ThermoCast: A Cyber-Physical Forecasting Model for Data Centers

Lei Li
Carnegie Mellon University
leili@cs.cmu.edu

Chieh-Jan Mike Liang
Microsoft Research Asia
liang.mike@microsoft.com

Jie Liu
Microsoft Research
jie.liu@microsoft.com

Suman Nath
Microsoft Research
suman.nath@microsoft.com

Andreas Terzis
Johns Hopkins University
terzis@cs.jhu.edu

Christos Faloutsos
Carnegie Mellon University
christos@cs.cmu.edu

ABSTRACT

Efficient thermal management is important in modern data centers as cooling consumes up to 50% of the total energy. Unlike previous work, we consider *proactive* thermal management, whereby servers can predict potential overheating events due to dynamics in data center configuration and workload, giving operators enough time to react. However, such forecasting is very challenging due to data center scales and complexity. Moreover, such a physical system is influenced by cyber effects, including workload scheduling in servers. We propose ThermoCast, a novel thermal forecasting model to predict the temperatures surrounding the servers in a data center, based on continuous streams of temperature and airflow measurements. Our approach is (a) capable of capturing cyber-physical interactions and automatically learning them from data; (b) computationally and physically scalable to data center scales; (c) able to provide online prediction with real-time sensor measurements. The paper's main contributions are: (i) We provide a systematic approach to integrate physical laws and sensor observations in a data center; (ii) We provide an algorithm that uses sensor data to learn the parameters of a data center's cyber-physical system. In turn, this ability enables us to reduce model complexity compared to full-fledged fluid dynamics models, while maintaining forecast accuracy; (iii) Unlike previous simulation-based studies, we perform experiments in a production data center. Using real data traces, we show that ThermoCast forecasts temperature 2× better than a machine learning approach solely driven by data, and can successfully predict thermal alarms 4.2 minutes ahead of time.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data mining; C.4 [**Performance of Systems**]: Measurement techniques;Modeling techniques

General Terms

Algorithms, Experimentation, Measurement, Performance

KDD'11, August 21–24, 2011, San Diego, California, USA.

Keywords

data center energy efficiency, time series forecasting, cyber physical modeling

1. INTRODUCTION

A modern data center hosts tens of thousands of servers used to provide reliable and scalable infrastructure for Internet-scale services. The enormous amount (in the order of tens of megawatts) of energy these facilities consume and the resulting operational costs have spurred interest in improving their efficiency.

Traditional data centers are over-provisioned; server rooms (usually called colos) are excessively cooled and the average server utilization is kept quite low (e.g., CPU utilization between 10% to 30%). As a consequence, a "well tuned" data center rarely has thermal alarms and it is sufficient to use *reactive* thermal management, where data center operators take necessary actions only after an over-heated server issues a protective shutdown. However, such a conservative approach leads to waste of computational resources and poor Power Utilization Efficiency (PUE) [1] (close to 2, with ≈ 40% of total data center energy used for cooling).

With increasing demand for improving data center efficiency, data center operators look into many ways to reduce cooling cost and increase server utilization. For example, a previous study confirms that fans consume most of the energy used by a Computer Room Air-Conditioning(CRAC) system [19]. A single Liebert Deluxe System/3 CRAC installed in our data center has three 7.57 kW·h fans for a total energy consumption of 22.71kW·h [17]. Since the power that fan motors consume increases with the *cube* of fan rotation speed [19], modern data centers use variable speed fans in order to reduce the CRAC's energy use: a mere 10% reduction in fan speed translates to 27% energy savings for the fan motor. Other energy-saving approaches taken by modern data centers include raising AC temperature set points, using outside air directly for cooling, consolidating workload using virtual machines, and leveraging statistical multiplexing to opportunistically oversubscribe the servers. However, as a result of such aggressive optimizations, the safety margin of data center operation is getting smaller. This trend requires data center monitoring to move from reactive to *proactive*, whereby the servers can predict potential overheating events early enough, giving operators enough time to react.

Central to any proactive thermal management approach is predicting temperature of different servers in a data center. This is extremely challenging due to large scale (a data center usually con-

[1] PUE is defined as the ratio between total facility energy consumption and the energy used by servers.

tains tens of thousands of servers, multiple CRAC units and fans), complex thermal interactions (e.g., due to server fans driving local air flow, by-pass air through gaps between servers and racks), and cyber effects (e.g., workload scheduling algorithms may have visible effects on temperature distribution). Previous works have considered two different approaches for data center thermal management. Thermodynamics-based solutions derive thermal models of different locations inside a data center using fundamental thermodynamics laws and data center layout [1, 21, 26, 31]. On the other hand, data-centric solutions use data-mining [27] or machine learning algorithms [22] to model and optimize cooling in a data center. All these existing solutions, however, provide static thermal models and are not adaptive to changes in workloads, CRAC fan speeds, data center layout, etc. Thus, these solutions are not adequate for modern data centers serving dynamic workload [4] or using power-efficient variable-speed fans.

In this paper we propose ThermoCast, a novel thermal forecasting model that addresses the above limitation. ThermoCast uses real-time workload information and measurements from a carefully deployed set of temperature and air-flow sensors to model and predict temperatures around servers. We assume that each server knows temperature of its cold inlet air and hot exhaust air. These data can be obtained from temperature sensors shipped with some servers, or a RACNet-like data center sensor network [16]. Two big challenges in building any such real-time adaptive model are *predictability* and *scalability*: the model should be able to predict overheating early enough, without many false positives/negatives, even when system configuration (e.g., workload, fan speed) changes and it should be able to handle millions of data points to monitor within a data center. Reducing false positives/negatives is important to reduce burden on human operators who must take some action following an alarm and predicting early enough is important to give operators enough time to react.

To achieve high predictability, ThermoCast uses a hybrid approach of aforementioned thermodynamics-based and data-centric approaches. ThermoCast is based on thermodynamics laws and cyber-physical interactions, however, it learns and adapts appropriate values of various parameters from real-time sensor data and workload information. Thus, it is able to provide online prediction even when configurations, such as servers' on/off state, workload, set of servers, air-conditioning equipment maintenance, change.

To achieve scalability, we use the insight that temperature around a server is affected mostly by configurations of its neighboring servers and not much by the servers far from it. Therefore, ThermoCast is based on a zonal thermal model that builds a relationship among the cold-aisle vent temperature, the location of the server, the local temperature distribution and the workload from nearby servers, to predict the intake temperature at each server. Because of such local nature, ThermoCast can distribute the modeling task among servers: each server learns and models the temperature around itself by using nearby sensor measurements and workloads of neighboring servers. Thus, ThermoCast is computationally and physically scalable for a large data center.

We have deployed and evaluated ThermoCast in a lab data center with a rack of 40 servers. Through dense data center instrumentation, we show the complex thermal dynamics with variable workload and CRAC activities. Our experiments show that ThermoCast is more effective than pure machine learning approach with better prediction accuracy and mean lookahead time. For example, with real data traces, we show that ThermoCast can predict thermal spikes 4.2 minutes ahead of time, comparing to 2.3 minutes using an auto-regression (AR) model. The extra two minutes can be crucial for thermal management. Previous studies have shown that it takes about a minute to safely suspend a virtual machine in cloud computing environment [33, 30]. For connection intensive servers, like Windows Live Messenger, a minute can safely drain 7% of total TCP connections [4].

In summary, we make the following contributions in the paper:

1. We provide a systematic approach to integrate the physical laws and sensor observations in a data center.

2. We provide an algorithm to learn from sensor data for such cyber-physical system, and it enables us to reduce complexity in full fluid models while still achieves good forecasting of future temperatures.

3. Unlike previous simulation-based studies, we perform experiments in a production data center. Using real data traces, we show that ThermoCast can forecast temperatures 2× better than the pure machine learning approach, and can successfully predict thermal alarms on average in 4.2 minutes ahead.

The rest of the paper is organized as follows. We review the literature in Section 2 and summarize the operation and energy cost of data center cooling using air conditioners in Section 3.1. Section 3 presents our findings about how temperature inside a data center changes as a function of server load and AC activity. Section 4 describes the proposed ThermoCast framework, while Section 5 presents evaluation results. We conclude in Section 6.

2. RELATED WORK

Our work is related to two areas of interest, thermal management in data centers, time series mining and prediction.

Data center thermal management. A number of recent papers have investigated methods for efficient thermal management in a data center. The methods can be broadly divided into two categories. The first category of solutions are based on fundamental thermal and air dynamics laws using computational fluid dynamic (CFD) simulators [1, 21, 26, 28, 31]. These solutions derive thermal models of different locations inside the data center during an initial profiling phase using data center layout and material thermo properties. The models are subsequently used by various energy-optimizing tasks. Cooling-aware workload placement algorithms [1, 21, 26, 31] use such models to place heavy computational workload in cooling-efficient locations. Energy-aware control algorithms [28] use such models to choose the best dynamics voltage and frequency scaling (DVFS) policy for each server to match its workload. Spatio-temporal scheduling algorithms [24] use the models with virtualization to improve cooling efficiency. Our work differs from these existing work in two important ways. First, rather than open-loop CFD models, ThermoCast is based on both thermodynamics laws and real-time measurements, and unlike previous solutions, it can adapt with dynamics in workload, fan speed, etc. Second, our focus is on predicting hot spots early enough, giving data center operators enough time to react. This requires ThermoCast to be scalable and predictable.

The second category of data center thermal management solutions use black-box data-driven approaches. Patnaik et al. [27] has proposed a temporal data mining solution to model and optimize performance of data center chillers, a key component of the cooling infrastructure. [22] proposed a thermal mapping prediction problem that learns the thermal map of a data center for different combinations of workload, cooling configurations, and physical topologies. The paper uses neural networks to learn this mapping from data derived from thermodynamic simulations of a data center. This model is then used for workload placement. This approach avoids the challenges of using thermodynamics to estimate server temperature through a data-driven approach that is amenable to online scheduling of computing workloads. However, the neural network is not dynamic and therefore temperature predictions might be affected by scheduling dynamics and lead to scheduling oscillations.

Mercury software suite [9] emulates single server temperatures based on utilization, heat flow, and air flow information. Mercury is then used by Freon, a system for managing thermal emergencies. Unlike Mercury and Freon, ThermoCast models the thermal relationship among nearby servers, which can be used to optimize computation and cooling.

Finally, work that proposes to improve data center energy efficiency through the use of low-power CPUs ([8]), smart cooling ([25]), and power-efficient networking ([10]) is orthogonal to our work that provides methods to improve the efficiency of existing infrastructure through data-driven thermal modeling and thermal-aware dynamic workload placement.

Time series mining and prediction. Since our data is collected from distributed sensors (temperature and airflow) in an online fashion. Our work also falls into the category of time series prediction. Autoregressive moving average (ARMA) are a standard family of models for time series analysis and forecasting (Box and Jenkins [2]), and are discussed in every textbook in time series analysis and forecasting (e.g., [3]). Kalman filters and state-space models are also previously used in mining motion capture sequences and sensor data [32]. We use AR model as a baseline in our experiments.

In this paper, we assume that we can obtain all sensor data. But one of the challenge in sensor data is the missing observations partly due to unreliability of wireless transmission. Li et al [14] proposed DynaMMo method to learn a linear dynamical system in presence of missing values and fill in them. Their method could then use the learned latent variables to better compress the long time sequences. Our system can leverage such approaches.

Remotely related is time series indexing, segmentation, classification [15] and anomaly detection [13]. A common approach for indexing time series is extracting a few features from time sequences and matching them based on the features [7], such as the Fourier transform coefficients, wavelet coefficients (Jahangiri et al. [11]), and local dimensionality reduction (Keogh et al. [12]). However, time series indexing does not offer predictability, which is key in data center management scenarios.

3. MOTIVATION

3.1 Background

To understand the challenges of thermal prediction, we overview the operation of a typical data center including its cooling systems.

There are many data center architectures, from ad hoc server cabinets to dedicated containers. However, most enterprise and Internet data centers use a cold-aisle, hot-aisle cooling design. Figure 1 illustrates the cross section of a data center server room that follows this design. Server racks are installed on a raised floor in aisles. Cool air is blown by the CRAC (Computer Room Air Conditioning) system to the sub-floor. Perforated floor tiles act as vents, making cool air available to the servers. The aisles with these vents are called *cold aisles*. Typically, servers in the racks draw cool air from the front and blow hot exhaust air to the back in *hot aisles*. To effectively use the cool air, servers are arranged face to face in cold aisles. As Figure 1 suggests, cool and hot air eventually mix near the ceiling, and this return air is drawn back into the CRAC.

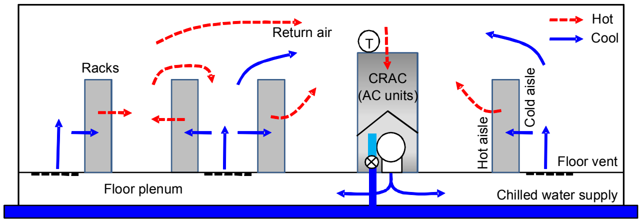


Figure 1: An illustration of the cross section of a data center. Cold air is blown from floor vents in cold aisles and hot air rises in hot aisles. Mixed air eventually returns to the CRAC where the chilled water cools the air.

In its simplest form, a CRAC consists of two parts: a heat exchange unit and one or more fans. To handle the large cooling capacity requirement of a data center, the heat exchange unit typically uses a chilled water-based design. Specifically, the CRAC is connected to water chillers outside of the facility with circulation pipes. These pipes deliver chilled water with which the return air exchanges heat inside the CRAC. The warm water then circulates back to the outside water chillers. Finally, the cooled air is blown by the CRAC's fans to the floor vents. To reduce the energy consumption of the cooling equipment, many CRACs offer adjustable cooling capacity by adjusting the chilled water valve opening and the fan speed according to the return air temperature reported by the temperature sensor at the CRAC's air intake [18].

3.2 Data Center Sensor Instrumentation

Liu et al. argued about the benefits of using wireless sensor networks (WSNs) for data center monitoring including the ease of deployment in existing facilities with minimal infrastructure requirements [20]. In our case, using a WSN to measure temperature and airflow speeds across a data center allows us to quickly reconfigure the measurement harness as we vary measurement locations across different experiments.

We deployed a network of 80 sensor nodes at an university data center hosting a high-performance scientific computing cluster. The cluster consists of 171 1U [2] compute nodes with eight CPU cores each connected to two file servers through a low-latency InfiniBand switch. The sensor nodes are equipped with low-power 802.15.4 radios and form a multi-hop routing tree rooted at a gateway. The network comprised 15 air flow velocity sensors [6] and 65 humidity/temperature sensors [29].

We used this network to instrument three server racks according to the following sensor configuration. First, a rack is divided into three sections: *top* (i.e., four top most servers), *middle* (i.e., five middle servers) and *bottom* (i.e., four servers closest to the floor). During the experiments we control the load on the server at the middle of each section (termed as the *controlled server*). Servers in all three sections are instrumented with two humidity/temperature sensors: one at the server's air intake grill, facing the cold aisle, and another at the air ventilation grill in the hot aisle. Second, to measure the velocity of the cold air flowing from the floor vent at different heights, we positioned 12 air flow sensors directly above the floor vent at a vertical interval of 5.25", or every 3U (cf. Fig 2). Furthermore, we placed one air flow sensor at the air intake grill of

[2] A *rack unit* or U is a unit of measure used to describe the height of equipment intended for mounting in a 19- or 23-inch rack. One rack unit is 1.75 inches.

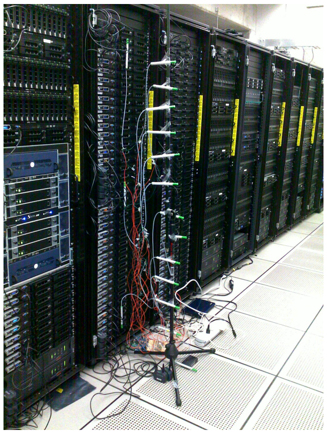


Figure 2: A picture of the air flow sensors setup. We positioned 12 air flow sensors directly above the floor vent at a vertical interval of 5.25", or every 3U.

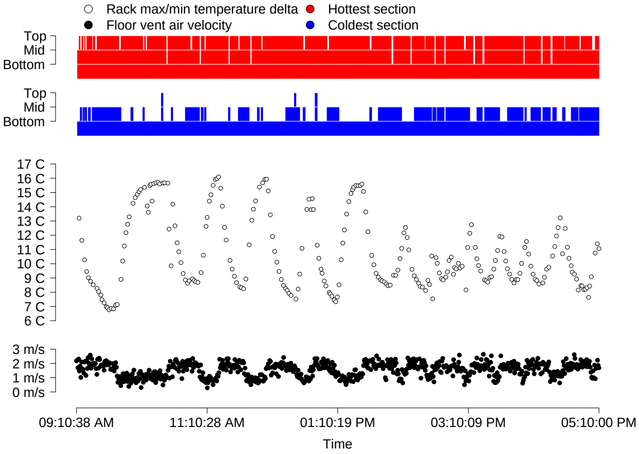


Figure 3: The relation between the cold air velocity from the floor vent and the server intake air temperatures of a single rack.

each controlled server. Finally, we used the servers' built-in monitoring facilities to monitor their CPU load, fan speeds, and power consumption.

3.3 Observations

This section presents insights derived from the WSN measurements which provide both the motivation and the intuition to the control framework presented in Section 4.

Figure 3 shows the relation between the cold air velocity from the floor vent and the range of server intake temperatures across a single rack. We make two observations from this figure.

First, the temperature difference cycle (termed as the *contraction and relaxation* cycle) is in antiphase with the air velocity cycle. In other words, the temperature variation of a rack is smallest when the air velocity is highest. When the air velocity is low, the air is colder closer to the floor vent but less cool air is available at the top of the rack. Hence, the high and low temperature at the top and bottom section respectively are significantly different. At high air velocities, the top section cools down as cold air is forced further up, but the temperature of the bottom section actually increases due to the Bernoulli effect. The implication of this effect, which dictates that fast moving air creates a decrease in pressure, is that hot air from the back of the rack is drawn to the front of the server as the speed of cold air increases [5]. Therefore, simply increasing the CRAC fan speed can lead to unexpected hotspots.

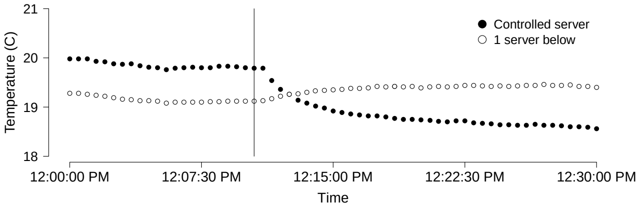


Figure 4: The intake air temperature change of the controlled server decreases after the server is shut down, while the temperature of the server below increases. The vertical line indicates when the controlled server was shut down.

Second, as the floor vent air velocity varies, the coldest section of the rack oscillates between the middle and the bottom section. On the other hand, the top section is almost always the hottest section. In addition to the fact that the CRAC needs to increase the fan speed to deliver cold air to the top section, the top section has a relatively higher initial temperature as it is close to the warm return air flow (cf. Figure 1).

Chen et al. suggested shutting down under-utilized servers to reduce the energy consumption of cooling system [4]. Intuitively, this approach applies well to servers in the top section which we just showed to frequently be the hottest. However, shutting down one server can impact the intake air temperature of its neighbors. Figure 4 illustrates an example of this interaction; shutting down the controlled server causes an increase in the intake air temperature of the server below it. While few servers are affected by the actions of one server, a framework that predicts temperatures should consider these interactions.

4. ThermoCast FRAMEWORK

ThermoCast faces the unique challenge of modeling the interaction between the computing and cooling systems. As we have seen in Section 3, duty cycles of the CRAC system affect the amount of cooling that the different rack sections receive and thus affect the temperature at the servers' intakes. Furthermore, turning a server on or off affects its nearby servers and reaching a new equilibrium can take as long as an hour.

4.1 Federated Modeling Architecture

The scale of mega data centers prevents us to use a centralized approach for model building and prediction. It is hard to even visualize tens of thousands of monitoring points on a screen. In ThermoCast, we use a federated modeling architecture that relies on each server to model its own thermal environment and make predictions. Only when local predictions exceed certain thresholds, the system draws the operators' attention, accumulates more sensor points, and possibly performs another tier of prediction and diagnosis.

The federated modeling architecture in ThermoCast takes advantage of the physical properties of heat generation and propagation. That is, heat diffuses locally and gradually following models of thermo- and fluid dynamics. Although the model parameters can be drastically different depending on the local configuration – rack heights, server locations, server types, on/off states etc. – the model structure remains the same. Based on this insight, we use a "gray-box" approach, in which the model is known but the parameters are unknown, as opposed to "white-box" modeling using CFD, and a completely data-driven "black-box" model such as neural network.

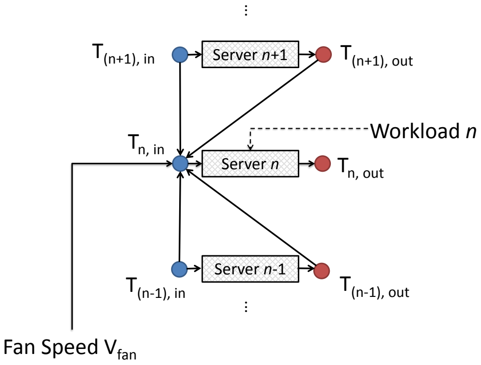


Figure 5: The ThermoCast modeling framework. Circles correspond to model variables, while the arrows indicate relationships among these variables.

Another advantage of the federated architecture is that model learning and prediction can be done in a distributed fashion. Figure 5 shows one section of the graphical model in ThermoCast. First of all, time is discretized into ticks. At every time tick, with step size t_s, server n uses its own intake and exhaust air temperatures, the intake and exhaust air temperatures of its immediate neighbor ($n-1$ and $n+1$), the air speed and temperature at the AC vent, and its own workload to build a model that computes its own intake and exhaust air temperature in the next time tick. The variable dependencies capture the air flow in different directions, as well as local heat generation.

Sensor data can be communicated efficiently in this architecture. If a wireless sensor network is used for monitoring, then each sensor only need to broadcast its value to the local neighbors. If the sensors are on the server chassis, then the data only needs to go through the local top-of-the-rack switch, rather than data center level routers.

4.2 Thermal Modeling

We build a model based on first principles from thermodynamics and fluid mechanics. While a comprehensive computational fluid dynamics model (CFD) is often complex and computationally expensive, we exploit a zonal model for the thermo/air dynamics near the server rack. The intuition is to divide the data center's indoor environment into a coarse grid of zones such that air and thermal conditions can be considered uniform within each zone. We divide the room into zones, as shown in Figure 6, and define the variables shown in Table 1.

We make the following assumptions to simplify the model during a prediction cycle:

A0: Incompressible air, which implies the density of air ρ is constant. We ignore dynamic pressure due to height and temperature differences, and care only about the Bernoulli effect caused by high-speed airflow.

A1: T_{RM}, the room temperature is constant in a short period of time.

A2: T_{FL}, the supply air temperature at the floor vent is constant within a short period of time.

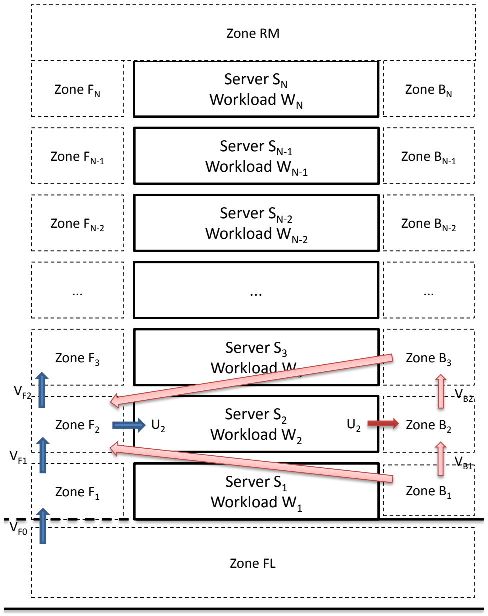


Figure 6: The zonal model for thermo dynamics around a rack.

A3: Constant server fan speed, thus $U_{F_i} = U_{B_i}$.

A4: The vertical air flow at the back of the server is negligible.

A5: The vertical air flow in the front scales linearly with the floor vent speed, although the scaling factor depends on server height and the on/off status of nearby servers. In other words $V_{F_i} = \delta_i V_{FL}$, where δ_i is constant during a short period of time.

We then model the following relationships between model variables.

Basic fluid dynamics: (Bernoulli's principle):

$$P = -\frac{1}{2}\rho\hat{V}^2 \tag{1}$$

where $\hat{V}$ is the total air speed, i.e. $\hat{V_z}^2 = U_z^2 + V_z^2$. Thus, for zone z

$$P_z = -\frac{1}{2}\rho(U_z^2 + V_z^2) \tag{2}$$

Now consider server s with front zone F_s and back zone B_s. By (2), and assumption [A4], $V_{B_s} = 0$

$$P_{F_s} - P_{B_s} \quad = \quad \frac{\rho}{2}(V_{B_s}^2 - V_{V_s}^2) = -\frac{\rho}{2}V_{F_s}^2 \tag{3}$$

This pressure difference drives the hot air to flow from the hot aisle to the cold aisle.

Basic thermodynamics:

Consider zone $z \in \{F_1, ...F_N\}$ of air mass M_z and temperature T_z. During time interval $[t, t+t_s]$, let $\Lambda_{i,z}(t)$ be the air mass flowing into z from zone i with temperature $T_i(t)$, then $\sum_i \Lambda_{i,z}(t)$ is the air flowing out of zone z (due to mass conservation and assumption about incompressible air [A0]) with temperature $T_z(t)$:

$$\begin{aligned} M_z \cdot T_z(t+1) \quad &= \quad M_z \cdot T_z(t) + \sum_i (\Lambda_{i,z}(t) \cdot T_i(t)) \\ &\quad -(\sum_i \Lambda_{i,z}(t)) \cdot T_z(t) \end{aligned} \tag{4}$$

Parameter	Description
$i = 1 \cdots N$	server index in the rack
Zone F_i	area in front of the server i; close enough to get the Bernoulli effect.
Zone S_i	area inside server i
Zone B_i	area immediately behind server i; only impacted by the heat generated by the server
Zone RM	zone for the room ambient air
Zone FL	zone below the vent
T_z	temperature of zone z
V_z	vertical airflow speed out of zone z
U_z	horizontal airflow speed out of zone z
P_z	dynamic air pressure in zone z, i.e. "measurable" pressure minus atmospheric pressure
W_s	Watts generated by server s, representing its workload
ρ	air density

Table 1: ThermoCast parameters and their description.

The air mass in exchange per unit time is proportional to air speed. So,

$$\Lambda_{i,z}(t) = \beta_{i,z}\sqrt{(P_i(t) - P_z(t))} \qquad (5)$$

where $\beta_{i,z}$ captures air density (ρ) and all geometric characteristics between i and z, such as gap size, server type, and server on/off etc., i.e. how hard it is to push air from i to z.

So, by (4) and (5)

$$T_z(t+1) = (1-\alpha_z)\cdot T_z(t) + \sum_i (\beta_{i,z}(P_i(t) - P_z(t)) \cdot T_i(t)) \quad (6)$$

where α_z captures the flowing out of the zone, including those going into the server and those going up/down to the next zone. Clearly, α_z depends on height, fan speed, and server on/off.

Plugging in (3) and using assumption [A5], we derive the following structure of the local thermo-dynamics model:

$$\begin{aligned} T_z(t+1) &= a \cdot T_z(t) \\ &+ \sum_{j \in \{B_{z-1}, B_z, B_{z+1}\}} (\beta_j \cdot (P_z(t) - P_j(t)) \cdot T_j(t)) \\ &+ \sum_{j \in \{F_{z-1}, F_{z+1}\}} (c_j \cdot V_{FL} \cdot T_j(t)) \qquad (7) \end{aligned}$$

where parameters $\{a, \beta_j\text{'s, and } c_j\text{'s}\}$ are server and location dependent and are learned by each server through parameter estimation.

Including Workload For each server s, the workload W_s converts into heat and effects the temperature at the back of the server. So for zone B_s, we have:

$$\begin{aligned} T_{B_s}(t+1) &= f_1 \cdot T_{B_s}(t) + f_2 \cdot T_{F_s}(t) + \\ & \quad f_3 \cdot U_s(t) \cdot W_s(t) + f_4 \cdot T_{B_{s-1}}(t) \qquad (8) \end{aligned}$$

When the server is on, the horizonal mass exchange from front zone F_z to back zone B_z in Eq. (5) is dependent on the server's fan speed ($= U_z$). We also assume the interaction between the server's intake and its neighbor's outtake is indirect, thus eliminating the corresponding terms in Eq. (7). Therefore, with such reasoning and the result of Eq (7), the workload dependent equation for intake temperature becomes,

$$\begin{aligned} T_z(t+1) &= a \cdot T_{F_i}(t) + b_1 \cdot U_i(t) \cdot T_{B_i}(t) \\ &+ b_2 \cdot (1 - U_i(t)) \cdot T_{B_i}(t) + b_3 \cdot V_{FL}(t) \cdot T_{F_{i-1}}(t) \\ &+ b_4 \cdot V_{FL}(t) \cdot T_{F_{i+1}}(t) \qquad (9) \end{aligned}$$

4.3 Parameter Learning

For each server in the rack, there are a total of eleven parameters in the above local model. To make things concrete, we use the notation, $\theta = \{a, b_1, b_2, b_3, b_4, c_1, c_2, f_1, f_2, f_3, f_4\}$. Let $\theta^{(i)}$ be the parameter set for server i, hence $\theta^{(-1)}$, $\theta^{(0)}$ and $\theta^{(1)}$ correspond to the server immediately below, the server itself, and the server directly above. Note that in our framework, the current local server does not know the temperatures and airflow status for neighbors that are two or more slots away on the rack, hence the corresponding parameters $b_3^{(-1)}$, $c_2^{(-1)}$ and $f_4^{(-1)}$ are explicitly made 0.

Base model

To estimate the parameters, we optimize the following objective function:

$$\hat{\theta}^{(i)} \leftarrow \arg\min f(\theta^{(i)}) = \sum_{t=1}^{t_{max}-1} g(\theta^{(i)}, t) \qquad (10)$$

where

$$\begin{aligned} g(\theta^{(i)}, t) &= \big(T_{F_i}(t+1) - a \cdot T_{F_i}(t) - b_1 \cdot U_i(t) \cdot T_{B_i}(t) \\ &- b_2 \cdot (1 - U_i(t)) \cdot T_{B_i}(t) - b_3 \cdot V_{FL}(t) \cdot T_{F_{i-1}}(t) \\ &- b_4 \cdot V_{FL}(t) \cdot T_{F_{i+1}}(t)\big)^2 \\ &+ \big(T_{B_i}(t+1) - f_1 \cdot T_{B_i}(t) - f_2 \cdot T_{F_i}(t) \\ &- f_3 \cdot U_i(t) \cdot W_i(t) - f_4 \cdot T_{B_{i-1}}(t)\big)^2 \qquad (11) \end{aligned}$$

Given the available measurements of temperature, server on/off status, workload, and floor vent air velocity, the objective function is convex and there is a global optimal solution. The solution can be obtained by minimizing the least square objective, i.e. by solving $\frac{\partial f(\theta^{(i)})}{\partial \theta^{(i)}} = 0$.

Proposed ThermoCast

The base model assign equal weights to the deviation of prediction and observation at all time ticks. However, in reality, temperatures can be more perturbed by temporal nearby events, e.g. shutdown of the server. Intuitively, a good model should forget the events or data in the distant age. In order to adaptively capture changes in dynamics, our proposed ThermoCast assign different weights for different time ticks, according to the temporal locality. We propose to use the following exponentially weighted loss

$$\hat{\theta}^{(i)} \leftarrow \arg\min f_\lambda(\theta^{(i)}) = \sum_{t=1}^{t_{max}-1} \exp(\lambda t) g(\theta^{(i)}, t) \qquad (12)$$

where λ is the forgetting factor, which can be tuned either manually or using cross-validation.

Again the solution of this optimization problem is obtained by solving $\frac{\partial f_\lambda(\theta^{(i)})}{\partial \theta^{(i)}} = 0$.

4.4 Prediction

In ThermoCast framework, the prediction component works as follows. Based on the learning results, each server predicts its local temperatures for the near future. The predictor uses a a past window of size T_w for training and predicts T_p minutes into the future.

Note that due to the structure of the model from (9), the server's intake temperature depends on its own past intake and its neighbors' intake and outtake, as well as the workload on the server. While the outtake temperature depends its intake, workload(fan status) and its neighboring outtakes. On the other hand, the neighbors' future environmental conditions (e.g. servers may shutdown) are unknown during the prediction process. This is a main source of prediction error and the reason that we cannot predict too far into the future.

In order to run the model forward in time, we extrapolate the neighbor's intake and output temperatures. Furthermore, we need the future floor air flow speed and temperatures. To this end, we use a separate autoregressive (second order AR) to predict the future floor vent air flow.

$$V_{FL}(t+1) = \eta_0 \cdot V_{FL}(t) + \eta_1 \cdot V_{FL}(t-1)$$

Where the parameters η_0 and η_1 are estimated using linear least square.

Since AC is the main external stimulus to the system we build a degenerate model for the bottom machine that depends only on the vent airflow. (The vent temperature is assumed to be a constant.) Using the same notation, the model for the bottom machine has the structure:

$$T(t+1) = \sum_{k=\{0,..m-1\}} a_k \cdot T(t-k) + b' \cdot V_{FL}(t) \qquad (13)$$

We introduce higher orders m in the regression to counterfeit unmodeled factors such as the node's neighbors. In practice, we found $m = 3$ to be adequate. We use the method described in Section 4.3 to estimate these parameters as well.

With the predicted floor vent air speed and bottom server temperature, it then straightforward to forecast the intake and outtake temperatures using Eq. 9 and Eq. 8.

5. EVALUATION

We evaluate ThermoCast using real data traces, controlled experiments, and trace driven simulations. In particular, we are interested in answering the following questions:

- How accurately can a server learn its local thermal dynamics for prediction?
- How much extra computing capacity can ThermoCast achieve compared to other approaches under the same cooling cost?

For environmental data such as temperature distributions and airflow speed, we use the data collected from the university testbed, as described in section 3. We use a total of 900 minutes of data traces, during which the AC has both high and low duty cycles. The sampling interval in the trace is 30 seconds. We choose one server at the top of a rack, one in the middle and one at the bottom of the rack to represent different server locations.

5.1 Model Quality

We are interested in how much historical training data a server needs to keep in order to obtain a good enough local thermal model. Obviously, less data means faster training speed, less storage, and less communication among servers. We evaluate the model accuracy in terms of its prediction accuracy. In the experiments, we choose a moving window T_w for training and prediction length T_p.

Figure 7 shows the prediction results in terms of Mean Square Error (MSE) as a function of training data length (in minutes). We can see that in general, the more data used the training, the more accurate the model is. The shorter prediction length, the better accuracy we can achieve. In fact, if we use 90 minutes of training data and predict 5 minutes into the future, we can obtain very good results. Figure 8(b) shows a time domain plot for one of the traces.

		Prediction length (minutes)			
		5	10	15	20
Training length (minutes)	15	5.1	5.0	5.0	5.1
	30	7.2	7.0	14.8	8.2
	45	10.1	10.3	10.7	11.4
	60	13.4	13.5	13.3	16.9
	75	16.0	17.6	17.7	178.0
	90	20.1	19.5	23.7	204.0

Table 2: Execution time (in milliseconds) for different training and prediction time combinations.

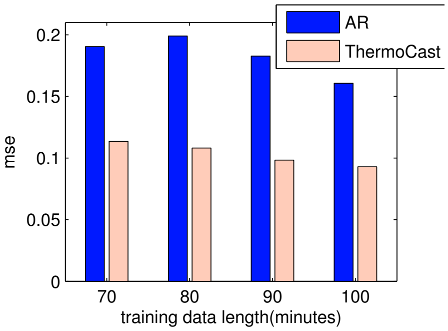


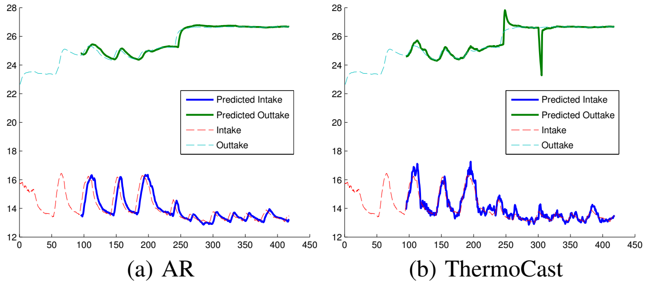
Figure 7: Forecasting error (MSE) of the thermal model as a function of training data length. All predictions are made at 5 minutes away from training. ThermoCast produces consistently lower error and is up to 2x better than the baseline AR method.

Table shows the computational overhead of prediction and learning on each server (Dual core 3.2GHz, 2G RAM, Win XP server). As the data shows, the overhead is small.

5.2 Preventive Monitoring

We did experiments on the real data set to test the capability of our model in case of thermal alarms. The major event of interest in data center is occasional over heating of servers. These event can be caused by a variety of factors such as insufficient cooling, blocking of intake air, fan error, and over placement of workload. Our goal is to exploit ThermoCast to continuously monitor and predict in advance cases of overheating of the intake air. Since we are not allowed to create actual overheating in a real data center, we use real traces of temperature readings and set an artificial threshold (=16). Any temperature higher than such a threshold will trigger an alarm.

The test process works as follows. We first obtain a true labeled trace by identifying "overheating" sections in the temperature sequences. Each section corresponds to a thermal event. In testing, we use ThermoCast or the baseline to forecast the temperatures in the future, and trigger alarms when such predicted temperature is above the thermal threshold. We then calculate two sets of metrics for both our model and the baseline, namely recall(R)/false alarm rate(FAR) and mean look-ahead time(MAT). Recall and false alarm rate are defined on all time ticks with or without alarms, using the



(a) AR (b) ThermoCast

Figure 8: A time domain trace for prediction quality using ThermoCast. T_w = 90 minutes; all predictions are made at 5 minutes away from training. The baseline AR uses second order auto-regressive model. ThermoCast: $\lambda = 0.006$. ThermoCast intake temperature forecasts closely resemble the actual observations. The spikes seen in the outtake temperature forecasts are due to change in CPU utilization (75% to 100%, and 100% to shutdown). Even though ThermoCast misses a few time ticks in the beginning of these transition, ThermoCast can adapt quickly as new observations become available.

	Baseline	ThermoCast
Recall	62.8%	71.4%
FAR	45%	43.1%
MAT	2.3min	**4.2 min**

Table 3: Alarm prediction performance. Better performance corresponds to higher recall, lower false alarm rate (FAR), and the larger Mean look-ahead time (MAT). $Tmax = 16°$C.

following equation:

$$Recall = \frac{\#truealarms}{\#truealarms + \#missedalarms}$$

$$FAR = \frac{\#falsealarms}{\#truealarms + \#falsealarms}$$

Mean look-ahead time (MAT) is to estimate how much time in advance the model can forecast future "overheating" events. It is only measured for the sections when true alarm happens.

$$MAT = \frac{\sum_i^K \max\{\Delta t | f(t_i - \Delta t) > T_{max}\}}{K}$$

where t_i is the starting time of i-th "overheating" section, T_{max} is the temperature threshold. $f(t_i - \Delta t)$ is temperature forecast using all the data before $t_i - \Delta t$ as training and predicting in next few minutes. The longer this time is, the better it predicts since it allows sufficient reaction time.

Table 3 show the performance of the alarm prediction based on our proposed ThermoCast and the baseline method. Note our method achieves nearly 10% better recall and forecasts the alarms twice earlier than the baseline approach.

5.3 Potential Capacity Gains

Better prediction implies better utilization of cooling capacity under the same CRAC load. To evaluate the computing capacity gain, we need to approximate the cooling effect of 1^oC temperature difference in intake temperature.

Our experimental servers are Dell PowerEdge 1950, with 300W peak power consumption. According to its specification, "the airflow through the PE1950III without the added back pressure from the doors is approximately 35 Cubic Feet Per Minute (CFM)." In [23], Dell Inc. recommended the following rules for estimating cooling capacity.

$$\text{CFM} = 1.78 \frac{\text{Power (W)}}{\text{Temperature Differences (°C)}} \quad (14)$$

In other words, under 35 CFM, 1^oC air flow can cool 20W of workload.

We compare predictive load placement with static profiling-based workload placement decisions. We use 5-minute forecast length, since it is long enough to change load balancer (or load skewer) policies or migrating virtual machines.

Let's assume that the profiling is tight. That is, we use the maximum measured temperature as the basis for profiling results and compute the difference between the static profiling (16.5^oC at the intake) and prediction results. In both cases, we add a 10% safety margin. With ThermoCast, we can operate the server at 13.75^oC on average, which leads to 53W computing power. That is, on average, the same server now can potentially take up to 53W more workload without adding any additional cooling requirement.

Assume that the servers consume 200W on average, we gain extra 26% compute power with the same cooling. Note that we assume that the 53W per server is moved from other places in the data center to this server. So, the overall CRAC duty cycle is unchanged. In other words, if the original PUE of the data center is 1.5 and there are enough work, then with ThermoCast, we can reduce the PUE to 1.4. If there are fixed amount of work, we can achieve better workload consolidation and shut down more servers from the whole data center perspective.

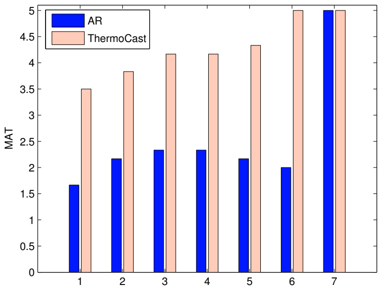


Figure 9: Mean look-ahead time(MAT) as a function of the thermal threshold. Higher MAT values provide more time to react. ThermoCast consistently outperforms the baseline AR method.

6. CONCLUSION

Data center temperature distribution and variations are complicated, which make most workload placement methods shy away from fine-graining thermal aware load scheduling. In this paper, through dense instrumentation and a gray-box thermo-dynamics model, we show that it is possible to predict servers' thermal condition in real time. The gray-box model is derived from a zonal partition of space near each server. In comparison to CFD models, zonal models are simple to evaluate and robust to unmodeled disturbances through parameter estimation.

To solve the scalability and coordination challenges, ThermoCast uses a federated architecture to delegate model building and parameter estimates to individual servers. Using predictions at each server, workload can be consolidated to servers with access to extra cooling capacity without changing CRAC setting.

This work is a building block towards a holistic data center load management solution that takes into account both the dynamic variation of workload and the responses of cooling system. We also plan to investigate in the future the dynamic server provisioning algorithm based on the local thermal effects due to turning servers on/off.

7. REFERENCES

[1] C. Bash and G. Forman. Cool job allocation: measuring the power savings of placing jobs at cooling-efficient locations in the data center. In *USENIX Annual Technical Conference*, 2007.

[2] G. E. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, Englewood Cliffs, NJ, 3rd edition, 1994.

[3] P. J. Brockwell and R. A. Davis. *Time Series: Theory and Methods*. Springer Verlag, New York, 1987.

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

[5] G. Craig. *Introduction to Aerodynamics*, volume 1. Regenerative Press, Anderson, IN, 1st edition, 2003.

[6] E. Elektronik. EE575 Series - HVAC Miniature Air Velocity Transmitter. Available at `http://www.epluse.com/uploads/tx_EplusEprDownloads/datasheet_EE575_e_02.pdf`.

[7] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *SIGMOD*, pages 419–429, Minneapolis, MN, May 25-27 1994.

[8] D. Grunwald, C. B. Morrey, III, P. Levis, M. Neufeld, and K. I. Farkas. Policies for dynamic clock scheduling. In *OSDI*, 2000.

[9] T. Heath, A. P. Centeno, P. George, L. Ramos, Y. Jaluria, and R. Bianchini. Mercury and freon: temperature emulation and management for server systems. In *ASPLOS*, 2006.

[10] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown. Elastictree: Saving energy in data center networks. In *NSDI*, 2010.

[11] M. Jahangiri, D. Sacharidis, and C. Shahabi. Shift-split: I/o efficient maintenance of wavelet-transformed multidimensional data. In *SIGMOD*, pages 275–286, 2005.

[12] E. J. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *SIGMOD*, pages 151–162, 2001.

[13] J.-G. Lee, J. Han, and X. Li. Trajectory outlier detection: A partition-and-detect framework. *ICDE*, pages 140–149, 2008.

[14] L. Li, J. McCann, N. Pollard, and C. Faloutsos. Dynammo: Mining and summarization of coevolving sequences with missing values. In *KDD*, New York, NY, USA, 2009. ACM.

[15] L. Li, B. A. Prakash, and C. Faloutsos. Parsimonious linear fingerprinting for time series. In *PVLDB*, volume 3, pages 385–396, 2010.

[16] C.-J. M. Liang, J. Liu, L. Luo, A. Terzis, and F. Zhao. RACNet: a high-fidelity data center sensing network. In *Sensys*, pages 15–28, 2009.

[17] Liebert. Liebert Deluxe System/3 - Chilled Water - System Design Manual. Available at `http://shared.liebert.com/SharedDocuments/Manuals/sl_18110826.pdf`, 2007.

[18] Liebert. Liebert Deluxe System/3 Precision Cooling System. Available at `http://www.liebert.com/product_pages/ProductDocumentation.aspx?id=13&hz=60`, 2007.

[19] Liebert. Technical Note: Using EC Plug Fans to Improve Energy Efficiency of Chilled Water Cooling Systems in Large Data Centers. Available at `http://shared.liebert.com/SharedDocuments/White%20Papers/PlugFan_Low060608.pdf`, 2008.

[20] J. Liu, B. Priyantha, F. Zhao, C.-J. M. Liang, Q. Wang, and S. James. Towards discovering data center genome using sensor net. In *Proceedings of the 5th Workshop on Embedded Networked Sensors (HotEmNets)*, 2008.

[21] J. Moore, J. Chase, P. Ranganathan, and R. Sharma. Making scheduling "cool": temperature-aware workload placement in data centers. In *USENIX Annual Technical Conference*, 2005.

[22] J. Moore, J. S. Chase, and P. Ranganathan. Weatherman: Automated, online and predictive thermal mapping and management for data centers. In *Proc. of the 2006 IEEE International Conference on Autonomic Computing*, 2006.

[23] D. Moss. Guidelines for assessing power and cooling requirements in the data center. Available at `http://www.dell.com/downloads/global/power/ps3q05-20050115-Moss.pdf`, 2005.

[24] T. Mukherjee, A. Banerjee, G. Varsamopoulos, S. K. S. Gupta, and S. Rungta. Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers. *Comput. Netw.*, 53(17):2888–2904, 2009.

[25] C. Patel, C. Bash, R. Sharma, and R. Friedrich. Smart cooling of data centers. In *ASME Interpack*, 2003.

[26] C. Patel, R. Sharma, C. Bash, and S. Graupner. Energy aware grid: Global workload placement based on energy efficiency. In *ASME International Mechanical Engineering Congress and R&D Expo*, 2003.

[27] D. Patnaik, M. Marwah, R. Sharma, and N. Ramakrishnan. Sustainable operation and management of data center chillers using temporal data mining. In *KDD*, 2009.

[28] L. Ramos and R. Bianchini. C-Oracle: Predictive Thermal Management for Data Centers. In *HPCA*, 2008.

[29] Sensirion. Datasheet SHT1x (SHT10, SHT11, SHT15) - Humidity and Temperature Sensor. Available at `http://www.sensirion.com/en/pdf/product_information/Datasheet-humidity-sensor-SHT1x.pdf`, 2010.

[30] K. R. Swalin. Evaluating microsoft hyper-v live migration performance using ibm system x3650 m3 and ibm system storage ds3400. Available at `ftp://public.dhe.ibm.com/common/ssi/ecm/en/xsw03091usen/XSW03091USEN.PD`, 2010.

[31] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach. *IEEE Transactions on Parallel and Distributed Systems*, 19(11):1458–1472, 2008.

[32] Y. Tao, C. Faloutsos, D. Papadias, and B. Liu. Prediction and indexing of moving objects with unknown motion patterns. In *SIGMOD*, pages 611–622, 2004.

[33] M. Zhao and R. Figueiredo. Experimental study of virtual machine migration in support of reservation of cluster resources. In *2nd International Workshop on Virtualization Technology in Distributed Computing*, 2007.