ALTM: Adaptive learning-based thermal model for temperature predictions in data centers

SeyedMorteza MirhoseiniNejad
Dept. of Computing and Software
McMaster University
Hamilton, Canada
mirhoses@mcmaster.ca

Fernando Martínez García
Dept. of Computing and Software
McMaster University
Hamilton, Canada
martinef@mcmaster.ca

Ghada Badawy
Computing Infrastructure Research Centre
McMaster University
Hamilton, Canada
badawyg@mcmaster.ca

Douglas G. Down
Dept. of Computing and Software
McMaster University
Hamilton, Canada
downd@mcmaster.ca

***Abstract*—To design effective control schemes for energy efficiency in data centers, it is crucial to have a thermal model of the system. Constructing thermal models of data centers for temperature prediction is extremely challenging, due to inherent complexity. Computational fluid dynamics (CFD) simulations or physical heat transfer equations are conventionally used to construct such thermal models. More recent approaches combine physical heat transfer rules and data-driven methods in an effort to obtain more accurate models.**

Our proposed adaptive learning-based thermal model (ALTM) is fast, adapts to thermal changes in the data center environment, and does not require prior knowledge of heat transfer rules between data center entities. Unlike other methods, ALTM is a holistic thermal model that predicts temperature of critical zones using data center operational variables as inputs. The operational variables are the controllable parameters and easily obtained measurements from IT and cooling units. A key use case for ALTM is that it can be effectively used for thermal-aware workload schedulers or cooling system controllers. Our results confirm the accuracy and adaptability of the model.

***Index Terms*—thermal model, thermal-aware workload scheduling, data center temperature prediction, adaptive cooling control, neural network thermal model**

I. Introduction

Air cooling systems continue to be the most common cooling systems in data centers. These can be simply building-designed coolers such as normal air conditioners (AC) or conventional heating, ventilation, and air conditioning (HVAC) units. Many large scale data centers use computer room air conditioning (CRAC) units, in the form of a raised-floor architecture [1]. In-row cooling units and rack mountable cooling units (RMCUs) are more recent and power-efficient cooling system designs [2].

Cooling systems should provide sufficient cool air for servers. Maintaining the intake air of servers below a certain temperature ensures a safe working environment for servers and does not compromise their performance (due to automatic throttling of computing nodes [3]) or reliability [4]. The current practice of today's data centers is to keep the maximum temperature of a zone affected by a cooling unit below a certain temperature. Implementing this practice inevitably results in many servers being far below the required temperature, such servers are said to be over-cooled. Reducing over cooling of servers is an obvious opportunity for power savings [5]. The key component in achieving the minimum amount of over-cooling is to have a holistic thermal model. This model should give the distribution of air temperatures inside a data center based on the operational parameters of the cooling units and the heat generation profiles of servers [6].

A thermal model simply answers the question "what will be the temperature at the front of each server?". The answer should be in the form of a vector containing the temperature distribution, at a given future time. This can be obtained with respect to the current status of a data center, such as cooling unit configurations or the arrangement of heat sources (servers). Tracking server temperatures is crucial for operational control of cooling systems and server workload management.

There are a number of works and methods presenting thermal models of data centers. In our previous work, we showed that using a holistic thermal model, a significant portion of the cooling power could be saved through an optimized assignment of workload and appropriate adjustment of cooling parameters [7]. Computational fluid dynamics (CFD) methods [8] can estimate the temperature of every point within a data center with high precision, however, they are very computationally intensive and are not appropriate for real-time decisions. There are a number of faster models using zonal-based methods and physical energy balance equations [6], [9]; however, these methods do not adapt with physical changes within data centers and also their accuracy deteriorates within large-scale settings. This is

This research was supported by a Collaborative Research and Development grant CRDPI506142-16 from the Natural Science and Engineering Research Council of Canada (NSERC), and Cinnos Mission Critical Incorporated.

because determining incoming and outgoing air flows of thermal zones becomes very complicated in such chaotic environments.

In this paper, we present a means to predict the inlet temperatures of servers with high precision. This is a transient model (as opposed to steady state) that adapts to physical changes and can estimate the temperature over a time horizon. The inputs to our model are the operational parameters of the cooling unit and server workload assignment. With this in mind, we set up an infrastructure monitoring tool to provide the required data to predict future temperatures. We compared a linear regression (least squares) model and a neural network approach and concluded that an appropriate neural network can predict the temperature more accurately, further into the future. An important application of our work is as a key component for holistic system management to both schedule the incoming workload and control the cooling unit parameters efficiently in order to minimize total power consumption.

The next section provides a review of works related to temperature prediction in data centers. Next, the details of our experimental data center architecture and data acquisition phase are explained. In Section (III-B), the framework to implement two model estimators is illustrated and discussed. Finally, results of implementing the framework are provided and analyzed.

II. Literature review

The literature lacks adaptive and/or practical solutions capturing all factors affecting air recirculation in a data center. *Computational fluid dynamics* (CFD) simulations are the predominant way of constructing thermal models for data centers. CFD simulations are based on thermodynamic laws and have heavy computational requirements. Although CFD methods have high precision and resolution, they cannot be evaluated at the time scales of data center dynamics [10].

The majority of works on thermal-aware workload assignment either simplified the effects of air recirculation using a static recirculation matrix [11], [12] or used a simple auto-regression method, simply based on IT load [13]. The drawback to these methods is that they have not considered the effects of all operational variables of a data center.

Moore et al. [14] used a neural network to compute the temperature of inlet air for all servers. The inputs of their model are pairs of power and heat profiles. Specifically, workload, cooling settings, and room layout measurements are used to train the model. However, it is a steady-state model that uses a limited number of influential parameters of the cooling unit as inputs to the model. So, the accuracy of the model can potentially be compromised by changes in parameters that have not been considered, such as a change in the air flow rates.

Zhang et al. [15], [16] developed a machine learning-based framework for temperature prediction of server cores. Several measurements of a running task are used as the features of the neural network model such as the CPU frequency, the number of instructions, floating-point operations, and cache hits or misses in different cache levels. Appropriate features are selected using a correlation feature selection (CFS) algorithm. They used this prediction model for application scheduling on different servers to reduce the maximum average core temperature.

Yao et al. [13] used a linear function that relates the outlet temperatures of IT-racks and CRACs to the inlet temperatures of IT-racks. $Y = WX$ is used as the linear model where X contains the outlet temperatures, Y contains the inlet temperatures, and W contains the weights. They used the recursive least squares (RLS) method to determine the weights (W) of the linear model.

Li et al. [17] presented an approach for energy-efficient thermal-aware workload scheduling. They used CFD methods to model the temperature distribution in a data center. Although it is asserted that the thermal model captures features of CRACs holistically, the simplified thermal model may be far from reality (for example, fan speeds are supposed to be constant, or the closest CRAC unit to a server is considered to be the only cooling unit that influences the server temperature) and the model lacks cooling unit details.

Li et al. [9] proposed ThermoCast, a thermal prediction model to predict temperatures in a data center, based on temperature and air flow measurements. This approach considers the most recent measurements of IT power consumption, temperature, and air flow rates to update the model coefficients. The main issue with this method is that the model requires a structure based on physical laws. They simplify the air flow equations to obtain the structure. So, errors due to simplification may propagate for large scale data centers. The other issue is that the model uses air flow measurements at the front of servers, which we have found to be problematic as directly controllable (or measurable) variables.

III. Thermal model

The main objective of constructing a thermal model is the estimation of the temperature distribution within a data center. The model should be able to predict the inlet temperatures of servers based on the operational parameters of cooling units and the data center workload. We start with illustrating outputs and inputs of the model. The outputs are the temperatures of thermal zones. A zone is the cubical volume at the front of a number of adjacent servers. For example, the data center shown in Fig. 1 has 25 thermal zones. Adjacent servers typically have small differences in their inlet temperatures; as a result, we use the inlet temperature of a server and the temperature of a zone interchangeably throughout this paper. Inputs of the model are manipulatable or controllable variables which here are workload profiles and cooling profiles; the former is related to the IT facilities and the latter corresponds to the cooling facilities.

a) Workload profile: For the sake of simplicity, this work simply considers the workload of a server to be its utilization.

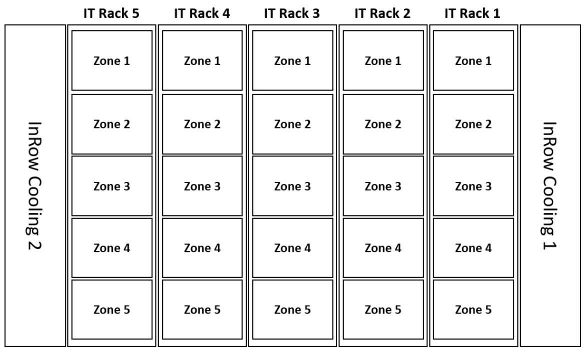

Fig. 1. Front view of an in-row cooling data center with two cooling units at two sides and five IT racks

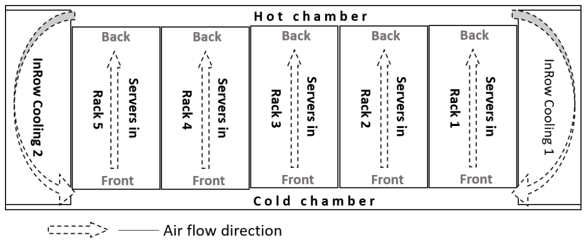

Fig. 2. The top view of the in-row cooling data center

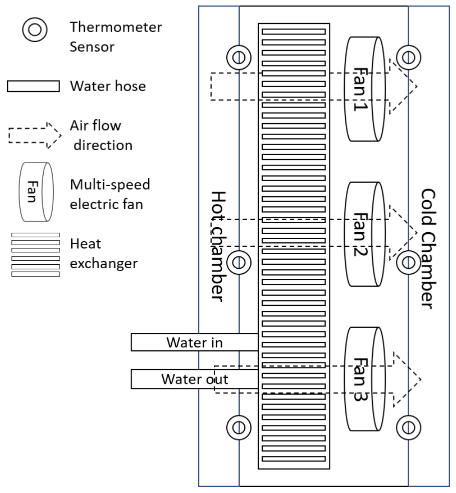

Fig. 3. In-row cooling schema

b) Cooling profile: The cooling profiles are the set of dynamic variables that can be measured and controlled and also affect the temperature distribution.

In this paper, we show that a reasonably accurate temperature prediction is gained using the suggested framework with the help of readily available inputs. The implementation of the framework is straightforward and there is no need for understanding the physics of the heat transfer within the data center. The determination of server inlet temperature estimates is both on-line and adaptive. To the best of our knowledge, this is the first thermal modeling approach that directly uses cooling and IT parameters for its predictions and adapts to changing thermal conditions. Changes in the thermal condition of a data center are to be expected. These changes can be initiated from component changes due to system maintenance, room alterations, device replacements, dust accumulation, modifications of the compartments and air vents, etc. An example later in the paper shows the necessity of being adaptive.

Next, the procedure for data acquisition for model estimation is described in detail. We then show the implementation of on-line model estimators and describe a framework for using them. Finally, we discuss the accuracy of using different model estimators, and illustrate the results.

A. Data acquisition

An important aspect of this work was setting up equipment and reporting tools to acquire data. The setup was implemented in a data center which has two in-row cooling units at two sides and five IT racks (Fig. 1). We developed a data acquisition tool to both apply our desired configurations and acquire all operational variables of cooling units and server profiles. Fig. 2 shows the top view of the data center under study, consisting of two major parts: IT and cooling units. IT is considered to be the servers and cooling units include the facilities that provide cool air at the front of servers.

Fig. 3 shows the architecture of each cooling unit. As shown, each cooling unit has a number of fans that draw hot air from the hot chamber, pass the air through a heat exchanger and blow the cold air to the cold chamber. Water flow within the heat exchanger transfers the generated heat out of the facility. In other words, cold water enters the heat exchanger, and warm water exits.

Cooling unit operational parameters can be controlled and monitored using the *simple network management protocol* (SNMP); these parameters include the speed of each fan and the water flow rate inside the heat exchanger of the cooling unit. On the other hand, the IT consists of servers that process the given workload. We can apply the given workload to servers and collect real-time reports using SSH commands. Each server is able to report the current utilization and temperature of its cores. Temperatures at the front of servers obtained via thermal sensors (DS18B20 digital thermometers) which are placed in each zone. The height of each rack is divided into five equal height thermal zones.

Our designed tool connects to cooling units using SNMP, to servers by SSH, and to thermal sensors via serial ports. It takes operational scenarios as an input. A scenario is a time series of values that needs to be applied to the controllable variables of the data center at specified times. The operational scenario should be rich in parameter variation to be suitable to train the model. Upon executing a scenario workload patterns are applied to servers and patterns of operational parameters are applied to cooling units. At the same time, reported data including measurements from the thermometers (installed at 25 thermal zones), the utilization and CPU temperature of servers, and operational parameters of the two cooling units are saved in a data base. The operational parameters of the cooling units consist of the inlet water temperature (T_{inlet}^{water}), the water flow (Q^{water}) and fan speeds (S_i^{fan}).

Gathered data is preprocessed and saved into corresponding matrices, input $\mathbf{X}$ and output $\mathbf{Y}$, to be applicable for a model estimator. Each row of $\mathbf{X}$ includes all input variables and each row of $\mathbf{Y}$ includes all output measures are obtained at the same time step. The values in $\mathbf{X}$ and $\mathbf{Y}$ are normalized to be in comparable scales. A bold capital letter, such as $\mathbf{Y}$, represents a matrix and a bold small letter, such as $\mathbf{y}_i$, denotes a vector corresponding to the i^{th} row of a matrix.

A top-level view of the thermal model can be formulated as (1). This is a transient model that predicts the inlet temperatures of servers at the next time step, shown by $\hat{\mathbf{y}}_{k+1}$. So, the model is a discrete function of the current and previous inputs and outputs. The output vector is represented by $\mathbf{y}_k$ and $\mathbf{x}_k$ is the input vector, both at the k^{th} time step. The vector $\mathbf{x}_k$ consists of the operational parameters of cooling units, $\bar{S}^{fan}$, $\bar{Q}_{water}$ and $\bar{T}_{in}^{water}$, and the workload profile given by server utilizations ($\bar{U}$). The vectors $\mathbf{x}_{k-i}$ and $\mathbf{y}_{k-i}$ are the input and output in the i^{th} previous time step.

$$\hat{\mathbf{y}}_{k+1} = f(\mathbf{x}_k, \mathbf{x}_{k-i}, \mathbf{y}_k, \mathbf{y}_{k-i}) \;\; i = 1, 2, \cdots \tag{1}$$

B. The model framework and algorithms

We explored two different approaches for temperature estimation. The first uses *weighted recursive least squares* (wRLS) for the estimation of a linear model, and the second trains a neural network model. We selected two off-the-shelf adaptive model estimators; one of them works well for a linear system and the other can better model a non-linear system. We arranged to make a fair comparison between them by exposing them to the same input data.

a) Weighted recursive least squares: We used wRLS for the parameter estimation of a linear thermal model [18]. Without loss of generality, the problem can be considered as a simple linear model $\mathbf{Y} = \mathbf{\Phi}\mathbf{X}$; here $\mathbf{X}$ and $\mathbf{Y}$ are matrices of inputs and measured output values, respectively. wRLS is an on-line model estimator which is able to adapt to changes in the system being estimated. The algorithm forgets the past data using a forgetting factor λ.

wRLS has an update phase that updates the model parameters (or $\mathbf{\Phi}$) upon receiving new data. For the sake of simplicity, we do not explain the wRLS process. We just denote the update phase in the form of $\mathbf{\Phi}_{new} = parameterUpdate(\mathbf{\Phi}_{old}, \mathbf{X}, \mathbf{Y}, \lambda)$. Here, $\mathbf{\Phi}_{new}$ is the newly calculated model parameters with respect to $\mathbf{\Phi}_{old}$ and updated $\mathbf{X}$ and $\mathbf{Y}$. $\mathbf{\Phi}_{old}$ is the latest calculation of the model parameters. The matrices $\mathbf{X}$ and $\mathbf{Y}$ are updated with new data during each iteration.

The wRLS algorithm considers a number of previous data samples p, often referred to as a p^{th} order filter. So, a window of length p is updated with the most recent data samples. $\mathbf{X}$ is a p by i matrix, $\mathbf{Y}$ is a p by o matrix, and $\mathbf{\Phi}$ is an i by o matrix in which i is the number of linear terms of the input and o is the number of outputs being estimated.

Algorithm 1 gives a simple form of wRLS to estimate the linear thermal model. In this algorithm, one iteration is performed upon receiving a new vector of data $\mathbf{d}$. The function *dataGeneration()* returns the vector of new samples of inputs and the corresponding outputs. The new data $\mathbf{d}$ is used to update input and output matrices using *dataInsertion()*. Finally, $parameterUpdate()$, using the new matrices of inputs and outputs, updates the previously obtained parameters in $\mathbf{\Phi}$.

```
Result: Estimation of the linear thermal model
X=$[0]_{p,i}$;
Y=$[0]_{p,o}$;
$\Phi$=$[0]_{i,o}$;
$\lambda$=0.9;
while true do
    d = dataGeneration();
    [X Y]=dataInsertion(X,Y,d);
    $\Phi$=parameterUpdate($\Phi$,X,Y,$\lambda$);
end
```

Algorithm 1: Adaptive linear thermal model

b) Neural Networks: The second method is training an adaptive neural network for the thermal model. For the neural network model, we used a MATLAB toolkit in which the standard back-propagation method uses the Levenberg-Marquardt algorithm to train the model. Our job is to see how well an off-the-shelf neural network performs. As a result, analyzing and comparing different neural network methods is out of the scope this paper. However, it is certainly an interesting topic for future work.

As explained previously, the model should be updated as time progresses. There are a number of methods that consider updating neural networks upon system changes [19]. For example, an update can be performed upon detecting a notable mismatch between the desired and estimated data. We chose the statistical batch selection method for updating [20], as it is straightforward to implement for our scenario.

Statistical batch selection updates the neural network model upon receiving a number of new data points. Randomly generated numbers are used as indexes to select data samples from the previously saved data. The batch selection approach is more likely to return recent data for the next iteration of the algorithm. To implement the adaptive neural network and batch selection method, we used Algorithm 2. The algorithm stores the recent data in a buffer of length i. It then selects the batch of data using the function $batchSel()$, as described. This batch is used to train the new neural network. The network uses the previous iteration weights and biases.

Algorithm 2 first initializes input and output data windows ($\mathbf{X}$ and $\mathbf{Y}$), and the internal neural network weights ($\mathbf{\Phi}$) using the function $initialize()$. It requires a specific number of data samples (l) at the beginning of each iteration. In our implementation, we set l to be 10. We chose

```
Result: Estimation of the neural network model
$\mathbf{X}$=$[0]_{p,i}$;
$\mathbf{Y}$=$[0]_{p,o}$;
$\mathbf{\Phi} = initialize()$;
$l = 10$;
$n = 1$;
$net = backpropagation(n)$;
while true do
    $\mathbf{D} = dataGeneration(l)$;
    $[\mathbf{X}, \mathbf{Y}] = dataInsertion(\mathbf{X}, \mathbf{Y}, \mathbf{D})$;
    $[\mathbf{X_b}, \mathbf{Y_b}] = batchSel([\mathbf{X}, \mathbf{Y}])$;
    $train(net, \mathbf{X_b}, \mathbf{Y_b}, \mathbf{\Phi})$;
    $\mathbf{\Phi} = net.weights()$;
end
```
Algorithm 2: Adaptive neural network thermal model

one hidden layer and the back-propagation method for the neural network (n).

In the loop, after receiving a certain number of data points (l), the new data samples (**D**) are inserted in the data window [**X**, **Y**] and the outdated data points are discarded from the window. After constructing [**X**, **Y**], the batch [$\mathbf{X_b}, \mathbf{Y_b}$] of selected data is constructed by the *batchSel()* function. The neural network is then trained using the selected batch and the previously calculated network weights.

IV. Results

We first compare the estimation results of the linear and neural network models. For the neural network the accuracy for the validation set is set to $0.001°C$. The termination of the neural network training happens after 9 epochs, on average. The neural network computational complexity was not limiting for our settings, however, this aspect should be be studied in the future. Fig. 4 depicts the estimation horizon of the neural network and a linear model. Curves represent the average temperature of the 25 temperature sensors. The solid line is the value of measurements and non-solid lines are the estimates. The figure shows that the neural network model has greater accuracy than the linear model.

To demonstrate the accuracy of the neural network model, the measured and estimated values are shown in the same plot. A box plot representation is chosen to plot 25 estimates and 25 measured values at each time step. For each box, the average is indicated by the central mark. The 75^{th} and the 25^{th} percentiles are shown by the top and the bottom edges of each box, respectively. In Fig. 5, the blank rectangles with red central marks show the measured values and filled blue rectangles with the central circle marks are the model estimates. The figure shows that the estimates follow the measured values accurately enough. The average estimation error for the 100 time step projection is $1.5°C$.

The neural network model is designed to be adaptive to changes that might occur in thermal conditions. We performed an experiment to demonstrate the adaptivity of the thermal model. We introduced thermal changes at the 1550^{th} time step. At that time, the front doors of the cooling

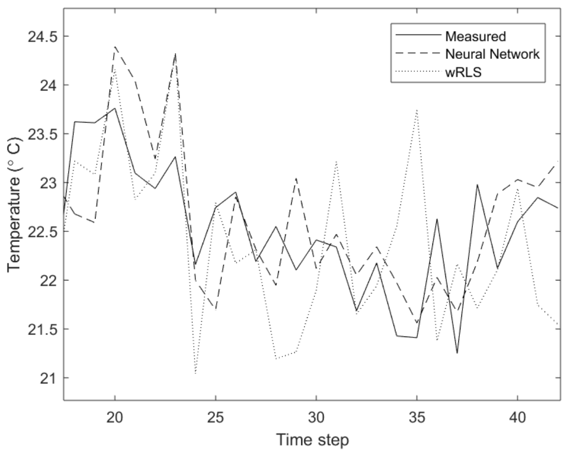

Fig. 4. Temperature prediction of the neural network vs wRLS models

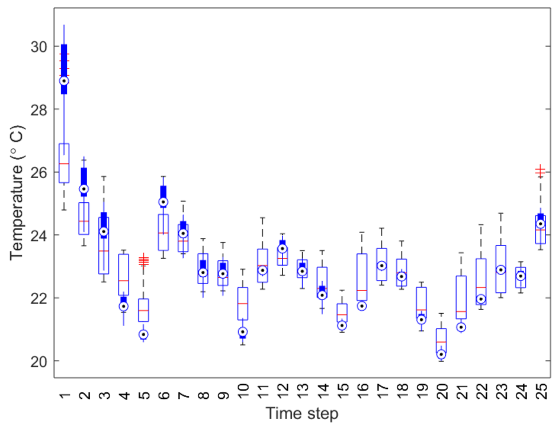

Fig. 5. The box plot representation of the 25 measured temperatures vs their estimates (blank rectangles show the measured values and filled blue rectangles are estimates of the temperatures using neural networks)

units were left partially open, having been closed before the 1550^{th} time step. Fig. 6 shows that at the time of the change a large error occurs in the estimates. The model then adapts to the new thermal conditions and the error decreases.

In Section II, CFD models and a number of physics-based thermal models were reviewed. It was stated there that the main issue with using these is that none of them are adaptive to the thermal changes in the data center environment. Fig. 7 shows the behavior of an adaptive and a non-adaptive neural network model. The figure clearly demonstrates the difference between these two models. The average error for the adaptive model is $1.15°C$ and for the non-adaptive model is $2.1°C$. The errors of non-adaptive models can potentially diverge for longer prediction horizons, so these errors are exacerbated as time increases.

V. Conclusion

We introduced a novel, low-complexity, easy to implement, and adaptive model estimator which captures the

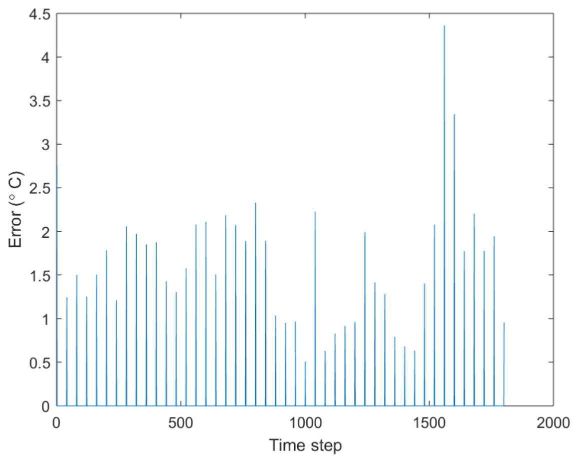

Fig. 6. 100 steps projection error for the neural networks model - An environmental change happened at the 1550^{th} time step

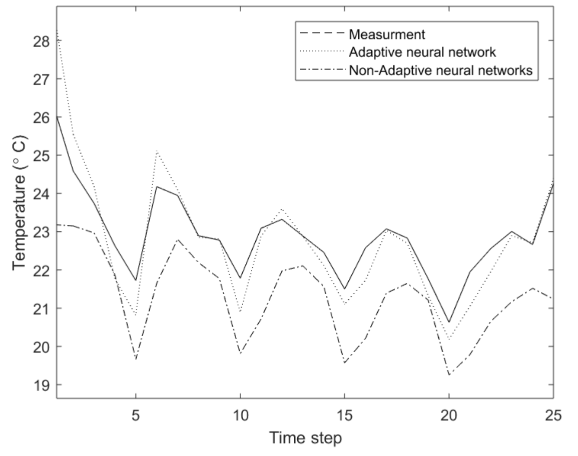

Fig. 7. The comparison between adaptive and non-adaptive thermal models

thermal dynamics of a data center. It can be applied in any data center and provides up-to-date information that could be used by a thermal-aware workload manager. The model is also attractive because it only requires readily available inputs. Other means of constructing thermal models have some deficiencies. Many of them are just fixed models that do not change with the changes within a data center, which is a serious drawback due to the dynamic nature of data centers. Some suggested adaptive thermal models do not consider the cooling infrastructure at the same level of detail as we have. Considering every operational variable of the cooling units provides the opportunity of controlling cooling together with the assignment of workload which can lead to significant power savings. Our adaptive thermal modeling approach appears to be an attractive option to incorporate into workload schedulers or control algorithms.

References

[1] K. Ebrahimi, G. F. Jones, A. S. Fleischer, A review of data center cooling technology, operating conditions and the corresponding low-grade waste heat recovery opportunities, Renewable and Sustainable Energy Reviews 31 (2014) 622–638.

[2] A. Capozzoli, G. Primiceri, Cooling systems in data centers: state of art and emerging technologies, Energy Procedia 83 (2015) 484–493.

[3] W. L. Bircher, L. K. John, Complete system power estimation using processor performance events, IEEE Transactions on Computers 61 (4) (2012) 563–577.

[4] X. Teng, H. Pham, D. R. Jeske, Reliability modeling of hardware and software interactions, and its applications, IEEE Transactions on Reliability 55 (4) (2006) 571–577.

[5] S. M. M. Nejad, G. Badawy, D. G. Down, Eawa: Energy-aware workload assignment in data centers, in: 2018 International Conference on High Performance Computing & Simulation (HPCS), IEEE, 2018, pp. 260–267.

[6] H. Moazamigoodarzi, S. Pal, S. Ghosh, I. K. Puri, Real-time temperature predictions in it server enclosures, International Journal of Heat and Mass Transfer 127 (2018) 890–900.

[7] S. M. M. Nejad, H. Moazamigoodarzi, G. Badawy, D. G. Down, Joint data center cooling and workload management: A thermal-aware approach, in: Future Generation Computer Systems, FGCS, 2019.

[8] C. D. Patel, C. E. Bash, C. Belady, L. Stahl, D. Sullivan, Computational fluid dynamics modeling of high compute density data centers to assure system inlet air specifications, in: Proceedings of IPACK, Vol. 1, 2001, pp. 8–13.

[9] L. Li, C.-J. M. Liang, J. Liu, S. Nath, A. Terzis, C. Faloutsos, Thermocast: a cyber-physical forecasting model for datacenters, in: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2011, pp. 1370–1378.

[10] X. Wang, X. Wang, G. Xing, J. Chen, C.-X. Lin, Y. Chen, Intelligent sensor placement for hot server detection in data centers, IEEE Transactions on parallel and distributed systems 24 (8) (2013) 1577–1588.

[11] Z. Abbasi, G. Varsamopoulos, S. K. S. Gupta, TACOMA: Server and workload management in internet data centers considering cooling-computing power trade-off and energy proportionality, ACM Trans. Archit. Code Optim. 9 (2) (2012) 11:1–11:37. doi:10.1145/2207222.2207227.
URL `http://doi.acm.org/10.1145/2207222.2207227`

[12] Q. Tang, S. K. S. Gupta, 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) (2008) 1458–1472.

[13] J. Yao, H. Guan, J. Luo, L. Rao, X. Liu, Adaptive power management through thermal aware workload balancing in internet data centers, IEEE Transactions on Parallel and Distributed Systems 26 (9) (2015) 2400–2409.

[14] J. Moore, J. S. Chase, P. Ranganathan, Weatherman: Automated, online and predictive thermal mapping and management for data centers, in: 2006 IEEE International Conference on Autonomic Computing, IEEE, 2006, pp. 155–164.

[15] K. Zhang, S. Ogrenci-Memik, G. Memik, K. Yoshii, R. Sankaran, P. Beckman, Minimizing thermal variation across system components, in: Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International, IEEE, 2015, pp. 1139–1148.

[16] K. Zhang, A. Guliani, S. Ogrenci-Memik, G. Memik, K. Yoshii, R. Sankaran, P. Beckman, Machine learning-based temperature prediction for runtime thermal management across system components, IEEE Transactions on Parallel and Distributed Systems 29 (2) (2018) 405–419.

[17] X. Li, P. Garraghan, X. Jiang, Z. Wu, J. Xu, Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy, IEEE Transactions on Parallel and Distributed Systems 29 (6) (2018) 1317–1331.

[18] S. Van Vaerenbergh, I. Santamaría, M. Lázaro-Gredilla, Estimation of the forgetting factor in kernel recursive least squares, in: 2012 IEEE International Workshop on Machine Learning for Signal Processing, IEEE, 2012, pp. 1–6.

[19] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, M. Woźniak, Ensemble learning for data stream analysis: A survey, Information Fusion 37 (2017) 132–156.

[20] I. Loshchilov, F. Hutter, Online batch selection for faster training of neural networks, arXiv preprint arXiv:1511.06343.