Self-Aware Thermal Management for High Performance Computing Processors

Andrea Bartolini *Member, IEEE*, Roberto Diversi, Daniele Cesarini, Francesco Beneventi
DEI, University of Bologna, 40136 Bologna, Italy

Abstract— Processors for high performance computing and server workload are today thermally constrained. To preserve a safe working temperature, state-of-the-art processors for this market segment integrates many cores on the same die and feature fine-grain power management and thermal management feedback loops implemented in hardware. However, to keep the control policy simple, these controllers fail in taking advantage on the underlining thermal heterogeneity, long thermal transients and specific user mode. In this paper, we present a self-aware thermal management framework for making each chip self-aware of its thermal and workload peculiarities and use it to maximize performance and thermal sustainability.

Index Terms— C.1.2.e Load balancing and task assignment < C.1.2 Multiple Data Stream Architectures (Multiprocessors) < C.1 Processor Architectures < C Computer Systems Organization, Temperature-aware design < B.9 Power Management < B Hardware, B.9 Power Management < B Hardware, C.1.4.d Scheduling and task partitioning < C.1.4 Parallel Architectures < C.1 Processor Architectures < C Computer Systems Organization

1. Introduction and Related Works

Nowadays it is well established that the pace dictated by the Moore's law on technological scaling comes at the cost of increasing power consumption and leads to thermally-bound computing systems. Supercomputers as well as data centres are on the cutting edge of this crisis, because of aggressive performance, integration density and sustainable power budget [1]. The top500 list ranks worldwide supercomputers based on their peak performance (Flops - floating point operations per second - when running a Linpack benchmark) [2]. Today the most powerful supercomputer in Top500 is Sunway TaihuLight which consumes 15.3 MW for delivering 93 Petaflops. The second one, Tianhe-2 (ex 1st) consumes 17.8 MW for "only" 33.2 Petaflops. However, the power consumption increases to 24 MW when considering also the cooling infrastructure [3]. Such an amount of cooling power serves to prevent thermal issues. Increasing the inlet coolant temperature reduces the cooling cost, but jeopardises the thermal budget and consequently the performance.

Modern multicore processors are equipped with thermal sensors and with fine-grain power management support to modulate the power consumption of the processor and peripherals based on current operating conditions. Intel RAPL modulates the frequency of the cores to maintain the processor power consumption below a user defined power budget. In novel Intel server processors, the frequency of each core can be scaled independently [5]. However, this flexibility is currently not fully explored as the operating systems in HPC installations (99.6% of Top500 supercomputers in November 2016 use Linux as O.S.) work by keeping all the cores at the same frequency to avoid performance-unbalance across the cores of the same processor. When it comes to hardware mechanisms to protect the die from over temperature (referred as thermal management) today's processors use two mechanisms: (i) dynamic voltage and frequency scaling (DVFS - ACPI P-states) as well as (ii) duty-cycling (Throttling - ACPI T-states). Thanks to turbo logic, cores can run at frequencies above the nominal TDP frequency without incurring in overheating and voltage drop problems. Duty-cycling is used as protection mechanism when die temperature exceed a critical threshold (in Intel Xeon E5-26XX v3 HPC class processors vary from 69°C to 101°C). However thermal throttling is strongly detrimental to core performance as the performance loss increases linearly with the power reduction. Differently, for DVFS low-power states the performance loss scales sub linearly with the power reduction. Moskovsky et al. [4] show that hardware built in mechanisms can use effectively the DVFS states for power and energy management but fail in preventing thermal throttling [4]. This is primarily due to the reactive nature of thermal controllers which takes corrective actions when the chip is already too hot.

In addition, there are several nonidealities in the thermal characteristics of large multicore processors targeting the high-performance computing market. Namely thermal heterogeneity, thermal capacitance and thermal noise. Beneventi et al [6] measures for a top end Intel Xeon computing node featuring 36 physical cores, on the same silicon die, up to 24°C of temperature difference between active cores and idle cores, and more than 7°C of thermal heterogeneity when all the cores execute identical workload and run at the same frequency. Precision of thermal sensors is bound to 1°C due to quantization. Moreover, there is an important thermal capacitance due to the large heat-sink and heat spreader which causes long thermal transients taking minutes to reach the steady-state after a change in the cores power consumption. These nonidealities worsen the efficiency of built in reactive controllers as the usage of a single frequency to control the temperature of a multicore chip - subjected to strong

thermal heterogeneity - is detrimental. This in addition to the difficulties in finding a stable but thermally safe operating point.

Finally, in a supercomputing environment, user's applications run in isolation on a portion of the machine and are accounted based on the execution time. The peculiar usage model of supercomputing hardware and applications, which are not interactive, nor latency critical, but require sustained peak performance, is not yet well understood in terms of unique opportunities for domain specific power and thermal management. Each user runs parallel workload with synchronization points on the multi-core and multi-node hardware. For this reason, slowing the frequency of the core running the critical thread may induce a slow-down in threads waiting for its completion to progress. Thread migration is prevented to avoid performance loss.

Authors of [7,8] survey thermal management and allocation strategies. From the survey we can conclude that most of the state-of-the-art works take advantage of (1) task migration to improve purely reactive approaches, (2) do not consider thread criticality but assumes deadlines or precedence constraints, (3) use thermal models obtained from thermal simulators and/or make simplifying assumptions on the intra-core temperature dependency, and neglect the thermal transients. These make them not directly applicable to supercomputers. Even if not directly applicable in the experimental results section we will compare the proposed strategy with thermal management ones which uses static thermal models for driving the task assignment, as well as purely reactive thermal controllers which uses short predictions to tune the power consumption dynamically.

In this paper, we present a self-aware optimization framework which combines noisy thermal sensor readings, robust system identification strategy, integer-linear programming optimization and application awareness to ensure a stable and safe working temperature. At the same time, the framework maximizes the application performance in a state-of-the-art high-performance computing nodes. We use this framework for discussing the design trade-offs of self-aware and application-aware proactive thermal management run-times targeting HPC systems. These includes: (i) Thermal capacitance; (ii) Noise and system identification; (iii) Workload awareness and job placement; (iv) HPC peculiarities.

2. SELF AND APPLICATION-AWARE THERMAL MANAGEMENT FRAMEWORK

In this section, we describe the proposed framework. Figure 1 shows its building blocks.

At the bottom, we have the HPC hardware which is composed of several compute nodes. Each compute node (*Node#i* in Figure 1) is composed of one to M computing engine. We consider as computing engine parallel processors, which are the most common case (more than 90% of Top500 systems use Intel Xeon Processors, as reported by the November 2016 Top500 list). Each parallel processor (CPU#i) is internally composed of N cores and uncore logic. The uncore logic accounts for the memory controllers, last level cache and I/O. Each processor, core and uncore are equipped with sensors which can be read periodically from the software stack. These sensors can monitor different architectural events as well as physical parameters. We focus on three main classes of sensors: per-core activity sensor, per-core thermal sensors, per-cpu power gauges.

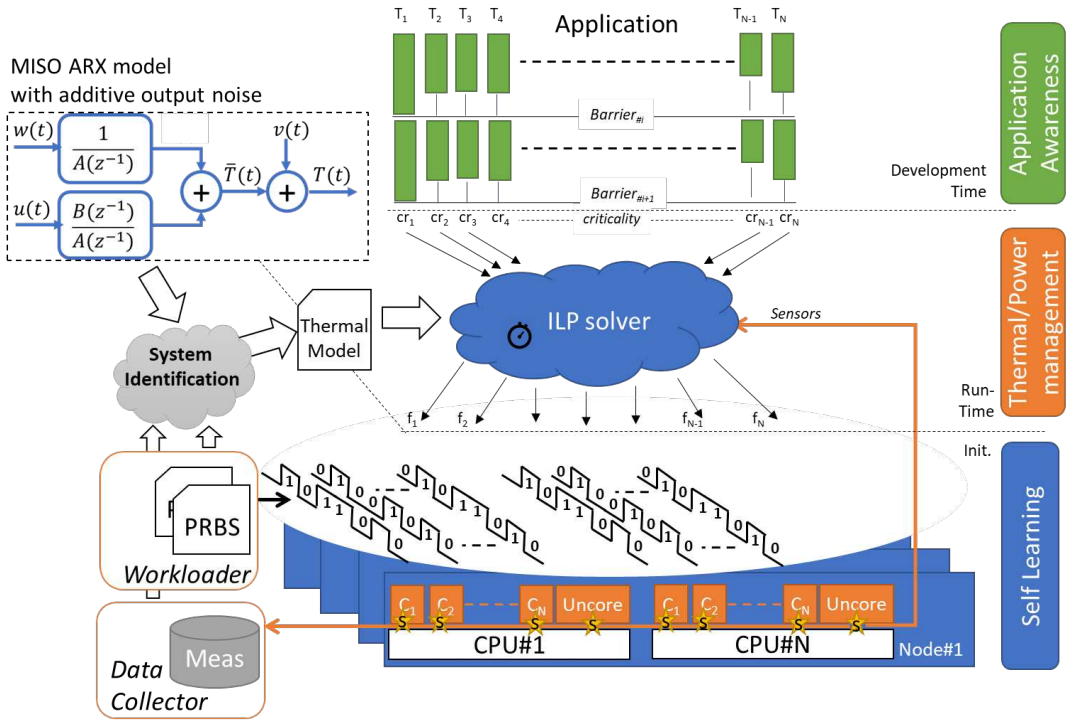

Figure 1: Block diagram of the self-aware and application aware power and thermal management framework.

These sensors are used for two purposes: self-learning a predictive thermal model through system identification and estimating the current state of the cores (silicon temperature, power consumption, activity) to be used as the basis for estimating the future core temperature and power consumption in the thermal and power management problem.

At the top of the stack we have the software applications running on the system which consist of parallel programs composed of several threads and processes which run exclusively on the computing resources (i.e. the cores). As effect of the computational patterns and data partitioning, threads of the same application running on different cores may require a longer computation before reaching a synchronization point. This makes the core running the thread which requires the longer computation the critical one in terms of completion time of the entire application. The programmer, or the run-time of the programming model, can estimate the criticality of the threads composing the application and flags the threads with a label which represent the application criticality (cr_x in Figure 1).

At the centre of the stack the proposed framework implements the *Self Learning* and the *Thermal/Power Management* policies. The *Self-Learning* policy consists of three stages. (i) First binary workload (1=active/0=idle) sequences are applied to each core to recreate a Pseudo Random Binary Sequence (PRBS) of power stress in each core. PRBS sequences have the characteristic of creating white noise spectrum with sequences of binary inputs. This guarantees that all the modes in the thermal removal path are equally exited. The power consumption values for each core is obtained by linear regression of each core's activity (cycles in active state and cycles in idle state) and per CPU power consumption. Our framework uses the Workloader to schedule one power stressmark in each core and uses POSIX signals to schedule and de-schedule it to follow the PRBS sequence. (ii) During this stress pattern, the Data Collector monitors the sensors present on each core and cpu synchronously with the PRBS sampling time. The values collected are stored in a database and pre-processed to translate each core activity in a per core power profile. Finally, (iii) the power traces and thermal responses are provided as input to a system identification algorithm which extracts a predictive thermal model. The thermal model is in the form of a Multi-Input Single-Output (MISO) AutoreRegressive eXogenous (ARX) dynamic model for each core.

The *Self-learning* policy is executed during system initialization and last for few minutes. The learnt thermal model is then fed as input to the *Thermal/Power Management* run-time. One of the main advantages of the proposed thermal modelling framework is that it is suitable to be integrated in a production environment. Indeed, it does not need to connect the computing node to lab equipment but it runs entirely in software on the inspected node. The running overhead is below the 1% and requires a small amount of data.

Differently the run-time executes periodically. At each iteration, it reads the application criticality level (cr_x) for each thread running in the CPU, the sensors' state, and based on these values it takes a decision. The decisions taken are different if taken at application start up or during its execution. At the *1st step* the *Thermal/Power Management* run-time decides the thread to core mapping and the frequency at which running each core. At the *n-th step* the run-time decides only the frequency at which running the core without exceeding the critical temperature. Indeed, in scientific HPC environment once the threads of a parallel application are pinned to a core their migration is not allowed to avoid performance losses. The proposed *Thermal/Power Management* run-time uses an Integer Linear Programming formulation to find the optimal thread to core mapping and cores frequencies which maximizes the application performance while preserving a safe working temperature for each core. Thermal interaction between the cores, thermal differences, as well as thermal capacitance effects (dynamics) are accounted in the optimization problem by mean of the thermal model self-learnt on the HW which is the result of the *Self Learning* policy. To account for the different decision variables available at the *1st step* and *n-th step* of the *Thermal/Power Management* run-time two different optimization problems are used.

3. THE SELF-LEARNING POLICY

The self-learning policy is implemented by adopting a distributed approach. Each core of a supercomputer node is represented by a MISO ARX model where the output is the actual core temperature $\bar{T}(t)$ and the inputs are the power dissipated by all the N cores $P_0(t), P_1(t), \ldots P_{N-1}(t)$ and the uncore power $P_{unc}(t)$. Note that the $P_{unc}(t)$ represents the power consumed by the uncore of the CPU as reported in Figure 1. This model is described by the following difference equation

$$A(z^{-1})\bar{T}(t) = B(z^{-1})u(t) + w(t) \tag{1.1}$$

where $u(t)$ is the input vector

$$u(t) = [P_0(t)\, P_1(t) \ldots P_{r-1}(t)\, P_{unc}(t)]^T \triangleq [u_0(t)\, u_1(t) \ldots u_{r-1}(t)\, u_r(t)]^T, \tag{1.2}$$

$w(t)$ is a white process with variance σ_w^{2*} and $A(z^{-1})$ and $B(z^{-1})$ are a polynomial and a $1 \times (r+1)$ matrix of polynomials in the backward shift operator z^{-1} (i.e. $z^{-1}x(t) = x(t-1)$):

$$\begin{aligned} A(z^{-1}) &= 1 + a_1 z^{-1} + \cdots + a_n z^{-n} \\ B(z^{-1}) &= \begin{bmatrix} B_0(z^{-1}) & B_1(z^{-1}) & \cdots & B_r(z^{-1}) \end{bmatrix} \\ B_i(z^{-1}) &= b_{i1} z^{-1} + \cdots + b_{in} z^{-n} \quad i = 0, \ldots, r. \end{aligned} \tag{1.3}$$

Previous works have shown that the relation between the core temperature and the dissipated power can be described by a purely dynamic model [9].

ARX models are widely used in system identification since they constitute the simplest way of representing a dynamic process in the presence of uncertainties [10]. Two important features of these models are the possibility of obtaining asymptotically unbiased estimates of their parameters by means of least squares and the absence of stability problems of the associated optimal one step-ahead predictors [10]. Nevertheless, it has been shown in [9] and [11] that the classic MISO ARX model (1.1) is not able to describe properly the thermal dynamics of the system because the estimated models are characterized by relevant negative poles and/or complex conjugate poles. This is in contrast with the physics of thermal systems, where only real positive poles can exist. As explained in [9], this problem is due to the presence of a significant level of measurement noise. To take into account the presence of this noise, MISO ARX models with noisy output have been considered. The actual core temperature $\bar{T}(t)$ is thus assumed as affected by an additive noise $v(t)$ so that the available measurement $T(t)$ is given by

$$T(t) = \bar{T}(t) + v(t) \tag{1.4}$$

where $v(t)$ is a white process with variance σ_v^{2*}. From (1.1) and (1.4) the available temperature $T(t)$ can be expressed as

(1.5) $$T(t)=\frac{B(z^{-1})}{A(z^{-1})}u(t)+\frac{1}{A(z^{-1})}w(t)+v(t)$$

This equation emphasizes the presence of both a process disturbance $1/A(z^{-1})w(t)$ and a measurement noise $v(t)$ as reported in Figure 1. The identification problem consists in estimating for each core the coefficients of $A(z^{-1})$, $B_i(z^{-1})$, $(i=0\ldots,r)$ i.e. the parameter vector

(1.6) $$\theta^{*}=\left[a_1\cdots a_n\, b_{01}\cdots b_{0n}\cdots b_{r1}\cdots b_{rn}\right]^{T}$$

starting from the input-output data $u(1),\ldots,u(N),T(1),\ldots,T(N)$.

The MISO ARX model with additive noise (1.5) cannot be identified by using the least squares method as the presence of $v(t)$ leads to asymptotically biased estimates [9]. For this reason, we propose an ad-hoc identification algorithm that exploits the properties of the dynamic Frisch scheme [12]. The rationale behind the Frisch scheme consists in searching for the solution of the identification problem within a locus of solutions which are compatible with the covariance matrix of the noisy data [12]. Let us introduce the vector $\varphi(t)=[-T(t)\ldots-T(t-n)u_0(t-1)\ldots u_0(t-n)u_1(t-1)\ldots u_1(t-n)\ldots u_r(t-1)\ldots u_r(t-n)]^{T}$ and its covariance matrix $\Sigma=E[\varphi(t)\varphi^{T}(t)]$.

It can be shown that in this context the above-mentioned locus of solutions, compatible with the covariance matrix of the noisy data Σ, consists in an interval describing a set of admissible additive noise variances. More precisely, consider the following two partitions of the same matrix Σ:

(1.7) $$\Sigma=\begin{bmatrix}\sigma_T^2 & \rho^{T}\\ \rho & R\end{bmatrix}=\begin{bmatrix}\Sigma_{TT} & \Sigma_{Tu}\\ \Sigma_{Tu}^{T} & \Sigma_{uu}\end{bmatrix}$$

where σ_T^2 is a scalar and Σ_{TT} is a $(n+1)$x$(n+1)$ matrix. We define also the scalar $\sigma^2_{v,\max}$ as the minimum eigenvalue of the matrix $\left(\Sigma_{TT}-\Sigma_{Tu}\Sigma_{uu}^{-1}\Sigma_{Tu}^{T}\right)$. It is possible to prove that every value σ_v^2 belonging to the interval $I(\Sigma)=[0,\ \sigma^2_{v,\max}]$ is an admissible additive noise variance from which it is possible to compute a possible parameter vector $\theta(\sigma_v^2)$ by means of $\theta(\sigma_v^2)=-\left(R-\tilde{R}(\sigma_v^2)\right)^{-1}\rho$, where $\tilde{R}(\sigma_v^2)=\mathrm{diag}[\underbrace{\sigma_v^2\cdots\sigma_v^2}_{n}\underbrace{0\cdots0}_{(r+1)n}]$ and diag denotes a diagonal matrix. It is also possible to compute a value of the process noise variance $\sigma_w^2=\sigma_T^2-\sigma_v^2+\rho^{T}\theta$. To find the true noise variance σ_v^{2*} within $I(\Sigma)$, and then the true parameter vector θ^{*}, it is necessary to introduce a suitable selection criterion. The criterion proposed in [11] exploits the properties of the instrumental variable (IV) approach and leads to asymptotically unbiased estimates when the covariance matrix Σ is replaced by its sample estimate $\hat{\Sigma}=\frac{1}{N-n}\sum_{t=n+1}^{t=N}\varphi(t)\varphi^{T}(t)$. A loss function $J(\sigma_v^2)$ based on the IV approach is minimized along $I(\hat{\Sigma})$ to get consistent estimates of θ^{*}, σ_v^{2*} and σ_w^{2*}, see [11] for the details. Finally, the identified models can be transformed into a state-space representation of the noisy ARX model (1.5) to be used in the ILP problem to proactively select the optimal usage of the resources.

The proposed identification method is more effective in the considered supercomputer system framework. Thanks to its robustness it can extract the faster dynamics in the presence of quantization noise even when working with a slow sampling rate as shown in [11].

4. THERMAL/POWER MANAGEMENT RUN-TIME

As introduced in Section 2, the thermal model self-learnt on the target HW is used to solve two optimization problems: the *1st step* and the *n-th step*. The Thermal/Power Management run-time operates at the node level and it is composed of two main components: the thermal-aware thread mapper and controller (TMC) and an energy-aware Message Passing Interface (MPI) wrapper. The TMC is triggered: (a) after the job scheduler has deployed the parallel application on the reserved portion of the HPC machine for the job execution; (b) periodically with period T_s . At job startup (a) the TMC specifies the thread to core mapping which will be maintained until application completion. Clearly, if a critical thread is mapped to a thermally inefficient core this will more likely cause a severe degradation of the final application performance. To abstract this requirement, we use a per-thread criticality level (cr_x). Higher criticality means higher impact of the thread performance on the final application.

The *1st step* optimization problem (FSP) is solved during the initialization of the application. Its purpose is to allocate the application's threads on the available cores and selecting for each of them the maximum frequency which meets the thermal constraint T_{max} in the prediction interval (PI_{FSP}). As we will see in the experimental results the prediction interval (i.e. the time horizon) plays an important role. Indeed, if it is too short, the TMC cannot predict the long-term impact of a thread allocation on the cores temperatures as its effect is hidden by the thermal capacitance. On the contrary, if the predicted interval is too long the TMC cannot take advantage of the thermal capacitance for sustaining short time power burst. In addition, as early introduced not all the threads have the same criticality. This is reflected in the optimization model which maximizes the frequency of the most critical thread penalizing the frequencies of others. The optimization problem considers K threads to be assigned to N cores where there are less or equal threads than cores i.e., K ≤ N. Each core can be configured with a frequency in a set of M levels. The Object Function (O.F.) maximizes the sum of frequencies of all active cores γ_{jf} weighted by the criticalities cr_i of the thread assigned on each core. To model the problem, we use two sets of binary decision variables:

(1.8) $$x_{jf}^{i} = \begin{cases} 1 \text{ if core j(j=1,…,N)works at frequency f(f=1,…,M) and executs job i(i=1,…,K)} \\ 0 \text{ otherwise} \end{cases}$$

(1.9) $$y_j = \begin{cases} 1 \text{ if core j(j=1,…,N)is active} \\ 0 \text{ otherwise, i.e., if it is idle} \end{cases}$$

We can formulate the following ILP model with three constraints to model the assignments and the thermal bounds:

1.10 $$O.F. = max \sum_{i=1}^{K} \sum_{f=1}^{M} \sum_{j=1}^{N} cr_i \gamma_{jf} x_{jf}^{i}$$

1.11 $$\sum_{j=1}^{N} \sum_{f=1}^{M} x_{jf}^{i} = 1, i = 1,\ldots,K$$

1.12 $$\sum_{j=1}^{N} \sum_{f=1}^{M} x_{jf}^{i} + y_j = 1, j = 1,\ldots,N$$

1.13 $$\sum_{j=1}^{N} GS_{jl} \left(\vec{p}_j y_j + \sum_{i=1}^{K} \sum_{f=1}^{M} p_{jf} x_{jf}^{i} \right) + T_l^0 + T^a \leq T_{MAX}, l = 1,\ldots,N$$

The constraint (1.11) specifies that a thread must be assigned only on a single core which works at a given frequency. In addition, it specifies that all the threads are assigned. The constraint (1.12) is needed to determine the decision variables which represent the idle cores. These variables are used in the constraint (1.13) in case there are less jobs than cores i.e., K ≤ N. This constraint guarantees that the temperature of each core does not exceed T_{MAX} during the next predicted interval (PI_{FSP}). In the last constraint (1.13) GS is a gain matrix with dimension $N \times N$. This matrix is used to calculate the increment of temperature of all the cores when a core is subjected to a constant power input for PI_{FSP} seconds. T_l^0 represents the dependency of the future temperature

(@ PI_{FSP}) from the current cores temperature. These values are derived from the state-space thermal model learnt in Section 3. T^a is the ambient temperature. When threads are less than cores the decision variable y_i is used in conjunction with the vector of idle powers $\bar{p}$, to add the idle power components.

After the threads have been assigned to the cores in the FSP, the TMC must solve periodically the assignment problem of frequencies to cores only, but at a finer time scale. The *i-th step* problem (ISP) has the same objective function as FSP (1.10) as well as the same thermal model formulation. However, the prediction interval for the ISP (PI_{ISP}) can be generally different from the FSP.
Differently from the previous case, the model considers only active cores (K) because the thermal constraints cannot be broken by an idle core. This reduces the overall complexity. As threads have been already allocated in FSP in this model, threads and cores do not need separate variables, thus a criticality level is referred to a core.

1.14 $$x_{rf} = \begin{cases} 1 \text{ if core r}(r=1,\ldots,K) \text{works at frequency f}(\text{f}=1,\ldots,\text{M}) \\ 0 \text{ otherwise} \end{cases}$$

The ISP model requires fewer constraints than FSP due the lower number of variables.

(1.15) $$O.F. = max \sum_{a \in A} \sum_{f=1}^{M} cr_a \gamma_{af} x_{af}$$

(1.16) $$\sum_{f=1}^{M} x_{af} = 1, \forall a \in \text{A}$$

(1.17) $$\sum_{a \in A} \sum_{f=1}^{M} GS_{la} p_{af} x_{af} + \sum_{i \in I} GS_{li} \vec{p}_i + T_l^0 + T^a \leq T_{MAX}, \forall l \in A$$

The constraint (1.16) bounds each core to a selected frequency. The constraint (1.17) guarantees the thermal limits imposed on the model, where the set $A = a_i$ contains the index of the active cores, the set $I = i_i$ contains the index of Idle cores directly defined from the solution of FSP and $A \cap I$ is empty. In general, the ISP problem is computationally simpler than the FSP problem due to the much lower number of decision variables and constraints.

In the next section, we will evaluate the performance of the proposed TMC in a realistic scenario and under different trade-offs in between the predicted horizons of the FSP and ISP problems.

5. EXPERIMENTAL RESULTS

In this section, we first describe the performance of the system identification step we carried on a supercomputing node based on dual socket Intel Haswell E5-2630 v3 CPUs, with 8 cores with 2.4 GHz clock speed and 130W Thermal Design Power (TDP). We will use this thermal model together with the proposed framework to study the implication of the prediction interval/horizon in the thermal-aware thread mapping and control of supercomputer nodes. For executing these tests we have assigned the highest criticality on the core running the MPI process with rank 0.

The TMC optimization problem proposed in Section 4 has been solved using IBM ILOG CPLEX 12.6.1. The solver call CPLEX each time there is a new TMC problem to be solved. At each CPLEX call, the run-time builds a new instance of the problem with the new thermal parameters and the criticality of the threads and it waits for CPLEX results. In our tests, we conducted the following experiments with different prediction intervals for both FSP and ISP problems. We considered PI_{FSP} and PI_{ISP} ={1s,10s,100s,SS}. In the following we name these tests with the notation $PI_{FSP} - PI_{ISP}$. It must be noted that 1s-1s represent state-of-the-art DTM solutions with no thermal-aware thread-to-core mapping, while SS-SS represents state-of-the-art static DTM solutions.

For all the experiments, we set the temperature limit 65% of maximum temperature which can be reached by the hottest core at the maximum frequency.

Figure 2a) shows the accuracy for one core of the predicted temperature in the target architecture. The residual is always below ±1°C.

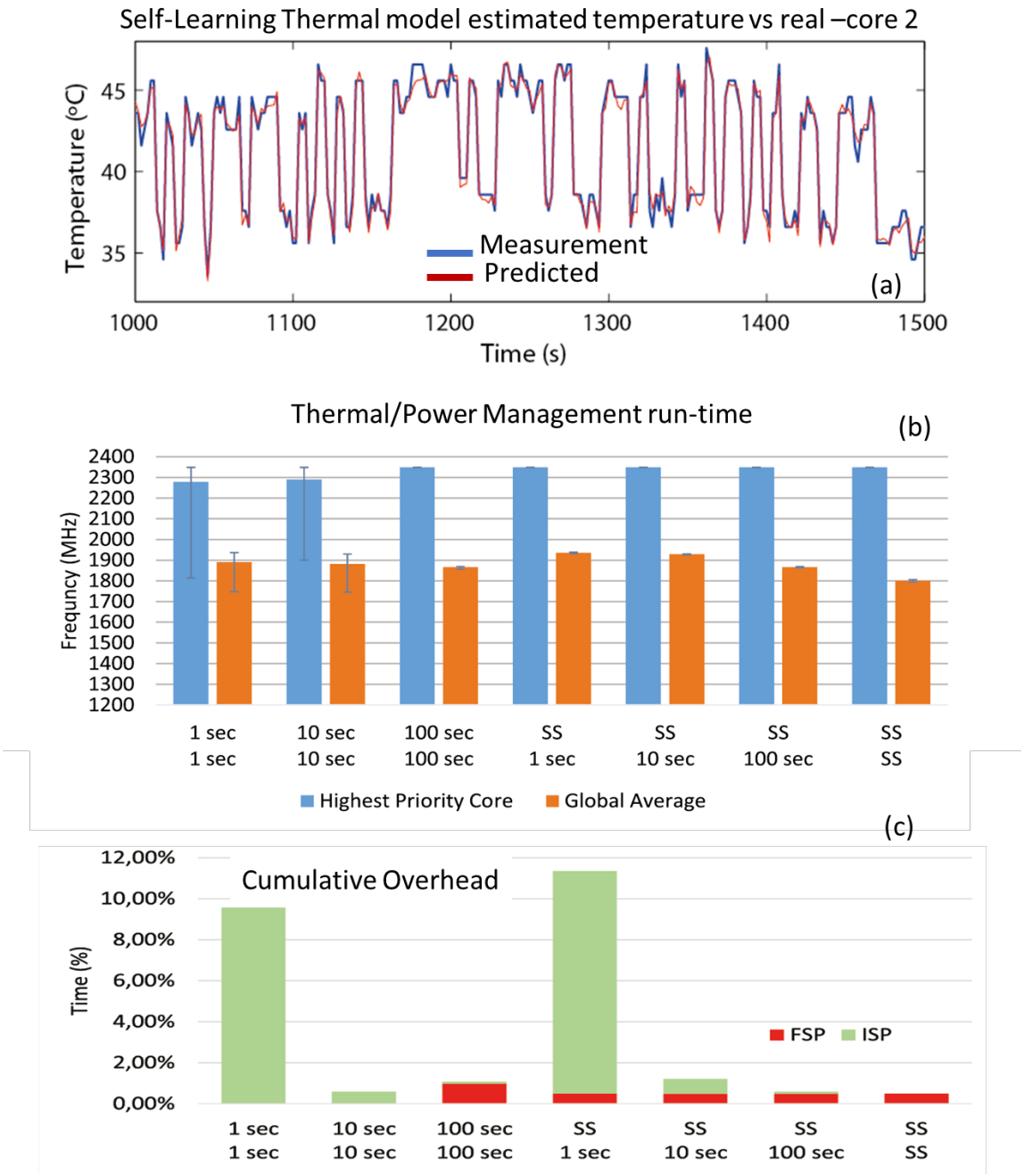

Figure 2 : Self-learned thermal model accuracy (a); Thermal/Power Manager performance vs. prediction horizon (b); Cumulative Overhead (c)

Figure 2b) depicts the average frequency of the core that hosts the highest criticality thread and the average frequency for all the cores in each configuration. The error bar shows the variance for each configuration among different executions. Larger error bars happen when FSP has a prediction horizon which is too short to see the effect of long term thermal evolution, and thus it cannot predict which core will hit the thermal constraint. For this case, the allocation problem FSP is trivial and threads are allocated following a simple numerical binding. If the most critical thread is lucky, it will be pinned on a ``cold'' core. Vice versa, if the most critical thread is unlucky, it will be mapped on a ``hot'' core. At the steady-state the frequencies of the cores will be limited by the ISP to

respect the thermal constraint. On the other cases, the PI_{FSP} is always enough to sense the thermal constraint. The optimization model will pin the highest criticality thread on a ``cold'' core, making it run at maximum frequency.

In the following analysis we take as a baseline the SS-SS configuration, which model the state-of-the-art solutions based on static allocation of jobs and frequency. The 1s-1s and 10s-10s induces performance penalties on the high criticality task, while they lead to an increase of performance of the 4.97% and 4.50% respectively in average in all the cores. For the remaining configurations, we measure no penalties for the high criticality tasks and a gain of to 7.46%, 7.06% and 3.65% respectively for the configuration SS-1s, SS-10s and SS-100s. These results show that short horizon predictive models pay off in the ISP as they allow to take advantage of the thermal capacitance. When considering also the problem solution overhead (Figure 2c) the best choice is the SS-10s configuration which induces an overhead of the 0.64% which in conjunction to the 7.06% of performance gain lead to an overall performance gain of the 6%.

6. CONCLUSIONS

In this paper we presented self-aware thermal management for high performance computing processors which features a robust self-calibrating thermal model policy and a ILP based application aware thermal controller. Our results show that our proposed framework is suitable to learn the thermal model with high accuracy and to take advantage of thermal heterogeneity and capacitance present in state-of-the-art HPC processors.

REFERENCES

[1] W. c. Feng and K. Cameron, "The Green500 List: Encouraging Sustainable Supercomputing," in Computer, vol. 40, no. 12, pp. 50-55, 2007.

[2] Dongarra, Jack, and Piotr Luszczek. "TOP500." Encyclopedia of Parallel Computing. Springer US, 2011.

[3] J. Dongarra. “Visit to the national university for defense technology changsha”. China, University of Tennessee, 2013.

[4] A. Moskovsky et al., “Server level liquid cooling: Do higher system temperatures improve energy efficiency?” Supercomputing frontiers and innovations, 3(1):67-74, 2016.

[5] D. Hackenberg et al., “An energy efficiency feature survey of the Intel haswell processor”. IPDPSW 2015.

[6] F. Beneventi et al., Cooling-aware node-level task allocation for next-generation green HPC systems management, HPCS 2016.

[7] Joonho Kong et al. Recent thermal management techniques for microprocessors. ACM Comput. Surv. 44, 3, Article 13, 2012

[8] Hafiz Fahad Sheikh et al., An overview and classification of thermal-aware scheduling techniques for multi-core processing systems, In Sustainable Computing: Informatics and Systems, Volume 2, Issue 3, 2012, Pages 151-169

[9] R. Diversi et al., Bias-compensated least squares identification of distributed thermal models for many-core systems-on-chip, IEEE Trans. on Circuits and Systems I, vol. 61, pp. 2663-2676, 2014.

[10] L. Ljung, System Identification – Theory for the User, Prentice Hall, Englewood Cliffs, NJ, 1999.

[11] R. Diversi et al., Thermal model identification of supercomputing nodes in production environment, IECON 2016.

[12] R. Guidorzi et al., The Frisch scheme in algebraic and dynamic identification problems, Kybernetika, vol. 44, pp. 585–616, 2008.

ACKNOWLEDGMENT

Work supported by the EU FETHPC project ANTAREX (g.a. 671623),EU project ExaNoDe (g.a. 671578), and EU ERC Project MULTI-THERMAN (g.a. 291125).

Andrea Bartolini (M13) is Post-Doctoral Researcher with the Integrated Systems Laboratory, ETH Zurich, Zurich, Switzerland. His current research interests include green computing from embedded to large-scale HPC systems with special emphasis on thermal and power-aware HW/SW co-design techniques.

Roberto Diversi is Associate Professor at the Department of Electrical, Electronic and Information Engineering, University of Bologna. His research interests include system identification, signal processing and fault diagnosis.

Daniele Cesarini is second year Ph.D. student in the Department of Electrical, Electronic and Information Engineering, University of Bologna. His research interests include parallel programming models and energy-efficient runtimes for embedded and HPC systems.

Francesco Beneventi is a Research Assistant in the Department of Electrical, Electronic and Information Engineering at the University of Bologna. His research interests include energy-aware and HPC, system identification and thermal modelling.