3D-TTP: Efficient Transient Temperature-Aware Power Budgeting for 3D-Stacked Processor-Memory Systems

Sobhan Niknam*
University of Amsterdam
Netherlands
s.niknam@uva.nl

Yixian Shen*
University of Amsterdam
Netherlands
y.shen@uva.nl

Anuj Pathania
University of Amsterdam
Netherlands
a.pathania@uva.nl

Andy D. Pimentel
University of Amsterdam
Netherlands
a.d.pimentel@uva.nl

ABSTRACT

The heat produced during computation severely limits the performance of multi-/many-core processors. High-performance 3D-stacked processor-memory systems stack cores and main memory on a single die. However, 3D-stacked systems suffer more severe thermal issues than their non-stacked planar 2D counterparts. Consequently, the aggressive thermal throttling required for their thermally-safe operation limits the potential performance gains.

Power budgeting is an effective thermal management technique that prevents thermal throttling in multi-/many-core processors by assigning a thermally-safe power budget to cores within the processors. State-of-the-art power budgeting techniques for 2D processors do not account for the vertical thermal coupling between the layers of the 3D-stacked system and will fail to prevent thermal throttling in them. Furthermore, estimating thermals for a 3D-stacked processor with power budgeting requires a finer-grained RC thermal model than non-stacked processors. This requirement inhibits the porting of existing power budgeting solutions for 2D processors to 3D-stacked processor-memory systems.

This work is the first to present the linear algebra-based algorithmic time-invariant transformations required to enable power budgeting in 3D-stacked systems. Based on the transformations, we propose the first transient-temperature-aware power budgeting technique, *3D-TTP*, for 3D-stacked systems. Detailed interval thermal simulations with the advanced *CoMeT* simulator designed for 3D-stacked systems also confirm no thermal violations with our *3D-TTP* technique. *3D-TTP* exhibits an average 11.41% speedup over the state-of-the-art reactive-based thermal management technique.

1 INTRODUCTION

State-of-the-art multi-/many-core processors, based on the Von Neumann architecture, require their compute (processing cores) and storage (main memory) to work in tandem for maximum performance. Therefore, the quest for higher performance in multi-/many-core processors requires moving the processing cores and memory closer. 3D die integration technology allows vertical stacking of silicon layers interconnected using technology such as Through-Silicon Vias (TSVs) [1]. 3D-stacked processor-memory systems (also referred to as 3D-stacked systems) use this technology to stack several layers of main (DRAM) memory directly on top of the logic (core) layer(s), as shown in Fig. 1. Each memory layer consists of several memory banks. 3D-stacked systems offer lower memory latency and higher memory bandwidth than their non-stacked counterparts. Therefore, cores in a 3D-stacked system can operate much closer to their maximum compute potential as they spend significantly less time waiting for data from memory.

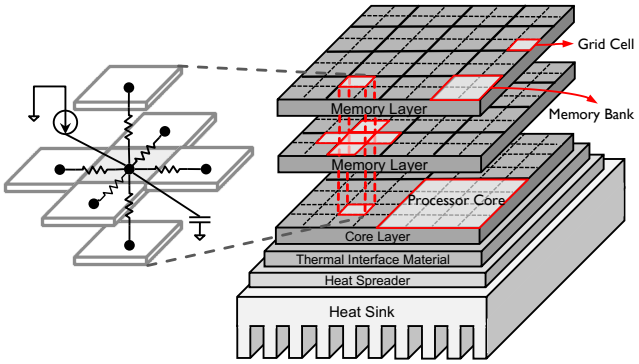
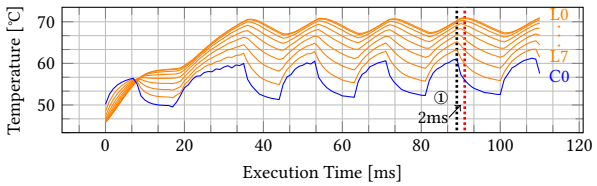


Figure 1: An abstract block diagram for a 3D-stacked processor-memory systems.

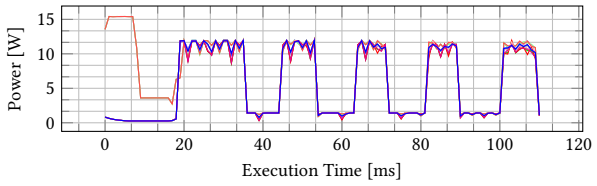
However, in practice, cores in 3D-stacked systems still struggle to achieve their compute potential because of accompanying thermal issues, which are much worse than for non-stacked processors [2]. 3D-stacked systems pump power for cores and memory within the same die. However, the surface area (heat sink) available to them to dissipate that power is much smaller due to stacking. Consequently, 3D-stacked systems require aggressive thermal throttling to keep them operating below the critical thermal threshold. Traditional thermal throttling [3] involves the reactive triggering of the Dynamic Thermal Management (DTM) unit by the on-chip power manager (Governor) as soon as the temperature of (part of) the processor goes beyond its safe limit. The DTM unit, when triggered, crashes the frequency of processing cores using Dynamic Voltage Frequency Scaling (DVFS) technology to bring down the processor's temperature back to safe operating limits. However, frequent triggering of DTM is detrimental to the processor's performance and degrades its reliability due to thermal cycling. DTM also introduces jitters in (streaming) applications.

Motivational Example: Fig. 2 shows the execution behavior of four four-threaded *lu.cont* benchmarks from the *Splash-2* benchmark suite running on a nine-layer 16-core 3D-stacked system with one core layer and eight memory layers. We set the thermal threshold for triggering DTM at 70 °C. Fig. 2(a) shows the peak temperature for each layer over the execution. Fig. 2(b) shows the power consumption of the cores in the core layer over the execution. Fig. 2(c) shows the operating frequency of the cores over the execution. Fig. 2 shows the frequent triggering of DTM wherein the processor frequency is lowered to the lowest frequency (1 GHz) from the highest frequency (4 GHz) as soon as the peak processor temperature

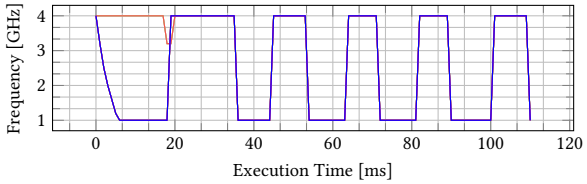
*Co-First Authors.



(a) Peak Temperature per Layer



(b) Power Consumption per Core



(c) Frequency per Core

Figure 2: Illustrative motivational example showing frequent triggering of DTM unit due to thermal emergency on 3D-stacked systems without proactive thermal management.

hits the thermal threshold of 70 ℃. Lowering the frequency lowers the cores' power consumption bringing down the processor's temperature. The temperature-aware Governor (from [4]) raises the processor frequency back to the highest frequency as soon as enough thermal headroom is available (i.e. when the peak processor temperature is less than 65 ℃). Raising the frequency raises the power consumption of cores. Consequently, the processor's temperature rises till it reaches the threshold wherein it retriggers the DTM. The pattern repeats till the benchmark finishes.

It is possible to manage the thermals of a 3D-stacked system using the DTM unit. However, designers of DTM designed it as a measure of last resort to resolve thermal emergencies. Therefore, a better thermal management strategy would be to prevent thermal emergencies on the processor and thereby avoid triggering the DTM unit. Power budgeting is a proactive thermal management technique that uses the RC-thermal model equivalent of a processor to calculate thermally-safe power budgets for individual cores in a processor. Power budgeting techniques guarantee the temperature within the processor will remain below the threshold as long as each core limits its power consumption to its assigned power budget. Therefore, instead of toggling between high and low frequencies, a Governor can choose intermediate stable frequencies for cores based on their power budgets.

Researchers have applied power budgeting extensively to manage the thermals of 2D processors with off-chip memories. Advanced power budgeting solutions for 2D processors account for several parameters, such as the number of active cores, spatiality of active cores on the processor's floorplan [5], and transient temperatures of the core themselves to calculate a core-level power budget [6]. State-of-the-art power budgeting solutions [5–7] can use the entire thermal headroom without violating the thermal threshold. However, we cannot apply these techniques directly to 3D-stacked systems for the following reasons.

Thermal hotspots form directly on the cores in 2D processors [8]. Therefore, they are straightforward to manage with power budgeting using core DVFS. On the contrary, in 3D-stacked systems, hotspots primarily form on the memory layers far from the core layer. These hotspots also take time to respond to power budgeting using core DVFS and are only weakly thermally correlated to the underlying cores directly below it. Furthermore, the memory banks within the memory layers also have a significant power consumption that we must also account for while calculating the budget for the cores in a 3D-stacked system.

Power budgeting also requires a detailed RC-thermal model (of the processor) to calculate the power budgets for the individual cores. One can use tools like *HotSpot* [8] to generate the RC-thermal model for a processor from its floorplan. A set of sub-component blocks constitute a floorplan. *HotSpot* accordingly can generate a block-level RC-thermal model for a 2D processor. However, several factors prevent a block-level RC-thermal model for a 3D-stacked system [1]. *HotSpot* resolves the problem by dividing the floorplan for each layer into equisized grids, as shown in Fig. 1. *HotSpot* uses these grid cells to generate a fine-grained grid-level RC-thermal model that works for the 3D-stacked system. However, existing power budgeting solutions for 2D processors can operate with only the coarse-grained block-level RC-thermal model. This limitation makes direct porting of existing solutions onto the next-generation interval thermal simulators for 3D-stacked systems like *CoMeT* [4] for empirical evaluation infeasible. Furthermore, since a core in grid mode consists of multiple grid cells, the power budget we calculate for individual cells must translate into a budget for the core. We can then use core DVFS for power budgeting in 3D-stacked systems.

Our Novel Contributions: This paper makes the following contributions in light of the discussion above.

- We present the linear algebra-based algorithmic time-invariant transformations required for power budgeting in 3D-stacked processor-memory systems.
- We propose *3D-TTP*, a transient temperature-aware power budgeting technique for 3D-stacked systems.
- We provide accurate power budgeting using a finer-grained grid-level RC-thermal model tailored to 3D stacked systems.
- We integrate *3D-TTP* with a state-of-the-art thermal simulator for the 3D-stacked system *CoMeT* [4] and evaluate it with *SPLASH* [9] and *PARSEC* [10] benchmark suites.

Open-Source Contributions: The code for the transient temperature-aware power budgeting scheduler, *3D-TTP*, as a *CoMeT* plugin is available for download at https://github.com/yixianUvA/3D-TTP.git.

2 RELATED WORK

Limited heat dissipation capability remains the primary performance bottleneck for processors. Therefore, thermal management

for processors remains an active subject of research. Power budgeting remains an important tool to drive proactive thermal management in processors. The authors of [11] were the first to introduce the concept of power budgeting in processors. They propose a power budgeting solution called *TSP* that computes power budgets for cores in a 2D processor. *TSP* uses the expected steady-state temperature of cores calculated based on the processor's floorplan (RC-thermal model) to calculate power budgets. *TSP* guarantees no thermal violations (DTM-free operations) in a processor if cores restrict themselves to the assigned power budgets.

The authors of [6] proposed a power budgeting solution called *T-TSP* that uses core(s) transient temperature for power budgeting. *T-TSP* provides better utilization of thermal headroom than *TSP* while still providing guarantees for a DTM-free operation. Authors of *T-TSP* were also the first to prove the efficacy of their power budgeting for 2D processors using detailed thermal interval simulations using *HotSniper* [12]. Similarly, the recently released *CoMeT* [4] simulator (evolved from *HotSniper*) for the first time provides the opportunity to empirically prove power budgeting for 3D-stacked systems using detailed interval thermal simulations.

Several works [13–15] combine power budgeting with knobs like task migration, DVFS, and power-gating for effective thermal management for 2D processors. Researchers have also proposed several reactive thermal management techniques for 3D-stacked systems using similar knobs [2, 3, 16]. However, to our knowledge, no work proposes proactive thermal management for 3D-stacked systems using power budgeting via knobs like DVFS.

3 SYSTEM MODELS

3.1 Architecture Model

In this work, we consider a 3D-stacked processor-memory system with homogeneous cores, as shown in Fig. 1. All cores have the same micro-architecture and share the memory address space. All cores are on the same layer stacked directly above the heat sink. Designers then stack multiple layers of memory (banks) above the core layer. The topmost memory layer goes into a secondary minor heat sink (PCB). Cores and memory banks communicate using TSVs. A core layer stacked directly above the heat sink contains all the cores. The processor operates in a thermally constrained environment, and its temperature should be below a given temperature threshold, i.e., T_{DTM}, to avoid triggering DTM. The processor executes multi-threaded benchmarks using a one-thread-per-core model as the workload.

3.2 Thermal Model

In this work, we use a well-established RC-thermal model [8] that has a basis in the well-known duality between thermal behavior and electrical circuits. In this model, we build an RC-thermal network with N thermal nodes for a 3D-stacked processor-memory system. The first n nodes in the network correspond to the processor's power-consuming components (e.g., cores, memory banks, etc.), while the remaining $N-n$ thermal nodes correspond to the cooling system. Each thermal conductance in the network interconnects two thermal nodes. Each thermal node is also associated with a thermal capacitance (except for the thermal node corresponding to the ambient temperature) to model transient temperature. Ambient temperature denoted as T_{amb} is constant. In this network, the power consumption of the active components acts as a heat source. We use *HotSpot* [8] to derive the RC-thermal network for the processor based on its floorplan, technology, material properties, cooling system parameters, etc.

With the above considerations, we can compute the temperature of each thermal node (a function of its power consumption, the temperature of its neighboring thermal nodes, and the ambient temperature) through a set of N first-order differential equations.

$$\mathbf{AT'} + \mathbf{BT} = \mathbf{P} + T_{\text{amb}}\mathbf{G}, \tag{1}$$

where $\mathbf{A} = [a_{i,j}]_{N\times N}$ contains the thermal capacitance, $\mathbf{B} = [b_{i,j}]_{N\times N}$ contains the thermal conductivity between neighboring nodes, $\mathbf{T} = [T_i(t)]_{N\times 1}$ contains the temperature on every node at a time t, $\mathbf{T'} = [T_i'(t)]_{N\times 1}$ contains the first-order derivative of the temperature on every node concerning time, $\mathbf{P} = [p_i]_{N\times 1}$ contains the power consumption of every node, and $\mathbf{G} = [g_i]_{N\times 1}$ contains the thermal conductivity between every node and the ambient temperature. By defining matrix $\mathbf{C} = -\mathbf{A}^{-1} \times \mathbf{B}$, we can rewrite Eq. (1) in the standard form given below.

$$\mathbf{T'} = \mathbf{CT} + \mathbf{A}^{-1}\mathbf{P} + T_{\text{amb}}\mathbf{A}^{-1}\mathbf{G}. \tag{2}$$

4 THERMAL SIMULATION FOR 3D-STACKED PROCESSOR-MEMORY SYSTEMS

Like 2D processors, thermal simulations remain the preferred means for studying the thermals of 3D-stacked systems. *HotSpot* [8] is a well-known widely-used thermal simulator that provides fast and accurate transient and steady-state thermal simulations for 2D and 3D-stacked systems, using the thermal model explained in Section 3.2. *HotSpot* provides two implementations of this model – *block* and *grid*. The *block model* allows for coarse-grain thermal simulation of 2D processors, with only a single thermal node for each component that results in a low-resolution heat map.

In contrast, the *grid model* enables more fine-grained but slower thermal simulation by supporting more thermal nodes per component, i.e., core or memory bank. In this model, the designer divides the floorplan of the chip into smaller portions called *grid cells*, each of which belongs to one or multiple components. An exemplary chip comprising three silicon layers - a single core layer containing four cores and two memory layers containing 32 memory banks - is illustrated in Fig. 1, in which each layer divides into an 8x8 grid. *HotSpot* computes the power consumption and temperature of grid cells using the following equations.

$$p_{n_g} = \sum_{n_b \in N_b} Area_{n_g} \times (P_{n_b}/Area_{n_b} \times Occupancy) \tag{3}$$

$$T_{n_g} = \sum_{n_b \in N_b} T_{n_b} \times Occupancy \tag{4}$$

where N_b and N_g are the set of all components/blocks and grid cells, T_{n_b} and P_{n_b} are temperature and power consumption of ${n_b}^{\text{th}}$ component, *Occupancy* is the ratio of occupying the grid cell by ${n_b}^{\text{th}}$ component, $Area_{n_b}$ and $Area_{n_g}$ are the area of ${n_b}^{\text{th}}$ component and ${n_g}^{\text{th}}$ grid cell. In the *grid model*, each grid cell is associated with a thermal node. *Hotspot* then uses Eq. (1) for thermal simulation.

In 3D-stacked systems, TSVs go through different silicon layers to provide connections between them. Therefore, silicon layers in

3D-stacked systems have non-uniform thermal properties. Consequently, the grid model is the only option for thermal simulations in 3D-stacked systems [1].

5 TRANSIENT TEMPERATURE-AWARE POWER BUDGETING FOR 3D-STACKED PROCESSOR-MEMORY SYSTEMS

Transient temperature-aware power budgeting represents the most advanced methodology in safely exploiting thermal headroom in 2D processors to improve their performance [6]. Therefore, in this paper, we propose a transient temperature-aware budgeting technique, *3D-TTP*, for power budgeting 3D-stacked systems.

The primary objective of *3D-TTP* is to derive the accurate power budget values of the active cores while accounting for their transient temperature. The main step toward this objective is to solve Eq. (1) to attain the relation between the transient temperature of the cores and their power consumption. We must have the initial conditions to solve the underlying differential equations. We define the column matrix $\mathbf{T}(t_s) = [T_i(t_s)]_{N_g \times 1}$ as the initial temperature of nodes at time $t = t_s$. The analytical solution of Eq. (1) - by using matrix exponential - is given by the following.

$$\mathbf{T}(t) = e^{\mathbf{C}(t-t_s)}\mathbf{T}(t_s) + \int_{t_s}^{t} e^{\mathbf{C}(t-\tau)}\mathbf{A}^{-1}\mathbf{P}(\tau)d\tau, \tag{5}$$

where we derive the temperature of grid cells of components, i.e., cores/memory banks, at the time t as a function of their initial temperature $\mathbf{T}(t_s)$ and instantaneous power consumption $\mathbf{P}(\tau)$ during time interval $\tau = [t_s, t_e)$. Now assuming the power consumption of components and thus their belonging grid cells to be *constant* during τ, i.e., $\forall t \in \tau:\ \mathbf{P}(t) = [p_i]_{N_g \times 1}$, we can rewrite Eq. (5) as

$$\mathbf{T}(t) = \mathbf{M} \times \mathbf{T}(t_s) + \mathbf{R} \times \mathbf{P}, \tag{6}$$

where

$$\mathbf{M} = e^{\mathbf{C}\tau},\ \mathbf{R} = -\mathbf{C}^{-1}(\mathbf{I} - e^{\mathbf{C}\tau})\mathbf{A}^{-1}. \tag{7}$$

Matrix $\mathbf{M} = [m_{i,j}]_{Ng \times Ng}$ and $\mathbf{R} = [r_{i,j}]_{Ng \times Ng}$ are static, and therefore we compute them only once at the design time. In the above equations, the term $e^{\mathbf{C}t} = [e^{\mathbf{C}t}_{i,j}]_{N_g \times N_g}$ is the *matrix exponential* of matrix $\mathbf{C}$ and can be analytically computed as $e^{\mathbf{C}t} = \mathbf{V}e^{\mathbf{D}t}\mathbf{V}^{-1}$, where $\mathbf{V}_{N_g \times N_g}$ represents a matrix containing the eigenvectors of matrix $\mathbf{C}$, $\mathbf{V}^{-1}_{N_g \times N_g}$ is the inverse of matrix $\mathbf{V}$, and

$$\mathbf{D} = \begin{pmatrix} \lambda_1 & & & & & & & \\ & \ddots & & & & & & \\ & & \lambda_\ell & & & & & \\ & & & \begin{matrix}\alpha_1 & \omega_1 \\ -\omega_1 & \alpha_1\end{matrix} & & & & \\ & & & & \ddots & & & \\ & & & & & \begin{matrix}\alpha_k & \omega_k \\ -\omega_k & \alpha_k\end{matrix} & \end{pmatrix}_{N_g \times N_g} \tag{8}$$

where $\lambda_1, \cdots, \lambda_\ell$ and $\alpha_1 \pm i\omega_1, \cdots, \alpha_k \pm i\omega_k$ are *real* and *complex* eigenvalues of matrix $\mathbf{C}$, respectively.

Now, having matrix $\mathbf{M}$ and $\mathbf{R}$, one can compute the temperature of all thermal nodes at the end of the interval τ, at time point t_e, using Eq. (6) concerning their initial temperature $\mathbf{T}(t_s)$, at time point t_s, and their constant power consumption $\mathbf{P}$ during the time interval τ. In addition, we can also adopt Eq. (6) to compute the power budget of cores – the maximum power that cores can consume safely without causing any thermal violations, i.e., $\forall n_g \in N_g:\ T_{n_g}(t) \leq T_{DTM}$

Algorithm 1: *3D-TTP* for 3D-stacked systems

Input: Floorplan, T_{DTM}, $\mathbf{T}_b(t_s)$, $\mathbf{P}_b(t_s)$, AC_b, τ
Output: Matrix $\mathbf{P}_{budget}$
/* Design-time phase */
Compute matrices $\mathbf{M}$ and $\mathbf{R}$ using Eq. (7)
/* Run-time phase */
Compute $\mathbf{P}_g(t_s) = [p_{g_i}]_{Ng \times 1}$ and $\mathbf{T}_g(t_s) = [T_{g_i}]_{Ng \times 1}$ of grid cells wrt. $\mathbf{T}_b(t_s)$ and $\mathbf{P}_b(t_s)$ using Eq. (3) and Eq. (4)
$AC_g \leftarrow$ grid cells belonging to active cores in AC_b
$AM_g \leftarrow$ grid cells in L_0 which are vertically located underneath of those in AC_g
$\mathbf{T}_{headroom} = [T_{headroom_i} = 0]_{|AC_g| \times 1}$
forall $n_{g_i} \in AM_g$ **do**
  $T_{headroom_i} = T_{\mathrm{DTM}} - \sum m_{i,j} \times T_{g_j} - \sum_{j \notin AC_g} r_{i,j} \times p_{g_j}$
$\mathbf{Q} = [q_{i,j} = 0]_{|AC_g| \times |AC_b|}$
forall $n_{g_i} \in AM_g$ **do**
  forall $n_{b_j} \in AC_b$ **do**
    forall $n_{g_k} \in AC_g$ *belonged to core* n_{b_j} **do**
      $q_{i,j} += r_{i,k}$
$\mathbf{T}_{headroom} = \mathbf{Q} \times \mathbf{P}_{budget}$
Solve $\mathbf{Q}^\top \times \mathbf{T}_{headroom} = \mathbf{Q}^\top \times \mathbf{Q} \times \mathbf{P}_{budget}$
forall $core_i \in AC_b$ **do**
  $p_{budget_i} = p_{budget_i} \times |AC_g|/|AC_b|$
return Matrix $\mathbf{P}_{budget}$

where T_{DTM} is the thermal threshold. These calculations occur at the run-time for every algorithm invocation.

Power budgeting in 3D-stacked systems has two new major challenges not present in 2D processors. (1) The temperature of thermal nodes on all layers cannot reach the thermal threshold simultaneously. This observation implies that all silicon layers cannot exploit their entire thermal headroom. Among all layers, peak temperature (thermal hotspot) is most likely to occur on the layer farthest from the primary heatsink, the closest to the PCB. The thermal conductivity among the silicon layers decreases with an increase in their distance from the heatsink, and dissipating the generated heat becomes more difficult. As a result, designers place the core layers that are the main source of power consumption (and heat generation) in the processor closer to the heatsink than the memory layers. Therefore, we assume in this work that the temperature of grid cells on the lowest memory layer (referred to as L_0) is critical and should never overshoot the thermal threshold, i.e.,

$$\forall n_{g_i} \in L_0 : \sum m_{i,j} \times T_j(t_s) + \sum r_{i,j} \times p_j \leq T_{DTM}. \tag{9}$$

(2) The heat transfer among silicon layers does not occur instantly. Fig. 2(a) shows the temperature on the core layer immediately reduces almost exponentially upon triggering DTM at 89 ms (point ①). However, it takes several milliseconds until the peak temperature on other layers reduces, during which a hotspot emerges on the lowest memory layer at 91 ms. So, we use this delay in the power budget calculation to avoid future overshooting of the thermal threshold on the lowest memory layer.

Algorithm 1 presents the pseudo-code for computing the power budget values with *3D-TTP*. This algorithm takes as inputs the floorplan including hardware-dependent matrices (computed once for every chip at design-time), a thermal threshold T_{DTM}, an initial block-level temperature matrix $\mathbf{T}_b(t_s)$, and power consumption matrix $\mathbf{P}_b(t_s)$, a set AC_b of active cores, and an epoch length of

Table 1: Parameters for the simulated 3D-stacked system.

Core Parameters	
Number of Cores	16, 1 layer
Core Model	x86, 4.0 GHz, 22 nm, out-of-order
Core Area	2.89 mm^2
L1 I/D cache	32/32 KB, 4/4-way,64 B block
L2 cache	private,512 KB, 8-way, 64 B block
L3 cache	512 KB, 16-way, 64B-block
Memory Parameters	
3D-stacked Memory	8 G, 8 layers, 16 channels, 128 banks
Memory Bank Area	2.89 mm^2
Memory Bandwidth	25.6 GB/s

τ. The algorithm returns as output a matrix $\mathbf{P}_{budget}$ containing the power budget values of the active cores on the chip. At design time, the algorithm computes the auxiliary matrices **M** and **R** using Eq. (7), in Line 1. Then, at run time, the algorithm first computes the grid-level initial power consumption and temperature using Eq. (3) and Eq. (4), respectively. In Line 3, the grid cells of all active cores in AC_b are stored in AC_g. Similarly, in Line 4, the grid cells located vertically underneath the grid cells in AC_g but on the lowest memory layer L_0 are stored in AM_g. These are the grid cells where hotspots are most likely to emerge. Thus, their temperature should be kept at a maximum at T_{DTM}. In Line 5, the matrix $\mathbf{T}_{headroom}$ is defined for storing the available thermal headroom on grid cells in AM_g. The thermal headrooms are then computed in Lines 6-7 using Eq. (9) when excluding the thermal contribution of grid cells in AC_g. Please note that we use the initial power consumption of memory banks in this calculation and assume they will remain constant during τ. Alternatively, we can empirically derive the maximum power consumption of memory banks and use it in the calculation, but it results in processor under-utilization due to conservative budgeting. We assume idle cores are also consuming constant power.

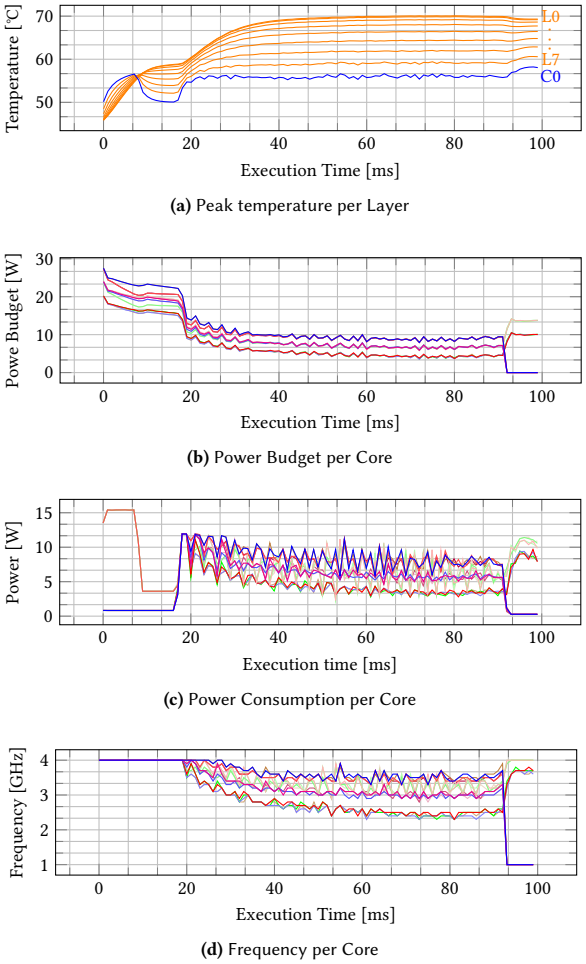
The grid-level power budgets can now be computed using

$$\forall i \in AC_g : T_{headroom_i} = \sum_{j \in AC_g} r_{i,j} \times p_{g_j}. \tag{10}$$

However, the Governor needs a core-level (or block-level) power budgeting for power budgeting with core DVFS. Therefore, we assume that grid cells belonging to each block/component must have the same power budget. To this end, in Lines 8-12 of Algorithm 1, the homogeneous system of linear equations in Eq. (10) is transformed into an overdetermined system, i.e., having more equations than unknowns (power budgets). The transformation results in the equation in Line 13. Such an overdetermined system may not have any exact solutions, but we can solve it approximately. We can transform the equation into a standard square system of linear equations in Line 14 using a technique like the least squares method in linear algebra. Then, we use the Gaussian elimination algorithm to solve the resulting equation. The computed **P** now contains the per-grid power budget value for the next epoch. Therefore, in Lines 15-16, each entry of **P** is multiplied by the number of grid cells per core block, i.e., $|AC_g|/|AC_b|$ (e.g., 16 in Fig. 1), for computing the per-core power budget values.

6 EVALUATION

We evaluate our work using interval thermal simulations via the state-of-the-art *CoMeT* simulator [4]. *CoMeT* integrates *Sniper* [17], *McPat* [18], *CACTI* [19], and *HotSpot* [8] into one toolchain for simulating thermals of 3D-stacked systems. Table 1 lists the parameters for the simulated 3D-stacked system. We use multi-threaded workloads from *PARSEC 2.1* [10] (with *sim-small* inputs) and *SPLASH-2* [9] (with *small* inputs) as the system workload. We used all the benchmarks that execute to completion using *CoMeT*. T_{amb} and $T_{critical}$ are set to 45 ℃ and 70 ℃ respectively.



(a) Peak temperature per Layer

(b) Power Budget per Core

(c) Power Consumption per Core

(d) Frequency per Core

Figure 3: Case-study showing thermally-safe DTM-free execution on a 3D-stacked system with transient temperature-aware power budgeting using *3D-TTP*.

Case Study: Fig. 3 shows the execution of four four-threaded *lu.cont* benchmarks on our simulated 3D-stacked system with transient temperature-aware power budgeting using *3D-TTP*. Fig. 3(a) shows the peak temperature for each layer over the execution. Fig. 3(b) shows the power budgets over the execution. Fig. 3(c) shows the power consumption of the cores in the core layer over the execution. Fig. 3(d) shows the operating frequency of the cores over the execution. In contrast to similar execution without power budgeting shown in Fig. 2, execution with power budgeting never hits the thermal threshold (70 ℃). Furthermore, *3D-TTP* keeps the peak temperature of the processor close to the thermal threshold

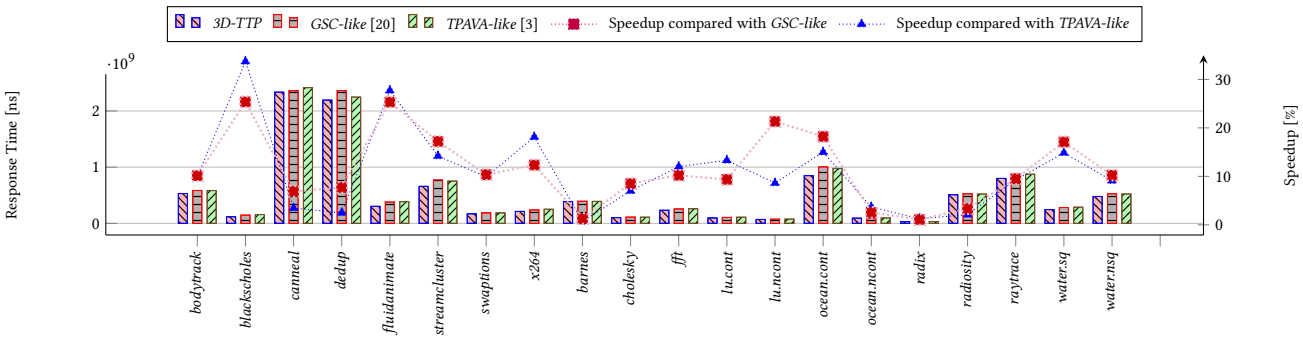


Figure 4: Response time for different benchmarks with *3D-TTP*, *GSC-like*, and *TPAVA-like* on a 3D-stacked system.

throughout the execution to use the entire thermal headroom. The execution of *lu.cont* under *3D-TTP* is also 10% faster due to no performance loss originating from DTM.

Comparative Baselines: Since we are unaware of any proactive power budgeting technique for 3D-stacked systems, we compare our proposed *3D-TTP* with reactive thermal management techniques designed for 3D-stacked systems. Authors of [3] propose a temperature-aware thermal management technique, called *TPAVA*, that uses DVFS to reduce the occurrence of hotspots in 3D-stacked systems. Authors of [20] propose a thermal manager, called *GSC*, that employs a Proportional-Integral-Derivative (PID) like controller to perform temperature-aware DVFS on 3D-stacked systems. Since both *TPAVA* and *GSC* predate the existence of the *CoMeT* thermal interval simulator used in this work, we have reimplemented both on *CoMeT*. We call the *CoMeT* implementations of *TPAVA* and *GSC* using the names *TPAVA-like* and *GSC-like*, respectively.

Comparative Evaluation: Figure 4 shows the response of different benchmarks with *3D-TTP*, *GSC-like*, and *TPAVA-like*. We load the system fully with multiple instances of each benchmark and then report the average response time in Figure 4. The figure also shows the speed up of *3D-TTP* over the baselines. The experimental results show that *3D-TTP*, on average, results in 10.82% and 11.41% better response time than *GSC-like*, and *TPAVA-like*, respectively.

The benchmark *blackscholes* is a compute-bound benchmark that generates lots of heat and requires precise thermal management to avoid triggering DTM. Consequently, *blackscholes* speed up by 33.69% and 25.37% with *3D-TTP* compared to *TPAVA-like* and *GSC-like*, respectively. On the other hand, memory-bound benchmarks like *canneal* hardly produce any heat. Consequently, the speedups observed with them using *3D-TTP* over the baselines is minimal.

Run-time Overhead: During 2000 runs under full load, *3D-TTP* requires 112.2 μs to calculate the power budget for the processor specified in Table 1. As a result, the projected overhead of *3D-TTP* for the power budget epoch of 2 ms is 5.61%.

7 CONCLUSION

In this work, we study the application of power budgeting in 3D-stacked processor-memory systems. We describe the challenges in applying the existing power budget solutions for 2D processors directly on 3D-stacked processor-memory systems. Consequently, we present a proactive transient temperature-aware power budgeting technique via linear algebra-driven time-invariant modeling, *3D-TTP*, for 3D-stacked processor-memory systems. We implement *3D-TTP* on a state-of-the-art interval thermal simulator, *CoMeT*. Detailed interval thermal simulations establish the efficacy of *3D-TTP* in using the thermal headroom and preventing thermal violations. We demonstrate that *3D-TTP* achieves an average 11.41% performance gain over current reactive thermal management methods in 3D-stacked processor-memory systems.

REFERENCES

[1] J. Meng and et al. Optimizing energy efficiency of 3-d multicore systems with stacked dram under power and thermal constraints. In *DAC*, 2012.
[2] D. Lee and et al. Performance and thermal tradeoffs for energy-efficient monolithic 3d network-on-chip. *TODAES*, 2018.
[3] C.H. Liao, C.H-P. Wen, and K. Chakrabarty. An online thermal-constrained task scheduler for 3d multi-core processors. In *DATE*, 2015.
[4] L. Siddhu and et al. Comet: An integrated interval thermal simulation toolchain for 2d, 2.5d, and 3d processor-memory systems. *TACO*, 2022.
[5] S. Pagani, H. Khdr, W. Munawar, J.J. Chen, M. Shafique, M. Li, and J. Henkel. Tsp: Thermal safe power: Efficient power budgeting for many-core systems in dark silicon. In *CODES+ISSS*, 2014.
[6] S. Niknam, A. Pathania, and A.D. Pimentel. T-tsp: Transient-temperature based safe power budgeting in multi-/many-core processors. In *ICCD*, 2021.
[7] H. Wang, D. Tang, M. Zhang, S.X.D Tan, C. Zhang, H. Tang, and Y. Yuan. Gdp: A greedy based dynamic power budgeting method for multi/many-core systems in dark silicon. *TC*, 2018.
[8] W. Huang, S. Ghosh, S. Velusamy, and et.al. Hotspot: A compact thermal modeling methodology for early-stage vlsi design. *VLSI*, 2006.
[9] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, and A. Gupta. The splash-2 programs: Characterization and methodological considerations. *SIGARCH*, 1995.
[10] C. Bienia, S. Kumar, J.P. Singh, and K. Li. The parsec benchmark suite: Characterization and architectural implications. In *PACT*, 2008.
[11] S. Pagani, H. Khdr, J.J. Chen, M. Shafique, M. Li, and J. Henkel. Thermal safe power (tsp): Efficient power budgeting for heterogeneous manycore systems in dark silicon. *TC*, 2016.
[12] A. Pathania and J. Henkel. Hotsniper: Sniper-based toolchain for many-core thermal simulations in open systems. *ESL*, 2018.
[13] H. Khdr, S. Pagani, E. Sousa, and et. al. Power density-aware resource management for heterogeneous tiled multicores. *TC*, 2016.
[14] M. Rapp, A. Pathania, T. Mitra, and J. Henkel. Neural network-based performance prediction for task migration on s-nuca many-cores. *TC*, 2020.
[15] H. Najibi, A. Levisse, G. Ansaloni, M. Zapater, and D. Atienza. Thermal and power-aware run-time performance management of 3d mpsocs with integrated flow cell arrays. In *GLSVLSI*, 2022.
[16] D.C. Juan, S. Garg, and D. Marculescu. Statistical peak temperature prediction and thermal yield improvement for 3d chip multiprocessors. *TODAES*, 2014.
[17] T.E. Carlson and et al. Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation. In *HPCA*, 2011.
[18] S. Li and et al. Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, 2009.
[19] N. Muralimanohar, R. Balasubramonian, and N.P. Jouppi. Cacti 6.0: A tool to model large caches. *HP laboratories*, 2009.
[20] B. Donyanavard, M. Rahmani, T. Muck, K. Moazemmi, and N. Dutt. Gain scheduled control for nonlinear power management in cmps. In *DATE*, 2018.