Prediction Horizon vs. Efficiency of Optimal Dynamic Thermal Control Policies in HPC Nodes

Prediction Horizon vs. Efficiency of Optimal Dynamic Thermal Control Policies in HPC Nodes

Daniele Cesarini†, Andrea Bartolini†, and Luca Benini†‡
†DEI, University of Bologna, 40136 Bologna, Italy
‡ IIS, Swiss Federal Institute of Technology, 8092 Zurich, Switzerland
{daniele.cesarini, a.bartolini}@unibo.it, lbenini@iis.ee.ethz.ch

***Abstract*—We are entering the era of thermally-bound computing: Advanced and costly cooling solutions are needed to sustain the high computing densities of high-performance computing equipment. To reduce cooling costs and cooling overprovisioning, dynamic thermal management (DTM) strategies aim at controlling the device temperature by modulating online the performance of processing elements. While operating systems allow the migration of threads between cores, in HPC systems the threads of parallel applications are pinned to the allocated cores at start-time to avoid job-migration overheads. In this scenario state-of-the-art DTM solutions, which use thermal models to map jobs to cores, are based on long-term predictions to map the most critical job to the coldest core. Instead, turbo-mode and DVFS controllers are based on short-term predictions to squeeze the thermal capacitance allowing for short period performance boosts which are thermally unsustainable.**

In this work we propose an integer-linear programming formulation and a fast solver for controlling, at the same time, the job mapping and cores frequency selections in HPC nodes, tested with real supercomputer workload. Our approach can be integrated with the MPI runtimes and OpenMP libraries and is capable of assigning high-performance cores to performance-critical threads. We show that by combining long and short term predictions with information of the programming model we can significantly improve the performance of final application w.r.t. state-of-the-art DTM solutions.

I. Introduction

Nowadays, it is well established that the pace dictated by the Moore's law on technology scaling of electronic devices come at the cost of increasing power densities and leads to thermally-bound computing systems. This is visible for a large variety of systems, ranging from smartphones to supercomputers and data centers.

In high-performance computing nodes, the maximum safe temperature depends on the cooling solution adopted. As an example, this ranges from 69^oC to 101^oC according to the package thermal resistance (cost) and the nominal thermal design power (TDP)[1] for Intel Xeon E5-26XX v3 server class.

Dynamic thermal management (DTM) reduces the cooling effort by controlling and limiting, when necessary, the heat generation. This is done by creating feedback loops between HW thermal sensors, the core's DVFS state, and workload allocation. Today's novel multi-cores allow to scale the frequency and voltage of each core independently, opening novel opportunities for fine-grained DTM solutions [1]. Operating systems use reactive controllers to maintain the processors under a critical temperature, while several approaches in the state-of-the-art explore proactive approaches to improve DTM performances [2], [3], [4], [5].

In this work, we bridge the gap between static solutions that are optimal from the thermal viewpoint and respect job-pinning constraints, and dynamic solutions that are very effective in exploiting thermal capacitances. We study the tradeoff between these solutions in terms of QoS, overhead, maximum temperature and DTM prediction horizon. We first characterize the thermal properties of a real supercomputer node and the impact of the control knobs on real application performance. We show that the dependency of the application walltime vs. the cores frequency is strongly impacted by the communication patterns, as Message-Parsing Interface (MPI) synchronization primitives do not exploit low-power states. Secondly, we propose an Integer Linear Programming (ILP) formulation and a time constrained approximate solving strategy for selecting the optimal job/frequency to core mapping accordingly to future thermal predictions with different time horizons. We show that significant performance improvements can be achieved in supercomputer environments by allocating the application tasks with long prediction horizons while job frequency allocation needs to be performed with short term prediction horizons.

Section II shows the state-of-the-art works on thermal management. Section III presents a characterization of supercomputing nodes and applications from the DTM perspective. Section IV presents our proposed thermal-aware mapping and control based on ILP formulation. Section V shows reports of experimental results.

[1]Intel Xeon®Processor E5 v3 Family Thermal Guide

II. Related Work

Several works have investigated thermal-aware workload allocation making use of DVFS mechanisms. Those approaches include: (i) on-line optimization policies [6], [7], [8], [9], based on predictive models and current temperatures directly read from embedded sensors; (ii) off-line allocation and scheduling approaches [10], [11] usually embedding a simplified thermal model of the target platform [8] or performing chip temperature assessment via a simulator [12].

State-of-the-art works in thermal management range from mobile to servers including supercomputer systems. Xie et. al. [13] show that smartphones are thermally bound and interestingly the thermal bound does not come from silicon limits but from user experience. Conficoni et al. [14] show that supercomputer cooling costs depends on several factors such as the total power consumption, ambient temperature and cooling control policy. Wang et. al. [15] show that fan power can account for up to 23% of typical server power and scales super-linearly with node utilization. Beneventi et. al. in [16] extract a predictive thermal model directly from the multicore device correlating power, performance and thermal sensors implemented in HW. They show that the thermal evolution of a multicore device can be modeled with a linear state-space representation.

Mutapcic et al. [11] formulate the problem of controlling the processor speed subject to environment thermal constraints as a convex optimization problem and they solve it with a specialized algorithm. However, their formulation cannot copes with the case in which the number of jobs to be assigned

is smaller than the number of cores, with some cores to remain idle and the case of heterogeneous jobs and HW resources.

Model predictive control combine thermal model, convex optimization and more rigorous optimal control theory to guarantee a reliable temperature capping in any working condition. Rudi et al. [17] propose an Integer Linear Programing (ILP) formulation which can combine frequency allocation and task allocation while ensuring a safe temperature bound. The solution proposed in [17] is capable of handling the case where not all the cores need to be active and "idleness opportunities" are allocated to the hottest cores. Unfortunately, [17] cannot handle job pinning and can potentially trigger a large set of migrations.

Few works show that when dealing with High Performance Computing (HPC) systems power management policies can take advantage of the peculiar workload model. Rountree et al. [18] create a framework (ADAGIO) for (i) discovering at run-time the communication percentage for each MPI, (ii) use this information to select a reduced frequency which minimizes the energy consumption. Eastep et al. [19] introduce (i) a set of APIs to be inserted in the application for finding at execution time the critical task and (ii) a framework for enforcing a power budget for the application while maximizing the performance of the critical task. While these approaches show how to combine power management with HPC workload (i.e. identifying at execution time the "relative importance" of each MPI tasks) they do not the case of thermal-bounded computing where each core performance is limited to maintain a safe-working temperature.

III. Workload and Thermal Modelling of HPC

Dynamic thermal management policies aim to reduce the cooling effort and power by adapting the processing element's performance to ensure a safe working temperature. In this section, we first introduce the nomenclature and the thermal properties of HPC nodes with direct measurements. Then, we extract from real scientific parallel workload a model linking the performance knob to the real performance of the final application.

We took as a target machine an HPC system based on an IBM NeXtScale cluster. Each node of the cluster is equipped with 2 Intel Haswell E5-2630 v3 CPUs, with 8 cores with 2.4 GHz clock speed and 85W Thermal Design Power (TDP, [1]). This supercomputer is ranked in the Top500 supercomputer list [20].

A. Thermal Model

We focus our attention on a single node of the cluster as the rack is composed by replication of the same node. To understand the thermal properties of a computing node, we have executed three main stress tests on which we have (i) kept the system in idle and measured the power and each core temperature after ten minutes, the (ii) we have executed a stressmark[2] in sequence on each core of each socket in the node, leaving idle the remaining ones. We maintained workload constant for ten minutes and measured power consumption and temperature. This test has been used to extract the maximum steady state temperature difference between cores. Finally (iii) we have simultaneously executed the stressmark for ten minutes in all the cores of the node and measured the temperature and the power consumption. In all the previous tests the temperature and power values are measured using an infrastructure similar to the one presented in [21], the Turbo mode was disabled to avoid power consumption to workload dependency. Results of our analysis are reported on table I.

[2]cpuburn stressmark by Robert Redelmeier: it is a single-threaded application which takes advantage of the superscalar architecture to load the CPU

TABLE I
Thermal Model

AVG temperature - Idle/Active cores	$15.93/33.39^oC$
Gradient - Idle/Active cores	$4.47/4.79^oC$
Gradient - Active core vs idle cores	8.05^oC
Stady-state time	$120sec$

As we will see in the experimental results section, we used the extracted characteristics to create a thermal model using a distributed RC approach [8], with an RC per core granularity tuned to have similar thermal characteristics.

B. Power Model

In addition to the previous tests, we have re-executed the stressmark in each core while scaling down the frequency for each core in all the available speed steps. We maintained each configuration for ten minutes and we measured the power consumed by each CPU. We collected these measurements in a set of Lookup-tables (LUTs), one for each node. We then used the LUTs to compute the power dissipated by each CPU in each Intel P-state. We measured a total power of 17.86 W when all cores in a computing node are idle. Power raises to 92.44 W when all the cores are active. We then extracted the power consumed by each core at each DVFS level: (2.51W @1.2GHz),...., (4,66W @2.4GHz) with an average standard deviation in between cores of 0.1Watts.

C. Workload Model

An HPC application can be seen as the composition of several tasks executed in a distributed environment, interconnected with a low-latency high-bandwidth network. HPC communications happen by sending explicit messages through a standard MPI programming model which takes advantage of the high-performance interconnect sub-system. Usually tasks are composed by computational intensive phases on independent data segments interrupted by synchronization points and communications. This characteristic impacts the sensitivity of the application to each core performance as computational imbalance can lead to longer synchronization phases.

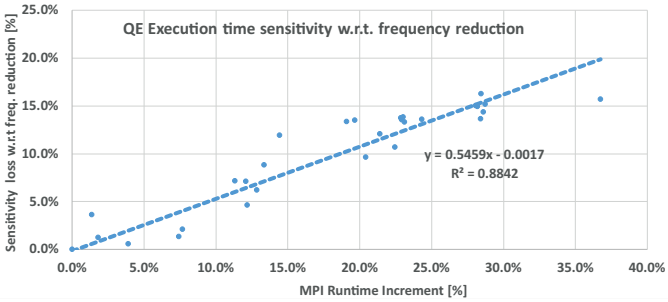

Fig. 1. Sensitivity loss w.r.t the reduction of frequency compared with the increment of the time spent into MPI runtime

As support to this statement, in this work we use as benchmark Quantum ESPRESSO (QE) [22], which is a real application widely used from the scientific community in high-end supercomputers. In the following experiment, we have explored how the different ratio of active code and MPI runtime for each QE task changes the impact of frequency scaling on the overall application execution time. We computed QE-CP on two computing nodes with 32 MPI tasks. We run QE-CP 32 times. At each run we configured sequentially one core of the 32 at minimum frequency while the other are maintained at the maximum. We compared it with the run in which all

the cores are at the nominal frequency. We then correlated the QE-CP slowdown and the MPI percentage of the slowed down task. Figure 1 shows that the impact of frequency reduction increases with percentage of MPI runtime present in each task. This result is inline with what was shown by the ADAGIO [18] and we can use it for extracting on-line the sensitive to frequency for each task. In this work, we take advantage of this information to assign priorities to the application tasks.

IV. HPC Optimal Thermal control

In this section, we present a DTM ILP formulation, namely the Optimal Thermal Controller (OTC), which matches all the requirements of HPC systems and proactive thermal control: (i) limiting the future temperature of all the cores below a critical threshold by selecting the frequency for each core; (ii) maximizing the application performance (frequency of all the cores); (iii) slowing down the core's frequency during communication phases; (iv) providing knobs to match the computation to communication ratio with the frequency selection.

The OTC operates on node level and it is composed of two main components: the thermal-aware task mapper and controller and an energy-aware MPI wrapper. The thermal-aware task mapper and controller (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 scheduling point (a) the TMC specifies the task to core mapping which will be maintained until the application completion. Clearly, if a critical task is mapped to a thermally inefficient core this will more likely cause a severe degradation of the final application performance. To capture this requirement, we use a per-task priority level. At scheduling point (b), the TMC selects the optimal frequency to be applied to the different cores for the following interval in order to maintain the future temperatures of all the cores below a maximum limiting value. Our OTC solution solves the scheduling points (a) and (b) with an ILP formulation and custom solver strategies respectively described in IV-A and IV-B.

The energy-aware MPI wrapper (EAW) is event-driven and acts as a bridge in between the MPI synchronization primitives and the core's frequency selection. This programming model interface is reactive and reduces the core's frequency when the MPI run-time is busy waiting.

When the execution flow returns to the application code, the frequency is restored to the one selected by the Thermal Controller. Figure 2 shows the main OTC components.

A. The First Step Problem - FSP

This optimization problem is solved during the initialization of the application. Its purpose is to allocate the application tasks 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 roles. Indeed if it is too short, the TMC cannot predict the impact of a task allocation on long term core's temperature as its effect is hidden by the thermal capacitance, making the problem trivial. On the contrary if the time horizon is too long the TMC cannot take advantage of the thermal capacitance for sustaining short time power burst.

In addition, not all tasks have the same priority. This is captured by the optimization model which maximizes the frequency of the highest priority task penalizing the frequencies of other ones in case a thermal limit is reached. The optimization model considers K tasks to be assigned to N cores where the number of tasks is lower or equal to the cores i.e., $K \leq N$. Each core can be configured with a frequency

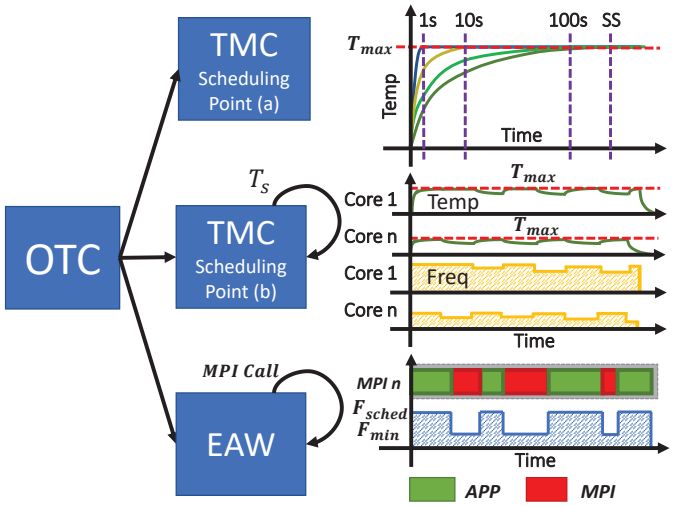

Fig. 2. This image shows the optimal thermal controller at node level

in a set of M level of frequencies. The Objective Function ($O.F.$) maximizes the sum of frequencies of all active cores γ_{jf} weighted by the priority δ_i of the task assigned on that core. To model the problem, we use two sets of binary decision variables:

$$x_{jf}^{i} = \begin{cases} 1 & \text{if core } j(j = 1, \ldots, N) \text{ works at frequency} \\ & f(f = 1, \ldots, M) \text{ executing job } i(i = 1, \ldots, K) \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$$y_j = \begin{cases} 1 & \text{if core } j(j = 1, \ldots, N) \text{ is idle,} \\ 0 & \text{otherwise, i.e., if it is working.} \end{cases} \tag{2}$$

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

$$O.F. = max \sum_{i=1}^{K} \sum_{f=1}^{M} \sum_{j=1}^{N} \delta_i \gamma_{jf} x_{jf}^{i} \tag{3a}$$

$$\sum_{j=1}^{N} \sum_{f=1}^{M} x_{jf}^{i} = 1 \tag{3b}$$

$$(i = 1, \ldots, K) \tag{3c}$$

$$\sum_{i=1}^{K} \sum_{f=1}^{M} x_{jf}^{i} + y_j = 1 \tag{3d}$$

$$(j = 1, \ldots, N) \tag{3e}$$

$$\sum_{j=1}^{N} GS_{jl} \left(\vec{p}_j y_j + \sum_{i=1}^{K} \sum_{f=1}^{M} p_{jf} x_{jf}^{i} \right) + T_l^0 + T^a \leq T_{MAX} \tag{3f}$$

$$(l = 1, \ldots, N) \tag{3g}$$

The constraint (3b) specifies that a task must be assigned only on a single core, which works at a given frequency. In addition, it specifies that all the N tasks must be assigned. Constraint (3d) is needed to determine the y decision variables which represent the idle cores. These variables are used in constraint (3f) in case there are less jobs than cores i.e., $K \leq M$n. Finally, constraints (3f) guarantee that the temperature of each core does not exceed T_{max} over the prediction

interval (PI_{FSP}). In the last constraint (3f), 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_0^l represents the dependency of the future temperature(@ PI_{FSP}) from the current core's temperature. These values can be derived from a state-space thermal model as described by [17]. T_a is the ambient temperature. When jobs 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.

B. The i-th Step Problem - ISP

After the tasks have been assigned to the cores in the FSP the TMC has to periodically solve, at a finer time scale, the assignment problem of frequencies to cores only. The ISP has the same objective function as FSP IV-A 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 (T) because the thermal constraints cannot be broken by an idle core. This reduces the overall complexity. As tasks have been already allocated in FPS in this model, tasks and core do not need separate variables, thus a priority is referred to a core.

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

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

$$O.F. = max \sum_{a \in A} \sum_{f=1}^{M} \delta_a \gamma_{af} x_{af} \tag{4a}$$

$$\sum_{f=1}^{M} x_{af} = 1 \tag{4b}$$

$$(\forall a \in A) \tag{4c}$$

$$\sum_{a \in A} \sum_{f=1}^{M} GS_{la} p_{af} x_{af} + \sum_{i \in I} GS_{li} \vec{p}_i + T_l^0 + T^a \leq T_{MAX} \tag{4d}$$

$$(\forall l \in A) \tag{4e}$$

The constraint (4b) bounds each core to a selected frequency. The constraint (4d) guarantees the thermal limits imposed on the model. Where the set $A = a_i$ contains the index of the active cores and the set $I = i_i$ contains the the index of idle cores directly defined from the solution of FSP. Where $A \cap I$ 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.

V. Experimental Results

In this section, we first describe an emulation framework we have created starting from the results of the characterization of computing nodes and real scientific workload conducted in Section III. We use this emulation framework to study the implication of the prediction interval/horizon in the thermal-aware task mapping and control of supercomputer nodes.

A. Emulation Framework

Our emulation framework is composed by the following components:

(i) The workload traces. The traces have been extracted using a tracing and profiling tool called Intel Trace Analyzer and Collector. The traces contain all the MPI activities (MPI call, data transfer, source/destination IDs) with a time instant. These have been extracted for the QE-CP running on a computing node.

(ii) The thermal simulator. We have created a first order discrete state-space model matched with the computing node as described in Section III-A. The model has a sample time of 10ms (Ts_{TM}), and as state variables has the temperature of each core of the node. Each core's power is computed with the power model presented in Section III-B. Workload traces which have higher temporal accuracy than the 10ms have been averaged among this period to produce the percentage of time in which each tasks was in the MPI runtime for each (Ts_{TM}) interval. We use this value to model the energy-aware MPI wrapper impact on core's power consumption.

(iii) The thermal-aware task mapping and control problem. The TMC optimization problem proposed in Section III has been solved using IBM Ilog CPLEX 12.6.1. The emulator call CPLEX each time there is a new TMC problem to be solved. This happens once at the application start (FSP) and periodically each ISP interval Ts_{ISP} which matches the prediction interval in the ISP problem (PI_{ISP}).

At each CPLEX call, the emulator builds a new instance of the problem with the new thermal parameters and the priority of the tasks and it waits for CPLEX results. During the waiting time the emulator is frozen, in this way the overhead time does not impact on the chronological MPI events. CPLEX has been executed on the same machine of the emulation framework which is a computing node, therefore the time overheads reflect real measurement. We use the same priority for all the tasks except for the root task, identified by the MPI rank 0. We noticed that this simple strategy to set priorities is very effective in our benchmark, as speeding-up the root core results in a performance gain for the entire application.

In our tests, we conducted the following experiments with different prediction intervals for both FSP and ISP problems. We considered PI_{FSP}=1s,10s,100s,SS and PI_{ISP}=1s,10s,100s,SS due the thermal propagation in our system is in the order of tens of seconds as we reported in table I. 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 task-to-core mapping, while SS-SS represents state-of-the-art static DTM solutions.

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

Figure 3 shows on the y-axis the temperature evolution of the coldest core (#0) for five cases. Namely no thermal control active, no thermal control active (NoTMC,NoEAW) but energy-aware MPI wrapper active (NoTMC,EAW), TMC active with (1s-1s), (SS-1s), (SS-SS). For the same configurations the figure 4 shows on the y-axis the temperature evolution of the hottest core. Clearly, according to the capability of the FSP problem, to predict the long term thermal evolution the higher priority (HP) task will be mapped on the coldest core. Indeed from figure 3, we can notice that if no TMC calls are executed, the coldest core executes a low priority task. When the FSP is empowered with a steady-state thermal predictor instead the TMC allocates the higher priority task on the coldest core and manages to run it always at the maximum frequency. Vertical spikes on the frequency allocated result from the energy-aware MPI wrapper which sets the minimum

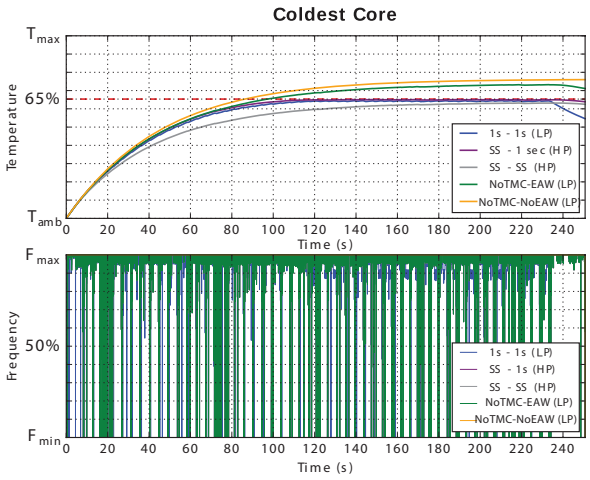

Fig. 3. This figure shows the coldest core of the system - core #0

frequency of the core during MPI synchronization calls. As a consequence of it, the maximum temperature reached by NoTMC-EAW is lower than NoTMC-NoEAW; showing its effectiveness in reducing the power consumption. Differently, short time horizons (1s-1s) in the FSP do not allow the solver to "see" the constraint and thus lead to a sub-optimal task mapping allocation. As a consequence, the high priority task need to be frequency limited to meet the thermal constraint as the thermal capacitance effect vanishes.

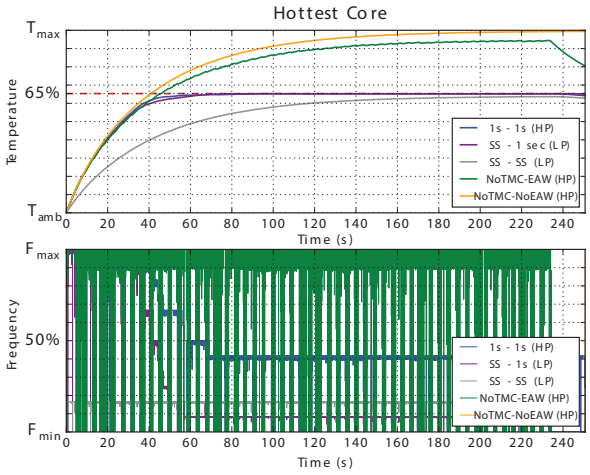

Fig. 4. This figure shows the hottest core of the system - core #14

B. Performance Gain

Figure 5 depicts the average frequency of the cores that host the highest priority tasks and the average frequency for all the cores in each configuration. Interestingly, in all the cases the highest priority task never reaches the maximum average frequency. This is the effect of the energy-aware MPI wrapper which reduces the core frequency during MPI calls.

The error bars show the variance for each configuration among different executions of the same QE-CP problem while moving the highest priority job from the MPI root task to another one. This means that if we shift the default position of highest priority job from 0 to 15 in the MPI rank all the configuration with predict interval in the FSP (PI_{FSP}) of 1 and 10 seconds we have a huge variation. This can be explained by the fact that in both experiments the 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 FSP problem is trivial and tasks are allocated on the first available core following a simple numerical binding where the job 0 will be allocate to the core 0 and so on. This binding is also the default on the Intel MPI runtime. In this particular case, if the highest priority task is lucky, it will be pinned on a "cold" core. Viceversa, if the highest priority task is unlucky, it will be mapped on a "hot" core. At the steady-state the frequency of the core 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 avoid the binding of the highest priority task on a "hot" core. In this case the highest priority job will be pinned on a "cold" core allowing the highest priority task to work at maximum frequency.

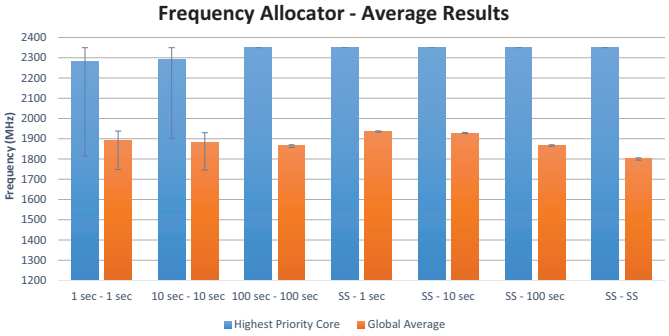

Fig. 5. Comparison between average core frequency and the frequency of the highest priority core using different configuration for the optimization problem.

We take as a baseline the SS-SS configuration, which model 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 priority 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 penalty for the high priority 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 pays off in the ISP as it allows to take advantage of the thermal capacitance. In the next section, we will add to this conclusion the solver overhead.

1) Overhead time: Figure 6 shows cumulative overhead for different configurations and quantify the induced performance loss as it sums up to the application execution time. The FSP bars represent the overhead time of the FSP problem solved only once at the application start, while the ISP bars are the sum of the overhead times of all iterations of the ISP solver.

For all the instances and the configurations, the solver is capable of finding the optimal solution. CPLEX allow to bound the solution time by the so called deterministic ticks, we use this approach to limit the solution time in case of harder problem. Authors of [17] show for a 60 core instance that the optimally gap always reduces below the 0.002% with a maximum number of 180 ticks.

We can see that for the 1s-1s and 10s-10s configuration the FSP solver time is negligible. After 1 second or 10 seconds the thermal transient has not reached the thermal constraint, for this reason the solution is trivial and consequently the solution

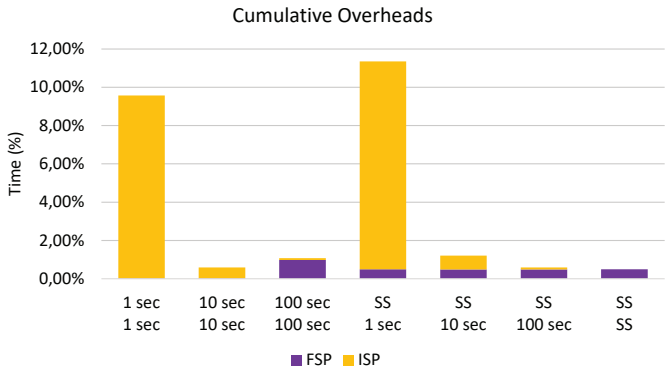

Fig. 6. Cumulative overhead induces by the optimization problem using different configuration for the optimization problem.

immediately converge. Instead, all the other configuration have an average overhead time of 0.59% of total execution time.

The total overhead time for the ISP significantly changes when we vary the PT_{ISP} and the Ts_{ISP}. Obviously, the ISP with a prediction interval of 1 second will be called hundred times more than a ISP with a prediction interval of 100 seconds. The results respect this trend, in particular for 1 seconds of prediction interval leads to an average penalty of 10.20% of total execution time, which makes this configuration worse than a static allocation (SS-SS) as cause of the solution overhead (7.46% of performance gain - 10.20% of overhead). Interesting the 10 seconds case (SS-10s) reduces the total penalty to the 0.64% which in conjunction to the 7.06% of performance gain w.r.t. the static-allocation lead to an overall performance gain of the 6%. At 100 seconds the total overhead penalty decreases to the 0.09%. However for this case the performance gain in only of the 3.46% making it less performing than the SS-10s case.

VI. Conclusion

In this paper, we presented a novel ILP formulation for the thermal-aware mapping and control of thermally constrained supercomputing nodes. Differently from state-of-the-art solutions, we focused our analysis on a real supercomputing system from which we modeled the real thermal characteristics as well as we extracted real scientific workload traces and we took into account the job pinning constraints induced by MPI runtime. We compared our solution with state-of-the-art dynamic thermal management which either dynamically control only the cores frequency or statically selects a core-to-job mapping and a specific frequency. Our results show that by selecting a long time horizon for the allocation of the MPI tasks on the different cores and short time horizons for the online DVFS selection our solution can lead up to 6% performance gain including overheads while ensuring that high priority tasks run always at the maximum frequency.

Acknowledgments

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

References

[1] P. Hammarlund, R. Kumar, R. B. Osborne, R. Rajwar, R. Singhal, R. D'Sa, R. Chappell, S. Kaushik, S. Chennupaty, S. Jourdan, *et al.*, "Haswell: The fourth-generation Intel core processor," *IEEE Micro*, no. 2, pp. 6–20, 2014.

[2] R. Ayoub, S. Sharifi, and T. S. Rosing, "Gentlecool: Cooling aware proactive workload scheduling in multi-machine systems," in *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 295–298, European Design and Automation Association, 2010.

[3] A. K. Coşkun, K. Whisnant, K. C. Gross, *et al.*, "Static and dynamic temperature-aware scheduling for multiprocessor SoCs," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 16, no. 9, pp. 1127–1140, 2008.

[4] H. Khdr, S. Pagani, M. Shafique, and J. Henkel, "Thermal constrained resource management for mixed ilp-tlp workloads in dark silicon chips," in *Proceedings of the 52nd Annual Design Automation Conference*, p. 179, ACM, 2015.

[5] H. Khdr, S. Pagani, E. Sousa, V. Lari, A. Pathania, F. Hannig, M. Shafique, J. Teich, and J. Henkel, "Power density-aware resource management for heterogeneous tiled multicores," *IEEE Transactions on Computers*, vol. 66, no. 3, pp. 488–501, 2017.

[6] A. K. Coskun, T. S. Rosing, and K. C. Gross, "Utilizing predictors for efficient thermal management in multiprocessor socs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 10, pp. 1503–1516, 2009.

[7] A. K. Coskun, T. S. Rosing, and K. Whisnant, "Temperature aware task scheduling in mpsocs," in *Proceedings of the conference on Design, automation and test in Europe*, pp. 1659–1664, EDA Consortium, 2007.

[8] A. Bartolini, M. Cacciari, A. Tilli, and L. Benini, "A distributed and self-calibrating model-predictive controller for energy and thermal management of high-performance multicores," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2011*, pp. 1–6, March 2011.

[9] F. Zanini, D. Atienza, L. Benini, and G. D. Micheli, "Thermal-aware system-level modeling and management for multi-processor systems-on-chip," in *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, pp. 2481–2484, May 2011.

[10] D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, "Temperature-aware distributed run-time optimization on mp-soc using game theory," in *Symposium on VLSI, 2008. ISVLSI'08. IEEE Computer Society Annual*, pp. 375–380, IEEE, 2008.

[11] S. Murali, A. Mutapcic, D. Atienza, R. Gupta, S. Boyd, and G. D. Micheli, "Temperature-aware processor frequency assignment for mp-socs using convex optimization," in *Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2007 5th IEEE/ACM/IFIP International Conference on*, pp. 111–116, Sept 2007.

[12] Y. Xie and W.-L. Hung, "Temperature-aware task allocation and scheduling for embedded multiprocessor systems-on-chip (mpsoc) design," *The Journal of VLSI Signal Processing*, vol. 45, no. 3, pp. 177–189, 2006.

[13] Q. Xie, M. J. Dousti, and M. Pedram, "Therminator: A thermal simulator for smartphones producing accurate chip and skin temperature maps," in *Low Power Electronics and Design (ISLPED), 2014 IEEE/ACM International Symposium on*, pp. 117–122, Aug 2014.

[14] C. Conficoni, A. Bartolini, A. Tilli, G. Tecchiolli, and L. Benini, "Energy-aware cooling for hot-water cooled supercomputers," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, DATE '15, (San Jose, CA, USA), pp. 1353–1358, EDA Consortium, 2015.

[15] Z. Wang, C. Bash, N. Tolia, M. Marwah, X. Zhu, and P. Ranganathan, "Optimal fan speed control for thermal management of servers," in *ASME 2009 InterPACK Conference collocated with the ASME 2009 Summer Heat Transfer Conference and the ASME 2009 3rd International Conference on Energy Sustainability*, pp. 709–719, American Society of Mechanical Engineers, 2009.

[16] F. Beneventi, A. Bartolini, A. Tilli, and L. Benini, "An effective gray-box identification procedure for multicore thermal modeling," *IEEE Transactions on Computers*, vol. 63, pp. 1097–1110, May 2014.

[17] A. Rudi, A. Bartolini, A. Lodi, and L. Benini, "Optimum: Thermal-aware task allocation for heterogeneous many-core devices," in *High Performance Computing Simulation (HPCS), 2014 International Conference on*, pp. 82–87, July 2014.

[18] B. Rountree, D. K. Lownenthal, B. R. De Supinski, M. Schulz, V. W. Freeh, and T. Bletsch, "Adagio: making dvs practical for complex hpc applications," in *Proceedings of the 23rd international conference on Supercomputing*, pp. 460–469, ACM, 2009.

[19] J. Eastep, S. Sylvester, C. Cantalupo, F. Ardanaz, B. Geltz, A. Al-Rawi, F. Keceli, and K. Livingston, "Global extensible open power manager: A vehicle for hpc community collaboration toward co-designed energy management solutions,"

[20] "Top500.org. top 500 supercomputer sites," 2017.

[21] A. Bartolini, M. Cacciari, C. Cavazzoni, G. Tecchiolli, and L. Benini, "Unveiling eurora - thermal and power characterization of the most energy-efficient supercomputer in the world," in *Proceedings of the Conference on Design, Automation & Test in Europe*, DATE '14, (3001 Leuven, Belgium, Belgium), pp. 277:1–277:6, European Design and Automation Association, 2014.

[22] P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo, *et al.*, "Quantum espresso: a modular and open-source software project for quantum simulations of materials," *Journal of physics: Condensed matter*, vol. 21, no. 39, p. 395502, 2009.