Sensor Based Real-Time Scheduling in Thermally Constrained Uniprocessor Systems

Rehan Ahmed*
Department of Electrical and Computer Engineering
University of Wisconsin Madison
Madison, Wisconsin 53706

Research Conducted Under Advisement of
Prof. Parmeshwaran Ramanathan† and Prof. Kewal K. Saluja‡

Fall 2010

*Email. rahmed3@wisc.edu
†Email. parmesh@ece.wisc.edu
‡Email. saluja@engr.wisc.edu

Abstract

Due to increasing power densities, thermal management is becoming an important issue in real-time systems. In particular, without thermal-aware scheduling, execution of tasks in a real-time application may increase the processor temperatures to levels at which the reliability of the underlying semiconductor devices is in jeopardy. In this work, I first present a model-based analysis of the thermal impact of a selected periodic task schedule. The proposed thermal analysis method is then extended to estimate the effect of executing a given aperiodic task at a specified time interval along with the periodic task schedule. Thermal sensors are periodically read to correct temperature estimation errors and account for process variations. One of the key features of the proposed method is that it can accurately account for the variations in power consumption during the execution of a task instance. The proposed method can be used at runtime to efficiently schedule an aperiodic task instance without violating its deadline constraint and the temperature constraints of the processor. The effectiveness of the proposed methods is compared with other approaches which do not consider temporal variation of task power consumption.

1 Introduction

Real-time applications have become pervasive with emergence of applications such as multimedia streaming, High Definition Television (HDTV), and interactive gaming in recent years. A task in a real-time application has deadline constraint by which they must complete its computation. Failure to meet the deadline constraint of a task may have severe or adverse consequences. For example, in aircraft control, real-time applications are expected to have stringent timing constraints. Failure to meet these constraints can result in system failure and possible loss of life.

In the past three decades, numerous algorithms have been developed for scheduling tasks in real-time application in such a way that their timing constraints are satisfied. A variety of models for both application tasks and their processing systems have been considered [1–5]. Specifically, real-time application task models considered in literature include periodic preemptive tasks, periodic non-preemptive tasks, aperiodic tasks, sporadic tasks, and/or tasks with precedence and resource constraints. Processing models considered in literature include single processor systems, multiple processor systems, multiple computer systems with significant inter-computer communication delays, and geographically diverse distributed systems.

However,there are certain critical issues which have not received as much attention. Specifically, with the scaling of Integrated Circuit (IC) technology, power and thermal constraints must be satisfied for the processing systems running the schedules of the real-time applications. The recent International Technology Roadmap for Semiconductors (ITRS) projects an increase in power density from about 0.57 W/mm^2 in 2007 to over 1.19 W/mm^2 by 2015. Higher power densities lead to increase in operating temperatures of the devices which may adversely affect device reliability and performance. For instance, it has been shown that a $10^o - 15^oC$ increase in device temperature can cause a factor of two reduction in device reliability [6]. Moreover, since electron mobility decreases with temperature, there is a corresponding increase in gate delays and thus a commensurate decrease in processor performance.

Several approaches have been proposed on thermal-aware task (or application) scheduling in general-purpose computing [7–9]. In most of these work, measurements from on-chip temperature sensors [10–12] are used to schedule tasks/applications under thermal constraint. However, the same scheduling scheme can not be adopted for real-time applications scheduling due to the presence of stringent timing constraints.

This work focuses on the thermal-aware scheduling problem of real-time non-preemptive periodic and aperiodic tasks. Thermal profile information of tasks is used to estimate the effect of execution of every task on the core temperature. Superposition along with an accurate cooling model is used to estimate core temperature. To ensure that temperature estimation errors do not accumulate over time, actual temperature values are read from on-chip sensors periodically. The schedule of periodic tasks is first determined statically using list scheduling heuristic. For that schedule, I derive a threshold parameter which can be used for admission control of aperiodic tasks. The computations for scheduling done on the fly are kept computationally simple so that their evaluation can be performed at runtime.

The report is organized as follows: Section 2 covers the related works followed by the system model and task model for real-time tasks in section 3. Next, I explain the temperature estimation scheme in section 4. Scheduling scheme of periodic tasks and the admission control for aperiodic tasks is covered in sections 5 and 6. I provide an illustrative example in section 7 in which I pick sample tasks and go through all the steps of the proposed estimation and scheduling approach. Section 8 compares the scheduling results of the proposed scheme with other works for various experiment setups.

2 Related Works

Extensive research has been done on scheduling of real-time applications: Starting with pioneering work by Liu and Layland [1] to the more recent but large body of work done by researchers such as Baruah [2, 13, 14], there have been numerous papers addressing different aspects of real-time scheduling. The mentioned work primarily focuses on satisfying timing constraints. However, as was mentioned earlier, a given schedule may not be feasible due to violation of thermal constraint even if the timing constraints are satisfied.

Recently, several approaches on task scheduling have been proposed to minimize the processor peak temperature. Dynamic Thermal Management (DTM) is widely used in these schemes. Brooks et al. [15] provided a comprehensive overview of thermal management schemes for high performance microprocessors. The techniques presented in their work include Dynamic Voltage/Frequency Scaling (DVFS) and reducing the activity of architectural units to reduce power consumption and consequently the resulting temperature. However, these hardware based DTM schemes have significant overhead. To counter this, there are several approaches [8], [16], [17] which tried to reduce temperature through task allocation at software level.

The problem becomes more complex for real-time systems since timing constraints for all tasks also have to be considered. In several works [18–23], Dynamic Voltage Scaling (DVS) is used to schedule tasks such that their timing and thermal constraints are satisfied. In these works, parameters such as response time and schedulable utilization are optimized while considering timing and thermal constraints. Thidapat et al. [24] used integer linear programming to minimize the maximum temperature for a given set of real-time tasks. They use an equivalent circuit model to estimate transient core temperatures and then an optimal static schedule is found based on the estimates. However, since their scheduling approach is entirely static, it cannot be used for scheduling aperiodic tasks since their arrival time is not known. Fisher et al. [25] use the thermal characteristics of processing cores to derive preferred speeds for processing elements in a homogeneous multicore system to minimize system temperature. The preferred speeds are then scaled to meet the timing requirements of sporadic tasks. In all of these approaches, Fourier model are used to estimate temperature and cooling of processing cores. The processor power function is assumed to be a cubic function of the processor speed.

A major limitation of all of these approaches is their inability to predict processor temperature accurately. Furthermore, these approaches do not account for temporal variations in the power consumption of a given real-time task. This proves to be a significant limitation since the power of each task must be conservatively assumed to be the maximum value. In reality, the power consumption of a task varies during its execution due to variations in activity. In addition, due to differences in manufacturing process parameter variations, the same task may have a different power consumption profile on two different processors. In this work, I account for all these variations in deciding whether to accept an aperiodic task.

3 System Model

The computer system considered in this work is a uniprocessor. The real-time application is comprised of non-preemptive periodic and aperiodic tasks. For simplicity of presentation this work assumes that the scheduler does not have flexibility to dynamically change the voltage and/or frequency. However, the presented approach can easily be extended to support a DVFS core capable of changing its frequency to discrete levels. This is because, the adopted model allows each task to have a separate power profile. Operating at a different frequency would simply imply a different power and thermal profile for the corresponding task.

Model of the Periodic Tasks. An application is comprised of a set Γ with n periodic tasks $\{\tau_1, \tau_2, \ldots, \tau_n\}$. Each periodic task τ_i is characterized by its worst-case non-preemptive computation time C_i, period T_i, and a hard deadline D_i. Let L denote the least common multiple of the periods T_1, T_2, ..., T_n. In the rest of this report, L will be referred to as the *hyperperiod*. In addition, each periodic task τ_i is also characterized by a power consumption profile, $p_i(u)$, $0 \leq u \leq C_i$, where $p_i(u)$ denotes the power consumption of τ_i in its computational interval $[u, u + \delta)$. Note that, most works in real-time computing literature assume that the power consumption of task is uniform throughout its execution. Power consumption is conservatively assumed to be the upper bound on the power consumed at any time during non-preemptive task execution. In contrast, in this work, the power profile can more accurately characterize the power consumption during the execution of the task instance. As a result, the scheduling results are expected to be less conservative and more accurate.

Since the power profile of each task is known a priori, one can also estimate the impact of executing its instance on the thermal profile of computer system. For instance, it has been shown that a thermal simulation tool, such as HotSpot, accurately estimates the thermal profile of a processor from the power profiles of the executing task instances [26]. Given a periodic task i, let $\eta_i(u)$, $0 \leq u \leq C_i$ denotes the temperature of the processor after u units of execution of one instance of τ_i when the ambient temperature is 0°C and the initial processor temperature, when the instance begins execution is also 0°C.

Model of the Aperiodic Tasks. As mentioned earlier, aperiodic tasks are activated in response to external events. As a result, their time of arrival is known only at runtime. Each aperiodic tasks A is characterized by its worst-case non-preemptive computation time C_A, a power consumption profile $p_A(u)$, $0 \leq u \leq C_A$, and a deadline D_A. As in the case of periodic tasks, $\eta_A(u)$, $0 \leq u \leq C_A$, denotes the temperature of the processor core after u units of execution of one instance of τ_i when the ambient temperature is 0°C and the initial processor temperature when the instance begins execution is also 0°C. Aperiodic tasks need deterministic runtime guarantees. That is, when the aperiodic task arrives, the scheduler must decide whether to admit the task. If admitted, the task's deadline must be guaranteed without violating the deadline constraints of the periodic tasks and temperature constraints of the processor.

4 Temperature Estimation Scheme

The framework used for estimating the processor temperature at a given time instant has three primary components. They are: 1) Superposition Principle 2) Cooling Model 3) Sensor Reading. These three components are discussed below.

4.1 Using the Superposition Principle

The proposed scheme relies on the well-known superposition principle [27] for linear systems, to reduce the computational overhead of the calculations to be performed in the scheme. Below, I briefly review the superposition principle as it applies to the problem at hand. Linear model of processors, RC model, is considered fairly accurate for the purpose of thermal modeling [26]. The linearity of the model allows us to exploit the superposition principle to rapidly compute the thermal impact of executing more than one task instance (at different times) on the processor. More specifically, consider the problem of computing the thermal impact of executing two task instances on a processor. Let $\xi_1(t)$, $\xi_2(t)$ denote the temperature of the processor at time t when only the first (second) instance is executed individually starting at time 0 with ambient and initial processor temperature at 0°C. Now, consider the temperature variation of the processor when both instances are executed; the first instance is executed starting at time s_1 and the second instance is executed

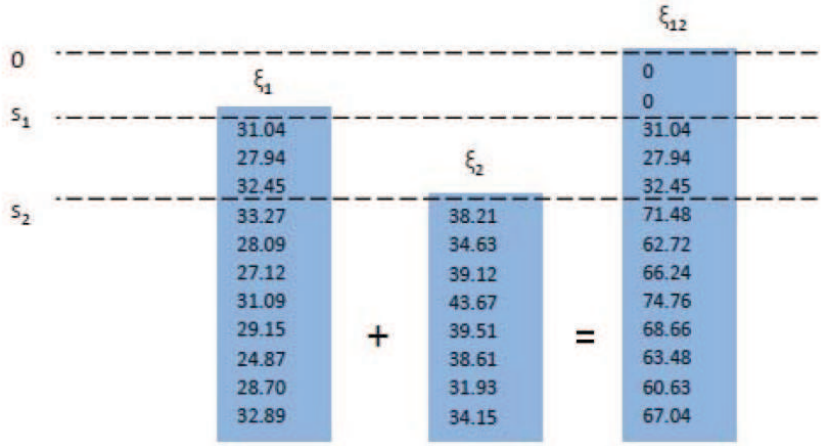

Figure 1: Superposition example

starting at time s_2. Then, from the superposition principle, the temperature of processor at time t is

$$\xi_{12}(t) = \begin{cases} 0 & 0 \leq t < s_1 \\ \xi_1(t - s_1) & s_1 \leq t < s_2 \\ \xi_1(t - s_1) + \xi_2(t - s_2) & s_2 \leq t. \end{cases} \quad (1)$$

when the ambient and initial processor temperature is 0°C. Figure 1 illustrates the use of superposition principle, in which $\xi_1(t)$ and $\xi_2(t)$ have been given factitious temperature values.

4.2 Cooling Model

This work assumes a simple first-order approximation of the cooling process. In this model, the rate of change in the processor temperature is proportional to the temperature difference between the processor and the ambient. Specifically, suppose that a processor has a temperature $\theta(t_0)$ at time t_0 and the ambient temperature is θ_a. Further, suppose that the processor is idle at all times $t > t_0$. Then, the temperature of the processor at the time $t > t_0$ is modeled as

$$\theta(t) = \theta(t_0) \cdot e^{-\beta(t-to)} + \theta_a(1 - e^{-\beta(t-to)}) \quad (2)$$

where β is a known cooling coefficient.

In practice, cooling does not follow such a simple first-order model. However, by choosing an appropriate value of β, one can conservatively bound the actual cooling function using the above first-order model.

4.3 Sensor Reading

On-chip sensors are an excellent resource of providing accurate temperature of the core. Therefore, they are used to periodically read actual temperature values of the processor. The sensors are read after every hyperperiod. Doing this helps correct any errors in the estimation scheme. Sensor readings are emulated in this work by running thermal simulations using HotSpot and reading simulated temperature values.

5 Thermal-aware scheduling of periodic tasks

Scheduling periodic real-time tasks with hard deadline constraints in the absence of power and thermal constraints is a well-studied problem. On a uniprocessor system, if all tasks are preemptive and there are no power and thermal constraints, then Earliest Deadline First (EDF) scheduling algorithm is optimal in the sense that: (i) if the total utilization of the task set is less than or equal to 1, then the EDF algorithm will find a schedule which meets all the hard deadline constraints, and (ii) there is no schedule which can meet all the deadline constraints if the total utilization of task set is greater than 1.

The scheduling problem is substantially more difficult if the tasks are non-preemptive. One of the earliest pioneering work on schedulability of non-preemptive, periodic hard real-time tasks on uniprocessor systems was done by Jeffay et al. [28]. The paper showed if the release of the first instance of each periodic task is specified, then problem of determining whether a given set of tasks is schedulable NP-hard. However, if the release time of the first instance is not specified, then there is pseudo-polynomial time algorithm to determine whether a task set is schedulable and the EDF algorithm can schedule every schedulable task set. The paper, however, did not consider power and/or thermal constraints.

Most works on scheduling periodic tasks using only power constraints make the following two assumptions: (i) power consumption of each instance of a task is constant (or conservatively, bounded by a given constant) throughout its execution, (ii) the power consumption bound is the same for all tasks, (iii) voltage and frequency of the processor can be scaled to achieve different processor speeds, and (iv) the power consumption bound is a cubic function of the processor speed and task computation time is inversely proportional to processor speed. In contrast, in this work, I do not assume (i) and (ii). Instead, as mentioned above, I assume that power consumption of a task instance varies during its execution and the power consumption profile of each task is known. This complicates the scheduling problem when thermal constraints are included. For clarity of presentation in explaining the solution to this more complicated problem, this work makes the following simplifying assumption. I assume that the scheduler does not have flexibility to dynamically change the voltage and/or frequency. Hence the processing speed is not under control of the scheduler. The processing speed is assumed to be known.

5.1 Theory

A schedule of a given set of periodic tasks, Γ, is a function $\Psi : R \rightarrow \Gamma \cup \{\phi\}$ such that if the processor is idle at time t, then $\Psi(t)$ equal to ϕ and otherwise it identifies the periodic task that being executed on the processor at time t. I assume that the schedule of a periodic tasks is constructed as successive repetitions of a schedule within the tasks' hyperperiod L, i.e, $\Psi(t) = \Psi(t \bmod L)$ for all t.

Definition 1 *A schedule Ψ of periodic tasks Γ is said to be* timing-feasible *if the instances of all periodic tasks are scheduled in such a way that they complete the execution prior to their respective deadlines.*

Given a schedule of periodic tasks Ψ, let $\eta_\Psi(t)$ denote the temperature of the processor at time t as result of executing the given schedule, when the ambient and initial temperatures are 0°C.

Definition 2 *A timing-feasible schedule Ψ of periodic tasks Γ is said to be* thermally-feasible *if the resulting maximum processor temperature is less than or equal to specified threshold Δ, i.e.,* $\max_{0 \leq t < \infty} \eta_\Psi(t) \leq \Delta$.

Theorem 1 *For some non-negative integer k, let $\Theta(kL)$ denote the temperature of the processor at time kL. Further suppose that only periodic tasks are executed in the time interval $[kL, (k+1)L)$ as per a schedule Ψ. Suppose that the ambient temperature is Θ_a. Then, the temperature of the processor at time $t \in [kL, (k+1)L)$ is*

$$\Theta(t) = \Theta(kL)e^{-\beta(t-kL)} + \eta_\Psi(t - kL) + \Theta_a(1 - e^{-\beta(t-kL)}).$$

Proof: Follows from the cooling model and the superposition principle. ■

Theorem 2 *For a given schedule Ψ of periodic tasks, let $\eta_\Psi(kL)$, $k = 0, 1, 2, 3, \ldots$, denote the processor temperatures at start of each hyperperiod when initial and ambient temperatures are $0°$. Let $\Theta(kL)$ denote the actual processor temperature and let Θ_a be the ambient temperature. Then,*

$$\lim_{k\to\infty} \Theta(kL) = \frac{\eta_\Psi(L)}{(1 - e^{-\beta L})} + \Theta_a.$$

Proof: From Theorem 1,

$$\begin{aligned}\Theta(kL) &= \Theta((k-1)L)e^{-\beta L} + \eta_\Psi(L) + \Theta_a(1 - e^{-\beta L}) \\ &= \Theta((k-2)L)e^{-\beta 2L} + \eta_\Psi(L)(1 + e^{-\beta L}) \\ &\quad + \Theta_a(1 - e^{-\beta 2L}).\end{aligned}$$

By further recursive substitution,

$$\begin{aligned}\Theta(kL) &= \Theta(L)e^{-\beta(k-1)L} + \eta_\Psi(L)(1 + e^{-\beta L} + \\ &\quad \cdots + e^{-\beta(k-2)L}) + \Theta_a(1 - e^{-\beta(k-1)L}).\end{aligned}$$

Since initial temperature is $\Theta(0)$, from Theorem 1,

$$\Theta(L) = \Theta(0)e^{-\beta L} + \eta_\Psi(L) + \Theta_a(1 - e^{-\beta L}).$$

Substituting for $\Theta(L)$ in the previous equation, we get

$$\begin{aligned}\Theta(kL) &= \Theta(0)e^{-\beta kL} + \eta_\Psi(L)(1 + e^{-\beta L} + \\ &\quad \cdots + e^{-\beta(k-1)L}) + \Theta_a(1 - e^{-\beta kL}).\end{aligned}$$

Taking the limit $k \to \infty$, the theorem follows. ■

Using proofs similar to that of Theorem 2, one can also prove the following two theorems.

Theorem 3 *If the processor temperature at time 0, i.e, $\Theta(0)$, is less than or equal to $\frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a$, then $\Theta(kL) \leq \Theta((k+1)L) \leq \frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a$ for all non-negative integers k.*

Theorem 4 *If the processor temperature at time 0, i.e, $\Theta(0)$, is greater than $\frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a$, then $\Theta(kL) \geq \Theta((k+1)L) \geq \frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a$ for all non-negative integers k.*

Theorem 5 *For a given schedule Ψ, define*

$$\begin{aligned} \Phi_\Psi(\Theta) &= \max_{0 \le u \le L} \left\{ \Theta e^{-\beta u} + \eta_{Psi}(u) + \Theta_a(1 - e^{-\beta u}) \right\}, \\ & \textit{and} \\ \Theta^*_\Psi &= \max \left\{ \Theta : \Phi_\Psi(\Theta) \le \Delta \right\}. \end{aligned} \tag{3}$$

Also let $\Theta(0)$ be the processor time at time 0. Then, schedule Ψ is thermally-feasible only if

$$\begin{aligned} \Theta(0) &\le \Theta^*_\Psi. \quad \textit{and} \\ \frac{\eta_\Psi(L)}{(1 - e^{-\beta L})} + \Theta_a &\le \Theta^*_\Psi \end{aligned}$$

Proof: Let $\Theta(0)$ denote the processor temperature at time 0. First consider the case when $\Theta(0) \le \frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a$. From Theorem 3, we know that $\Theta(kL) \le \frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a \le \Theta^*_\Psi$. Hence, from the hypothesis of the theorem, we know that the processor temperature will never exceed Δ in the interval $[kL, (k+1)L)$. Since this result holds for any non-negative k, the resulting schedule is thermally-feasible in this case.

Now consider the complementary case when, $\Theta^*_\Psi \ge \Theta(0) \ge \frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a$. From Theorem 4, we know that $\Theta^*_\Psi \ge \Theta(kL) \ge \frac{\eta_\Psi(L)}{(1-e^{-\beta L})} + \Theta_a$. Hence, from the hypothesis of the theorem, we know that the processor temperature will never exceed Δ in the interval $[kL, (k+1)L)$. Since this result for any non-negative k, the resulting schedule is thermally-feasible in the case.

Hence, the theorem. ■

5.2 Algorithm

Above observations lead us to the following algorithm for scheduling periodic tasks without violating thermal constraints. Let τ_{ij} denote the j^{th} instance of periodic task τ_i. That is, τ_{ij} is activated at time $j \cdot T_i$, has a computation time of C_i with a hard deadline of $(j \cdot T_i + D_i)$. Let $J = \{\tau_{ij} : 1 \le i \le n, 0 \le j \le L/T_i\}$ denote the set of all task instances in a hyperperiod. The algorithm works as follows:

Algorithm 1: Thermal-aware Scheduling of Periodic Tasks

---

1. Create an ordered list of all task instances in J in ascending order with respect to their deadlines.
2. Schedule one task instance at a time in the above list order.
3. Assume the first l task instances have already been scheduled and we are in the process of scheduling the $l+1$ instance on the list. Suppose the $l+1$ instance on the list is τ_{ij}. Then, schedule τ_{ij} from the time $[s, s + C_i)$ if and only if s is the smallest time satisfying the following three conditions:
   - $jT_i \le s$ and $s + C_i \le (jT_i + D_i)$,
   - None of the first l tasks instances in the list are previously scheduled to execute at any time in the interval $[s, s + C_i)$,
   - Executing τ_{ij} in this interval along with the previously scheduled l task instances will not violate the thermal constraints of the processor.
4. Once all task instances in J are scheduled, check whether all conditions in Theorem 5 are satisfied. If yes, then the resulting periodic schedule is both timing-feasible and thermally-feasible. Otherwise, the task set is considered to be infeasible.

Using above algorithm, the thermal profile for the entire hyperperiod, comprising only of periodic tasks, is computed and stored off-line, prior to the start of the system. This is done to reduce the runtime computational complexity of the proposed scheduling approach. This phase of the scheduling algorithm has no runtime complexity.

6 Thermal-aware scheduling of aperiodic tasks

The proposed approach shall now be extended for determining whether a given aperiodic task can be scheduled to execute in specified time interval without violating its deadline and the temperature constraints of the processor; in the presence of periodic tasks.

There are two primary conditions for executing aperiodic tasks. 1) Execution of an aperiodic task must finish before its deadline. 2) Aperiodic task must not cause thermal constraint to be violated during and after its execution. Note that, once an aperiodic task instance executes, the processor temperature at any future time instant is always higher than the corresponding temperature had the aperiodic task not executed. As a result, the challenge for the scheduler is to ensure that the temperature increases due to an aperiodic task execution will not prevent the timely execution of any future periodic task instances due to thermal violations.

Due to thermal sensor readings, the processor temperature is assumed to be known accurately at the start of every hyperperiod. For all other time instances, the proposed approach uses the superposition principle and cooling model to estimate the processor temperature.

6.1 Theory

Suppose a new aperiodic task A arrives at time $t \in [kL, (K+1)L)$ and the scheduler is evaluating whether the task can be executed in the interval $[S, S+C_A)$ without violating the thermal constraints. Assume that $S \in [lL, (l+1)L)$ for some $l \geq k$. To determine whether thermal constraints will be violated, we need to estimate the processor temperature at any time $u \geq S$. Let $\Theta_{\text{meas}}(kL)$ be the processor temperature measured using the thermal sensors at time kL. For any time $u \geq S$, let $V(u)$ be the set of aperiodic tasks which have already completed or scheduled to complete in the interval $[kT, u-L)$. Similarly, for any time $u \geq S$, define $W(u)$ to be the set of aperiodic tasks which are scheduled to complete in the interval $[u-L, u)$. For simplicity, assume that the new task A is included in $W(u)$. For each aperiodic task, $\tau \in E(u) \bigcup W(u)$, let s_τ denote the scheduled start time of τ. Let m be the smallest integer such that $s_\tau \leq mL$ for all $\tau \in E(u) \bigcup W(u)$.

Using superposition principle, one can show that the processor temperature at time u is

$$\begin{aligned} &\Theta_{\text{meas}}(kL)e^{-\beta(u-kL)} + \theta_a(1 - e^{-\beta(u-kL)}) + \eta_\Psi(u - lL) \\ &+ \sum_{j=k}^{l-1} \eta_\Psi(L)e^{-\beta(u-jL)} + \sum_{\tau \in E(u)} \eta_\tau(L)e^{-\beta(u-s_\tau-L)} \\ &+ \sum_{\tau \in W(u)} \eta_\tau(u - s_\tau) \end{aligned} \tag{4}$$

Theorem 6 *There will be no thermal violation if the new aperiodic task is scheduled for execution in the interval* $[S, S + C_A)$ *if*

$$\Delta \geq \max_{s \leq u < mL} \left\{ \begin{array}{l} \Theta_{\text{meas}}(kL)e^{-\beta(u-kL)} + \theta_a(1 - e^{-\beta(u-kL)}) \\ + \sum_{j=k}^{l-1} \eta_\Psi(L)e^{-\beta(u-jL)} + \sum_{\tau \in W(u)} \eta_\tau(u - s_\tau) \\ + \sum_{\tau \in E(u)} \eta_\tau(L)e^{-\beta(u-s_\tau-L)} + \eta_\Psi(u - lL) \end{array} \right\} \tag{5}$$

and

$$\begin{aligned} \Theta^*_\Psi \geq & \Theta_{\text{meas}}(kL)e^{-\beta(mL-kL)} + \theta_a(1 - e^{-\beta(mL-kL)}) \\ & + \sum_{j=k}^{l-1} \eta_\Psi(L)e^{-\beta(mL-jL)} + \sum_{\tau \in W(u)} \eta_\tau(mL - s_\tau) \\ & + \sum_{\tau \in E(u)} \eta_\tau(L)e^{-\beta(mL-s_\tau-L)} + \eta_\Psi(mL - lL) \end{aligned} \tag{6}$$

Proof: Note that, by definition, no aperiodic task is scheduled to executed after time mL. Since Equation 5 is satisfied, there will not be a thermal violation until time mL. Since Equation 6 is satisfied, from Theorem 5, we can conclude that, future executions of the periodic schedules, will not result in a thermal violation at any time beyond mL. Hence the theorem.

6.2 Algorithm

The above observations lead us to the following algorithm for making admission control decisions for aperiodic tasks. Let A be an aperiodic task arriving at t with computation time C_A, deadline D_A. The algorithm for making admission control decision works as follows.

Algorithm 2: Thermal-aware Admission Control of Aperiodic Task

1. Generate a list of all possible start times S for task A in the interval $[t, D_A - C_A)$.
2. Find the smallest value of S such that conditions in equation 5 and 6 are satisfied.
3. If such a value of S exists, A is scheduled for execution at time S
4. Task A is rejected if no such value is found.

Note that the steps conducted to make admission control decisions are done at runtime. This is because the arrival time of aperiodic tasks is not known a priori. Specifically, the runtime computations are simple and include 1) Calculation of an exponentials for generation of cooling profile 2) Superposition of aperiodic task thermal profile with hyperperiod and cooling thermal profiles.

7 Illustrative Examples

This section covers an illustrative example of the proposed scheduling and estimation methodology. I consider two periodic tasks and construct the hyperperiod thermal profile based on their individual thermal profiles and scheduling algorithm presented in section 5. The periodic tasks are defined by the following parameters:

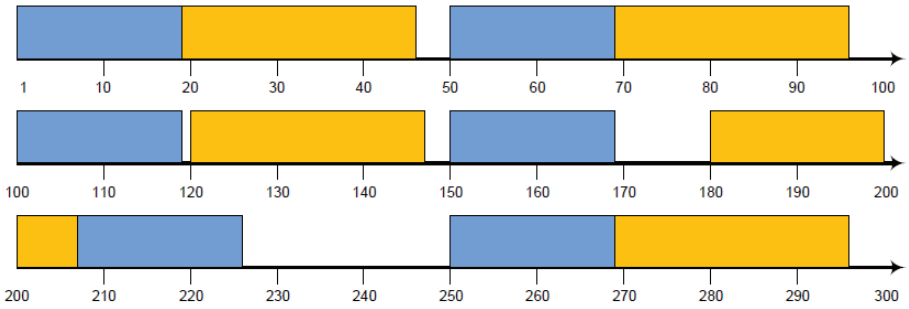

Figure 2: Periodic task schedule

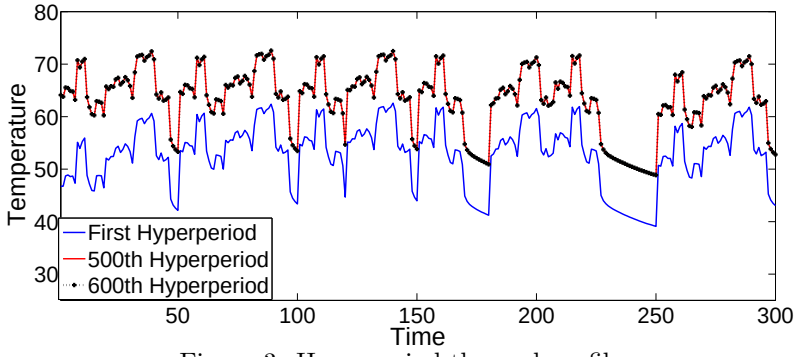

Figure 3: Hyperperiod thermal profiles

$\tau_1 = (C_1 = 19, D_1 = 50)\ \forall u = 0 \rightarrow C_1 : p_1 = 50 \rightarrow 200$
$\tau_2 = (C_2 = 27, D_2 = 60)\ \forall u = 0 \rightarrow C_2 : p_2 = 50 \rightarrow 200$
For this set of periodic tasks, the hyperperiod length (L)= 300 units of execution. Figure 2 shows the timing schedule for the entire hyperperiod. As shown in the figure, the hyperperiod has a total of seven slack intervals which are 46→50, 96→100, 119→120, 147→150, 169→180, 227→250 and 296→300. The periodic task set considered has a utilization of 83%.

Figure 3 shows the thermal profiles for 1^{st}, 500^{th} and 600^{th} hyperperiods at $\Theta_a = 32°C$ and illustrates how the temperature has increased due to execution in multiple hyperperiods. The difference between the thermal profile of 500^{th} and 600^{th} hyperperiods is minimal; indicating that the system has reached steady state. We also note that:
$\lim_{k\to\infty} \Theta(kL) = 52.68\ \Theta_a = 35\ \eta_\psi(L) = 8.11$

Using these values, we can use theorem 1 to calculate the value of β such that there is no steady state error. This value of β is 2.046×10^{-3}. Based on the periodic task thermal profile we also try to find the value of Θ^*_ψ. This value is used to determine if an aperiodic task will be accepted or rejected for execution (admission control) at a given time. Through experiments, I deduced that for $\Delta = 100°$, $\Theta^*_\psi = 72°C$.

I shall now try to illustrate how admission control decisions for aperiodic tasks are made. Assume that the ambient temperature is 35°C. Further assume that time 0 is a start of a hyperperiod and the processor has been executing the periodic tasks for a long time before time 0. Hence, at time 0, $\Theta(0) = 52.68$, i.e., $\Theta(0) = lim_{k\to\infty}\Theta(kL)$.

1. At time 160, an aperiodic task τ_{A1} arrives and has a computation time of 8 and deadline of 250. The thermal profile of τ_{A1} when ambient and initial temperatures are zero is shown in Figure

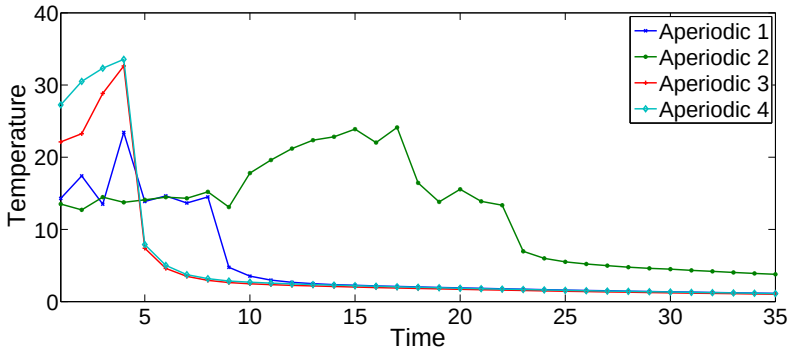

Figure 4: Thermal profiles: Aperiodic tasks

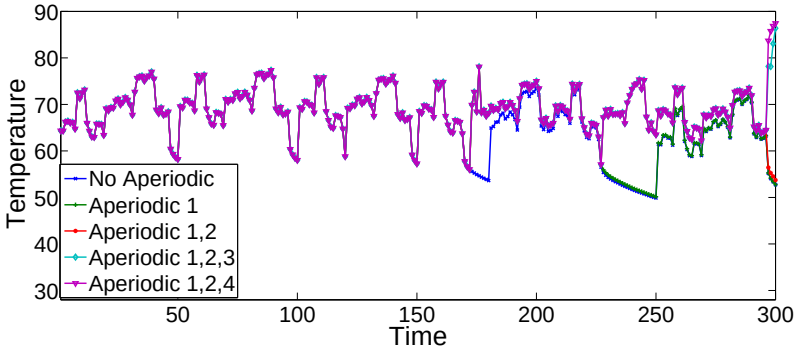

Figure 5: Thermal profile: Aperiodic task thermal feasibility check

4. In the periodic schedule, the first possible time slot when τ_{A1} can be scheduled is the interval $[169, 177)$. Using Equation (4), we can compute the resulting thermal profile of the processor from time 169 to the end of the hyperperiod at time 300. This thermal profile is shown as a curve in Figure 5. Similarly, from the right hand side of Equation 5, we can verify that the maximum processor temperature in $[169, 300)$ if τ_{A1} is executed in $[169, 176)$ is 78.1°C. Clearly, this maximum temperature is less than Δ and Equation 5 is satisfied. Similarly, the calculation in the right hand side of Equation 6, results in 52.77°C, which is less than $\Theta* = 72$. Hence, Equation 6 is satisfied. Therefore, the scheduler will accept τ_{A1} and schedule it for execution in the interval $[169, 177)$.

2. At time 226, an aperiodic task τ_{A2} arrives and has a computation time of 23 and a deadline of 320. The thermal profile of τ_{A2} when ambient and initial temperatures are zero is shown in Figure 4. In the schedule including τ_{A1}, the first possible time slot when τ_{A2} can be scheduled is the interval $[226, 249)$. Using Equation (4), we can compute the resulting thermal profile of the processor from time 226 to the end of the hyperperiod at time 300. This thermal profile is shown as a curve in Figure 6. Similarly, from the right hand side of Equation 5, we can verify that 75.35°C is the maximum processor temperature in $[226, 300)$ if τ_{A1} is executed in $[169, 177)$ and τ_{A2} is executed in $[226, 300)$. Clearly, this maximum temperature is less than Δ and Equation 5 is satisfied. Similarly, the calculation in the right hand side of Equation 6, results in 53.73°C, which is less than $\Theta* = 72°C$. Hence, Equation 6 is satisfied. Therefore, the scheduler will accept τ_{A2} and schedule it for execution in the interval $[226, 249)$.

3. At time 290, an aperiodic task τ_{A3} arrives and has a computation time of 4 and a deadline of 320. The thermal profile of τ_{A3} when ambient and initial temperatures are zero is shown in Figure 5. In the schedule including τ_{A1} and τ_{A2}, the first possible time slot when τ_{A3} can be scheduled is the interval $[296, 299)$. Using Equation (4), we can compute the resulting thermal profile of the

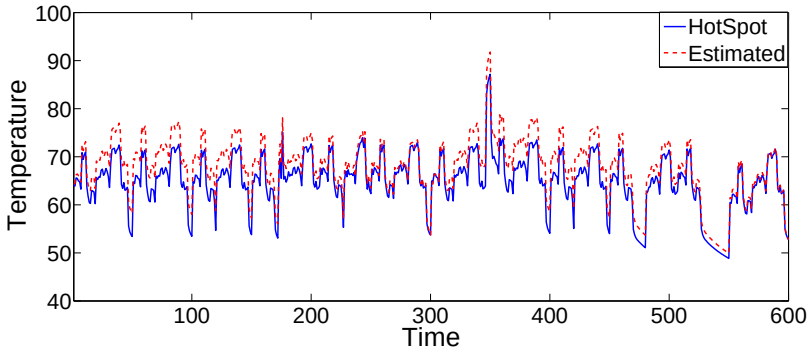

Figure 6: Thermal profile. All aperiodic tasks executed

processor from time 226 to the end of the hyperperiod at time 300. This thermal profile is shown as a curve in Figure 6. Similarly, from the right hand side of Equation 5, we can verify that is 86.37°C is the maximum processor temperature in $[296, 300)$ if τ_{A1} is executed in $[169, 177)$, τ_{A2} is executed in $[226, 249)$ and τ_{A3} is executed in $[297, 299)$. Clearly, this maximum temperature is less than Δ and Equation 5 is satisfied. Similarly, the calculation in the right hand side of Equation 6, results in 86.37°C, which unfortunately is greater than $\Theta^*_\psi = 72°C$. Hence, Equation 6 is not satisfied. Therefore, the scheduler cannot schedule τ_{A3} in the interval $[296, 300)$. The next possible time interval at which τ_{A3} can be executed is $[346, 350)$. However, the this slack interval is after τ_{A3}'s deadline of 320. Since both conditions in theorem 5 can not be satisfied for any start time before D_{A3}, τ_{A3} is rejected.

4. At time 296, an aperiodic task τ_{A4} arrives and has a computation time of C4 and a deadline of 400. The thermal profile of τ_{A4} when ambient and initial temperatures are zero is shown in Figure 4. In the schedule including τ_{A1} τ_{A2}, the first possible time slot when τ_{A4} can be scheduled is the interval $[296, 299)$. Using Equation (4), we can compute the resulting thermal profile of the processor from time 296 to the end of the hyperperiod at time 300. This thermal profile is shown as a curve in Figure 5. Similarly, from the right hand side of Equation 5, we can verify that is 87.42°C is the maximum processor temperature in $[296, 300)$ if τ_{A1} is executed in $[169, 177)$, τ_{A2} is executed in $[226, 249)$ and τ_{A3} is executed in $[296, 300)$. Clearly, this maximum temperature is less than Δ and Equation 5 is satisfied. Similarly, the calculation in the right hand side of Equation 6, results in 87.42°C, which is greater than $\Theta^*_\Psi = 72°C$. Hence, Equation 6 is not satisfied. Therefore, the scheduler cannot schedule τ_{A4} in the interval $[296, 300)$. The next possible time interval at which τ_{A4} can be executed is $[346, 350)$. This thermal profile is given in Figure 6. We can verify that 91.77°C is the maximum processor temperature in $[346, 600)$ if τ_{A1} is executed in $[169, 177)$, τ_{A2} is executed in $[226, 300)$ and τ_{A4} is executed in $[346, 350)$. Clearly, this maximum temperature is less than Δ and Equation 5 is satisfied. Similarly, the calculation in the right hand side of Equation 6, results in the temperature of 52.74°C at 600, which is less than $\Theta^*_\Psi = 72°C$. Hence, Equation 6 is satisfied. Therefore, the scheduler will accept τ_{A4} and schedule it for execution in the interval $[346, 350)$.

Figure 6 also illustrates the accuracy of my temperature estimation scheme by displaying the temperature from HotSpot simulator. For this thermal profile, the proposed estimation scheme is conservative in estimating temperature with a maximum error of less than 6°C.

8 Results and Conclusion

In this section, I demonstrate through a series of experiment that the proposed scheduling and estimation scheme can give us significant gains in scheduling of aperiodic tasks. First, I shall char-

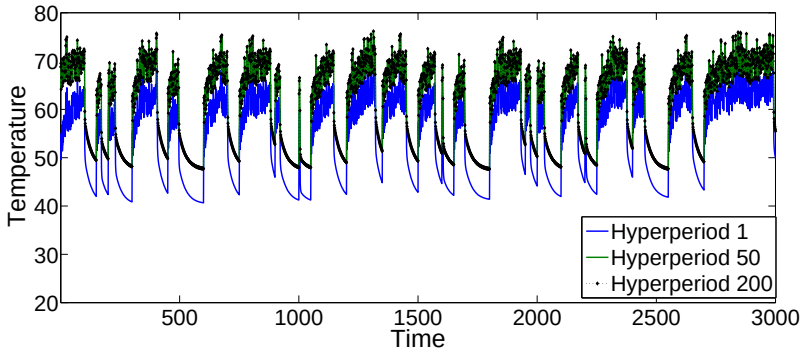

Figure 7: Thermal Profiles: Hyperperiods

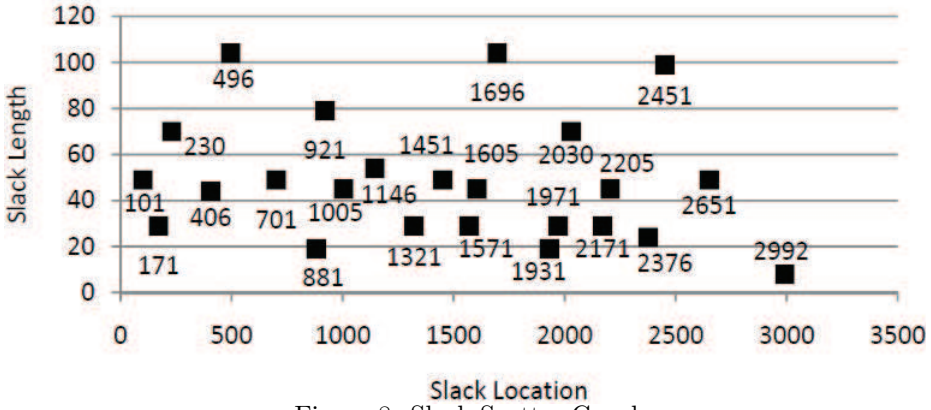

Figure 8: Slack Scatter Graph

acterize a periodic task-set and aperiodic tasks considered in terms of their computational times, deadlines, power and thermal profiles.

Periodic Task Set The periodic task set used in this study comprises of nine tasks. The power consumption of each periodic task follows a three state Markov model, with power consumption varying between 60W and 150W. It is assumed that the power profiles generated are the worst case power profiles. In actual execution, a given periodic task instance may consume less power. Hyperperiod length for the task set 3000 and the utilization of periodic task set is 61%. Figure 7 shows the thermal profile during the 1^{st}, 50^{th} and 200^{th} hyperperiod. The figure shows that the periodic task set has reached steady state at the 50^{th} hyperperiod since the thermal profiles for 50^{th} and 200^{th} hyperperiods overlap completely. In this example, the scheduling for periodic tasks was done at an ambient temperature of 40°C.

From the thermal profile in figure 7 we notice that the temperature at the end of the first hyperperiod is 49.82°C. whereas the steady state temperature for this task set at the end of every hyperperiod beyond the 50^{th} hyperperiod is 55.2°C. These values can be used to evaluate the cooling coefficient β used by the estimation scheme as follows.
$\lim_{k\to\infty} \Theta(kL) = 55.2$
$\Theta_a = 40^\circ$
$\eta_\psi(L) = 49.82 - 40 = 9.82^\circ$
theorem 1 is used to evaluate value of $\beta = 3.32 \times 10^{-4}$ The size and location of all slack intervals is illustrated in figure 8 using a scatter graph. Also for making scheduling decisions, $\Delta = 100^\circ$, and Θ^*_ψ is conservatively set to $65^\circ C$

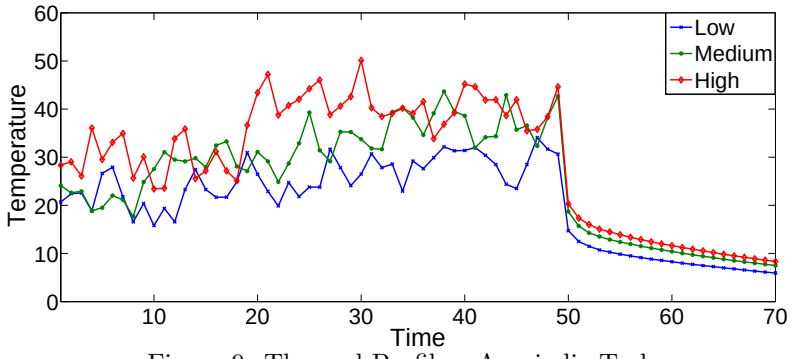

Figure 9: Thermal Profiles: Aperiodic Tasks

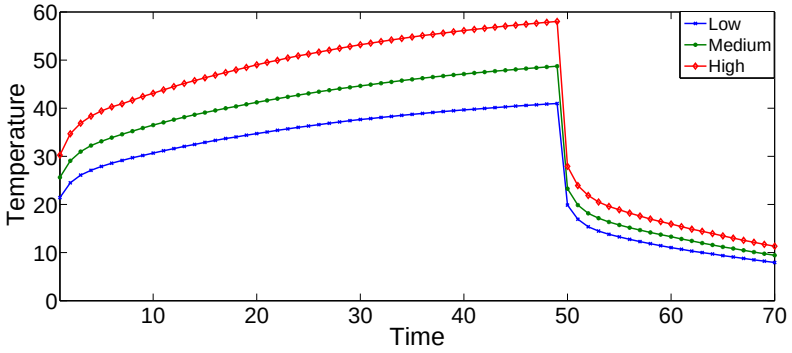

Figure 10: Thermal Profiles: Aperiodic Tasks-Constant Power

Aperiodic Tasks I have considered multiple instances of three aperiodic tasks to perform evaluation. They can be characterized as low, medium and high load tasks. Their power consumption ranges are given below.

$$\begin{array}{lll} \tau_{Low} & : & C = 49 \quad \forall u = 0 \rightarrow C : p(u) = 80 \rightarrow 200 \\ \tau_{Medium} & : & C = 49 \quad \forall u = 0 \rightarrow C : p(u) = 100 \rightarrow 250 \\ \tau_{High} & : & C = 49 \quad \forall u = 0 \rightarrow C : p(u) = 120 \rightarrow 300 \end{array}$$

As was the case with periodic tasks, the power profile of all aperiodic tasks follow a three state Markov model. This again is assumed to be the worst case power consumption. The thermal profiles for all three aperiodic tasks are given in figure 9. As shown in the figure, Low activity aperiodic task reaches a maximum temperature of 34.06°C starting at $\Theta_a = 0$°C. Similarly, medium and high activity aperiodic tasks have a maximum temperature of 43.67°C and 50.12°C respectively.

Experiment 1 This experiment compares the proposed scheduling and estimation approach with approaches that do not allow power consumption of tasks to have temporal variations. This experiment consists of three sets of tasks. In the first set, all aperiodic tasks executed are low activity tasks (τ_{Low}). The arrival times of aperiodic task instances follow a Poisson distribution with mean task inter arrival time of 750 (average of 4 aperiodic task instances in a hyperperiod). Each instance has an associated deadline. Deadlines follow an exponential distribution with mean of 3000 (1 Hyperperiod). Simulations are conducted with the following setups:

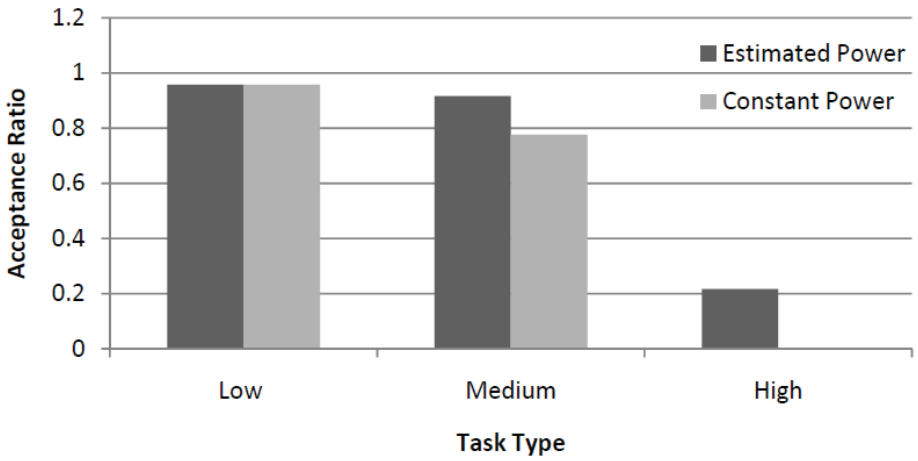

Figure 11: Exp 1:Task Acceptance Ratio. Estimated Power vs Constant Power

1. *Estimated Power*: Periodic and aperiodic tasks consume their worst case estimated power. The proposed estimation scheme uses the thermal profiles of all tasks and cooling model to determine temperature. After every hyperperiod, the simulated temperature values are read from HotSpot and the errors in the proposed estimation scheme are corrected. The simulated temperature values from HotSpot are used to emulate the reading of circuit sensors to correct errors.

2. *Constant Power*: Periodic tasks consume their worst case estimated power. However, the power consumption of all aperiodic task instances is set to a constant value equal to the 95^{th} percentile of its corresponding worst case power profile. The thermal profiles for constant 95^{th} percentile power profiles of all three task types are also generated and stored off-line. These thermal profiles are shown in figure 10 for $\Theta_a = 0°C$. My estimation scheme considers worst case thermal profile for all periodic tasks and thermal profiles corresponding to constant power profiles for all aperiodic task instances. HotSpot considers worst case power profiles for both periodic and aperiodic tasks and simulated values are read after every hyperperiod to correct estimation errors.

This gives us the first set of results in which all aperiodic task instances are low activity tasks. For the second and third set of results I repeat this experiment with all aperiodic task instances assigned to medium activity (τ_{Medium}) and high activity (τ_{High})tasks. Figure 11 shows how the acceptance ratio of each task type varies for *estimated power* and *constant power* setups. The results indicate that the acceptance ratio of *estimated power* is consistently higher compared to the *constant power* scheme. For τ_{Low}, both *estimated power* and *constant power* schemes have overlapping curves and show high acceptance rates. This is because the temperature profiles corresponding to τ_{Low}'s worst case and constant power profiles do not show high maximum temperatures. In case of medium load tasks, there is a significant difference in task acceptance ratios and the proposed estimation approach provides significant gains. The *constant power* scheme does not allow acceptance of any high activity tasks due to high temperature. *Estimated Power* approach, on the other hand, allows nearly 21% of high activity aperiodic tasks to be accepted.

Experiment 2 In this experiment, I use the same arrival times and deadlines for aperiodic task instances as in experiment 1. However, each arrival time/deadline pair is assigned randomly to one of the three aperiodic task types (τ_{Low},τ_{Medium},τ_{High}). The assignment of all three task types is equally

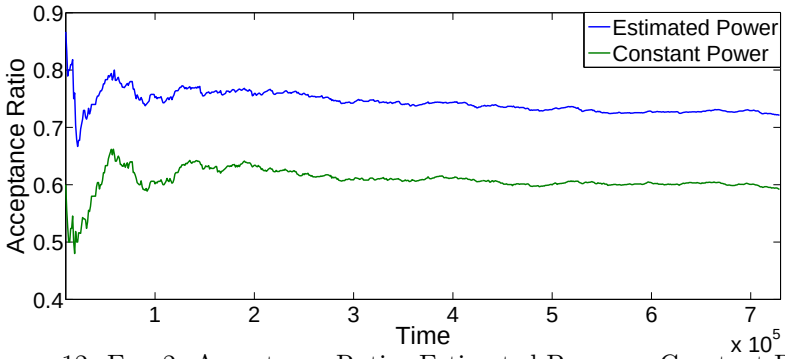

Figure 12: Exp 2: Acceptance Ratio. Estimated Power vs Constant Power

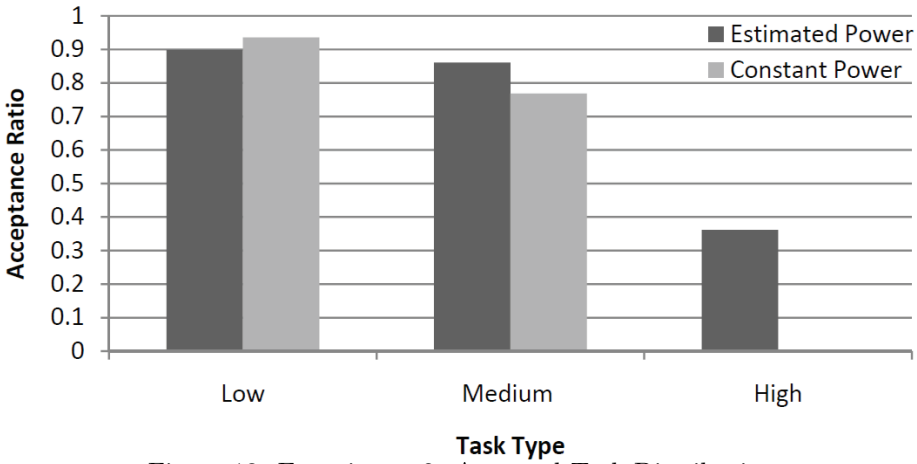

Figure 13: Experiment 2: Accepted Task Distribution

likely. This experiment is designed to illustrate the operation and comparison of the proposed estimation scheme and constant power scheme when there are multiple types of aperiodic tasks present in the system. As in Experiment 1, the two steps of setup are: *Estimated Power*: Periodic and aperiodic tasks consume worst case estimated power for both HotSpot and estimator. *Constant Power*: Estimator assumes 95^{th} percentile power and temperature profile for aperiodic tasks and worst case temperature and power profile for periodic task. Hotspot considers worst case power profile for all tasks. Again, simulated temperature values are read from HotSpot to emulate circuit sensor reading.

Figure 12 shows how aperiodic task acceptance ratio changes with time. Figure 13 shows the acceptance ratio for each type of aperiodic task along with the total number of instances for each type. Figure 13 shows that *constant power* scheme is highly biased towards accepting τ_{Low} and τ_{Medium}. Also, as expected (knowing the results of Experiment 1), no τ_{High} instances are accepted. Due to this bias, the maximum simulated temperature reached with *constant power* setup (94.74°C) is less than the maximum simulated temperature reached in *estimated power* experimental setup (99.56°C). *Constant power* scheme has a marginally higher acceptance rate for τ_{Low} instances. This is because of the lower circuit temperature caused due to low aperiodic task acceptance rate, as well as non acceptance of high activity tasks by the constant power scheme. If scheduling decisions are based on worst-case power consumption and corresponding thermal profiles, the number of accepted medium and high activity aperiodic task instances are considerably higher compared to the corresponding results for *constant power* scheme. Figure 12 shows that, despite having higher circuit temperatures and accepting considerable number of high activity aperiodic tasks, the *estimated power* scheme

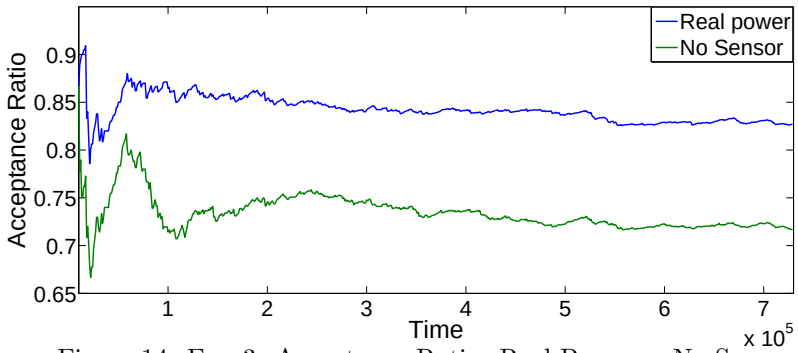

Figure 14: Exp 3: Acceptance Ratio. Real Power vs No Sensor

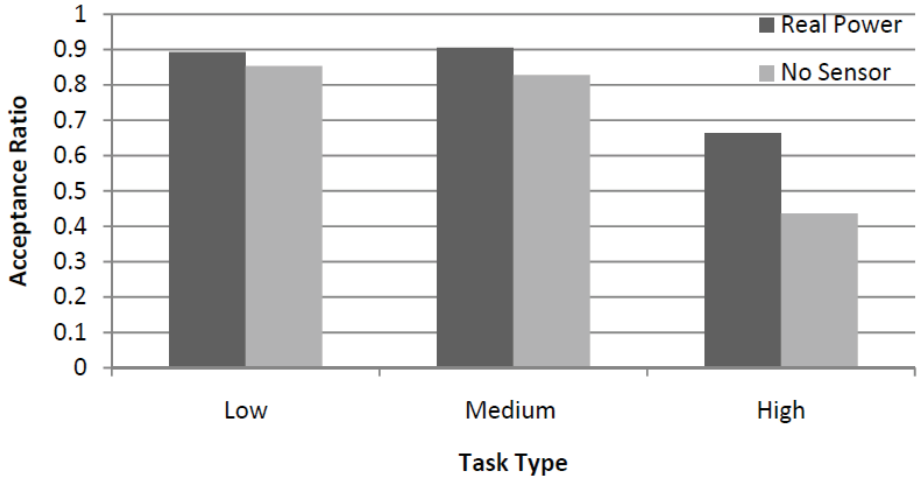

Figure 15: Experiment 3: Accepted Task Distribution

consistently has a higher acceptance ratio compared to *constant power* scheme.

Experiment 3 This experiment shos the advantage of reading temperature sensor values periodically as it offers significant gains in scheduling aperiodic tasks when process variations are present. For this experiment, the aperiodic task schedule generated in experiment 2 is used. Simulations are conducted for the following two setups:

1. *Real Power*: The proposed estimation approach assumes the worst case estimated power consumption for both periodic and aperiodic tasks. However, HotSpot assumes that each task instance consumes 70% $\rightarrow$ 80% of its worst case estimated power. Simulated temperature values are read after every hyperperiod to emulate circuit sensors.

2. *No Sensor*: The estimation approach considers worst case thermal profiles for all periodic and aperiodic task instances. However, simulated values are not read from HotSpot and therefore, no temperature corrections are possible in the absence of sensors.

Figure 14 shows how rejection ratio for both setups varies over time. Figure 15 shows the acceptance ratio for each type of aperiodic task. Figure 15 shows that the *no sensor* scheme is biased towards accepting lower activity aperiodic tasks compared to the *real power* scheme. However,

the bias is not as profound as the one observed in experiment 2. Despite the bias, *real power* scheme exhibits a considerably high acceptance ratio consistently; as shown in figure 14. Maximum simulated temperature for real power approach reaches 68.88°C compared to the maximum estimated temperature of 98.1°C for no sensor scheme.

The results in this section show that the proposed estimation and scheduling scheme can give significant gains in scheduling aperiodic tasks. This is because this scheme allows temporal variations in task power consumption to be modeled less conservatively. Furthermore, the results show that reading thermal sensors to correct estimation errors is important, and can provide significant gains.

References

[1] C. Liu and J. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM (JACM)*, vol. 20, no. 1, pp. 46–61, 1973.

[2] S. Baruah and G. Lipari, "Executing aperiodic jobs in a multiprocessor constant-bandwidth server implementation," in *Proceedings of the Euromicro Conference on Real-time Systems*, 2004, pp. 109–116.

[3] T. P. Baker, "Stack-based scheduling of realtime processes," *Journal of Real-Time Systems*, vol. 3, no. 1, pp. 67–99, Mar. 1991.

[4] K. Ramamritham, J. Stankovic, and W. Zhao, "Distributed scheduling of tasks with deadlines and resource requirements," *Computers, IEEE Transactions on*, vol. 38, no. 8, pp. 1110 –1123, Aug. 1989.

[5] K. G. Shin and P. Ramanathan, "Real-time computing: A new discipline in computer science and engineering," *Proceedings of the IEEE*, vol. 82, no. 1, pp. 6–24, Jan. 1994, (Invited paper).

[6] R. Viswanath, V. Wakharkar, A. Watwe, and V. Lebonheur, "Thermal performance challenges from silicon to systems," *Intel Technology Journal*, vol. 3, pp. 1–16, 2000.

[7] A. Coskun, T. Rosing, and K. Whisnant, "Temperature aware task scheduling in MPSoCs," in *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 2007, p. 1664.

[8] J. Choi, C. Cher, H. Franke, H. Hamann, A. Weger, and P. Bose, "Thermal-aware task scheduling at the system software level," in *Proceedings of the International Symposium on Low power Electronics and Design*. ACM, 2007, p. 218.

[9] A. K. Coskun, T. v. Rosing, K. A. Whisnant, and K. C. Gross, "Static and dynamic temperature-aware scheduling for multiprocessor socs," *IEEE Transactions on Very Large Scale Integrated Systems*, vol. 16, no. 9, pp. 1127–1140, 2008.

[10] J. Clabes, J. Friedrich, M. Sweet, J. DiLullo, S. Chu, D. Plass, J. Dawson, P. Muench, L. Powell, M. Floyd *et al.*, "Design and implementation of the POWER5 microprocessor," in *ISSCC*. IEEE, 2004, pp. 56–57.

[11] S. Memik, R. Mukherjee, M. Ni, and J. Long, "Optimizing thermal sensor allocation for microprocessors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 3, pp. 516–527, 2008.

[12] "Quad-core intel xeon processor 5400 series," http://download.intel.com/design/xeon/guides/318611.pdf.

[13] S. Baruah and N. Fisher, "The partitioned multiprocessor scheduling of sporadic task systems," in *Proceedings of the Real-time Systems Symposium*, Dec. 2005.

[14] S. K. Baruah, "Resource sharing in edf-scheduled systems: A closer look," in *Proceedings of the Real-time Systems Symposium*, Dec. 2006, pp. 379–387.

[15] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *High-Performance Computer Architecture, 2001. HPCA.*, 2001, pp. 171 –182.

[16] J. Yang, X. Zhou, M. Chrobak, Y. Zhang, L. Jin, and N. Corporate, "Dynamic thermal management through task scheduling," in *IEEE International Symposium on Performance Analysis of Systems and Software*, 2008.

[17] H. Liu, Z. Shao, M. Wang, and P. Chen, "Overhead-aware system-level joint energy and performance optimization for streaming applications on multiprocessor systems-on-chip," in *Proceedings of the Euromicro Conference on Real-time Systems*, July 2008, pp. 92–101.

[18] N. Bansal, T. Kimbrel, and K. Pruhs, "Speed scaling to manage energy and temperature," *J. ACM*, vol. 54, no. 1, pp. 1–39, 2007.

[19] S. Wang and R. Bettati, "Reactive speed control in temperature-constrained real-time systems," in *Proceedings of the Euromicro Conference on Real-time Systems*, 2006, pp. 160–170.

[20] J.-J. Chen, S. Wang, and L. Thiele, "Proactive speed scheduling for real-time tasks under thermal constraints," in *Proceedings of the Real-Time and Embedded Technology and Applications Symposium*, April 2009, pp. 141–150.

[21] S. Wang and R. Bettati, "Delay analysis in temperature-constrained hard real-time systems with general task arrivals," in *Proceedings of the Real-time Systems Symposium*. IEEE Computer Society, 2006, pp. 323–334.

[22] J. Chen, C. Hung, and T. Kuo, "On the minimization fo the instantaneous temperature for periodic real-time tasks," in *13th IEEE Real Time and Embedded Technology and Applications Symposium, 2007. RTAS'07*, 2007, pp. 236–248.

[23] J.-J. Chen and T.-W. Kuo, "Voltage scaling scheduling for periodic real-time tasks in reward maximization," in *Proceedings of the Real-time Systems Symposium*. IEEE Computer Society, 2005, pp. 345–355.

[24] T. Chantem, R. Dick, and X. Hu, "Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs," in *Proceedings of the conference on Design, automation and test in Europe*, 2008, pp. 288–293.

[25] N. Fisher, J.-J. Chen, S. Wang, and L. Thiele, "Thermal-aware global real-time scheduling on multicore systems," in *Proceedings of the Real-Time and Embedded Technology and Applications Symposium*, April 2009, pp. 131–140.

[26] K. Skadron, M. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 1, no. 1, pp. 94–125, 2004.

[27] C. Yao, K. Saluja, and P. Ramanathan, "Partition based SoC test scheduling with thermal and power constraints under deep submicron technologies," in *Proceedings of the Asian Test Symposium*. IEEE Computer Society, 2009, pp. 281–286.

[28] K. Jeffay, D. Stanat, and C. Martel, "On non-preemptive scheduling of period and sporadic tasks," in *Proceedings of the Real-time Systems Symposium*. IEEE, 2002, pp. 129–139.