General Behavioral Thermal Modeling and Characterization for Multi-core Microprocessor Design *

Thom J. A. Eguia†, Sheldon X.-D. Tan†, Ruijing Shen†, Eduardo H. Pacheco‡ and Murli Tirumala‡
†Department of Electrical Engineering, University of California, Riverside, CA 92521
‡Intel Corporation, 2200 Mission College Blvd, Santa Clara, CA, 95052

Abstract – *This paper proposes a new architecture-level thermal modeling method to address the emerging thermal related analysis and optimization problem for high-performance multi-core microprocessor design. The new approach builds the thermal behavioral models from the measured or simulated thermal and power information at the architecture level for multi-core processors. Compared with existing behavioral thermal modeling algorithms, the proposed method can build the behavioral models from given arbitrary transient power and temperature waveforms used as the training data. Such an approach can make the modeling process much easier and less restrictive than before, and more amenable for practical measured data. The new method is based on a subspace identification method to build the thermal models, which first generates a Hankel matrix of Markov parameters, from which state matrices are obtained through minimum square optimization. To overcome the overfitting problems of the subspace method, the new method employs an overfitting mitigation technique to improve model accuracy and predictive ability. Experimental results on a real quad-core microprocessor show that ThermSID is more accurate than the existing ThermPOF method. Furthermore, the proposed overfitting mitigation technique is shown to significantly improve modeling accuracy and predictability.*

Index Terms – **Thermal analysis, architecture thermal modeling, multicore processor**

1. INTRODUCTION

In the nanometer regime, temperature becomes the major concern for high-performance microprocessor design. The exponential power density increase will, in turn, lead to a rapid rise in average chip temperature [3]. Higher temperatures have significant adverse effects on chip packaging cost, performance, and reliability. Excessive on-chip temperature leads to slower transistor speed owing to reduced carrier mobility, more leakage power consumption as leakage currents grow exponentially with temperature, higher interconnect resistance, and reduced reliability [6, 4].

Multicore techniques mitigate the exponential growth of temperatures resulting from the exponential power density increase from to clock rate increase [10, 1, 2]. But multicore microprocessors have new thermal problems and related package design issues as the temperature of different cores can differ drastically. This can affect the package design costs with different placement of cores and shared caches. Therefore, it is very important to consider temperature effects during the floor planning, architecture design and package configuration of multi-core microprocessors.

Existing work on HotSpot [7, 13] attempts to solve this problem by generating the architectural thermal model in a bottom-up manner based on the internal structure/architecture of the microprocessor. However, bottom-up compact models may suffer from accuracy loss, and have to be calibrated with hardware if more accurate models are required. Also, the generated models may not easily accommodate changing parameters such as thermal conductivity, thermal conditions (ambient temperatures) and packaging configurations when producing parameterized thermal models [14, 15]. Recently, a top-down behavioral architecture level thermal modeling method, *ThermPOF*, was proposed [9], where temperature impulse responses are used to build the thermal models through the matrix pencil method. This method, however, requires the step temperature responses for each core to build the models.

In this paper, we propose a more general thermal behavioral modeling approach for fast temperature estimation at the architecture level for multicore microprocessors. The new approach builds the system matrices from measured or simulated thermal and power information. However, unlike existing thermal modeling methods like *ThermPOF* [9], where only step thermal responses from one power input can be accepted, the proposed method can accept general transient power and temperature waveforms. This makes the new method much more training friendly and data-tractable as step responses are typically difficult, even impossible (intractable), to obtain from the measurement. The new method, called *ThermSID*, consists of two techniques. First, it uses a more general subspace system identification method to build the model, which eliminates the need for step temperature responses. The subspace system identification first generates the states of the desired models in terms of a Hankel matrix of Markov parameters from the measured input and output data via subspace projection and reduction. Then, the discrete state matrices are obtained through state matrix realizations. Second, it employs an over-fitting mitigation technique to pick up the best model among several that are built on partial training data to overcome the unavoidable overfitting problem associated with training-based modeling processes.

Experimental results on a practical multicore microprocessor show that *ThermSID* can provide thermal behavioral models that closely match the measured data, which compares favorably with the existing *ThermPOF* method [9] in terms of accuracy. In addition, *ThermSID* can be much more amenable and less restrictive for more general training data. We also show that the over-fitting mitigation technique can significantly improve modeling accuracy and predictability over the simple subspace based modeling method.

The rest of this paper is organized as follows: Section 2 presents the thermal modeling problem we are trying to solve. Section 3 presents the new thermal modeling method – *ThermSID*. Section 4 shows the experimental results and Section 5 concludes the paper.

2. ARCHITECTURE-LEVEL THERMAL MODELING PROBLEM

*This work is supported in part by NSF Grant CCF-0902885, in part by Semiconductor Research Corporation (SRC) grant under No. 2009-TJ-1991, in part by a grant from Intel Corporation

978-3-9810801-6-2/DATE10 © 2010 EDAA

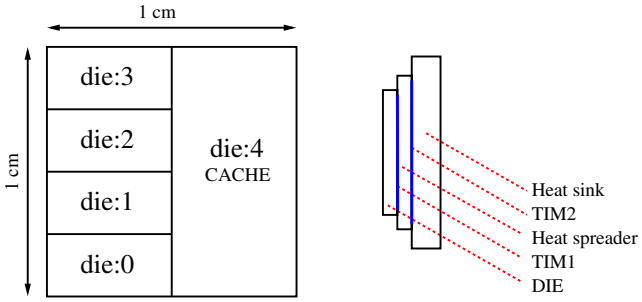

Figure 1: The quad-core architecture.

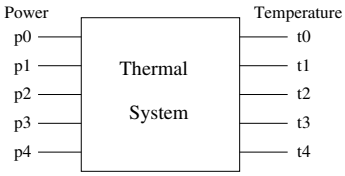

Figure 2: The abstracted model system.

We first present the new thermal behavioral modeling problem. In general, we want to build a behavioral model which is excited by the power input, and produces the temperature outputs in specific locations at the architectural level of the multi-core microprocessor. Our behavioral models are created and calibrated with the measured or simulated temperature and power information from the chip.

Since the given temperature data are transient and changing over time, we need to capture the transient behavior of the temperature. For the new modeling problem, we should not have any restriction for the given power and temperature data in general (the only catch is that the transient training data is long enough). What we want is to build a linear time invariant dynamic system (although strictly speaking, the thermal systems of integrated systems are not linear dynamic systems in general).

In this paper, we study a quad-core microprocessor architecture from our industry partner, Intel Corporation, to validate the new thermal modeling method. The architecture of the multi-core microprocessor is shown in Fig. 1, where there are four CPU cores (die 0 to die 3) and one shared cache core (die 4). TIM stands for thermal interface material. The temperature of each is measured on the bottom face of the center of each die. We can abstract this quad-core CPU into a linear system with 5 inputs and 5 outputs as shown in Fig. 2 (the input p_i and output port t_i will be shared as shown later). The inputs are the power traces of all the cores, while the outputs are the measured temperatures.

Instead of using transfer functions to model the systems [9], we propose representing a thermal system using a discrete linear time invariant system with state space models

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k + Du_k, \end{aligned} \tag{1}$$

where $A \in \mathbb{R}^{nxn}$ is a stable matrix, $B \in \mathbb{R}^{nxm}$, $B \in \mathbb{R}^{lxn}$, and $D \in \mathbb{R}^{lxm}$. The input vectors $u_k \in \mathbb{R}^{mx1}$ and output vectors $y_k \in \mathbb{R}^{lx1}$ are the measured power input traces and temperature responses, respectively.

Given input u_k and output y_k, the problem at hand is how to generate state matrices A, B, C, and D, where D is typically considered a matrix of zeros. Note that given the discrete system matrices, the continuous system matrices can be obtained using methods such as Zero-Order-Hold, Impulse Invariance and Tustin Approximation [5]. In the following section, the subspace system method for finding these state matrices are described.

3. THE NEW THERMAL MODELING METHOD: THERMSID

To determine the state matrices of a core's thermal system, we apply a recently proposed subspace system identification method called Numerical Algorithm for Subspace State Space System Identification (N4SID) [11]. The new method, *ThermSID* is based on the N4SID as the baseline algorithm. It then applies an new overfitting remedying algorithm to overcome the overfitting issues in the N4SID method to improve the model predictability. We will first introduce some concepts used in N4SID.

3.1 Review of the subspace system method

We suppose a core's thermal system is linear time invariant (LTI) with noise, where the state space model is described in (1). For the system in (1), $h_k = CA^{k-1}B$ for $k \geq 1$ are called Markov parameters, which are the values of discrete-time impulse responses. The Hankel matrix of the Markov parameters can be defined as

$$\mathcal{H}_{i,j} = \begin{bmatrix} h_i & h_{i+1} & \dots & h_{i+j} \\ h_{i+1} & h_{i+2} & \dots & h_{i+j+1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{i+j} & h_{i+j+1} & \dots & h_{i+2j} \end{bmatrix}. \tag{2}$$

It can be proven that

$$\mathcal{H}_{1,N-1} = \mathcal{M}_O(N)\mathcal{M}_C(N), \tag{3}$$

where $\mathcal{M}_C$ and $\mathcal{M}_O$ are the extended controllability and observability matrices, which have the following form

$$\mathcal{M}_C(j) = \begin{bmatrix} B & AB & \dots & A^{j-1}B \end{bmatrix}, \mathcal{M}_O(j) = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{j-1} \end{bmatrix}. \tag{4}$$

Considering s points of history and r points of future data, there are three main steps for the subspace algorithm:

1) Develop an estimate $\hat{x}_k$ for the state x_k,

$$y_{future}(k) = \mathcal{M}_O(r)\hat{x}_k + S_{future}u_{future}(k) \tag{5}$$

where

$$y_{future}(k) = [y_k, y_{k+1} \dots, y_{k+r-1}]^T, \tag{6}$$

$$u_{future}(k) = [u_k, u_{k+1} \dots, u_{k+r-1}]^T, \tag{7}$$

$$S_{future} = \begin{bmatrix} 0 & 0 & \dots & 0 \\ CB & 0 & \dots & 0 \\ \vdots & & \ddots & 0 \\ CA^{r-2}B & CA^{r-3}B & \dots & 0 \end{bmatrix}. \tag{8}$$

2) Use $\hat{x}_k$ in (5) to estimate $\hat{y}_{future}(k)$.

3) Form block Hankel matrices for data and predicted responses, and select parameters to

$$\min ||Y^{data}_{future} - \hat{Y}^{pred}_{future}||^2_F, \tag{9}$$

where $||A||^2_F = \text{Trace}(A * A)$.

Here we choose the best linear mean-square for x_k,

$$\hat{x}_k = K_1 y_{past}(k-s) + K_2 u_{past}(k-s) + K_3 u_{future}(k), \tag{10}$$

where

$$\begin{aligned} y_{past}(k-s) &= [y_{k-s}, y_{k-s+1}, \dots, y_{k-1}]^T, \\ u_{past}(k-s) &= [u_{k-s}, u_{k-s+1}, \dots, u_{k-1}]^T. \end{aligned} \tag{11}$$

From (5) and (10), $\hat{y}_{future}(k)$ should be a linear combination of $y_{past}(k-s)$, $u_{past}(k-s)$ and $u_{future}(k)$. Rewrite it in matrix form as follows,

$$\hat{Y}^{pred}_{future} = L_1 Y_{past} + L_2 U_{past} + L_3 U_{future}, \qquad (12)$$

where

$\hat{Y}^{pred}_{future} = [\hat{Y}_{future}(s+1), \hat{Y}_{future}(s+2), \ldots, \hat{Y}_{future}(N-r+1)]$,

$$\begin{aligned} Y_{past} &= [y_{past}(1), y_{past}(2), \ldots, y_{past}N - r - s + 1], \\ U_{past} &= [u_{past}(1), u_{past}(2), \ldots, u_{past}(N - r - s + 1)], \end{aligned}$$

$U_{future} = [u_{future}(s+1), u_{future}(s+2), \ldots, u_{future}(N-r+1)]$.

It can be proven that $[L_1 \quad L_2] = \mathcal{H}$, where L_i is chosen to optimize (9).

$$W = [Y^T_{past} \quad U^T_{past} \quad Y^T_{future}]^T, \qquad (13)$$

then the solution of the optimization problem in (9) can be obtained by the LQ-factorization of W,

$$W = \begin{bmatrix} * & 0 & 0 \\ * & * & 0 \\ L_1 & L_2 & * \end{bmatrix} [\ Q_1 \quad Q_2 \quad Q_3\ ]. \qquad (14)$$

We perform SVD on $\mathcal{H} = [L_1 L_2] = U\Sigma V^*$, where Σ is used to identify the system order n. Afterwards, we can define the first n columns of U and V as U_1 and V_1, respectively, and the diagonal matrix with the first n singular values in Σ as Σ_1. Then, we define

$$\mathcal{M}_O = U_1 \Sigma_1^{1/2},\ \mathcal{M}_C = \Sigma_1^{1/2} V_1^*. \qquad (15)$$

System matrix C is the first n rows of $\mathcal{M}_O$, while B is retrieved from the first n columns of $\mathcal{M}_C$. The matrix A can be obtained by

$$A = (\mathcal{M}_O^{\downarrow})^{\dagger} \mathcal{M}_O^{\uparrow}, \qquad (16)$$

where

$$\mathcal{M}_O^{\uparrow} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{M-1} \end{bmatrix},\ \mathcal{M}_O^{\downarrow} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{M-2} \end{bmatrix}. \qquad (17)$$

3.2 The overfitting problem

Simply applying the subspace-based method, however, can lead to overfitting problems during the modeling process. Overfiting occurs when a model describes random error or noise instead of the underlying behavior of the system, thus leading to poor predictive performance on new data sets.

Typically, overfitting occurs when a model is built to follow its training data so closely that it harms the model's predictive capabilities. The same phenomena occurs when using the subspace system method to generate the model. Fig. 3 provides an example of this using a model of order 10. We can see that the model provides a near exact match of the training data used to generate it. However, simulating a new set of inputs shows that the model is unable to follow the trend of the verification data.

3.3 The mitigating schemes

Improving the model's predictive capabilities requires minimizing the complexity of the model itself. Fig. 4 shows the simulation results of three models of different order that are generated using the same training set. The model at order 20 exhibits the worst fit, but improves progressively as the model order is reduced. At order 5, the fit of the model has significantly improved. Furthermore, our tests indicate that order 5 is the optimal setting for our model, with no improvement gains to be made at lower orders. Lowering

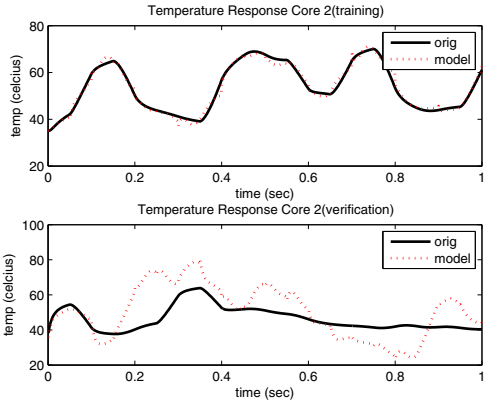

Figure 3: Training and verification waveform comparison, model order = 5

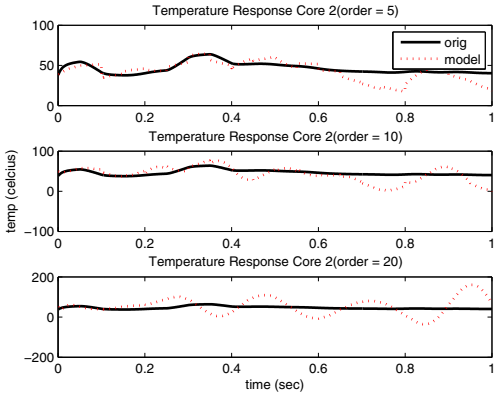

Figure 4: Verification waveform comparison, model order = 5, 10, 20

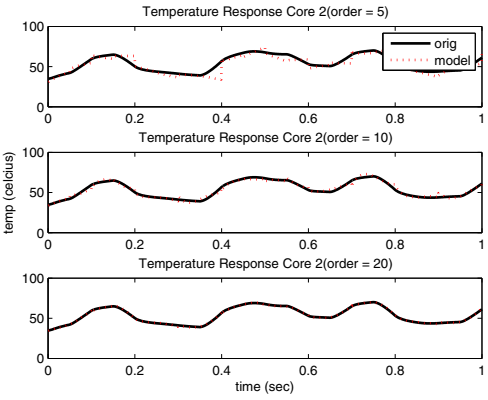

Figure 5: Training waveform comparison, model order = 5, 10, 20

the complexity of the model, however, adversely affects the fit of the training data. Fig. 5 shows this result, where the quality of the fit decreases with the order of the model. The significance of this result will be explained later.

Despite lowering the complexity, the subspace system method is still unable to provide a reasonably accurate model. This result suggests that overfitting is still occurring and may be explored further. Several methods have been developed in the statistics community to mitigate the overfitting problem, which include early stopping, regularization, cross-validation,

and Bayesian priors [8, 12]. For our problems, we observe that some sections of training data are more misleading than other sections due to highly correlated data that mask the underlying behavior of the system.

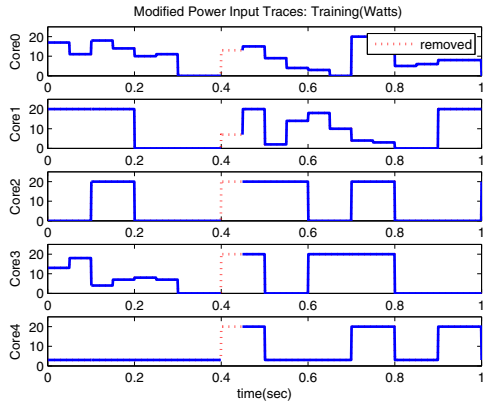

Figure 6: Power input traces with section removed

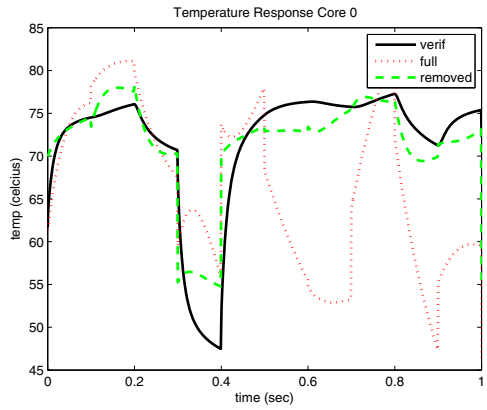

Figure 7: Temperature response comparison between model 1 and model 2 for Core 0

To show the effects of these correlated data, we shall observe the differences in accuracy of a model created by the training data presented in Fig. 6. First, we shall create model 1 using the entire data set, and then generate model 2 with the dotted red section removed. Fig. 7 shows the simulation results of Core 0 on a separate verification set. The dotted red line is the waveform created by model 1, while the dashed green line is created by model 2. The results show a distinct improvement in accuracy when a particular section of data is removed. This misleading section of data appears to be detrimental to the predictive capabilities of the model. In this particular section, a step input value is present in each of the four cores.

Based on such observations, we design a specific overfitting mitigation scheme in which we iteratively remove some portions of the training data to search for better models. An obvious solution may be to remove all sections where there are simultaneous step inputs. However, doing this will remove most of the data, leaving us with an unusable training set. Furthermore, not all simultaneous step input occurrences are necessarily misleading. Removing certain sections indiscriminately may, in fact, harm the accuracy of the generated model. Therefore, a method of detecting and removing these misleading sections of data proves necessary.

To detect misleading sections of data, we employ a methodology that requires both a training set and validation set.

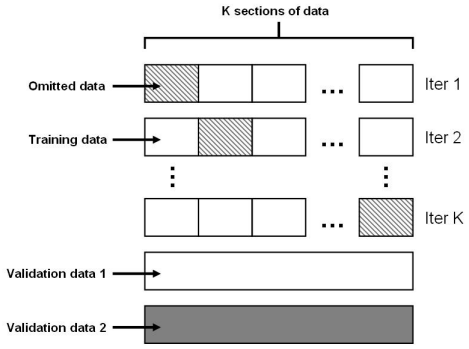

Figure 8: Overfitting solution algorithm

This could be done by splitting an existing data set in half, with one portion used for training and the other for validation. Such partitioning scheme is similar to the Early Stopping method [8]. As described in Fig. 8, the training set is further partitioned into K sections. In each iteration, a section of data is omitted, with the rest used as training data. A modeling error ϵ is computed in each iteration by comparing the model created using the training data i with the validation data set. In addition, the model is also compared to the entire training data set, resulting in a second modeling error value. The motivation for this comparison stems from the prior observation that the training waveform accuracy is lowered with low ordered models. It is further observed that removing sections of misleading data results in the improvement of the training waveform accuracy. Therefore, verification on two separate data sets provides a much greater insight into the predictive ability of the model. The final modeling error ϵ_i is thus calculated as the average of the errors for validation sets 1 and 2. Afterwards, the improved model can be chosen as the one with the lowest error value.

3.4 The the flow of the proposed algorithm

An outline of the proposed *ThermSID* algorithm is presented in Alg. 1. The algorithm requires a training set and verification set as inputs, which can be obtained by splitting an existing data set in half. These data sets are composed of power input traces and their corresponding measured temperature responses. In each iteration, two error calculations are performed against the verification and training temperature response waveforms. The final error for that iteration is calculated as the average of the two errors. The algorithm returns the state space matrices A, B, C, D of the improved system model.

Algorithm 1 *ThermSID with Overfitting Solution*(u_{trn}, y_{trn},u_{ver}, y_{ver})

Choose partition size K
Partition u_{trn} and y_{trn} into K sections
for $i = 1$ to K **do**
  Remove section i from u_{trn} and y_{trn} to form u_i and y_i
  Perform *ThermSID* using u_i and y_i to generate $model_i$
  Simulate $model_i$ using u_{ver}
  Compare results with y_{ver} and y_{trn} to obtain error ϵ_1 and ϵ_2
  Calculate error ϵ_i by taking the average of ϵ_1 and ϵ_2
end for
Choose smallest ϵ_s and obtain corresponding u_s and y_s
Perform *ThermSID* using u_s and y_s
Return state space matrices A, B, C, D

Compared to the previously proposed thermal modeling algorithm, *ThermPOF* [9], thermal modeling using *Therm-*

SID is more flexible and training data friendly. This is especially the case for measured data as it is difficult, if not impossible, to power just one component in many complex thermal systems with step like inputs. Also in *ThermPOF*, a change to log-scale is needed to better capture step response behavior. On the contrary, random power input traces and the resulting temperature responses of the desired system can be used directly as the input and output data sets for *ThermSID*, thus avoiding a log scale change for time during the training process.

4. EXPERIMENTAL RESULTS

This section verifies the effectiveness of the proposed thermal modeling method. We call the simple implementation of subspace system identification method, *Sid*. The proposed method is called *ThermSID* with the overfitting mitigation algorithm. We generate a model with *Sid* using the entire training data set, and then generate a second model by employing *ThermSID*. We then compare the accuracy of both models using a separate verification data set. We perform our tests on a quad-core microprocessor architecture shown in Fig. 1 from our industry partner Intel Corp. Both training and verification data sets were also provided by Intel. All algorithms were implemented in Matlab 7.6.

The training data set excites each core with step power inputs that range between 0 to 20 Watts for a duration of 0 to 1 seconds. The verification data also excites the system with step inputs, but only with values of 0 or 20 Watts. Fig. 9 and Fig. 10 show the training and verification power traces, respectively. We partition the training data set into 20 sections during the overfitting mitigation process.

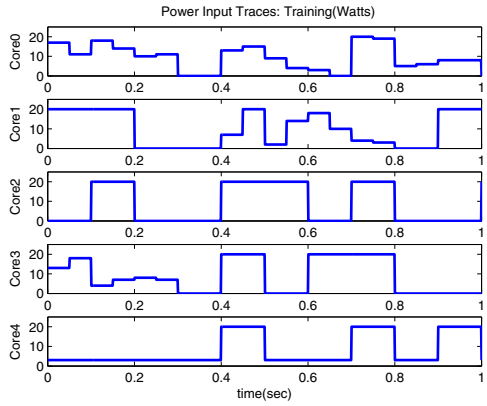

Figure 9: Random power input traces for cores 0 to 3 and cache (set 1)

We verify our algorithms by comparing their results with the temperature response data of the verification set. Fig. 11 and Fig. 12 show the temperature response waveforms on Core 2 and the cache. The dotted line represents the simulation results generated by *Sid*, while the dashed line represents the results generated by *ThermSID*. Lastly, the black line is the measured temperature response from the verification data. The figures clearly show the improvements in accuracy when the overfitting mitigation process is employed in generating the models.

Table 1 provides the percentage error statistics of cores 0 to 3 and the cache. The mean error and standard deviation are calculated for both *Sid* and *ThermSID* models. Results show that *ThermSID* significantly reduces the mean and standard deviation of errors in all regions. The most notable improvement occurs in the cache, where the mean error drops from 21.17 percent to 2.21 percent.

We then compare the performance of *ThermSID* against a previously proposed thermal modeling method *ThermPOF*.

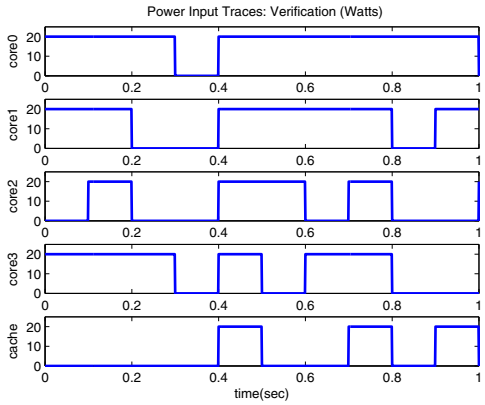

Figure 10: Random power input traces for cores 0 to 3 and cache (set 2)

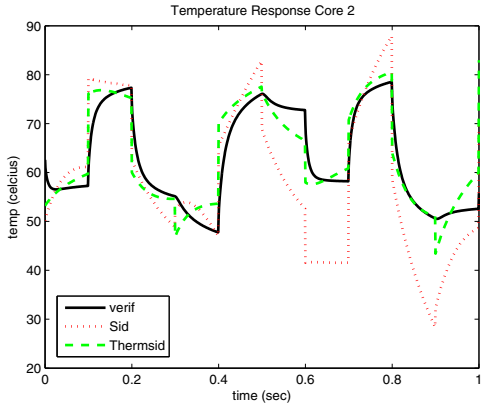

Figure 11: Temperature response of the verification data, *Sid*, and *ThermSID* for Core 2

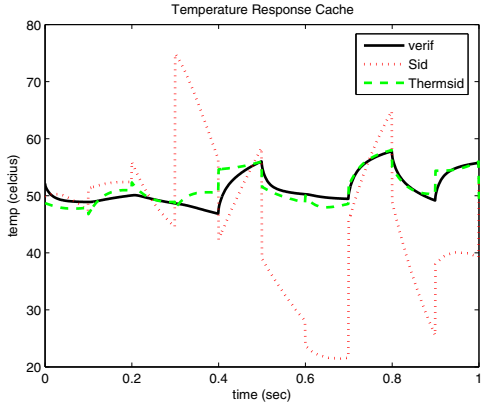

Figure 12: Temperature response of the verification data, *Sid*, and *ThermSID* for cache

Table 3 presents the error statistics of both methods. Note that since *ThermPOF* utilizes step responses to generate its model, we use a training data set with power inputs similar to that of Fig. 10 and their corresponding temperature response outputs. The models are then verified on a separate data set. With distinctly smaller mean error and standard deviation values, our results suggest that *ThermSID* exhibits better accuracy than *ThermPOF*.

We further verify the overfitting mitigation method by applying it to a second training data set with greater pre-

Table 1: Statistics of errors (%) between gvien and computed temperatures for verification set 1

core	Sid mean error	ThermSID mean error	Sid error std	ThermSID error std
Core 0	13.67	3.63	10.53	4.29
Core 1	8.51	4.03	7.10	4.32
Core 2	12.68	4.55	10.78	4.63
Core 3	15.17	5.72	13.50	4.97
Cache	21.17	2.21	18.64	2.10

Table 2: Statistics of errors (%) between given and computed temperatures for training set 2

core	Sid mean error	ThermSID mean error	Sid error std	ThermSID error std
Core 0	6.55	3.54	4.85	4.00
Core 1	6.06	2.70	5.94	2.96
Core 2	7.81	5.82	5.86	3.38
Core 3	9.40	7.58	6.61	7.19
Cache	9.37	4.77	5.12	3.32

dictive ability. As with the first test, the training data set is divided into 20 sections, and *Sid* is used to generate the model. Fig. 13 compares the temperature response of Core 1, from which we can see that the model created by *ThermSID* exhibits an improvement in accuracy. Table 2 shows that the improvements carry over to all regions of the core, with a decrease in both mean error and standard deviation. The results of our tests on training set 2 show that the overfiting mitigation scheme is still applicable even on data sets that exhibit a fair amount of accuracy.

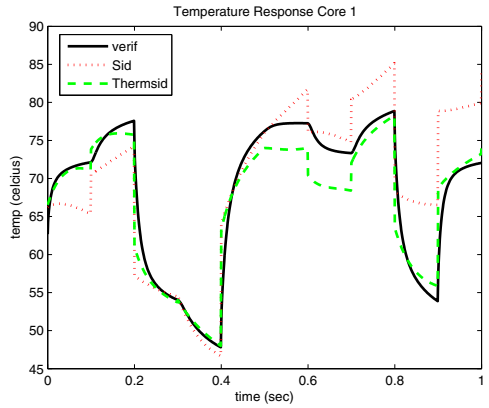

Figure 13: Temperature response of the verification data, *Sid*, and *ThermSID* for Core 1

Our experimental results show that utilizing the overfitting mitigation method in conjunction with the subspace based modeling method significantly increases the accuracy of its generated model. Such an optimization approach is useful when sets of training data are scarce, and the present data is unable to generate an acceptably predictive model. While more optimal methods of model verification exist, the overfitting mitigation approach enables the creation of an improved model under such restrictive circumstances.

Table 3: Statistics of errors (%) between given and computed temperatures: ThermSID vs ThermPOF

core	ThermPOF mean error	ThermSID mean error	ThermPOF error std	ThermSID error std
Core 0	0.74	0.16	0.29	0.11
Core 1	0.98	0.24	0.69	0.38
Core 2	3.11	0.34	1.31	0.46
Core 3	2.10	0.20	0.88	0.26
Cache	1.23	0.62	0.43	0.37

5. CONCLUSION

In this paper, we proposed a new architecture-level thermal modeling method, called *ThermSID*, which builds the thermal behavioral models from the measured or simulated thermal and power information. Based on the subspace system identification scheme, *ThermSID* builds the thermal models by first generating a Hankel matrix of Markov parameters, from which state matrices are obtained through minimum square optimization. Compared to existing behavioral thermal modeling algorithms, the proposed method is much more general in that it does not require step temperature responses, leading to greater flexibility during the modeling process. To overcome the overfitting problems of the subspace based method, an overfitting mitigation technique has been proposed to improve the model accuracy and predictive ability. Experimental results on a practical quad-core microprocessor show that *ThermSID* compares favorably with the existing *ThermPOF* method in accuracy, but offers much more flexibility and tractability for training data. Furthermore, the proposed overfitting mitigation technique is shown to significantly improve the predictability of the resulting thermal models.

6. REFERENCES

[1] "Intel multi-core processors (white paper)," 2006, http://www.intel.com/multi-core.

[2] "Multi-core processors - the next evolution in computing (white paper)," 2006, http://multicore.amd.com.

[3] "International technology roadmap for semiconductors(itrs) 2007 edition," 2007, http://public.itrs.net.

[4] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *Proc. of Intl. Symp. on High-Performance Comp. Architecture*, 2001, pp. 171–182.

[5] G. F. Franklin, M. L. Workman, and D. Powell, *Digital Control of Dynamic Systems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

[6] S. Gunther, F. Binns, D. Carmean, and J. Hall, "Managing the impact of increasing microprocessor power consumption," in *Intel Technology Journal*, First Quarter 2001.

[7] W. Huang, M. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy, "Compact thermal modeling for temperature-aware design," in *Proc. Design Automation Conf. (DAC)*, 2004, pp. 878–883.

[8] M. J. Kearns and U. V. Vazirani, *An introduction to computational learning theory*. Cambridge, MA, USA: MIT Press, 1994.

[9] D. Li, S. X.-D. Tan, and M. Tirumala, "Architecture-level thermal behavioral characterization for multi-core microprocessors," in *Proc. Asia South Pacific Design Automation Conf. (ASPDAC)*, 2008, pp. 456–461.

[10] Y. Li, B. C. Lee, D. Brooks, Z. Hu, and K. Skadron, "CMP design space exploration subject to physical constraints," *Proc. IEEE Int. Symp. on High Performance Computer Architecture (HPCA)*, pp. 15–26, 2006.

[11] P. V. Overschee and B. D. Moor, "N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems," *Automatica*, vol. 30, no. 1, pp. 75–93, 1994.

[12] P. D. Pedrod, "Bayesian averaging of classifiers and the overfitting problem," in *in Proc. 17th International Conf. on Machine Learning*. Morgan Kaufmann, 2000, pp. 223–230.

[13] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature aware microarchitecture," in *Proc. IEEE International Symposium on Computer Architecture (ISCA)*, 2003, pp. 2–13.

[14] W. Wu, L. Jin, J. Yang, P. Liu, and S. X.-D. Tan, "A systematic method for functional unit power estimation in microprocessors," in *Proc. Design Automation Conf. (DAC)*, June 2006, pp. 554–557.

[15] ——, "Efficient power modeling and software thermal sensing for runtime temperature monitoring," *ACM Trans. on Design Automation of Electronics Systems*, vol. 12, no. 3, pp. 1–29, 2007.