Fast Thermal Simulation for Run-Time Temperature Tracking and Management

Pu Liu, *Student Member, IEEE*, Hang Li, *Student Member, IEEE*, Lingling Jin, Wei Wu, Sheldon X.-D. Tan, *Senior Member, IEEE*, Jun Yang, *Member, IEEE*

***Abstract*— As power density increases exponentially, runtime regulation of operating temperature by dynamic thermal managements becomes necessary. This paper proposes two novel approaches to the thermal analysis at the chip architecture level for efficient dynamic thermal management. The first method, TMMSpectrum, is based on the observations that the power consumption of architecture level modules in microprocessors running typical workloads presents strong nature of periodicity. Such a feature can be exploited by fast spectrum analysis in frequency domain for computing steady state response. The second method, TMMPWC, is based on the observation that average power consumption of architecture level modules in microprocessors running typical workloads determines the trend of temperature variations. As a result, we can further speed up the thermal analysis by using piecewise constant average power inputs. To obtain the transient temperature changes due to initial condition and constant/average power inputs, numerically stable moment matching methods with enhanced pole searching methods is carried out to speed up on-line temperature tracking with high accuracy and low overhead. The resulting thermal analysis algorithm has linear time complexity in run-time setting when average power inputs are applied. Experimental results show that the resulting thermal analysis algorithms lead to 10x-100x speedup over the traditional integration-based transient analysis with small accuracy loss.**

***Index Terms*— Thermal analysis, temperature tracking, dynamic thermal management. moment matching, model order reduction.**

I. Introduction

As current IC technology enters nanometer regime, extremely high package density and operating frequency will lead to drastic increase of power density. The exponential power density increase will in turn lead to average chip temperature to raise rapidly [2]. Furthermore, local hot spots, which have much higher power densities than the average, make local temperature even higher.

Higher temperature has significant adverse impacts on chip performance and reliability. Excessive on-chip temperature leads to slower transistor speed, more leakage power consumption, higher interconnect resistance, and reduced reliability. It is believed that prompt real-time regulation of on-chip temperature by dynamic thermal management (DTM) is required for today's high-performance microprocessor and embedded systems [3], [21].

The basic idea of DTM is to dynamically reduce the temperature of some hot units (spots) in a chip via a suite of techniques such as activity migration, local toggling, dynamic voltage/frequency scaling [3], [21]. Architecture level thermal behavior cannot be seen at circuit or gate level at design time since different workloads generate different thermal profiles. Performing DTM at architecture level is advantageous in that it can capture the run-time behavior of the program, and quickly adapt to different features within or across different programs. Further, recent studies show that architecture level thermal managements at small performance degradation cost can significantly reduce the packaging costs typically designed for worst cases [9], [21], [22].

One of the most critical aspects of thermal modeling and simulation for DTM is to efficiently capture the temperature changes due to the variations of the power consumption caused by the runtime variation of an application or the differences across different applications at chip architecture level. DTM performed at run-time requires accurate real-time sensing of temperature for each functional blocks. Previous research relies on thermal diode-based sensor for on-line temperature tracking, which renders imprecision, delay and space overhead for hardware implementation [3], [8], [11]. These sensor noises could degrade DTM performance significantly due to conservative triggering of DTM [21]. One viable alternative solution to this problem is to use fast on-chip thermal estimation technique in software form to complement or even replace the thermal sensors for effective DTM application.

In this paper, we propose fast transient thermal simulation algorithms at architecture level for fast dynamic monitoring and thermal management. We present two algorithms for different application requirements. The first algorithm, **TMMSpectrum** (Thermal Moment Matching with Spectrum Analysis), is suitable for short time and more accurate temperature tracking with strong periodic power inputs; while the second method, **TMMPWC** (Thermal Moment Matching based on Piecewise Constant Power Inputs), is more efficient for long time and on-line temperature estimation for general power inputs. In the run-time setting, the TMMPWC method can achieve linear time complexity, which makes it very suitable for on-line thermal estimations. The two methods

Manuscript received August 8, 2005; revised on January 5, 2006. Some preliminary results of this paper appeared in Proc. Int. Conf. Computer-Aided Design (ICCAD'05) [14] and *Proc. Int. Conf. Computer Design (ICCD'05) [13]*. This work was recommended by Associate Editor Wim Schoenmaker.

Pu Liu, Hang Li, and Sheldon X.-D. Tan are with Department of Electrical Engineering, University of California, Riverside, Riverside, CA 92521 USA (e-mail: {uliu,hli,stan}@ee.ucr.edu)

Lingling Jin, Wei Wu and Jun Yang are with University of California, Riverside, Riverside, CA 92521 USA (e-mail: {ljin,wwu,junyang}@cs.ucr.edu)

This work is supported by NSF CAREER Award under Grant No. CCF-0448534 and NSF under Grant No. CCF-0541456, UC Senate Research Funds (05-06).

are mainly different in how the power inputs are modeled and handled. Experimental results show that the resulting thermal analysis algorithms lead to at least 10x-100x speedup over traditional integration-based transient analysis with small accuracy loss [21], [22].

The rest of the paper is organized as follows: Section II reviews existing thermal analysis algorithms, especially those at architecture level. Then we present the rationale behinds the new thermal analysis methods. Section III briefly mentions the architecture thermal modeling in [22], and our modification of the architecture as opposed to [22]. In section IV, we first describe basic computation steps in the proposed methods. Then we present the two proposed thermal analysis methods TMMSpectrum and TMMPWC. Section V shows how the proposed method can be applied to on-chip dynamic thermal management. Section VI present the theoretical time complexity analysis. The experimental results are summarized and compared to SPICE-like simulator in section VII to validate our method. The conclusion and future works are presented in section VIII.

II. Thermal Analysis Algorithm Reviews and Rationales for the New Algorithms

Many previous research works have been concentrated on thermal modeling and simulation at the circuit or gate level [5]. Due to the large volume of thermal components and power sources at full circuit or gate level, different schemes were proposed to increase the efficiency of thermal circuit simulation. Those schemes can be roughly classified into two categories. The first type of methods are based on the discretization on differential operators (finite deference method) or the field quality (finite element method). Examples are [24], [25], where the entire chip is discretized, and the heat transfer equation in partial differential form is solved by finite difference or finite element method. The main drawback of those methods is the huge sizes of the resulting thermal circuits due to volume meshing. Different techniques were proposed to solve with the extremely large thermal circuit, such as ADI in [25], and model order reduction in [24]. The second type of methods are based on Green function method [23], [26], which provides a faster yet less accurate thermal simulation than the above methods due to the simplified two dimensional modeling of the thermal problem.

Although many efficient algorithms have been proposed for circuit or gate level thermal analysis, less attention has been paid to thermal modeling and simulation at the chip-architecture level. Architecture level thermal behavior can not be seen at circuit or gate level at design time since different workloads generate different thermal profiles. Performing DTM at the architecture level is advantageous in that it can capture the run-time behavior of the program, and quickly adapt to different features within or across different programs. Further, recent studies show that architecture level thermal managements at small performance degradation cost can significantly reduce the packaging costs typically designed for worst cases [9], [21], [22]. An architecture level thermal modeling and simulation tool called HotSpot [21] was developed to exploit and study different DTM techniques in regulating microprocessor operating temperature for representative benchmark programs. HotSpot provides an accurate architecture level thermal modeling based on equivalent thermal circuit of thermal resistances and capacitances that correspond to the micro-architecture blocks and essential aspects of packaging. Component-wise temperatures are derived from the power consumptions generated by power simulations.

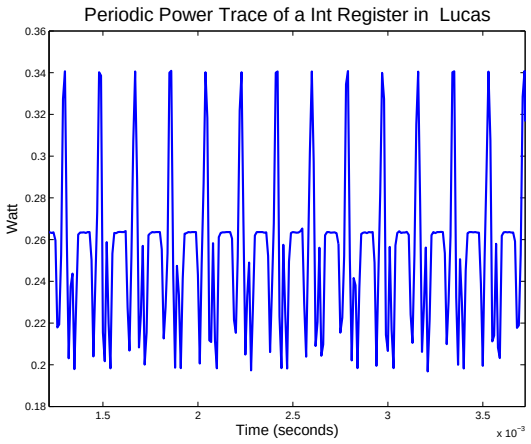

Fig. 1. The power trace of an integer register under *lucas* benchmark.

However, the efficiency of the HotSpot method for evaluating different DTM techniques depends on the execution time of transient thermal simulation throughout the program execution. HotSpot models the thermal behaviors based on the equivalent thermal circuit which consists of thermal resistors and capacitors. It uses conventional integration-based transient simulation conducted at each execution interval in order to get the whole temperature profile. When a program is loaded into HotSpot, its power consumption is first obtained at regular intervals. Then, temperature at every interval is calculated taking the temperature at the last interval and power values at the past few intervals. To obtain the temperature at certain running point of the program, all the previous temperature points should be generated since every point depends on its previous points. For a modern benchmark program which has tens to hundreds of billions of instructions, this method is not suitable for runtime temperature monitoring since it will bring significant performance and thermal overhead as studied in [11]. In other words, traditional integration-based numerical techniques are not suitable for fast run time thermal estimation.

Our first thermal analysis method, TMMSpectrum, is inspired by recent discoveries in the runtime behavior of programs over long periods of time. It has been shown that most program behavior is not random and actually presents strong periodic patterns due to the existence of loops or *phases* [18]–[20]. Many programs execute as a series of *phases* which characterizes certain program behavior at different times. Each phase may be very different from the others, while having a fairly homogeneous and periodic behavior within itself. This also reflects a periodic nature of each computation unit's output power consumption. Fig. 1 shows a snapshot of a typical power trace of an integer register running under program

Lucas over 14 time periods in SPEC CPU 2000 suite [1]. Each period contains about 20 power values collected over 0.2 million-instruction interval. This trace was obtained from simulating a 3GHz processor and thus, each period translates to $\sim$0.06ms and the entire trace represent $\sim$0.84ms of program execution. As we can see, there exists very strong periodicity even in this short amount of time.

TMMSpectrum exploits such periodic power traces of many hot modules at the architecture level to speed up the transient thermal simulation. For instance, the integer register file is typically the hottest module for most benchmarks [22]. Our proposed new algorithm is based on the fact that the transient behavior of the linear thermal system is the sum of the zero-input natural response and zero-state forced response. We applied two efficient algorithms to compute the two responses: (1) for periodic power trace input, frequency domain spectrum analysis is performed to calculate the steady-state response. (2) for transient behaviors of temperature, moment matching technique is used with the consideration of the initial states and the DC input values computed in the first step. Since the analysis is performed in pure frequency domain and the resulting system transient response is in an analytical closed-form expression in terms of time, the run time has been improved significantly.

To further improve the accuracy of our temperature simulation results over a long time for general power inputs, we propose the second fast thermal simulation algorithm, TMM-PWC, which is more suitable for on-line thermal tracking. The method is based on the observation that the average power in a certain amount of time determines the trend of temperature variations. This is especially true for power inputs with very large DC components and smaller high-frequency harmonics as we see in typically power inputs of hot modules. As a result, we can partition the simulation intervals into several intervals (pieces) and each interval is simulated sequentially based on its start and end times. By selecting appropriate interval lengths, we can capture the DC component change of the power trace over a long time without much overhead. Since the poles of the thermal circuits can be pre-computed off-line or at the initial stage, only the changing moments are computed on-line in the run time, which leads to linear-time thermal analysis method in the on-line setting.

III. Architecture Level Thermal Modeling

Generally speaking, the heat transfer phenomena is governed by the following differential equation [5]:

$$\rho C_p \frac{\partial T(\vec{r},t)}{\partial t} = \bigtriangledown \cdot [\kappa(\vec{r},T) \cdot \bigtriangledown T(\vec{r},t)] + g(\vec{r},t) \qquad (1)$$

where $T(K)$ is the temperature, $\rho(Kg/m^3)$ is the density of the material, $C_p(J/m^3 \cdot K)$ is the specific heat, $\kappa(W/m \cdot K)$ is the thermal conductivity, and $g(W/m^3)$ is the heat energy generation rate. The heat flow described by this differential equation has the similar format as that for electrical current, and there is a well-known duality between them. The heat flow passing through a thermal resistor is equivalent to the electrical current, and the temperature difference corresponds to the voltage difference. There is also the thermal-equivalent capacitance through which the heat is absorbed. Based on these observations, an equivalent thermal RC circuit will be derived and solved in dealing with thermal issues.

In the circuit level thermal RC circuit modeling, volume meshing is used to discretize the entire circuit structure, and the finite difference or finite element method is used to discretize equations(1). The resulting RC circuit is typically huge. At the architecture level, however, due to the limited components at floor-plan and unknown details of physical implementation, the corresponding RC model is compact, and the accurate extraction of thermal resistance and capacitance is critical to the application of thermal analysis.

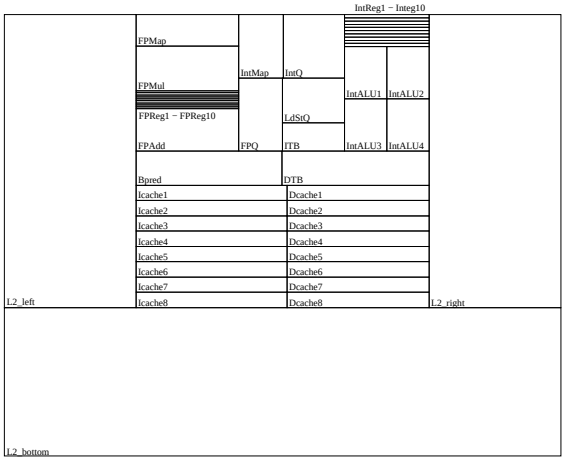

Fig. 2. Modified architecture of Compaq Alpha 21364.

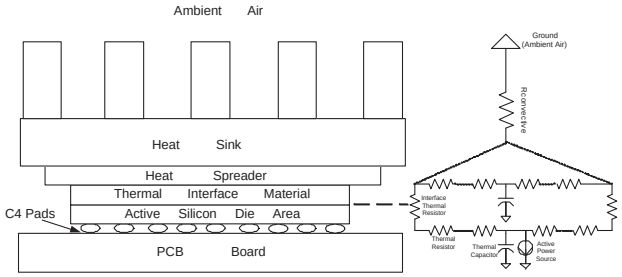

Fig. 3. Modern chip packaging structure and equivalent RC circuit modeling.

In this paper, we follow thermal modeling method at the architectural level thermal in [9], where a fairly accurate equivalent RC model, which is verified by other commercial tools, is developed from the floor-plan information. For a modern chip with CBGA packaging, heat sinks and cooling systems as shown in Fig. 3, there exists two main heat conduction paths, where the heat generated by active silicon die area can flow either through the convective ambient air, or the printed-circuit board. The primary RC circuit lies in the silicon die area, where the floor-plan information is provided to obtain the equivalent thermal resistance and capacitance. The floor-plan example we used in this paper is depicted in Fig. 2. The difference between our floor-plan model and the one in [22] is that we divide some critical computing components, such as Integer Register (IntReg) and Floating Point Register(FPReg), into more detailed pieces so that more

accurate temperature variation could be obtained in these temperature-critical components used during DTM. As shown in Fig. 2, the thermal resistance between two adjacent modules are determined by the common border length shared by them. Spreading/constriction resistances are also considered as in [12]. Each unit has a thermal capacitance to the thermal ground, which is determined by individual unit's area. And a scaling factor is needed to bridge the gap between this single-lumped capacitance and a distributed one. Besides the active die area, there are two additional heat spreader and heat sink layers lie underneath it. More component units are developed in the model corresponding to these two layers, but without active power sources for each component as appeared in the active die area.

Finally, in modeling the package to air interface, an equivalent convection resistance $R_{convection}$ is assigned, and a sustained power source is attached between the thermal ground (temperature of ambient air) and the package bottom. Calibration of this resulting model parameters is done as in [22], to provide a convergent results as compared to other commercial tools, as well as a good distribution of benchmarks behaviors in the final experimental results. The thermal circuit netlist is stamped into matrix representation, and processed in the circuit simulation phase discussed in Section IV.

IV. New Algorithm for Fast Transient Thermal Simulation

In this section, we first present the algorithm preliminary in subsection IV-A. Then we present the basic steps used in the two methods in subsections IV-B, IV-C and IV-D respectively. After this we present the outlines of TMMSpectrum and TMMPWC methods in subsections IV-E and IV-F respectively.

A. Algorithm Preliminary

For a general dynamic system

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \tag{2}$$

where $\mathbf{A}$ is the system matrix and $\mathbf{B}$ is the input selection matrix. $\mathbf{x}$ and $\mathbf{u}$ are the state variable and input vectors. The complete response is the sum of the zero-input response and the zero-state response starting from time t_0 as shown in Eq.(3).

$$\mathbf{x(t)} = \mathbf{e}^{\mathbf{A(t-t_0)}}\mathbf{x_0} + \int_{\mathbf{t_0}}^{\mathbf{t}} \mathbf{e}^{\mathbf{A(t-\tau)}}\mathbf{B}\mathbf{u}(\tau)\mathbf{d}\tau. \tag{3}$$

The first term on the right-hand side is the zero-input response due to the initial condition and the second term on the right-hand side is the zero-state response due to input sources $\mathbf{u(t)}$ only.

Our spectrum analysis of the typical power trace of many architecture modules shows that most of energy in the power trace concentrates on the DC as shown in Fig. 4 (Y axis is plotted in logarithm scale). As a result, we partition the power trace $\mathbf{u(t)}$ into two parts $\mathbf{u_1}$ and $\mathbf{u_2(t)}$. $\mathbf{u_1}$ is the DC component of the power input and $\mathbf{u_2(t)}$ is the periodic component of the power input where

$$\mathbf{u_2(t+T)} = \mathbf{u_2(t)}, \tag{4}$$

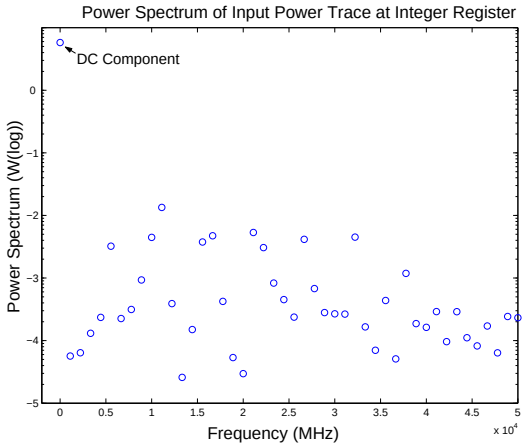

Fig. 4. The power spectrum of an integer register file in Lucas program.

and T is the time of the period in power inputs. We stress here that the practice power traces are not exact periodic over long time. But typically in a given phase (short time interval), changes between different periods are so small that approximating them as periodic inputs will not cause significant errors. As a result, Eq.(3) can be written as

$$\mathbf{x(t)} = \mathbf{e}^{\mathbf{A(t-t_0)}}\mathbf{x_0} + \int_{\mathbf{t_0}}^{\mathbf{t}} \mathbf{e}^{\mathbf{A(t-\tau)}}\mathbf{d}\tau\mathbf{B}\mathbf{u_1} + \int_{\mathbf{t_0}}^{\mathbf{t}} \mathbf{e}^{\mathbf{A(t-\tau)}}\mathbf{B}\mathbf{u_2}(\tau)\mathbf{d}\tau. \tag{5}$$

Now the first two terms on the right-hand side are only functions of the initial condition and the DC inputs of power trace. The third response of the circuit is stimulated by periodic power inputs.

The main idea of this work is to efficiently compute the first two responses by using moment matching method and the third periodic response by using fast spectrum analysis method in frequency domain to exploit the periodic patterns of power inputs. In the next two subsections, we present how the first two items are computed using the improved moment matching method.

One way to improve the simulation efficiency is to perform model order reduction (MOR) on the thermal circuits and simulate the circuits using the reduced models. However the projection based MOR methods is not suitable for circuits with many input and outputs like thermal and power grid circuits. One reason is that MOR need to compute the transfer functions between all the terminals. The number of transfer functions is n^2, where n is the number of terminals. In other words, we need to compute n moment series at each node.

For our problem, we only need to compute one moment series (due to the initial conditions at all the nodes) in each node as we do not need to compute all transfer functions (or admittances in the reduced admittance matrix). Also to overcome the numerical issue of the direct moment match method, a subspace projection method is used to compute the poles, which is to be discussed in Subsection IV-C. But we can do this by computing one transfer function as poles are shared by all the transfer functions. So the moment information is sufficient for the pole computation in the projection method.

B. Moment Matching Considering Initial Condition and DC Inputs

For the equivalent thermal circuits with thermal resistors and capacitors and power trace inputs, we use can use Modified Nodal Analysis to formulate the thermal circuit:

$$\mathbf{Gx} + \mathbf{C}\dot{\mathbf{x}} = \mathbf{Bu_1}, \tag{6}$$

here we only consider the DC component of power trace $\mathbf{u_1(t)}$. $\mathbf{C}$ and $\mathbf{G}$ are capacitive and conductive circuit matrices, $\mathbf{x}$ is the vector of node temperature, $\mathbf{u}$ is the vector of independent power sources, and $\mathbf{B}$ is the input selection matrix. In frequency domain, the Laplace transformation of the state equation (6) can be rewritten as

$$\mathbf{GX}(s) + s\mathbf{CX}(s) - \mathbf{CX_0} = \frac{1}{s}\mathbf{Bu_1}. \tag{7}$$

where $\mathbf{X_0}$ is the initial condition at $t = 0$.

In the traditional asymptotic waveform evaluation (AWE) based moment matching method [16], all transfer functions (the admittances in the reduced admittance matrices) between designated sources or ports are computed. As a result, n moment series have to be computed for each node for n input sources as each source stimulates a moment vector at every other node. In our problem, we only compute one moment series at each node as we only consider the response from the initial conditions and constant DC power inputs at all the nodes. As a result, the computation costs of the proposed method is not related to the number of sources.

Specifically, let $\tilde{\mathbf{X}}(\mathbf{s}) = \mathbf{sX}(\mathbf{s})$, then the above equation becomes to

$$\mathbf{G}\tilde{\mathbf{X}}(s) + s\mathbf{C}\tilde{\mathbf{X}}(s) = s\mathbf{CX_0} + \mathbf{Bu_1} \tag{8}$$

We then expand the $\tilde{\mathbf{X}}(s)$ using Taylor's series at $s = 0$, to have

$$\begin{aligned}\mathbf{G}(\mathbf{m_0} + \mathbf{m_1 s} + \mathbf{m_2 s^2} + ...) + \mathbf{sC}(\mathbf{m_0} + \mathbf{m_1 s} + \mathbf{m_2 s^2} + ...) \\ = s\mathbf{CX_0} + \mathbf{Bu_1}\end{aligned} \tag{9}$$

We then obtain the recursive moment computation formula as follows:

$$\begin{aligned}&\mathbf{m}_0 = \mathbf{G}^{-1}\mathbf{Bu_1}\\ &\mathbf{m}_1 = -\mathbf{G}^{-1}\mathbf{C}(\mathbf{m}_0 - \mathbf{X_0})\\ &\mathbf{m}_2 = -\mathbf{G}^{-1}\mathbf{Cm}_1\\ &\vdots\\ &\mathbf{m}_{2q} = -\mathbf{G}^{-1}\mathbf{Cm}_{2q-1}\end{aligned} \tag{10}$$

After all the moments are computed, the response at each node can be written as

$$\mathbf{X}(s) = \frac{1}{s}\mathbf{m}_0 + \mathbf{m}_1 + s\mathbf{m}_2 + s^2\mathbf{m}_3 + \cdots + s^{2q-1}\mathbf{m}_{2q} + \ldots \tag{11}$$

The first term on the right-hand side is a step response in time domain and the rest of the moments then are used to find the rational approximation via Padé approximation. In order to find a q^{th} order Padé approximation, the first $2q$ moments are needed. Then we obtain $2q$ moment matching equations of the response at node l:

$$\begin{aligned}&-(k_1 + k_2 + \cdots + k_q) = m_0 - x_0\\ &-(\tfrac{k_1}{p_1} + \tfrac{k_2}{p_2} + \cdots + \tfrac{k_q}{p_q}) = m_1\\ &\vdots\\ &-(\tfrac{k_1}{p_1^{2q-1}} + \tfrac{k_2}{p_2^{2q-1}} + \cdots + \tfrac{k_q}{p_q^{2q-1}}) = m_{2q-1}\end{aligned} \tag{12}$$

where p_i and k_i are the ith pole and residue in the partial fraction form of the response at node l

$$x_l(s) = \frac{1}{s}m_0 + \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} + ... \tag{13}$$

Classic AWE method [16] computes the poles and residues directly from the moments at each node. Specifically AWE first computes the q poles by writing $x_l(s)$ into the following polynomial form:

$$x_l(s) = \frac{1}{s}m_0 + \frac{a_{q-1}s^{q-1} + ... + a_1 s + a_0}{b_q s^q + ... + b_1 s + 1}$$

and then equaling it with $x_l(s)$ in the moment form

$$x_l(s) = \frac{1}{s}m_0 + m_1 + m_2 s + ... + m_{2q}s^{2q-1}$$

Then we can obtain q equations to solve for the q coefficients $b_1 ... b_q$ of the q^{th} order denominator polynomial. Once we know the denominator polynomial, its q roots are the poles $p_1, ...p_q$ and are found by any rooting finding numerical method. Once the q poles are known, it uses (12) (using only q equations) to solve for the q residues. We repeat the process to compute the residues and poles for all other nodes as we obtain the moments at all the nodes in (10). However, AWE method suffers the numerical problems as high order moments numerically lose the large pole information very quickly as shown in (12) where moments are inverse power function of poles.

Instead we propose a more numerical stable method to compute the poles, which is based on the subspace projection based method as shown in next subsection. After the poles are computed, all the residues k_i are still computed using q equations from Eq.(12). The time domain responses are trivially obtained by taking inverse Laplace transformation of $x_l(s)$. Note also that since the transient responses start with an initial condition, the initial conditions need to explicitly be enforced as shown in the first equation in Eq.(12).

Since most of the power energy is in the DC inputs, the response computed by moment matching can be close to the exact response, and the spectrum analysis solution typically adds small transient changes in the node temperature as observed by our experiments.

C. Numerically Stable Estimation of Poles by the Projection Based Model Reduction

Traditional moment matching method [17] may produce unreliable poles (positive poles) when computing (13) from (11) due to numerical problems. A better way of finding poles is by projection based model order reduction, where moments are orthonormalized and are used to build a projection matrix. The projection matrix then is used to reduce the original circuit matrix by congruence transformation, which can ensure that

the reduced system is passive (thus stable) [6]. Also by using this method, we only require q moments to find q poles.

Specifically, we obtain the first q moment vectors through (10). Then we form the following $N \times q$ matrix where each moment vector is a column.

$$M = [\mathbf{m}_0, \mathbf{m}_1, \cdots, \mathbf{m}_{q-1}]_{N \times q} \tag{14}$$

where $q \ll N$, and N is the number of temperature variables (nodes) in the thermal circuit and also the dimension of the moment vectors. Then we orthonormalize M into a $N \times q$ projection matrix V such that columns in V are mutually orthogonal, i.e. $v_i^T v_j = \delta_{ij}, i \neq j$. Such an orthogonalization process can be easily carried out by using *Gram-Schmidt* method or other numerically stable processes like Arnoldi methods [7]. Once we obtain the projection matrix V, the original circuit matrix $\mathbf{G}$ and $\mathbf{C}$ in (6) can be reduced to two $q \times q$ order reduced matrices by the *congruence transformation*:

$$\hat{\mathbf{G}} = V^T \mathbf{G} V, \quad \hat{\mathbf{C}} = V^T \mathbf{C} V \tag{15}$$

After this reduction process, the eigenvalues of matrix $\hat{\mathbf{G}}^{-1}\hat{\mathbf{C}}$ will be related to the dominant poles we are looking for as:

$$p_i = -\frac{1}{\lambda_i} \tag{16}$$

where p_i and λ_i are the ith pole and eigenvalue. This can be easily obtained by performing the eigen-decomposition of $\hat{\mathbf{G}}^{-1}\hat{\mathbf{C}}$. Once all the poles are computed, we then compute the residues at node x using equations in (12).

We note that the orthonormalization process can also be carried out by more numerically stable methods like Arnoldi method with modified Gram-Schmidt and double orthogonalization, Lanczos method with modified Gram-Schmidt to further improve the numerical stability of the proposed method [4], [7].

The proposed method guarantees stability of the responses as all the poles computed are stable pole (less than zero in their real part) [10], [15] due to the nature of congruence transformation and the MNA formulation of the original thermal circuit matrices.

D. Spectrum Analysis in Frequency Domain

In this subsection, we discuss spectrum analysis method for periodic zero state steady response of the thermal circuits under periodic input power traces. The basic idea is to transform the input signals into frequency domains via discrete Fourier transformation (DFT) and then compute the responses of dominant Fourier coefficients or harmonics. The resulting response Fourier coefficients are then transformed back to time-domain to obtain the steady-state time responses.

In discrete Fourier transformation, if we sample k points in the time domain, we have k harmonics in the frequency domain. Then we solve for responses of each harmonics on the thermal circuits as shown below:

$$\begin{array}{c} \mathbf{A}(\omega_1)\mathbf{X}(\omega_1) = \mathbf{P}(\omega_1) \\ \mathbf{A}(\omega_2)\mathbf{X}(\omega_2) = \mathbf{P}(\omega_2) \\ \vdots \\ \mathbf{A}(\omega_k)\mathbf{X}(\omega_k) = \mathbf{P}(\omega_k) \end{array} \tag{17}$$

where $\omega_i, i = 1...k$ are the harmonic frequencies, $\mathbf{A}(\omega)$ is the MNA matrix stamped with these thermal elements at frequency ω. $\mathbf{X}(\omega)$ is the temperature vector at each component at frequency ω, and $\mathbf{P}(\omega)$ is the computed harmonics of the power input vector from DFT.

By this transformation, the computational cost can be reduced significantly especially for the long cycle simulation. The reason is that we only compute the stead state using one period and the number of MNA equations depends on only the number of sampling points. If the faster simulation is required, we can reduce the sampling frequency to speed up the calculation at the cost of more accuracy loss. But at the same time we make sure that the sampling rate should satisfy the Shannon's sampling theorem. Finally we convert the frequency response to time domain responses by inverse discrete Fourier transformation.

Note that the final steady state temperature will be the temperature we computed using spectrum analysis and environmental temperature, which is treated as the ground voltage in our equivalent thermal circuits.

E. TMMSpectrum Method

In this subsection, we present the flow of the TMMSPectrum method.

TMMSPECTRUM ALGORITHM

1. Compute the poles (using the steps in Subsection IV-C) and residues based on the moment matching method in Subsection IV-B.
2. Compute the steady response from periodic power inputs in Subsection IV-D

The actual temperature response is the sum of the zero-input response (stage 1) and the zero-state response (stage 2).

Since the solution from the spectrum analysis in frequency domain is the steady state solution, which means it will happen at infinity time when system response becomes stable, it is less accurate for response at $t = 0$. But the difference is very small as energy in the periodic inputs are typically very small compared to the DC components. This has been observed and verified by our experimental results.

F. TMMPWC Method

In this subsection, we present the flow of the TMMSpectrum method.

TMMPWC ALGORITHM

1. Partition the time interval into many smaller intervals based on the input power patterns.
2. Compute the average power inputs for each intervals.
3. Compute the poles (using step in Subsection IV-C) and residues based on the moment matching method in Subsection IV-B for each time intervals. The ending responses of one interval will be the initial conditions for the next interval.

The final transient results are the simulated responses from all intervals for the whole time interval.

In the run-time setting, the pole computation step in stage 3 can be pre-computed in the initial stages and the equations

Eq.(12) used to solve residues can also be LU decomposed in the initial stage. The resulting TMMPWC will be linear in terms of number nodes as shown Section VI.

V. Application to Dynamic Thermal Management

Our techniques have great benefits to performing DTM at the architecture level. For instance, given an initial thermal setting and the fact that a phase of the program is periodic, our technique can predict if the current phase will reach a critical temperature and if yes, how soon it will happen. Thus, before such a critical temperature is reached, effective DTMs such as dynamic voltage/frequency scaling, local toggling, and activity migration can be carried to cool down the soon-to-be hot modules preventing them from entering the temperature critical stage. This can be easily computed using our proposed algorithms. In some situations, the DC component stripped periodic power trace only adds small distributions on the general transients of the temperature. The moment matching responses can be used to give fairly accurate time estimation.

Another important question of DTM is whether a program having long time of periodic behavior will ever reach a critical temperature, i.e., what is the steady thermal state. If the steady state is well below a temperature threshold, then there is no need to perform DTM as long as the same periodicity holds. Our moment matching algorithm can directly compute such an asymptotic steady state from the very beginning of the period, producing fast and accurate prediction. On the other hand, if the steady state is very close to some critical temperature, only the pure spectrum analysis is needed to find out the perturbations around the steady temperature. In that case, DTM is necessary whenever the temperature pulses surpass the threshold.

If there are several phases involved in the input power traces, we can compute the responses for each phase using the proposed methods and combine them to report the temperature profile as done in the TMMPWC method.

VI. Time Complexity Analysis

A. TMMSpectrum Method

For thermal circuit with N nodes, if only a few moments ($q \ll n$) are required, the time complexity of TMMSpectrum is about

$$O(\text{TMMPWC}) + O(kN^{1.5}) \tag{18}$$

Since the TMMPWC will first be performed and the only overhead here is the complexity of spectral analysis. $kN^{1.5}$ is the cost of the spectrum analysis on k sampling points in Eq.(17), and $O(\text{TMMPWC})$ is the cost of TMMPWC, which will be analyzed next.

B. TMMPWC Method

For TMMPWC, the time complexity analysis is more involved. As pointed earlier, the major computation cost of TMMPWC is the moment computation, the orthonormalization process for pole computation, and residue computation. Still, for thermal circuit with N nodes, if only a few moments ($q \ll n$) are required, the time complexity of TMMPWC can be approximated as:

$$O(N^{1.5})+O(qN^*)+O(q^2N)+O(qN)+O(2q^3)+O(q^3)+O(q^2N)+O(qN) \tag{19}$$

where $N^{1.5}$ is the time complexity for LU decomposition for a sparse matrix. The first qN^* is the cost of q forward and backward substitutions for solving q moments in (10) for sparse matrices, where N^* is the number of non-zeros in the L and U matrices, which typically are $2N$ for sparse matrices. q^2N is the cost for the orthonormalization process. The next qN term is the cost for the congruence transformation in (15). $2q^3$ is the cost to get the inverse matrix of the reduced dense matrix $\hat{G}$ and to perform the eigen-decomposition of $\hat{\mathbf{G}}^{-1}\hat{\mathbf{C}}$. The second q^3 is the cost for the LU decomposition of the residue-solving matrix in (12). The second q^2N term is the cost for solving residues for N nodes using backward and forward substitutions in (12). The last term is the cost to compute the time domain response for all N nodes in specific time point.

Above we show the computation cost of the entire TMMPWC process. However, the poles are thermal circuit information, and therefore do not change with the power inputs. Therefore, the poles only need to be computed once, and reused for each following interval. Hence, the analysis process for each interval only involves the computation of moments and residues. Given the fact that we can reuse matrices $\mathbf{G}^{-1}$, and $\mathbf{G}^{-1}\mathbf{C}$ in (10) and (12) in these processes, the serial processing of partitioned input power trace should be controllable, and linear to the number of intervals. For each interval, the computing cost of the temperature changes in real time can be written as

$$O(qN^*) + O(q^2N) + O(qN) \tag{20}$$

where qN^* for computing the moments and q^2N computes the residues and qN computes the time domain response at specific time point. Notice that the time complexity become linear in terms of the size of thermal circuits in run-time temperature computation.

Notice that the time complexities of both analysis are independent of time intervals or number of time steps used in the traditional integration based transient simulation, which is the major advantage and speedup over traditional methods.

VII. Experimental Results

The proposed algorithm has been implemented in Matlab. We use the modified Compaq Alpha 21364 microarchitecture in Fig. 2 for generating the power traces and equivalent thermal models similar to that in [22]. To perform fairly comparison, we also implemented traditional integration based thermal simulation method used in HotSpot in Matlab in order to compare the accuracy and the speed difference between the two approaches.

A. Results for TMMSpectrum

We evaluate our results using benchmarks from the SPEC CPU 2000 suite [1]. We select 3 programs `Art`, `Lucas`, and

Wupwise and run for 10 billion instructions and simulate the periodical portions. Since the hottest unit is the integer register file, we compute the temperature changes of one typical integer register file as an example.

Table I summarizes the statistics of the three programs and experimental parameters. Column #II is the number of instruction intervals, which is about $10\mu s$, Column *#S* is the number of samplings used in the spectrum analysis. Columns *CPU (HotSpot)* and *CPU (TMMSpectrum)* are the CPU times for the traditional simulation method used in HotSpot and our proposed method. The CPU time is in seconds. The simulated thermal circuit consists of about 166 nodes. Although the thermal circuit is small, given the very long power input trace (billions of instruction cycles), the simulation time of HotSpot will still be long as shown in Table I. In the test cases, 6 poles are computed from projection based MOR method to compute the transient response. We find 6 poles typically are good enough to give fairly good results.

TABLE I
PERFORMANCE EVALUATION OF TMMSPECTRUM ON SPEC CPU2000 PROGRAMS.

Program	#II	#S	CPU (HotSpot)	CPU (TMMSpectrum)
Lucas	20000	18	183.74	0.78
Lucas	30000	18	219.43	0.77
Art	10000	614	72.23	15.67
Art	30000	614	242.49	15.55
Wupwise	10000	24	66.31	1.11
Wupwise	30000	24	515.02	1.10

From Table I, we can see that the proposed method have 10x to 100x speedup over the traditional simulation method.

Fig. 5 shows the transient temperature change comparison under Lucas benchmark for about 20000 simulation points where each point is $3.3\mu s$. Fig. 6 shows the transient temperature change comparison under Art benchmark for about 10000 points. Fig. 7 shows the transient temperature change under Wupwise benchmark. It can be seen that the results from our new algorithm match fairly well with the HotSpot simulation results but with at least 10X shorter CPU time as shown in Table I. Note that given longer simulation interval, the speedup will be further increased as the new method dose not depend on the number of simulation intervals.

For the results shown in Fig. 7, we only use 6 poles to compute the transient response at each node. However, we can reliably compute more poles via projection based method we described in section IV-C. If we use more poles to represent the system model, for example, 8 poles, the transient simulation results are better with respect to the responses from HotSpot (especially when t increases), which is shown in Fig. 8.

To show the detail waveforms due to periodic power trace, we enlarge Fig. 6, which is shown in Fig. 9. It can be seen that the temperature indeed changes periodically over the time. The HotSpot and the proposed new method match each other very well.

B. Results for TMMPWC

As mentioned earlier, proper interval partitioning should be performed for extremely long power trace to avoid the error

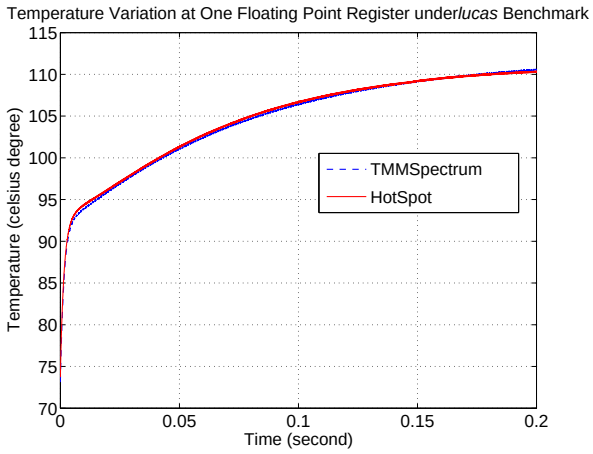

Fig. 5. The temperature comparison between TMMSpectrum and HotSpot under Lucas program.

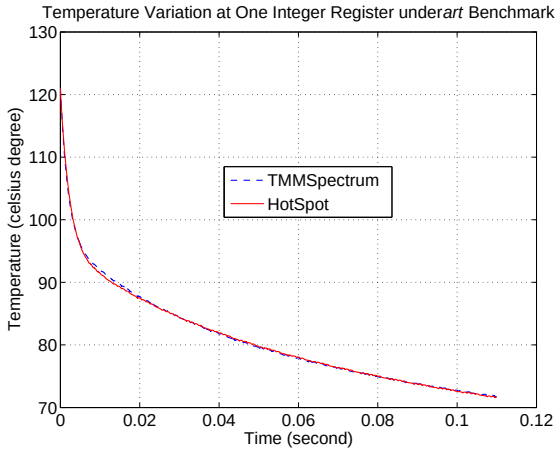

Fig. 6. The temperature comparison between TMMSpectrum and HotSpot under Art program.

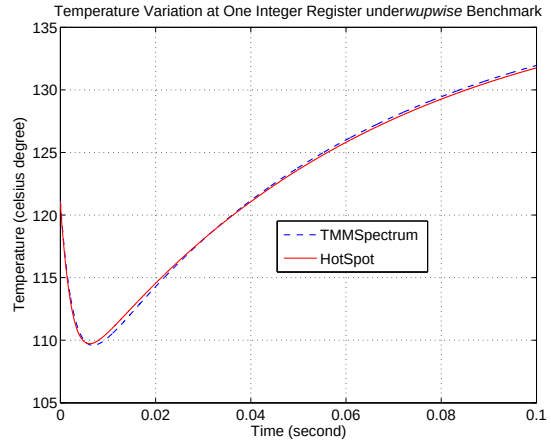

Fig. 7. The temperature comparison between TMMSpectrum and HotSpot under Wupwise program using 6 poles.

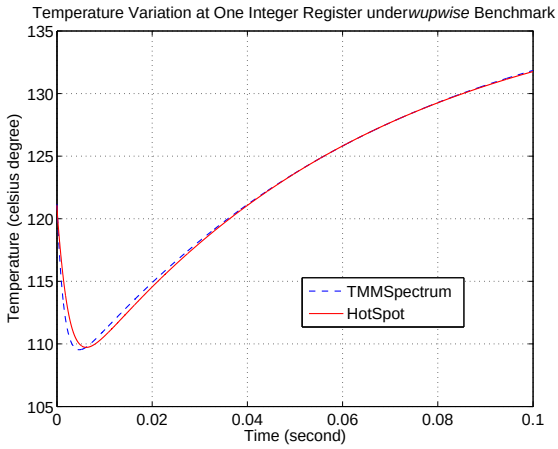

Fig. 8. The temperature comparison between TMMSpectrum (using 8 poles) and HotSpot under `Wupwise` program using 8 poles.

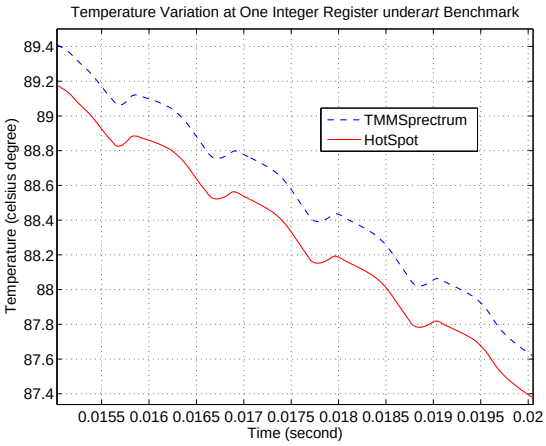

Fig. 9. The detailed periodic temperature comparison between TMMSpectrum and HotSpot under `Art` program.

caused by average power drifting, otherwise the temperature computed by the proposed method may have noticeable differences from the ones given by HotSpot for very long power traces. The main reason is that the periodic power inputs may change slowly over the time. As a result, the DC components may drift from the value we use for the moment matching at $t = 0$. To resolve this problem, we can partition the simulation intervals into several intervals and each interval is simulated sequentially based on its start and end times. The goal here is to make sure that the DC components from periodic power trace drifts are very small so that the moment matching method is accurate enough. For very long simulation times, interval-by-interval simulation improves the simulation accuracy at a small computing cost, as we analyzed in section VI.

We evaluate our results by running 10 benchmark programs from SPEC CPU 2000 suite [1]. In our experiments, we run all the benchmarks on a 3GHz processor, which yields $0.33ns$ per cycle period. We sample each component's output power at 10K-cycle intervals, corresponding to $3.3\mu s$ in each time interval. We first arbitrarily set the interval window size to be 1500 intervals to evaluate the performance of TMMPWC. On the other hand, SPICE and HotSpot will run the entire power trace interval-by-interval to derive the whole temperature profile.

Table II summarizes the statistics of these programs and experimental results. Column 2 is the number of instruction cycles. Columns 3 and 4 summarize the CPU times for TMMPWC and Matlab-based SPICE, with speed-up ratio listed in column 5. The last two columns give the average and maximum errors of our algorithm compared to that given by HotSpot for each benchmark program.

We notice that our algorithm achieves almost 100X speed-up over SPICE. Considering the faster run time provided by Linux workstation for HotSpot, the actual run time speed-up of TMMPWC (using Matlab) over HotSpot will be larger. For accuracy evaluation, the average and maximum errors compared to HotSpot among all these 10 benchmarks are only 0.13°C and 0.37°C, which provides a highly accurate temperature prediction for on-line DTM application. Here we changed the moment matching method to the projection method described in section IV-C and in all the test cases, 7 poles are computed for the transient response analysis.

To show the detailed waveforms comparison of TMMPWC with HotSpot results, we pick a run-time window of the temperature variation at Integer Register File under program `gcc`. Also, we didn't apply the spectrum analysis in this experiment, because as mentioned earlier, in most cases the dominant energy of the power trace is in its DC component, and ignoring all the high frequencies will have very little impact on the temperature change trend. Indeed, as illustrated in Fig. 11, each point calculated by TMMPWC matches very well with the corresponding temperature point in HotSpot curve, and the level of precision will be more than enough for most DTM techniques.

We also study the effect of interval window sizes on the performance of TMMPWC. Fig. 10 depicts the run-time, average and maximum errors for each TMMPWC run under different window sizes. We select benchmark program `wupwise` as an example. From the figure we can see, that the execution time will decrease rapidly with the increasing window size. This is because of the decreased interval number to be calculated by TMMPWC. The maximum error from TMMPWC as compared to HotSpot will increase with a larger window size, which is due to a less accurate average power estimation within that period of time. However, the error is less than 0.7°C for the maximum window size of 6000. And the average error almost stays the same for all window sizes around only 0.1°C. Compared to a real temperature deviation of 2°C offered by temperature sensor in previous DTM scheme [22], our new algorithm provides a viable and reliable on-line temperature estimation for DTM applications.

VIII. Conclusions and Future Works

In this paper, we proposed two efficient thermal analysis methods for the architecture level dynamic thermal monitoring and management. The first method, TMMSpectrum, exploits the periodic patterns in power consumptions of the architecture modules in microprocessors and embedded high-performance

TABLE II

PERFORMANCE EVALUATION OF TMMPWC ON SPEC CPU2000 PROGRAMS.

Program	Ins. Cycles(billion)	CPU(TMMPWC)(s)	CPU(SPICE)(s)	Speed-up	Avg. Err.(°C)	Max. Err.(°C)
gcc	28.5	55	4357	79	0.09	0.37
wupwise	56.3	126	8615	68	0.10	0.42
eon	27.3	55	4166	75	0.11	0.27
gzip	24	55	3669	66	0.11	0.30
bzip	55.7	116	8514	73	0.05	0.34
lucas	46.7	90	7131	79	0.16	0.53
mesa	20.0	42	3065	72	0.17	0.34
parser	30.7	67	4700	70	0.13	0.43
swim	11	23.6	1690	72	0.3	0.49
vortex	74.3	147	11360	77	0.074	0.27
Average	-	-	-	-	**0.13**	**0.37**

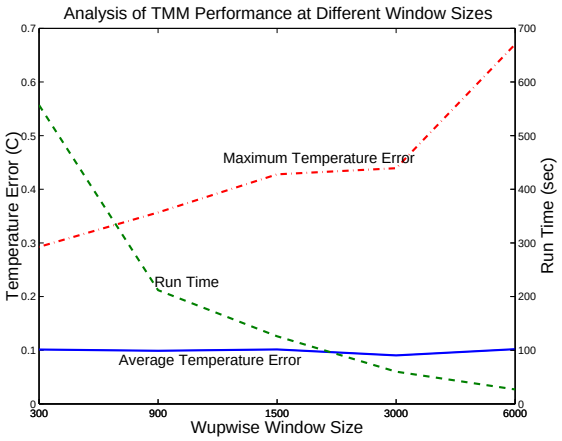

Fig. 10. TMMPWC performance evaluation under different window sizes.

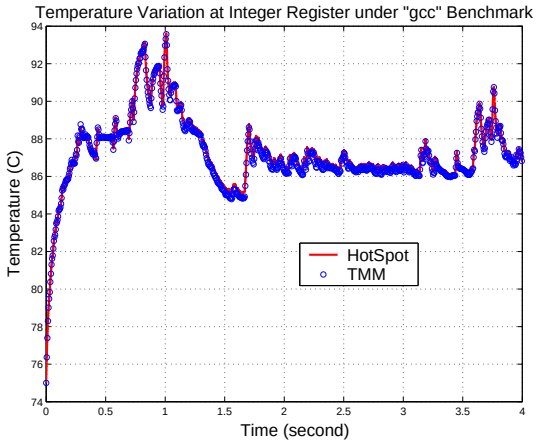

Fig. 11. Temperature comparison between TMMPWC and HotSpot under `gcc` program.

systems. We proposed to use spectrum analysis in frequency domain to compute the periodic responses of temperatures. The second method, TMMPWC, exploits the fact that average power consumption of architecture-level modules in microprocessors running typical workloads determines the trend of temperature variations. As a result, we can further speed up the thermal analysis by using piecewise constant average power inputs. To compute transient response due to initial conditions and constant/average power inputs, numerically stable moment matching method with enhanced pole searching methods has been carried out. The resulting fast thermal analysis algorithms lead to 10x-100x speedup over traditional integration-based SPICE-like HotSpot transient simulation with small accuracy loss. TMMPWC in the run-time setting becomes linear in terms of circuit sizes and is well suited for on-line thermal estimation for the dynamic thermal management.

In the future, we will integrate our thermal simulation engine with the DTM optimizer described in section V to complete the architecture level DTM framework.

IX. ACKNOWLEDGEMENTS

The authors would like to thank Associate Editor Dr. Wim Schoenmaker for his encouragement and appreciation of this work. We would also be grateful to two the anonymous reviewers of this paper for their comments and suggestions that helped improve this work.

REFERENCES

[1] Http://www.spec.org/cpu2000/CFP2000/.

[2] "International technology roadmap for semiconductors(itrs), 2004 update," 2001, http://public.itrs.net.

[3] 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.

[4] M. Celik, L. Pileggi, and A. Odabasioglu, *IC interconnect analysis*. Kluwer Academic Publishers, 2002.

[5] Y.-K. Cheng, C.-H. Tsai, C.-C. Teng, and S.-M. Kang, *Electrothermal Analysis of VLSI Systems*. Kluwer Academic Publishers, 2000.

[6] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by pade approximation via the lanczos process," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 5, pp. 639–649, May 1995.

[7] G. H. Golub and C. F. V. Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: The Johns Hopkins University Press, 1989.

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

[9] 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.

[10] K. J. Kerns and A. T. Yang, "Stable and efficient reduction of large, multiport RC network by pole analysis via congruence transformations," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, no. 7, pp. 734–744, July 1998.

[11] K. Lee and K. Skadron, "Using performance counters for runtime temperature sensing in high performance processors," in *the Workshop on High-Performance, Power-aware Computing(HP-PAC), in conjunction with the 2005 International Parallel and Distributed Processing Symposium*, Apr. 2005.

[12] S. Lee, S. Song, V. Au, and K. Moran, "Constricting/spreading resistance model for electronics packaging," in *Proc. ASME/JSME Thermal Engineering Conference*, Mar. 1995, pp. 199–206.

[13] H. Li, P. Liu, Z. Qi, L. Jin, W. Wu, S. X.-D. Tan, and J. Yang, "Efficeint thermal simulation for run-time temperature tracking and management," in *Proc. IEEE Int. Conf. on Computer Design (ICCD)*, Oct. 2005, p. to appear.

[14] P. Liu, Z. Qi, H. Li, L. Jin, W. Wu, S. X.-D. Tan, and J. Yang, "Fast thermal simulation for architecture level dynamic thermal management," in *Proc. Int. Conf. on Computer Aided Design (ICCAD)*, Nov 2005, pp. 639–644.

[15] A. Odabasioglu, M. Celik, and L. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 645–654, 1998.

[16] L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 352–366, April 1990.

[17] L. T. Pillage, R. A. Rohrer, and C. Visweswariah, *Electronic Circuit and System Simulation Methods*. New York: McGraw-Hill, 1994.

[18] T. Sherwood, E. Perelman, and B. Calder, "Basic block distribution analysis to find periodic behavior and simulation points in applications," in *the International Conference on Parallel Architectures and Compilation Techniques(PACT2001)*, 2001, pp. 3–14.

[19] T. Sherwood, E. Perelman, G. Hamerly, S. Sair, and B. Calder, "Discovering and exploiting program phases," in *IEEE Micro:Micro's Top Picks from Computer Architecture Conferences*, 2003, pp. 84–93.

[20] T. Sherwood, S. Sair, and B. Calder, "Phase tracking and prediction," in *Proc. IEEE International Symposium on Computer Architecture (ISCA)*, 2003, pp. 45–57.

[21] 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.

[22] ——, "Temperature aware microarchitecture: Extended discussion and results," in *University of Virginia, Dept. of Computer Science, Technicial Report CS-2003-08*, Apr. 2003.

[23] B. Wang and P. Mazumder, "Fast thermal analysis for vlsi circuits via semi-analytical green's function in multi-layer materials," in *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS)*, 2004.

[24] T. Y. Wang and C. C. Chen, "3-D thermal-ADI: a linear-time chip level transient thermal simulator," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 12, pp. 1434–1445, Dec. 2002.

[25] ——, "Spice-compatible thermal simulation with lumped circuit modeling for thermal reliability analysis based on model reduction," in *Proc. Int. Symposium. on Quality Electronic Design (ISQED)*, 2004, pp. 357–362.

[26] Y. Zhan and S. Sapatnekar, "Fast computation of the temperature distribution in vlsi chips using the discrete cosine transform and table look-up," in *Proc. Asia South Pacific Design Automation Conf. (ASPDAC)*, Jan. 2005, pp. 87–92.