Systematic Temperature Sensor Allocation and Placement for Microprocessors

Rajarshi Mukherjee and Seda Ogrenci Memik
Department of Electrical Engineering and Computer Science
Northwestern University, Evanston, IL 60208
{rajarshi, seda}@ece.northwestern.edu

ABSTRACT

Modern high performance processors employ advanced techniques for thermal management, which rely on accurate readings of on-die thermal sensors. As the importance of thermal effects on reliability and performance of integrated circuits increases careful planning and embedding of thermal monitoring mechanisms into these systems will be crucial. Systematic tools for analysis of thermal behavior and determination of best allocation and placement of thermal sensing elements is therefore a highly relevant problem. In this paper, we propose novel optimization techniques for determining the optimal locations and allocations for thermal sensors to provide a high fidelity thermal profile of a complex microprocessor system. Our algorithm identifies an optimal physical location for each sensor such that the sensor's the attraction towards steep thermal gradient is maximized. We also present a hybrid allocation and placement strategy showing the trade-offs associated with number of sensors used and expected accuracy. Our results show that our tool is able to create a sensor distribution for a given microprocessor architecture providing thermal measurements with maximum error of 3.18°C and average maximum error of 1.63°C across a wide set of applications.

Categories and Subject Descriptors: J.6 [**Computer Applications**]: Computer-aided Engineering (CAD): B.8.2 [**Hardware**]: Performance and Reliability – Performance Analysis and Design Aids.

General Terms: Algorithms, Measurement, Experimentation.

Keywords: Temperature, Sensor, Allocation, Placement.

1. INTRODUCTION

Steady miniaturization and large-scale integration are vital to meet aggressive performance targets in high performance circuits. They have lead to rapidly increasing power densities on microprocessors [1]. Power dissipated on a chip is converted to heat, which creates reliability threats, adversely impacts leakage power and increases cost of cooling. The challenge is to provide high performance and reliability at lowest cost possible. Effective assessment and analysis of the thermal behavior of microprocessors is crucial to overcome this challenge. Modeling and simulation tools and thermal-aware design methodologies are one avenue of efforts towards this goal [2-5]. A major hurdle in this direction is the fact that thermal behavior is input dependent and also sensitive to environmental conditions. Thus, a highly accurate thermal profile of a complex system can only be established once it is deployed.

Thermal monitoring using on-die thermal sensors provides the means to assess the run-time thermal profile of a system. Thermal monitoring is already in use in modern processor architectures to assist Dynamic Thermal Management (DTM) mechanisms. For instance, Intel Pentium 4, Pentium M and IBM PowerPC processors are equipped with thermal sensors that trigger alerts if the junction temperature exceeds a specified limit. Based on these alerts the processor power consumption is regulated via clock throttling [6]. POWER5, IBM's next generation POWER microprocessor employs 24 digital temperature sensors [7]. In this paper, we propose a tool that helps the designer to determine the best allocation and placement of thermal sensors in a complex microprocessor system. In the following, we will first elaborate on the need for optimization in this problem.

The thermal behavior of complex systems is affected by various factors. For example localized heating on a processor is application dependent. Moreover, different architectural choices may result in diverse thermal profiles. Also, the specific purpose of monitoring will determine what specific form of sensing setup should be used. For instance, if we are primarily interested in worst case operating temperatures and design of emergency intervention mechanisms, then the spots that reach the highest temperatures on the processor are most relevant. On the other hand, if we are interested in controlling temperature dependent leakage power and developing dynamic mechanisms to manipulate the configuration of certain components, then we might need to capture the thermal profile of an individual component even if its absolute temperatures are not threatening reliability. Similarly, if temperature dependent delay violation is the focus of attention, the thermal condition of the critical path of the processor must be monitored closely.

It is intriguing to observe that several recent studies aiming to identify hottest regions of microprocessors seemed to have reached different conclusions. It is reported that the single thermal sensor on the Intel Pentium 4 processor is placed near the rapid integer ALU, which was identified as the likeliest candidate to cause a hotspot [8]. Skadron et al. [5] report that in the Alpha 21364 architecture the register file appears to be the hottest component consistently across a large set of SPEC CPU2000 [9] benchmarks. We performed yet another set of experiments with the same architecture (with a slightly different configuration), benchmark suite, and thermal simulator. Our detailed results from this analysis will be presented in Section 4. Our thermal analysis revealed that the Instruction Queue generated the hottest points consistently.

Figure 1 depicts the map of hotspots at the level of individual processor blocks. This map was obtained from our thermal simulations across the SPEC2000 benchmarks in the same Alpha 21364 architecture. Across 25 benchmarks and 18 different components of the processor, the number of unique block-level hotspots is 184 (out of possible distinct 450 points, some hotspots re-occur due to correlation of activity and power density, which leaves us with 184 distinct points). This means that the location of the hottest point within each block shifts across applications. Thus, the number of distinct hotspots and their relative spatial distribution within each individual block can present unique characteristics. For example, in the floating-point multiplier hotspots appear in widely disperse locations. In contrast, in the instruction queue (IntQ) the local hotspots are concentrated near the center. If we aim to setup a network of thermal sensors to capture the thermal behavior at individual block-level, there is a need to determine the optimal locations for best accuracy by considering these characteristics in a systematic framework.

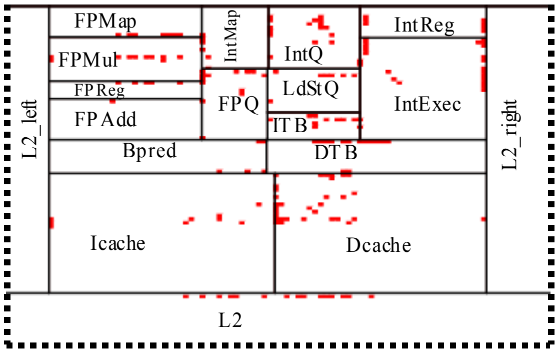

Figure 1. Distribution of the hotspots (marked in red) for each processor block for SPEC2000 benchmarks.

With the increasing impact of process technologies, process variation specifically; it will become increasingly harder to depend on the measurements of sensors placed "near" a component. In addition, there is an increasing interest in design of dynamic thermal management techniques that are applied at the granularity of individual blocks. Several local optimization techniques applied to various processor components such as "Heat and Run" thread assignment for chip multiprocessors [10], activity migration to reduce hotspots [11], temperature-aware steering, clustering and thermal-aware renaming, and committing mechanisms have been proposed [12]. All such efforts will require accurate local thermal monitoring for individual processor components to support fine grain dynamic optimizations. Therefore, techniques for strategic placement of thermal sensors will become indispensable aids to processor component designers and architects.

In conclusion, the thermal behavior of a microprocessor can be affected by a variety of factors and the presentation of the thermal behavior itself is highly purpose-specific. Therefore, there is a clear need to establish a thermal monitoring mechanism that can capture different aspects of the thermal behavior with high fidelity. In this paper, we address this challenge and propose a method to determine the best allocation and placement of thermal sensors to maximize the accuracy of measurements. We formulate the problem of thermal sensor instantiation as a clustering problem. We propose a k-means-based algorithm to cluster hotspots into groups and assign a sensor to monitor each group. Our algorithm identifies an optimal physical location (we refer to as cluster centers) for each sensor such that the sensor's the attraction towards steep thermal gradient is maximized. This essentially translates into placing a thermal sensor near a location of potentially high thermal stress. We also present a hybrid allocation and placement strategy showing the trade-offs associated with number of sensors used and expected accuracy. Our specific contributions in this paper are as follows. We,

- Formulate of the sensor instantiation problem for thermal monitoring of microprocessors,
- Propose efficient clustering-based algorithms for allocation and placement of sensors,
- Present extensive simulation results to study the impact of different placement strategies on measurement accuracy.

To the best of our knowledge this is the first attempt to develop an automated and optimized tool to determine the distribution and locations of thermal sensors for microprocessors and other similar complex integrated circuits.

In the following, we give an overview of related work. In Section 3 we discuss the thermal sensor insertion paradigm. We provide an overview of k-means clustering in Section 3.1 and introduce our sensor placement algorithm in Section 3.2. Section 4 presents our experimental results. We conclude with a summary in Section 5.

2. RELATED WORK

While design of highly accurate efficient sensors has been studied extensively, the problem of automated sensor placement has not been investigated in detail. Lee et al. [13] presented an analytical model that describes the maximum temperature difference between a hotspot and a region of interest. Gunther et al. [14], present observations on thermal maps and point to opportunities for optimized decisions on sensor placement. In this regard, our work is the first attempt to address this opportunity.

On a different track, Bratek et al. [15] present sensor placement for fault diagnosis of integrated circuits, by linking temperature sensors and power modules in pairs. In various other fields the problem of "sensor placement" has been studied. In the context of wireless sensor networks, coverage refers to the quality of surveillance. This is generally formulated as a decision problem verifying that a given sensor deployment successfully covers a certain area [16]. In contrast our problem is to place sensors to minimize the error margin of the resulting "coverage", hence, it is an optimization problem that tries to create the best coverage. Sensor networks also encounter problems similar to the Art Gallery Problem [17]. Art Gallery Problem is defined as setting minimal number of observers required for complete visual coverage of a polygonal gallery. Placing sensors for thermal monitoring is constrained by the range of the sensor or the tolerable error margin in contrast to the "line of sight" of the observer. Hence, the meaning of establishing sight (i.e. coverage) in our problem also requires satisfaction of a distance constraint between the sensor and the point of interest.

Partitioning and clustering techniques are widely used in the placement of VLSI circuits [18]. The objective of clustering in such problems is to minimize the net cut cost and balance the cluster sizes. The clusters are not evaluated considering any spatial relationship between the elements contained in them. In our problem, the goodness of a clustering solution must consider the spatial and thermal properties of the hotspots.

3. THERMAL SENSOR INSERTION

Our goal is to systematically analyze thermal maps to identify locations, which lie close to a maximal number of eventful thermal spots across a range of applications. We formulate this problem as a clustering of the points of interest in the spatial domain. The center of each cluster will indicate the physical location of a sensor. This sensor will monitor the points associated with that cluster. Hence, the temperature reading from that sensor is representative of the entire coverage area. We developed algorithms based on k-means clustering to carry out this task. In the next section we will first introduce the k-means clustering concept.

For the remainder of the discussion we will focus on accurate monitoring of hotspots on a microprocessor. We define a hotspot as a point, which reaches the highest temperature during the execution of an application. This point can be defined *globally* over the entire processor (to monitor reliability threats) or *locally* for each individual component (to assist in dynamic thermal management of local components). If we consider a collection of applications, each application can create a distinct hotspot with a distinct temperature value. This yields a set of hotspots observed across a range of applications. Such a set of hotspots can be obtained via thermal simulation of a given architecture configuration. We will describe our methodology to perform such a simulation in Section 4.

3.1 K-means Clustering-Based Sensor Placement

K-means clustering technique can be defined as: given a set of *n* data points in d-dimensional space and an integer *k*, determining *k* centers, such that the mean squared distance from each data point to its nearest center is minimized [19]. The k-means clustering works by iteratively refining the position of the *k* cluster centers. Initial cluster centers are assigned to any *k* points from *n* elements and the membership set (based on the Euclidean distance) of each cluster is calculated. Next, each element h_i is assigned to a cluster C_j such that the Euclidean $E(O_j, h_i)$ distance between h_i and j^{th} cluster center O_j is minimum. The Euclidean distance calculation is shown in Equation 1, where $\{h_{ix}, h_{iy}\}$ and $\{O_{jx}, O_{jy}\}$ represent the locations of h_i and O_j in the (x,y) plane, respectively.

$$E(O_j, h_i) = (O_{ix} - h_{ix})^2 + (O_{iy} - h_{iy})^2 \qquad (1)$$

The elements assigned to a cluster C_j comprise the membership set M_{sj} of the cluster *j*. The coordinates of the cluster center O_j are then updated as shown in Equation 2.

$$O_{jx,y} = \sum_{i \in M_{sj}} h_{ix,y} / size(M_{sj}) \qquad (2)$$

In the next iteration, Euclidean distance $E(O_j, h_i)$ for $i \in 1\ldots n$ and $j \in 1\ldots k$ changes due to new positions of cluster centers leading to new membership sets of clusters. Based on the new membership sets the cluster centers are recomputed (Equation 2). Adding elements at minimum Euclidean distance from cluster center O_j to the membership set M_{sj} for cluster C_j ($\forall j \in 1\ldots k$), and refining the position of center O_j to the centroid of those elements belonging to M_{sj}, are iterated until there is no reassignment of an element from its present cluster to another or the sum of Euclidean distances does not decrease significantly. We refer to this as *basic k-means*.

For our purpose the elements are hotspots on the die. For number of sensors equal to *k*, *k* clusters need to be created to monitor *n* hotspots. The *k* cluster centers corresponding to each of the *k* clusters specify the location of a thermal sensor. We refer to the location of the sensor as *cluster center* in the remainder of this discussion. The final cluster centers O_j would indicate the positions of the sensors S_j, for all $j \in 1\ldots k$. The membership set M_{sj} of cluster C_j corresponds to the set hotspots for which the sensor S_j will be the one expected to provide the most accurate measurement.

Localized heating happens much faster than the chip-wide heating due to slow rate of lateral heat propagation. Considering two distinct hotspots, the one at a relatively higher temperature will exhibit a steeper thermal gradient with respect to its neighbors as compared to the other hotspot. Thus, the sensor needs to be placed closer to the former hotspot than to the later to maximize overall accuracy. If we were to apply the basic k-means directly to sensor placement problem (we will refer to this as the *2-Dimensional placement*), such clustering in the (*x*, *y*) plane would minimize average distances of the sensors from all the hotspots. However, it would not differentiate between hotspots in terms of their thermal gradients. An optimal sensor placement should place an emphasis on minimizing the distance of the sensors from steep temperature gradients on the die and prioritize among different hotspots accordingly. We propose a *thermal gradient-aware* clustering and sensor placement algorithm to overcome this challenge.

3.1.1 Thermal Gradient-Aware Sensor Placement

Considering the thermal characteristic *t* of the hotspots, each such element can be considered distributed in the 3-D space and defined by the tuple (*x*, *y*, *t*). Using this representation, our sensor placement algorithm operates in two stages. In the first stage, we group hotspots into clusters where elements in the same cluster exhibit both spatial and thermal correlation. In the second stage, we identify the physical location within each such cluster where a thermal sensor should be placed. The sensor placed at this location would provide the most reliable information regarding the thermal condition of any hotspot within a certain cluster.

Hotspot Cluster Formation: We apply clustering to a set of hotspots in 3-D space. The *k* cluster centers are initially set to *k* hotspots. Euclidean distance $E(O_j, h_i)$ between h_i and O_j is computed according to Equation 3 shown below. The difference between temperature (*t*) coordinates of the jth cluster center O_j and hotspot h_i are considered in calculating this distance.

$$E(O_j, h_i) = (O_{ix} - h_{ix})^2 + (O_{iy} - h_{iy})^2 + (O_{it} - h_{it})^2 \qquad (3)$$

The hotspots h_i ($\forall$i, $i \in 1\ldots n$) are then assigned to cluster C_j ($\forall$j, $j \in 1\ldots k$) such that $E(O_j, h_i)$ ($\forall$i, j) is minimum. Using 3-dimensional distance evaluation is more likely to create clusters containing hotspots having similar thermal characteristics.

Determination of Physical Sensor Location within Clusters: The key idea behind this approach is to move cluster centers or sensors closer to the relatively higher temperature hotspots. This is equivalent to the sensor being *attracted* to hotspots with higher temperature values with a larger force. For each addition of hotspot h_i to M_{sj} of cluster C_j, the cluster center coordinates are the cumulative sum of the corresponding member coordinates. The cumulative sum computation is shown in Equation 4.

$$O_{jx,y} = O_{jx,y} + h_{ix,y} + \alpha(O_{jx,y} - h_{ix,y}) \times (h_{it} - O_{jt} / size(M_{sj}))$$
$$O_{jt} = O_{jt} + h_{it} \qquad (4)$$

The (*x*, *y*) coordinates of the cluster center $O_{jx,y}$ are closer to the hotspot h_i if the *t* dimension of the center O_j is less than that of h_i. If $h_{it} < O_{jt}$, the cluster center moves further from the position of h_i. We represent this phenomena using an attraction coefficient α in Equation 4. We have determined experimentally that an attraction coefficient value α = 0.1 performs best. After iterating over all the hotspots the final position of O_j is updated as shown in Equation 5.

$$O_{jx,y,t} = O_{jx,y,t} / size(M_{sj}) \qquad (5)$$

The cluster centers determined using Equation 4 and Equation 5 are then iteratively refined such that the mean squared distances of the hotspots from their respective cluster centers is minimized. Note that computing the cluster center using this method moves the sensor location physically closer to steeper thermal gradients. We would also point to the fact that although we use the temperature dimension of the cluster centers (as the average of the cluster members), this dimension (*t*) is used for modeling attraction towards points exhibiting high thermal stress. The temperature at the sensor's physical location determines its thermal reading, which has no physical relation with the temperature coordinate of the cluster center. Next, we will show how our proposed algorithm can be applied for different sensor placement scenarios.

3.2 Sensor Placement Strategies

In this section we discuss two different strategies for sensor placement. Thermal monitoring can be performed in the global scope for the entire processor or locally per processor block.

3.2.1 Global Sensor Placement

In this strategy, the global hotspots are considered. The global hotspots generally emerge in the same block (although shifted inside the block across applications). However, there can be reasons for hotspots to move into different components of the processor. For instance, in a superscalar processor multiple copies of the floating point unit can be selectively activated. During the intervals, where a single floating point unit is active it can contain the hotspot. In another interval, where multiple copies of this unit are active, the load would be distributed evenly and the power density would be low in this location. At that point a different unit, such as the instruction queue can be the origin of the hotspot. Our strategy to place sensors to capture such events works as follows. First an initial number of sensors are estimated and thermal gradient-aware sensor placement is performed with that number. Then, the sensor allocation is changed iteratively until the results are within a given accuracy. A good starting point is to select the number of sensors to be equal to the total number of blocks and then increase or decrease that number of sensors.

3.2.2 Local Sensor Placement

In this case our goal is to determine the placement of the sensors for each individual processor component. Effective local monitoring can be vital in various dynamic optimizations. Activity migration and thread assignment techniques [10, 11] can be assisted by local thermal monitoring mechanisms. Temperature information regarding local components can be exploited by dynamic cache optimizations [20, 21] to reduce leakage. There can be different approaches for local sensor placement.

Naïve Placement: The naïve approach is to place a fixed number of sensors per processor block. There are different ways to place sensors based on the geometry and alignment of the block. The main idea is to recursively bisect the block into smaller units, until the number of units is equal to the number of desired sensors. For example, this will involve placing a single sensor at the geometric center of the processor block. For two sensors, the block is bisected along the longest edge and a sensor is placed at the center of the each bisected rectangle.

Single Sensor Placement at Weighted Cluster Center: This technique involves placing a single sensor for each processor component. This method is equivalent to applying the thermal gradient-aware placement without performing clustering. All hotspots are by default contained within one single cluster. The location of the center of this cluster is determined using Equation 4 and Equation 5. We experimentally determined that attraction coefficient $\alpha = 0.05$ gives best results for this local placement.

Thermal Gradient-Aware Placement Applied to Local Blocks: In this method multiple sensors are placed per processor block. Such placement is performed by our thermal gradient-aware technique, where k is the number of sensors and their physical locations are the k cluster centers. Increasing the number of sensors increases monitoring accuracy. We observed that selecting number of sensors equal to 2 gives good accuracy thermal sensing for the processor configuration in our experiments.

Hybrid Sensor Placement: In this method the sensor placement is determined by a synergy of weighted cluster center and thermal gradient aware placement strategies. In contrast to the previous approaches where all blocks have equal number of sensors (either single or multiple), we allow allocation of variable number of sensors in different blocks to maximize accuracy. By customizing the number of sensors required for accurate temperature measurement for each block, the number of sensors is also reduced. At first for each processor unit, single sensor placement by weighted cluster center method is performed and temperature error is determined. Then thermal gradient-aware placement is repeatedly applied with increasing number of sensors until no significant improvement in accuracy is observed.

4. EXPERIMENTAL RESULTS

In the following sections we describe our experimental methodology and then we present our results.

4.1 Experimental Setup

We simulated the SPEC2000 benchmark (13 floating point and 12 integer benchmarks) suite [9] using SimpleScalar [22]. SimpleScalar simulates a super-scalar processor with out-of-order issue and execution. We simulate for 10 million instructions after fast-forwarding application-specific number of instructions as proposed by Sherwood et al. [23]. We have chosen Alpha 21364 as our base processor. The base processor is a 4-way processor with a load store queue and register update unit of size 32 and 64 respectively. The level 1 instruction and data cache are 64 KB, 4-way associative with 32-byte block size and 2 cycle latency. Unified level 2 cache is 512 KB, 4-way associative cache with 128-byte line size and 15 cycle latency. Wattch infrastructure [24] is used for architectural level power modeling. The access patterns of the processor blocks from SimpleScalar are then used by Wattch (version 1.02) to compute the power dissipation of the blocks. The power data for 1.6 V at 1 GHz at 180nm node was scaled using Wattch's linear scaling to obtain power for 130nm node, V_{dd}=1.3V, and a clock speed of 3 GHz. We have used HotSpot version 3.0 [5] for performing thermal simulation. The floorplan of Alpha 21364 and the power dissipation from Wattch are used as input to HotSpot. HotSpot can perform thermal simulation in block level and also in the grid level. The processor floorplan is divided into grids and the temperature of each grid element is calculated. Depending on the grid size, a block can have multiple grid elements. This type of grid level thermal modeling is useful for capturing spatial temperature variation within a processor unit. During static thermal simulation the initial die temperature was assumed to be 60°C. This represents the die temperature if the processor was already executing instructions prior to execution of benchmarks to model the warm up period. The ambient temperature is set to 40°C. For 3Ghz clock frequency, a rise in 0.1°C of temperatures takes 100K cycles. Sampling rate of 10K cycles gives the best tradeoff between precision and overhead [5].

For our experiments the point of interest is the hottest point per component. For each benchmark, each component will exhibit a hotspot. For different applications, the location of this hotspot may change. We first combine these locations to find the distribution of hotspots across different benchmarks. Based on this distribution we make decisions of the location of sensors. When a sensor is placed at a certain location, we assume that it reads the exact temperature at that location. We adjusted our grid size to ensure that each component contains a large number of distinctly monitored grid points, i.e., the granularity of the thermal simulation is much finer than the number of components. One of our main goals is to measure the error of temperature readings provided by the sensors. To calculate this error, we consider the temperature read by the sensor (i.e., the temperature of its location) and compare it against all the temperature values of the hotspots within the corresponding component. If there are multiple sensors within a component, we find the error by comparing the temperature reading and all the hotspots that the sensor is associated with.

4.2 Results

Our first set of results presents the maximum error in sensor reading for each benchmark using the global placement technique described in Section 3.2.1 for allocating 16 sensors. We observed that allocating number of sensors equal to the number of blocks in the processor is a good starting point. Then, the number of sensors can be increased or decreased depending on the desired accuracy. Since the starting number of sensors is few, a linear increase or decrease in the number of sensors for meeting accuracy proved to be a good method. For our experiments we found that 16 sensors provide good accuracy. We compared our method against 2 alternatives - 2-dimensional placement (g-*2D*), and 3-dimensional placement (*g-3D*). The 3-dimensional placement computes the distance function using Equation 3 and updating the cluster center as shown in Equation 6.

$$O_{jx,y,t} = \sum_{i \in M_{sj}} h_{ix,y,t} / size(M_{sj}) \qquad (6)$$

Figure 2 shows the maximum error and average of maximum errors for each benchmark using 2-dimensional placement (g-*2D*), 3-dimensional placement (*g-3D*) and our thermal gradient-aware approach (*g-T-GA*). The maximum error for *g-2D* can be as high as 13.7°C whereas that for *g-T-GA* is 4.5°C. *g-T-GA* also performs better than *3D*, which has a maximum error of 6.1°C. For all benchmarks except *wupwise-ref*, *g-2D* exhibits higher maximum error than *g-3D*. For majority of benchmarks it can be observed that the maximum error for *g-T-GA* is less than *g-3D*. The average of the maximum error for *g-2D*, *g-3D*, and *g-T-GA* are 4.5°C, 2.6°C and 2.1°C respectively. This shows *g-T-GA* on average shows 19% improvement in accuracy over *g-3D*.

Our second set of results present the thermal monitoring accuracy for local sensor placement as discussed in Section 3.2.2. In our experiments we observed placing 1 to 2 sensors per block gives good accuracy for our hotspot distribution. Of the different local placement techniques, we present the thermal monitoring accuracy results for weighted-centroid (*l-WC*) for a single sensor per unit, thermal gradient-aware approach (*l-T-GA*) for 2 sensors per block, and hybrid assignment (*l-H*) of one to two sensors per block. These set of results shown in Figure 3 represent the most interesting trends for our proposed methods. For hybrid method, initially a single sensor is placed at the weighted centroid per block and the sensor errors are computed. At this point, we identified the blocks responsible for maximum sensor error. We found that L2_left, FPAdd, FPReg, FPMul, FPMap, IntExec, and FPQ contributed the largest sensor errors. Note that out of the 7 processor units, only L2_left belongs to memory subsystem and the rest are functional units. It can be observed from Figure 1 and Figure 4, L2_left has 2 distinct hotspot clusters – one at the boundary of Icache and another at FPMul. The *l-WC* found the sensor location for L2_left at the center of 2 such hotspot clusters showing large error.

Then, we applied *l-T-GA* for placing 2 sensors in those blocks. Using *l-WC* method, the maximum and average error is 5.2°C, and 9.5°C. When we selectively assigned 2 sensors in 7 blocks namely L2_left, FPAdd, FPReg, FPMul, FPMap, IntExec, and FPQ, the maximum and average error reduced to 3.8°C, and 1.9°C. The maximum error is comparable with *l-T-GA* using 2 sensors per block. The average of the maximum error is slightly better for *l-T-GA* (1.63°C) as compared to 1.9°C in *l-H*. We would like to point out that this increased accuracy for *l-T-GA* over *l-H* comes at an increased number of sensors, which we illustrate in Table 1.

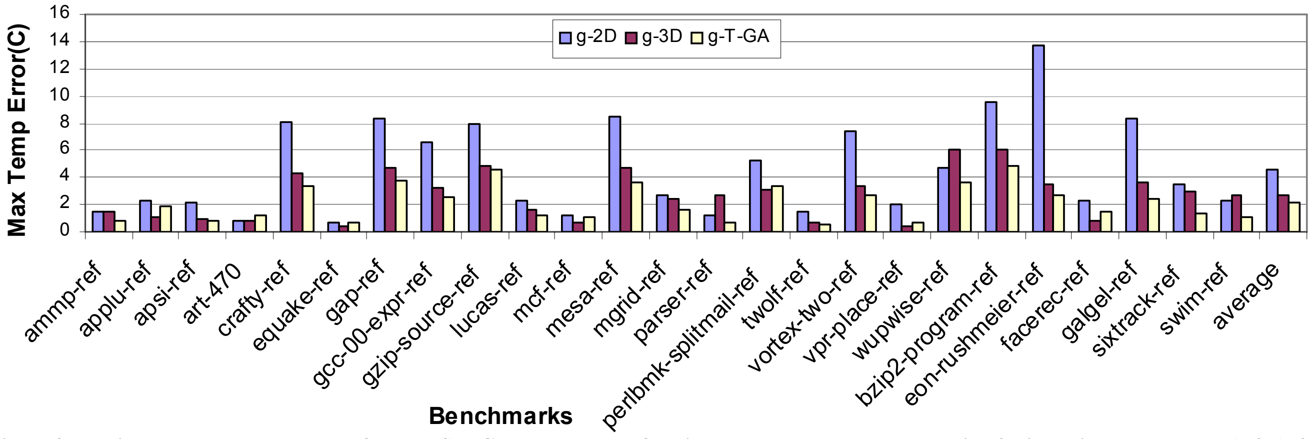

Figure 2. Maximum Temperature Error for each SPEC benchmark performing global sensor placement using 2-dimensional placement (g-2D), 3-dimensional placement (g-3D) and thermal gradient-aware (g-T-GA) techniques.

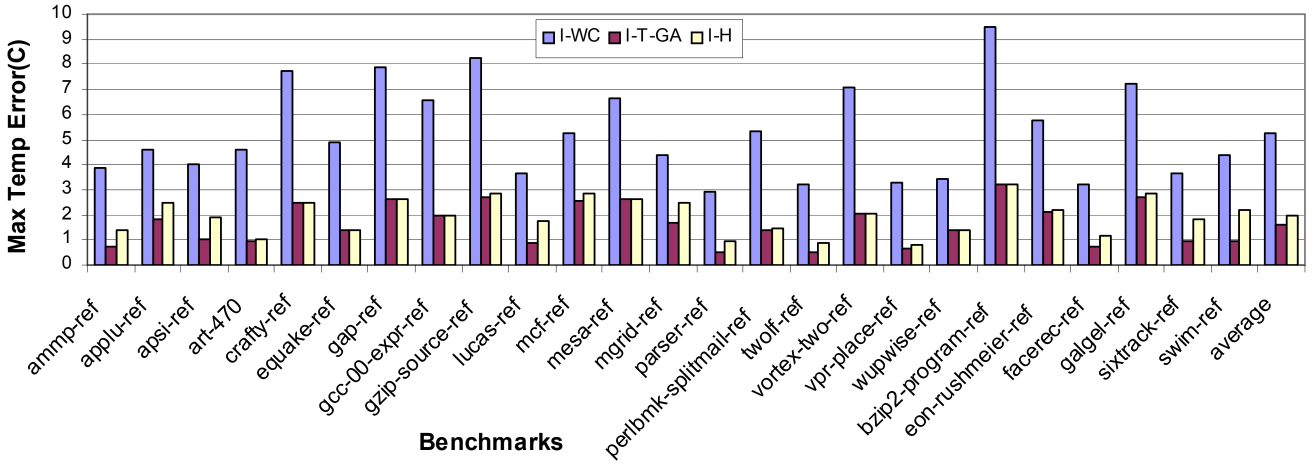

Figure 3. Maximum Temperature Error for each SPEC benchmark performing local sensor placement using one sensor per component using weighted centroid (l-WC), 2 sensors per component using thermal gradient-aware (l-T-GA) technique, and hybrid approach (l-H).

Table 1 summarizes the maximum error and average of maximum errors for each of our sensor placement techniques. We have also listed the number of sensors required for each approach. It is evident that performing *l-T-GA* sensor placement (modeling *attraction* of the sensor to relatively hotter points) performs best. For *g-T-GA* placement the maximum and average of maximum error across benchmarks is higher (maximum: 4.85°C, average: 2.10°C) when compared to *l-T-GA* placement (maximum: 3.18°C, average: 1.63°C). Local sensor placement shows more accuracy. However, for global placement we would require 16 sensors whereas local sensor placement would require 36 sensors (we have assigned 2 sensors per component for 18 processor components). We observe the most interesting results for *l-H* placement. It achieves comparable accuracy to *l-T-GA* with 11 sensors fewer. The maximum errors for *l-H* and *l-T-GA* are same.

Table 1. Maximum error and average of the maximum error for SPEC benchmarks using different global and local thermal sensor placement strategies.

	# of Sensors	Type	Max Err	Avg Err	Sensors
Global	16	2-D placement	13.69°C	4.58°C	16
		3-D placement	6.11°C	2.66°C	16
		T-GA	4.85°C	2.10°C	16
Local (sensor / block)	1	Weighted-centroid	9.46°C	5.25°C	18
	2	Naïve	22.96°C	10.79°C	36
		2-D placement	7.27°C	3.05°C	36
		3-D placement	4.05°C	2.01°C	36
		T-GA	3.18°C	1.63°C	36
	1-2	Hybrid	3.18°C	1.95°C	25

Figure 4 shows the sensor distribution using hybrid approach for a distribution of hotspots. This floorplan zooms onto the core of the processor. Dotted lines represent that the majority of the cache blocks have not been included in this floorplan. It can be seen from the figure that 2 sensors were assigned to the blocks L2_left, FPAdd, FPReg, FPMul, FPMap, IntExec, and FPQ and single sensor assigned to the rest.

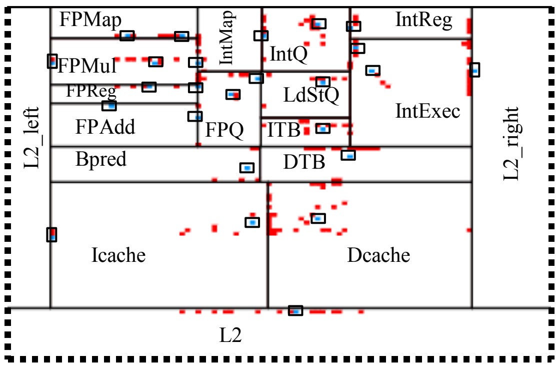

Figure 4. Hotspot distribution (marked in red) across processor blocks and the sensor locations (marked in blue and squares) determined by our hybrid strategy.

5. CONCLUSIONS

We have introduced a systematic technique for thermal sensor allocation and placement in microprocessors. Our algorithm identifies an optimal physical location for each sensor such that the sensor's the attraction towards steep thermal gradients is maximized. The sensor allocation is determined by the required accuracy in temperature measurement. Such accurate monitoring schemes will have implications for dynamic thermal and power management schemes in microprocessors.

The results show that our technique has average accuracy improvement of 19% over 3-dimensional placement for global sensor placement strategy. For local placement strategy thermal gradient aware placement and hybrid methods provide average error for 1.63°C and 1.95°C respectively. The number of sensors is 36 for the former and 25 for the later, which can be used to trade-off number of sensors and accuracy. The results show that our allocation and placement techniques have been indeed effective in minimizing thermal sensing error.

6. ACKNOWLEDGEMENTS

We would like to thank Dr's Gokhan Memik and Russ Joseph for their help in setting up our experimental flow and their general comments and suggestions. This research is supported in part by NSF Career Award #0546305.

7. REFERENCES

1. Borkar, S., *Design Challenges of Technology Scaling.* IEEE Micro, July-August 1999. **19**(4): p. 23--29.
2. Cong, J., et al. *A Thermal-Driven Floorplanning Algorithm for 3D ICs*. in *ICCAD*. 2004.
3. Huang, W., et al. *Compact Thermal Modeling for Temperature-Aware Design*. in *DAC*. 2004.
4. Sankaranarayanan, K., et al., *A Case for Thermal-Aware Floorplanning at the Microarchitectural Level.* The Journal of Instruction-Level Parallelism, 2005. **7**: p. 1--16.
5. Skadron, K., et al. *Temperature-Aware Microarchitecture*. in *ISCA*. 2003.
6. Rotem, E., et al. *Analysis of Thermal Monitor features of the Intel® Pentium® M Processor*. in *TACS Workshop*. 2004.
7. Clabes, J., et al. *Design and Implementation of the POWER5 Microprocessor*. in *DAC*. 2004.
8. Krinitsin, V. *Pentium 4 and Athlon XP: Thermal Conditions*. http://www.digit-life.com/articles/pentium4athlonxpthermalmanagement/.
9. SPEC-CPU2000. *Standard Performance Evaluation Council, Performance Evaluation in the New Millennium, Version 1.1*. 2000.
10. Powell, M.D., et al. *Heat-and-Run: Leveraging SMT and CMP to Manage Power Density Through the Operating System*. in *ASPLOS*. 2004.
11. Heo, S., et al. *Reducing Power Density through Activity Migration*. in *ISLPED*. 2003.
12. Chaparro, P., et al. *Distributing the Frontend for Temperature Reduction*. in *HPCA*. 2005.
13. Lee, K.J. and K. Skadron. *Analytical Model for Sensor Placement on Microprocessors*. in *ICCD*. 2005.
14. Gunther, S., et al., *Managing the Impact of Increasing Microprocessor Power Consumption.* Intel Technology Journal, February 2001.
15. Bratek, P. and A. Kos. *Temperature Sensors Placement Strategy for Fault Diagnosis in Integrated Circuits*. in *SEMI-THERM*. 2001.
16. Meguerdichian, S., et al. *Coverage Problems in Wireless Ad-hoc Sensor Networks*. in *INFOCOM*. 2001.
17. Chvatal, V., *A Combinatorial Theorem in Plane Geometry.* Journal of Combinatorial Theory 1975. **18**: p. 39-41.
18. Sherwani, N.A., *Algorithms for VLSI Physical Design Automation*. 1995, Norwell, MA: Kluwer Academic Publisher.
19. MacQueen, J. *Some Methods for Classification and Analysis of Multivariate Observations*. in *Fifth Berkeley Symposium on Mathematical Statistics and Probability*. 1967.
20. John, J.K., et al. *Optimizing the Thermal Behavior of Subarrayed Data Caches*. in *ICCD*. 2005.
21. Kaxiras, S., et al. *Cache Decay: Exploiting Generational Behavior to Reduce Cache Leakage Power*. in *ISCA*. 2001.
22. Burger, D.C. and T.M. Austin, *The SimpleScalar Tool Set, Version 2.0.* Computer Architecture News, 1997. **25**(3): p. 13-25.
23. Sherwood, T., et al. *Basic Block Distribution Analysis to Find Periodic Behavior and Simulation Points in Applications*. in *PACT*. 2001.
24. Brooks, D., et al. *Wattch: A Framework for Architectural-Level Power Analysis and Optimizations*. in *ISCA*. 2000.