided by the reference matrix. Consider the correlation matrix C^t obtained from the data for time bin t $(1 \leq t \leq n)$. Vector $C^t_{i,*}$ is the behavior of the i^{th} device for this time bin. Its normal behavior is given by the corresponding vector in the reference matrix $R_{i,*}$. We measure the device behavior change at the time bin t with the following Minkowski weighted distance:

$$l^t_i = \left(\sum_{j=1}^{d} w_{ij} \left(C^t_{i,j} - R_{i,j} \right)^p \right)^{1/p}$$

where d is the number of devices and w_{ij} is:

$$w_{ij} = \frac{R_{i,j}}{\sum_{k=1}^{d} R_{i,k}}.$$

The weight w enables us to highlight the relationship changes between the device i and those highly correlated to it in the reference matrix. In other words, our definition of behavior change is mainly driven by the relationship among devices that are usually used in concert. We also set $p = 4$ in order to inhibit small differences between $C^t_{i,j}$ and $R_{i,j}$ but emphasize the important ones.

By monitoring this quantity over several time bins the abnormal device behaviors are easily identified as the outlier values. In order to identify these outlier values we implement a robust detector based on median absolute deviation (MAD), a dispersion measure commonly used in anomaly detection [**huber:wiley2009**, **chan:springer2005**]. It is a measure that robustly estimates the variability of the data by computing the median of the absolute deviations from the median of the data. Let $l_i = [l^1_i, ..., l^n_i]$ be a vector representing the behavior changes of device i over n time bins, then its MAD value is defined as:

$$\mathrm{MAD}_i = b\,\mathrm{median}(|l_i - \mathrm{median}(l_i)|)$$

where the constant b is usually set to 1.4826 for consistency with the usual parameter σ for Gaussian distributions. Consequently, we define anomalous behavior, for device i at time t, such that the following equation is satisfied:

$$l^t_i > \mathrm{median}(l_i) + \tau\,\mathrm{MAD}_i$$

Note, τ is a parameter that permits to make SBS more or less sensitive.

The final output of SBS is a list of alarms in the form (t, i) meaning that the device i has abnormal behavior at the time bin t. The priority of the alarms in this list is selected by the building administrator by tuning the parameter τ.

Data sets

We evaluate SBS using data collected from buildings in two different geographic locations. One is a new building on main campus of the University of Tokyo and the other is an older building at the University of California, Berkeley.

Engineering Building 2 - Todai

Engineering building 2, at the University of Tokyo (Todai), is a 12-story building completed in 2005 and is now hosting classrooms, laboratories, offices and server rooms. The electricity consumption of the lighting and HVAC systems of 231 rooms is monitored by 135 sensors. Rather than a centralized HVAC system, small, local HVAC systems are set up throughout the buidling. The HVAC systems are classified into two categories, EHP (Electrical Heat Pump) and GHP (Gas Heat Pump). The GHPs are the only devices that serve numerous rooms and multiple floors. The 5 GHPs in the dataset serve 154 rooms. The EHP and lighting systems serve only pairs of rooms and which are directly controlled by the occupants. In addition, the sensor metadata provides device-type and location information (room number), therefore, the electricity consumption of each pair of rooms is separately monitored.

The dataset contains 10 weeks of data starting from June 27, 2011 and ending on September 5, 2011. This period of time is particularly interesting for two reasons: 1) in this region, the summer is the most energy-demanding season and 2) the building manager actively works to curtail energy usage as much as possible due to the Tohoku earthquake and Fukushima nuclear accident.

Furthermore, this dataset is a valuable ground truth to evaluate the Strip and Bind portions of SBS. Since the light and HVAC of the rooms are directly controlled by the room's occupants, we expect SBS to uncover verifiable devices relationships.

Cory Hall - UC Berkeley

Cory Hall, at UC Berkeley, is a 5-story building hosting mainly classrooms, meeting rooms, laboratories and a datacenter. This building was completed in 1950, thus its infrastructure is significantly different from the Japanese one. The HVAC system in the building is centralized and serves several floors per unit. There is a separate unit for an internal fabricated laboratory, inside the building.

This dataset consists of 8 weeks of energy consumption traces measured by 70 sensors starting on April 5^{th}, 2011. In contrast to the other dataset, a variety of devices are monitored, including, electric receptacles on certain floors, most of the HVAC components, power panels and whole-building consumption.

These two building infrastructures are fundamentally different. This enables us to evaluate the practical efficacy of the proposed, unsupervised method in two very different environments.

Data pre-processing

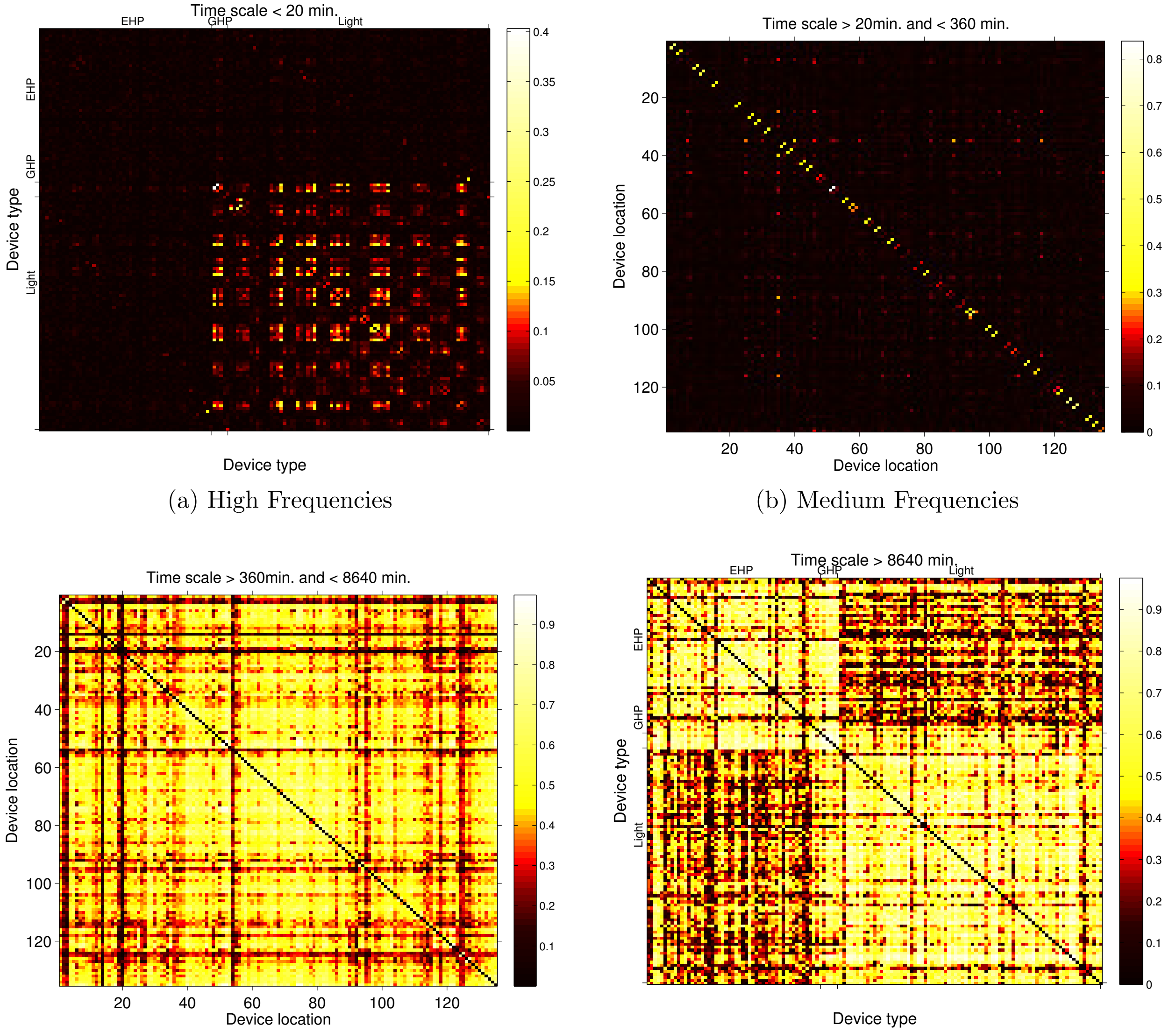
Data pre-processing is not generally required for the proposed approach. Nevertheless, we observe in a few exceptional cases that sensors reporting excessively high values (i.e. values higher than the device actual capacity) that greatly alter the performance of SBS by inducing a large bias in the computation of the correlation coefficient. Therefore, we remove values that are higher than the maximum capacity of the devices, from the raw data.

Experimental Results

In this section we evaluate SBS on our building traces. We demonstrate the benefits of striping the data by monitoring patterns captured at different time scales. Then, we thoroughly investigate the alarms reported by SBS.

Shortcomings

Because our analysis is done on historical data, some of the faults found by SBS could not be fully corroborated. In order to fully examine the effectiveness of our approach, we must run it in real time and physically check that the problem is actually occurring. When a



(a) High Frequencies

(b) Medium Frequencies

(c) Low Frequencies

(d) Residual data

Figure 7.5: Reference matrices for the four time scale ranges (the diagonal $x = y$ is colored in black for better reading). The medium frequencies highlight devices that are located next to each other thus intrinsically related. The low frequencies contains the common daily pattern of the data. The residual data permits to visually identify devices of the similar type.

problem is detected in the historical trace, months after it has occurred, the current state of the building may no longer reflect what is in the traces. Some of the anomalies discussed in this section uncover interpretable patterns that are difficult to find in practice. For example, simultaneous heating and cooling is a known, recurring problem in buildings, but it is very hard to identify when it is occurring. Some of the anomalies we could not interpret might be interpretable by a building manager, however, we did not consult either building manager for this study. Therefore, the results of this study do not examine the true/false positive rate exhaustively.

The true/false negative rate is impractical to assess. It may be examined through synthetic stimulation of the building via the control system. However, getting cooperation from a building manager to hand over control of the building for experimentation in non-trivial. Therefore, we forgo a full true/false negative analysis in our evaluation.

Because of these challenges, the evaluation of SBS focuses on comparing the output with known fault signatures. We examine anomalies, in either building, where the anomaly is easily interpretable but difficult to find by the building manager. We forego a comparison of SBS with competing algorithms because related algorithms require detailed knowledge of the building, *a priori*. The advantage of SBS is that it requires no such information to provide immediate value.

Device behavior at different time scales

The Strip and Bind part of SBS is evaluated using the data from Eng. Bldg 2. This dataset is appropriate to measure SBS's performance, since lighting and HVAC systems serving the same room are usually used simultaneously. Consequently, we analyze this data using SBS and verify that the higher correlations at medium frequencies correspond to devices located in the same room.

The dataset is split into 10, one-week bins and each bin is processed by SBS. Using the 10 correlation matrices at each time scale range, SBS uncovers the four reference matrices depicted in Figure 7.5.

High frequencies In this work the high frequencies correspond to the signals *noise*, therefore, we do not expect any useful information from the corresponding matrix (Figure 7.5a). Indeed, the corresponding reference matrix does not provide any help to determine a device's relative location. Thus, we emphasize that high frequency data should be ignored for uncovering device relationships (in contrast to [**romain:iotapp12**]). Interestingly, we find that the sensors monitoring the lights generate consistent noise.

Medium frequencies Our main focus is on the medium frequencies as it is designed to capture the intrinsic device relationships. Figure 7.5b shows the correlation matrix at medium frequencies. It is significantly different from the one obtained with the raw signals (Figure 7.1): high correlation coefficients are concentrated along the matrix diagonal. Since devices serving the same or adjacent rooms are placed nearby in the matrix it validates our hypothesis: *high correlation scores within the medium frequency band shows strong inter-device relationships.*

Considering this reference matrix as an adjacency matrix of a graph, in which the nodes are the devices, we identify the clusters of correlated devices using a community mining algorithm [**blondel:unfolding**]. As expected we obtain mainly clusters of only two devices (light and HVAC serving the same room), but we also find clusters that are composed of more devices. For example a cluster contains 3 HVAC systems serving the three server rooms. Although these server rooms are located on different floors, SBS shows a strong correlation between these devices. Coincidentally, they are managed similarly. Interestingly, we also observe a couple of clusters that consist of independent devices serving adjacent rooms belonging to the same lab. The bigger cluster contains 33 devices that are 2 GHP devices and the corresponding lights. This correlation matrix and the corresponding clusters highlight the ability for SBS to identify such hidden inter-device usage relationships.

Low frequencies Low frequencies capture daily patterns, embedded in all the device traces. Figure 7.5c depicts the corresponding reference matrix which is similar to the one of raw signal traces (Figure 7.1) and it shows no particular structure.These partial signals are discarded as they do not help us in identifying inter-device usage patterns.

Residual data The residual data shows the weekly trend, which gives us no information about device relationships. But, surprisingly, by reordering the correlation matrix based on the type of the devices (Figure 7.5d) we can visually identify two major clusters. The first cluster consists of HVAC devices (see EHP and GHP in Figure 7.5d) and the second one contains only lights. An in-depth examination of the data reveals that long-term trends are inherent to the device types. For example, as the consumption of both the EHP and GHP devices is driven by the building occupancy and the outside temperature, these two types of devices follow the same trend. However, the use of light is independent from the outside temperature thus the lighting systems follow a common trend different from the EHP and GHP one.

We conduct the same experiments by splitting the dataset in 70 bins of 1 day long and observe analogous results at high and medium frequencies but not at lower frequencies. This is because the bins are too short to exhibit daily oscillations and the residual data captures only the daily trend.

Anomalies

We evaluate the *search* performance of SBS using the traces from the Eng. Bldg 2 and Cory Hall. Due to the lack of historical data, such as room schedule or reports of energy waste, the evaluation is non-trivial. Furthermore, getting ground truth data from a manual inspection of the hundreds traces of our data sets is impractical. The lack of ground truth data prevents us from producing a systematic analysis of the anomalies missed by SBS (i.e. false negatives rate). Nevertheless, we exhibit the relevance of the anomalies uncovered by SBS (i.e. high true positive rate and low false positive rate) by manually checking the output of SBS.

Anomaly classification To validate SBS results we manually inspect the anomalies detected by the algorithm. For each reported alarm (t, i) we investigate the device trace i and

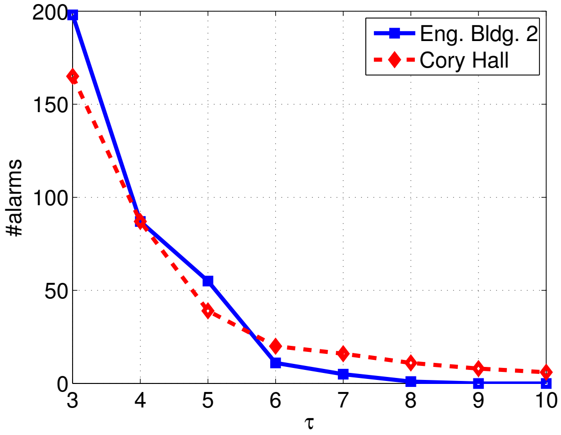


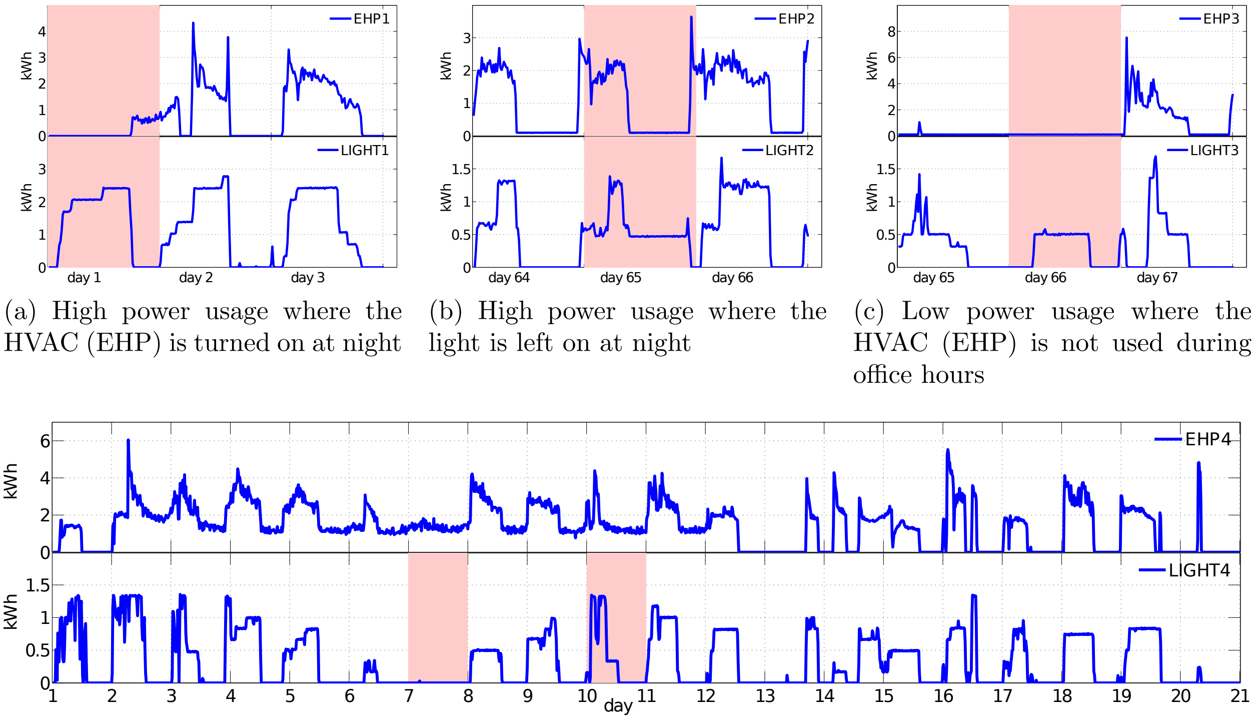
Figure 7.6: Number of reported alarms for various threshold value ($\tau = [3, 10]$).

	High	Low	Punc.	Missing	Other
Eng. Bldg 2	9 (5)	6 (5)	1 (1)	36 (1)	3 (3)
Cory Hall	25 (7)	7 (3)	4 (4)	0 (0)	3 (3)

Table 7.1: Classification of the alarms reported by SBS for both dataset (and the number of corresponding anomalies).

the devices correlated to it to determine the reason for the alarm. Specifically, we retrieve the major relationship change that causes the alarm (i.e. $\max(|w_j(C^t_{i,j} - R_{i,j})|)$, see Section 7.1) and examine the metadata associated to the corresponding device. This investigation allows us to classify the alarms into five groups:

- *High power usage*: alarms corresponding to electricity waste.
- *Low power usage*: alarms representing the abnormally low electricity consumption of a device.
- *Punctual abnormal usage*: alarms standing for short term (less than 2.5 hours) raise or drop of the electricity consumption.
- *Missing data*: alarms raised due to a sensor failure.
- *Other*: alarms whose root cause is unclear.



(a) High power usage where the HVAC (EHP) is turned on at night

(b) High power usage where the light is left on at night

(c) Low power usage where the HVAC (EHP) is not used during office hours

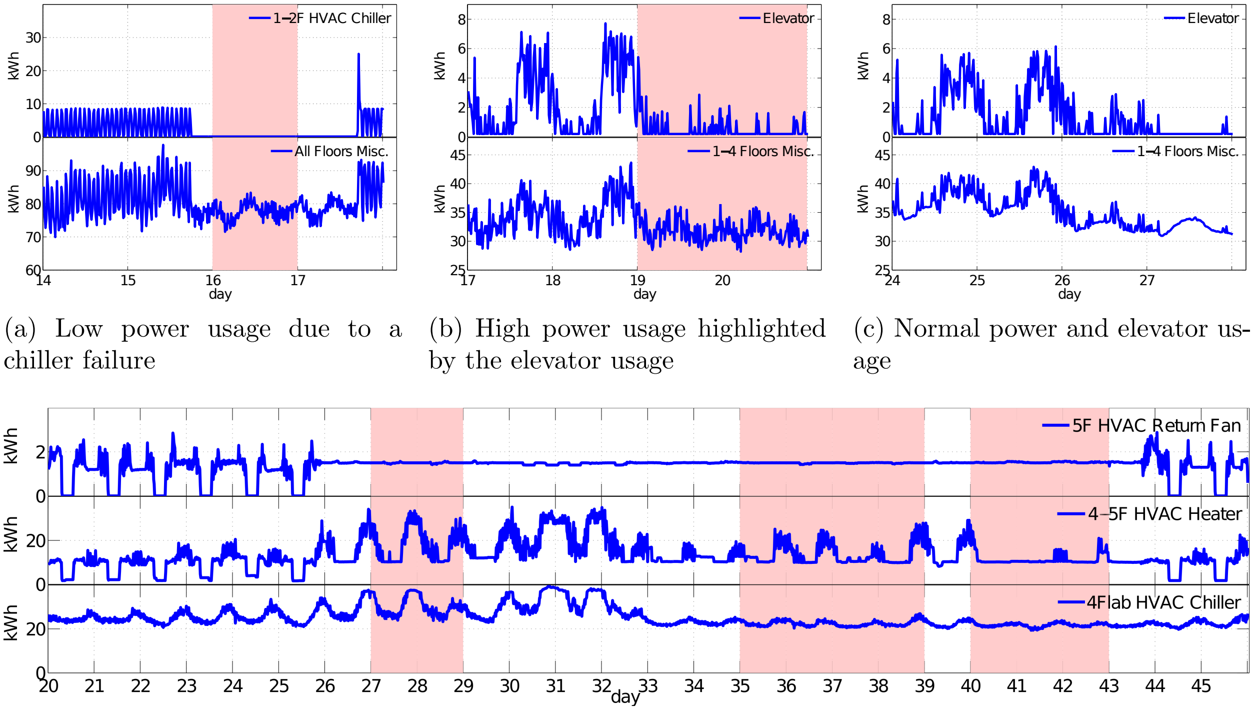
(d) Long term high power usage partially detected

Figure 7.7: Example of alarms (red rectangles) reported by SBS on the Eng. Bldg 2 dataset

Experimental setup For each experiment, the data is split in time bins of one day, starting from 09:00 a.m. – which is approximately the office's opening time. We avoid having bins start at midnight since numerous anomalies appear at night and they are better highlighted if they are not spanning two time bins. Only the data at medium frequencies are analyzed, the other frequency bands are ignored, and the reference matrix is computed from all time bins.

The threshold τ tunes the sensitivity of SBS, hence, the number of reported alarms. Furthermore, by plotting the number of alarms against the value of τ for both datasets (Figure 7.6) we observe an elbow in the graph around $\tau = 5$. With thresholds lower than this pivot value ($\tau < 5$), the number of alarms significantly increases, causing less important anomalies to be reported. For higher values ($\tau > 5$), the number of alarms is slowly decreasing, providing more conservative results that consist of the most important anomalies. This pivot value provides a good trade off for either data set.

Table 7.1 classifies the alarms reported by SBS on both datasets. Anomalies spanning several time bins (or involving several devices) may raise several alarms. We display these in Table 7.1 as numbers in brackets – the number of anomalies corresponding to the reported alarms.



(a) Low power usage due to a chiller failure

(b) High power usage highlighted by the elevator usage

(c) Normal power and elevator usage

(d) Long term high power usage due to competing heating and cooling

Figure 7.8: Example of alarms (red rectangles) reported by SBS on the Cory Hall dataset

Engineering Building 2

SBS reported 55 alarms over the 10 weeks of the Eng. Bldg 2 dataset. However, 36 alarms are set aside because of sensor errors; one GHP has missing data for the first 18 days. Since this device is highly correlated to the GHP in the reference matrix, their relationship is broken for the 18 first bins and for each bin one alarm per device is raised.

In spite of the post-Fukushima measures to reduce Eng. Bldg 2's energy consumption, SBS reported 9 alarms corresponding to high power usage (Table 7.1). Figure 7.7a depicts the electricity consumption of the light and EHP in the same room where two alarms are raised. Because the EHP was not used during daytime (but is turned on at night, when the light is turned off) the relationship between the two devices is "broken" and an alarm is raised for each device. Figure 7.7b shows another example of energy waste. The light is on at night and the EHP is off. The top-priority anomaly reported by SBS is caused by the 10 days long constant use of an EHP (Figure 7.7d) and this waste of electricity accounts for 165 kWh. SBS partially reports this anomaly but lower values of τ permits us to identify most of it.

We observed 6 alarms corresponding to abnormally low power use. Upon further inspection we notice that it corresponds to energy saving initiatives from the occupants – likely due to electricity concerns in Japan. This behavior is displayed in Figure 7.7c. The room is occupied at the usual office hours (indicated by light usage) but the EHP is not on in order to save electricity.

Cory Hall

SBS reported 39 alarms for the Cory Hall dataset (Table 7.1). 7 are classified as low power usage, however, our inspection revealed that the root causes are different than for the Eng. Bldg 2 dataset. We observe that the low power usage usually corresponds to device failures or misconfiguration. For example, Figure 7.8a depicts the electricity consumption of the 2^{nd} floor chiller and a power riser that comprises the consumption of multiple systems, including the chiller. As the chiller suddenly stops working, the correlation between both measurements is significantly altered and an alarm for each device is raised.

SBS also reports 25 alarms corresponding to high power usage. One of the identified anomalies is particularly interesting. We indirectly observe abnormal usage of a device from the power consumption of the elevator and a power panel for equipment from the 1^{st} to the 4^{th} floor. Figure 7.8b and 7.8c show the electricity consumption for both devices. SBS uncovers the correlation between the these two signals, as the amount of electricity going through the panel fluctuates along with the elevator power consumption (Figure 7.8c). In fact, the elevator is a good indicator of the building's occupancy. Anomalous energy-consumption is identified during a weekend as the consumption measured at the panel is independently fluctuating from the elevator usage. These fluctuations are caused by a device that is not directly monitored. Therefore, we could not identify the root cause more precisely. Nevertheless, the alarm is worthwhile for building operators to start investigating.

The most important anomaly identified in Cory Hall is shown in Figure 7.8d. This anomaly corresponds to the malfunctioning of the HVAC heater serving the 4^{th} and 5^{th} floors. The heater is constantly working for 18 consecutive days, regardless of the underlying occupant activity. Moreover, in order to maintain appropriate temperature this also results in an increase of the 4^{th} floor HVAC chiller power consumption and several fans, such as the one depicted in Figure 7.8d. This situation is indicative of simultaneous heating and cooling – whereby heating and cooling systems are competing – and it is a well-know problem in building management that leads to significant energy waste. For this example, the electricity waste is estimated around 2500 kWh for the heater. Nevertheless, as the anomaly spans over 18 days, it is hidden in the building's overall consumption, thus, it is difficult to detect by building administrators without SBS.

Discussion

SBS is a practical method for mining device traces, uncovering hidden relationships and abnormal behavior. In this paper, we validate the efficacy of SBS using the sensor metadata (i.e. device types and location), however, these tags are not needed by SBS to uncover devices relationships. Furthermore, SBS requires no prior knowledge about the building and deploying our tool to other buildings requires no human intervention – neither extra sensors nor a training dataset is needed.

SBS is a best effort approach that takes advantage of all the existing building sensors. For example, our experiments revealed that SBS indirectly uncovers building occupancy through device use (e.g. the elevator in the Building 2). The proposed method would benefit from existing sensors that monitor room occupancy as well (e.g. those deployed in [**agarwal:ipsn2011**, **erickson:ipsn2011**]). Savings opportunities are also observable with a minimum of 2 monitored devices and building energy consumption can be better understood after using SBS.

SBS constructs a model for normal inter-device behavior by looking at the usage patterns over time, thus, we run the risk that a device that constantly misbehaves is labeled as normal. Nevertheless, building operators are able to quickly identify such perpetual anomalies by validating the clusters of correlated devices uncovered by SBS. The inspection of these clusters is effortless compare to the investigation of the numerous raw traces. Although this kind of scenario is possible it was not observed in our experiments.

In this paper, we analyze only the data at medium frequencies, however, we observe that data at the high frequencies and residual data (Figure 7.5) also permits us to determine the device type. This information is valuable to automatically retrieve and validate device labels – a major challenge in building metadata management.

This paper aims to establish a methodology to identify abnormalities in device power traces and inter-device usage patterns. In addition, we are planning to apply this method to online detection using, for example, a sliding window to compute an adaptive reference matrix that evolve in time. However, designing such system raises new challenges that are left for future work.

Conclusions

The goal of this article is to assist building administrators in identifying misbehaving devices in large building sensor deployments. We proposed an unsupervised method to systematically detect abnormal energy consumption in buildings: the Strip, Bind, and Search (SBS) method. SBS uncovers inter-device usage patterns by striping dominant trends off the devices energy-consumption trace. Then, it monitors device usage and reports devices that deviate from the norm. Our main contribution is to develop an unsupervised technique to uncover the true inter-device relationships that are hidden by noise and dominant trends inherent to the sensor data. SBS is used on two sets of traces captured from two buildings with fundamentally different infrastructures. The abnormal consumption identified in these two buildings are mainly energy waste. The most important one is an instance of a competing heater and cooler that caused the heater to waste around 2500 kWh.

Related work

The research community has addressed the detection of abnormal energy-consumption in buildings in numerous ways [**katipamula:1review2005**, **katipamula:2review2005**].

The rule-based techniques rely on a priori knowledge, they assert the sustainability of a system by identifying a set of undesired behaviors. Using a hierarchical set of rules, Schein et al. propose a method to diagnose HVAC systems [**schein:hvacr2006**]. In comparison, state machine models take advantage of historical training data and domain knowledge to learn the states and transitions of a system. The transitions are based on measured stimuli identified through a domain expertise. State machines can model the operation of HVAC systems [**patnaik:toist2011**] and permit to predict or detect the abnormal behavior of HVAC's components [**bellala:buildsys2012**]. However, the deployment of these methods require expert knowledge and are mostly applied to HVAC systems.

In [**seem:energybldg2007**], the authors propose a simple unsupervised approach to monitor the average and peak daily consumption of a building and uncover outlier, nevertheless, the misbehaving devices are left unidentified.

Using regression analysis and weather variables the devices energy-consumption is predicted and abnormal usage is highlighted. The authors of [**brown:buildperf2012**] use kernel regression to forecast device consumption and devices that behave differently from the predictions are reported as anomalous. Regression models are also used with performances indices to monitor the HVAC's components and identify inefficiencies [**zhou:wiley2009**]. The implementation of these approaches in real situations is difficult, since it requires a training dataset and non-trivial parameter tuning.

Similar to our approach, previous studies identify abnormal energy-consumption using frequency analysis and unsupervised anomaly detection methods. The device's consumption is decomposed using Fourier transform and outlier values are detected using clustering techniques [**Bellala˙buildsys11**, **wrinch:pes2012**, **chen:aaaiw2011**]. However, these methods assume a constant periodicity in the data and this causes many false positives in alarm reporting. We do not make any assumption about the device usage schedule. We only observe and model device relationships. We take advantage of a recent frequency analysis technique that enables us uncover the inter-device relationships [**romain:iotapp12**]. The identified anomalies correspond to devices that deviate from their normal relationship to other devices.

Reducing a building's energy consumption has also received a lot of attention from the research community. The most promising techniques are based on occupancy model predictions as they ensure that empty rooms are not over conditioned needlessly. Room occupancy is usually monitored through sensor networks [**agarwal:ipsn2011**, **erickson:ipsn2011**] or the computer network traffic [**kim:buildsys2010**]. These approaches are highly effective for buildings that have rarely-occupied rooms (e.g. conference room) and studies show that such approaches can achieve up to 42% annual energy saving. SBS is fundamentally different from these approaches. SBS identifies the abnormal usage of any devices rather than optimizing the normal usage of specific devices. Nevertheless, the two approaches are complementary and energy-efficient buildings should take advantage of the synergy between them.

7.2 Visualization and Analysis of Streaming Data

7.3 Energy Audition with Mobile Phones

Introduction

The United States leads the world in per-capita energy consumption. Our electricity use has consistently increased over the last 40 years [**oecd2011**] and other parts of the world are rising all too rapidly. With the specter of climate change and the increasing cost of energy, we must explore new ways for individuals to gain visibility and insight into their energy consumption in order to optimize and reduce it. With the increasing penetration of embedded sensors in the environment and the continued rise in smartphone adoption, we see an opportunity for smartphones to bridge the physical world to our computational infrastructure and provide an 'energy lens' on the physical world.

We use mobile phones to construct an entity-relationship graph of the physical world and combine it with streaming sensor data in order to perform detailed energy-attribution. We limit the scope of the world to a single building domain. We have designed and implemented a real-time, mobile energy auditing application, called the 'Energy Lens', that allows us to collect information about things throughout the building and how they are related to each other. For example, computer X is inside room Y and connected to meter Z. Then, we use these relationships to guide our data look-up and analytical calculations. For example, the load curve of room Y consists of the sum of all the power traces for loads inside room Y. We use the mobile smartphone as the main input tool. Our work examines *three main challenges* in setting up and deploying a real, whole-building infrastructure to support real-time, fined grained energy analytics.

The first challenge is related to tracking and mobility. The use of mobile phones presents classical, fundamental challenges related to mobility. Typically, mobility refers to the phone, as the person carrying it moves from place to place. However, in the energy-attribution context, we are also referring to the movement of energy-consuming objects. Tracking their relationships to spaces and people is as important as tracking people. We describe how we deal with *both moving people and moving objects* and show that these historically difficult problems can be addressed relatively easily, if the proper infrastructure is in place.

The second challenge is about capturing the inter-relationship semantics and having these inform our analytics. We adopt the general notion of physical tags that identify objects in the world. Our system uses *QR codes* to tag things and locations in the physical world. However, *any tag that provides a unqiue identifier for an object could serve the same purpose.* Once tagged, there are three types of interactions – registration, linking, and scanning – which establish important relationships. Registration is the act of creating a virtual object to represent a physical one. Linking captures the relationship between pairs of objects.

Scanning is the act of performing an item-lookup. Each of these interactions requires a set of swiping gestures. Linking requires two tag swipes while the other two actions require a single tag swipe. Internally, we maintain a *entity-relationship graph (ERG)* of things, people, and locations, that gets updated through these sets of gestures.

The third challenge is about indoor network connectivity and access. In order to connect these components, we rely on having 'ubiquitous' network connectivity. However, in practice, network *availability* is intermittent and our system must deal with the challenges of intermittency. We discuss how caching and logging are used to address these challenges. Moreover, when connectivity is re-established, we must deal with applying updates to the ERG, as captured by the phone while disconnected.

Motivating scenario

Imagine having the ability to walk through a building and see live, detailed energy data as you point your phone at various things and locations. As you enter the building, you scan its tag and see the live breakdown of energy consumption traces, including HVAC, lighting, and plug-loads. You continue your walk through the building as you head to your office. When you arrive to your floor you scan the tag for the floor and observe similar figures, only this time they are in relation to that floor alone. Since there are several meeting rooms on that floor, you are curious how much is consumed by occupants versus visitors. You choose to view the total load curve co-plotted with the occupant load curve, specifically for that floor. You see that approximately half the total energy is consumed by visitors during the day.

Curious about what portion of total are attributed to you, you select the personalized attribution option and you see *your* personal load curve plotted with the total load curve – as well as accompanying statistics, such as the percent of total over time. As you quickly examine the data on your phone, you see that you consumed energy during hours that you were not there. You choose to see a more detailed breakdown. You enter your office, scan various items that you own, and see that your computer did not shut down properly and your light switch was set to manual. You immediately correct these.

Being able to interact with your environment and get a complete energy break-down can provide a useful tool for tracing and correcting rampant energy consumption. In buildings, having the occupants actively participate allows for localized, personal solutions to efficiency management and is crucial to scaling to large buildings. However, providing this detailed level of attribution is challenging. There's lots of data coming from various systems in the building, and integrating them in real time is difficult. Furthermore, attribution is non-trivial. We must be able to answer to following: How much of the total consumed on this floor went to charging laptops? How many of those charging laptops belong to registered occupants of this floor? For centralized systems, multiple locations are served simultaneously. It is non-trivial to determine the exact break-down for each location. At the plug-load level, some plug loads move from place to place throughout the building over the course of the day. Tracking where they are at any given time is difficult.

Answering these queries is relatively easy once the information is available, however, collecting the information is non-trivial, especially over time. Historically, it has been difficult to collect plug-load information. Various studies have used wireless power meters to accomplish just this [**stephscale**, **lanz**, **aceee**]. All previous work collected the data and performed post-processing to analyze it. We want to take the next natural set of steps: perform processing in real-time and present the occupants with live information.

There are several systems challenges that must be overcome in order to achieve this vision. We examine those dealing capturing and maintaining a view of the physical world. We must record which things belong to whom, where people and things are over time, how to deal with the mobile phone as the main interactive modality, and how to do this at the scale of hundreds to thousands of users and meters. We argue that *without addressing these systems issues, this vision cannot be achieved.* In our work, we start by deploying a network of wireless power meters and use the mobile phone to re-create a model of people, things, and locations in the building. We also use it to assist in tracking of people and things over time.

Related Work

Our work touches on several areas from logisticsto context-aware mobile applications [**ACE**]. In the building space, there has been some interest in building various kinds of energy-related visualization and control applications. HBCI [**hbci**] proposes a high level archiecture that also relies on QR codes, mobile phones, and ubiquitous network access. HBCI introduces the notion of object capture through the mobile phone and individual services provides by the object, accessible via an object lookup. The proposed service model is *object-centric*, such as individual power traces or direct control accesss. Their "query" service is a tag lookup mechanism realized through QR code scanning of items. The 'Energy Lens' also embodies the "query" via a tag lookup, however we focus on context-related services rather than object-centric services. We build and maintain an entity-relationship graph (ERG) to capture the inter-relationships between items. The ERG informs our analytical processing. We use an eventual-consistency model to maintain the inter-relationship graph over time. HBCI does not address the challenges faced in realizing an indoor, interactive application that relies on ubiquitous network connectivity. Our architecture directly addresses this challenge, as we observe that indoor connectivity characteristics do not comply with the ubiquitous connectivity requirement for this class of application.

Challenge 1: Tracking People and Things

The Energy Lens app consists of a web application that displays timeseries data and an Android-based smartphone app. The android app is relatively simple; consisting of a menu with only two options: Update deployment state, scan to view services. Swipe gestures manipualte a local portion of the entity-relationship graph – local with respect to a user's current location. Since each location (room, floor) has a QR code atextttached to it and items are associated with those locations, we can identify the location by name (/buildings/SDH/spaces/4F/toaster).

The first set is called a 'registration' swipe and we use it to register new items. The user scans a QR code and the item it is atextttached to. This creates an 'atextttached-to' link between them. Adding, removing, binding, and atextttaching items is done with a pair of swipes. A lookup is done with by swiping the QR code atextttached to an item.

We have designed a set of heuristics for setexttting the location during an update, that piggybacks on the swipe gesture. The following is a list of rules for automatically setexttting the location of people and things:

- When a user swipes at a location L, they are presumed to be at L for fixed period τ. An "association timer" is set to release this association after τ seconds.
- If the user swipes anything that is associated with a location l at time $t \leq \tau$, and $l(t) \neq$ L, then we set the new location of the *thing* they swiped to l(t) and reset the association timer.
- If the user swipes anything at location l at time $t \geq \tau$, we set the location of the *person* to l(t). We reset the association timer to τ.
- If a user registers a new location, they are presumed to be at that location.

For each of these, we provide an interactive option to ask for location-change confirmation from the user. So if we think the user/item has moved but they have not, the preset action can be overridden. The guiding principal we follow in our design is to leverage the swipe gesture for as much contextual information as possible. Furthermore, we do not explicitly track users. Context is only set on the phone and used in operations sent to the server.

We construct the entity relationship graph through naming in StreamFS. StreamFS uses filesystem constructs, such as symbolic links and hierarchical naming which are useful for expressing an acyclic graph structure (StreamFS checks for cycles when symlinks are created). Registered meters are placed in the device path, /dev. Items are stored in /inventory. QR codes are stored in /qrc. When an item is registered a symbolic link is created from the specific qr code directory to the item. /spaces contains a hierarchy of floors, rooms, and sub-spaces. /users contains the list of usernames. We also have a /tax directory, where we construct an device hierarchy for access by plug-load category. Placement (location) is also captured with symbolic links.

Challenge 2: Consistency Management

We use an eventual-consistency model for maintaining the ERG over time. Naturally, the spatial inter-relationships change over time as items are moved and replaced. In order to deal with this we offer two options: 1) we periodically re-scan the items and their locations, essentially re-capturing the inventory collection portion of the audit or 2) we allow building occupants to participate as auditors, capturing their own personal items and shared items. This should provide at least as much value as a periodic energy audit and can be completed in a fraction of the time [**aceee˙mobileaudit**].

Challenge 3: Disconnected Operation

Although connectivity is ubiquitous, network access is not. This occurs due to dead zones, idle-disconnect and failed hand-off between access points. When encountered in practice, especially while editing deployment state, it can be quite frustrating and discourage use of the application. We designed a mechanism that does smart caching, not only to improve performance, but also to allow for disconnected operation.

The process is demonstrated in Figure 7.9. The components shown are the *ERG cache*, the *operation log (OpLog)*, and the *prefetcher*. We separate the steps in the figure as a READ sequence and a WRITE sequence. All reads go to the cache (steps 1 and 2 on the left hand side of the figure). Writes go through the OpLog (steps 1 - 5 on the right side of the figure). For writes, the application makes a write request (1) and it is forwarded to StreamFS (2). If StreamFS is reachable and the write is successful (3), the operation is applied to the ERG cache (4) and the response is sent the application (5). If the operation is not successfuly, step 4 is skipped. If StreamFS could not be reached, step 3 is skipped, and the operation is writexttten to the OpLog. The OpLog is flushed to the server, by the prefetcher, upon re-connection.

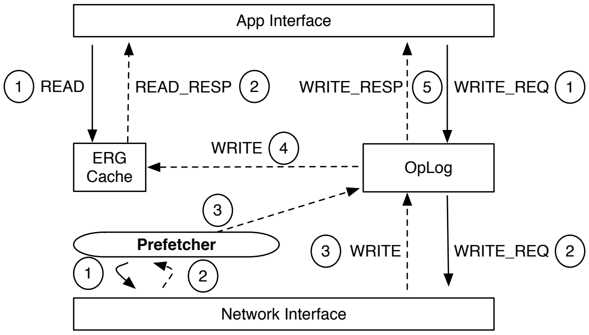


Figure 7.9: Standard mechanisms for consistency management on the phone. All READ request go to the local cached version of the ERG. All WRITES must go through the OpLog.

Deployment Experience

We deployed 20 ACme power meters [**acme**] on a single floor of a building on campus. The data was made available through sMAP [**smap**] and forwarded to our processing and data management layer, StreamFS [**streamfs**]. We distributed the ACmes throughout a single floor in our building and registered various plug loads as being measured by them. We also tagged hundreds of items and locations throughout the entire building. In total we tagged 20 meters, 20 metered items, 351 un-metered items, and 139 rooms over 7 floors. Figure 7.10 shows three screen shots of power traces obtained from the ACme deployment, and displayed through the Energy Lens.

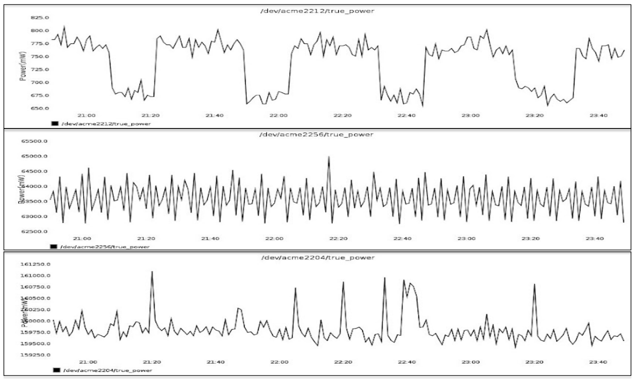


Figure 7.10: Power traces obtained from power meters atextttached to various plug load on one of the floors of a building on campus. These show screen shots of the Energy Lens timerseries data display.

In our initial deployment we found that the use of our tracking scheme to be effective, especially in conjunction with interactive confirmation. The ERG was effective at capturing deployment state, although highly mobile items, such as laptops, were particularly difficult to keep track of. Finally, our disconnected operation mechanism was effective at masking intermitexttttent connectivity. For future work we will baseline the floor's energy consumption before and after the deployment and measure if more visibility and analytics indeed motivates occupants to use less.

Evaluation

In this section we measure prefetch download times and discuss strategies for providing scalability. We also look at the transaction manager and discuss conflict resolution.

Sensing and tag layer

We deployed 20 ACme power meters [**acme**] on a single floor of a building on campus. The data was made available through sMAP [**smap**] and forwarded to our processing and data management layer, StreamFS [**streamfs**]. We distributed the ACmes throughout a single floor in our building and registered various plug loads as being measured by them. We also tagged hundreds of items and locations throughout the entire building. In total we tagged 20 meters, 20 metered items, 351 un-metered items, and 139 rooms over 7 floors.

No. nodes	**F**etch time (sec)	**S**td. Error (sec)
1	0.8902	0.0756
10	5.7342	1.7087
100	52.3145	14.1146

Table 7.2: Shows the time to fetch nodes based on the size of the fetch. The fetch time increased linearly with the number of nodes. Caching maintain fetch time near that of fetching a single node. A callback is used when cache is invalidated.

Prefetching

Prefetching occurs when a user enters a new floor, as detected by a floor scan or an item scan. Table 7.2 shows that the prefetch times scale linearly with the number of items (and data) to prefetch. Each node holds approximate 100 bytes or information and for a 20-node deployment of power meters, producing 100 bytes of data per stream (three streams per ACme) every 20 seconds, we fetch approximately 1 MB of data.

These prefetch times are non-trivial to deal with, especially since they cause the phone application to slow down until the data is received and loaded into the local cache. The overhead is dominated by the query in StreamFS that constructs the entire sub graph to send to the application. For future work, we are will implement a callback facility and pass the application a reference to it. The app can then periodically check back until the query completes and the data is ready to be downloaded. We can also include partial responses to the query in the prefetch-loop response. This also allows users to continue using the application without any frustrating waits.

Log replay latency

Table 7.3 shows the operations that the transaction manager calls on the StreamFS server. Log replay and transaction processing is entirely dependent on the time to execute these operations on StreamFS. There are five types of transactions, a *move*, a *un/bind*, *un/attach*. A move is a combination of a 'delete' and a 'create link', a bind is a 'create link' and an 'update tags', an unbind and unattach is a 'delete' and 'update tag'. The transaction latency is the sum of these operations. By far the most expensive operation is a 'create node' operation. This occurs when a user adds a new item/space/person to the graph. The time to apply the operation also scales linearly with the size of the logs.

All logs dumps are processed sequentially. However, for future work we look to parallelize processing into parallel processes updating different portions of the graph. For example, log updates rooted at different floors could occur simultaneously.

Operation	**A**vg. exec. time (ms)
fetch	250
delete	326
update tags	267
create link	250
create node	1036

Table 7.3: Average operation execution time in StreamFS.

Future work and conclusion

In this paper we examined two system challenges – mobility and consistency management – for enabling and energy analytics applications in buildings. We also offered initial solution approaches. We see these as crucial barriers to solve in order to provide the kinds of services described in Section 7.3. However, other challenges remain, particularly those related to scaling to an entire buildings, integrating many more streaming data sources, and providing streaming analytics for immediate display to building occupants. We see an opportunity to combine these with control in order to empower building occupants to literally take control of their energy footprint. The components of our architecture are simple, and simplicity is important for scale and generalizability. We hope that with the right tools and information, people will be motivated to act, and large energy waste reduction can be achieved.

Chapter 8

Lessons Learned and Future Work

Chapter 9

Conclusion