2013 IEEE International Conference on Embedded and Real-Time Computing Systems and Applications

On Thermal Utilization of Periodic Task Sets in Uni-Processor Systems

Rehan Ahmed, Parameswaran Ramanathan, and Kewal K.Saluja
Department of Electrical and Computer Engineering
University of Wisconsin Madison
Madison, Wisconsin 53706
rahmed3@wisc.edu, parmesh@ece.wisc.edu, saluja@ece.wisc.edu

Abstract—**In this paper we introduce a novel characterization of real-time tasks based on their temperature impact. This characterization is used to analyze the schedulability of periodic real-time tasks in thermally constrained systems. The proposed characterization is important because thermal constraints are becoming increasingly vital due to rapid and increasing rise in power densities of modern architectures. As part of this work, we introduce the concept of "Accumulated Thermal Impact" (ATI), which represents cumulative temperature increase due to execution of a given task. The concept of ATI is then used to determine thermal utilization of a periodic task set, which is somewhat analogous to traditional computation utilization, and perform schedulability analysis. We also propose a speed scaling scheme for minimizing thermal utilization/system temperature. Our results show that thermal utilization of a periodic task set is strongly correlated to its thermal feasibility. The proposed speed scaling scheme is also shown to perform significantly better than current schemes in terms of thermal utilization/temperature minimization.**

***Index Terms*— Real-Time Systems, Thermal Aware Scheduling, Periodic Tasks, DVFS.**

I. Introduction

THE primary difference between real-time and general-purpose computing is that, tasks in real-time systems have timing/deadline constraints. Therefore, the correctness of a given operation depends on its logical correctness and the time at which output is delivered. Real-time computing systems are primarily found in embedded applications such as process control, multimedia communication, automotive, and biomedical systems.

Real-time systems have been an active area of research for the past four decades starting with pioneering work by Liu and Layland [1]. In the existing literature, various task (periodic, aperiodic, sporadic) and system (uni-processor, uniform multiprocessor, homogenous multiprocessor, heterogeneous multiprocessor) models have been considered. However, for real-time systems running on modern computing platforms, several critical issues/constraints have to be considered.

Specifically, with scaling of IC technology, considering thermal constraints of the processing platform has become increasingly important. This is because of the rapid and continuing rise in power densities of processors. The "Assembly and Packaging Tables" in International Technology Roadmap for Semiconductors (ITRS) 2010 update [2] project nearly doubling of average power density from 0.65−0.8 W/mm^2 in 2013 to 1.2−1.35 W/mm^2 by 2024. Furthermore, the peak power densities can be two orders of magnitude higher than these average power density numbers [3]. Higher power densities lead to increase in operating temperatures of the devices which may adversely affect device reliability and performance [4]. The loss of reliability or performance cannot be tolerated in most real-time systems, since failure in such a system may have severe consequences; including loss of life.

In general purpose computing, Dynamic Thermal Management (DTM) [5] is widely used to ensure that system's thermal constraints are not violated. However, the problem becomes more complex for real-time systems, since timing constraints of all tasks also have to be considered. Traditionally, research on real-time systems has focused on satisfying timing constraints of tasks. More recently, there have been works which consider power/thermal constraints of the system. Several of these works consider Dynamic Voltage and Frequency Scaling Strategies [6]-[13]. These speed scaling techniques can be divided into reactive schemes [6][7] and proactive schemes [9][12][13]. The difference between proactive and reactive schemes is that proactive schemes set the processor temperature judiciously based on the current temperature. Reactive schemes, on the other hand, "react" to the system temperature getting above a certain threshold, by altering speed.

Reactive speed scaling schemes are studied/analyzed by Wang et al. [6][7]. In their work, schedulability bounds for periodic real-time tasks are presented in [6] and delay analysis for a job stream is given in [7]. Chaturvedi et al. [8] propose an m-oscillating scheme for scheduling periodic tasks under thermal constraints. In this scheme, the processor speed oscillates between m levels to keep the system temperature below threshold. The scheme is shown to perform better than the two-speed reactive schemes. The reactive schemes are also improved upon by Chen et al. [9] using proactive speed scheduling to find speed schedules such that the timing and thermal constraints of periodic tasks are satisfied. They also derive approximation bounds on the maximum temperature for EDF schedule compared to a thermally optimal schedule in

[10]. Speed scaling is used in [11] to minimize energy for periodic real-time tasks. In their work, Quan et al. [12][13] derive thermal feasibility checks for a given periodic task schedule and corresponding speed schedule. They also formulate constructive scheduling algorithms for periodic tasks under thermal constraints.

More general real-time task models have also been considered in literature. In [14][15], Schor et al. perform worst case temperature analysis for real-time tasks running on multi-core systems. They use the concepts of real-time calculus [16] to model arrivals and computation demands of real-time tasks. The worst case computation methodology is further used in [17] to perform task and frequency assignment for real-time tasks executing on multi-core systems. The problem is formulated and solved as a binary optimization problem. Kumar et al. [18] perform worst case delay analysis on a stream of jobs following arrival and computation patterns based on real-time calculus.

Contributions: In contrast to other works in this area, this work proposes a novel characterization of real-time tasks called Thermal Utilization. Thermal Utilization of a given periodic task set can be used to discern the likelihood of the task set to be *thermally feasible*. A given task set is considered thermally feasible if repeated execution of tasks in the periodic task set do not cause any thermal violations. This concept is similar to computation utilization, which has been addressed by several works on real-time systems in the past three decades. In this work, we perform thermal/schedulability analysis of periodic real-time task sets.

Furthermore, the task power model used in this work is less restrictive compared to the power model commonly used in other works on thermal aware scheduling of real-time tasks. Most works assume a simplistic power model where processor power consumption is only a function of processor speed. This is not true in practice, and the power consumption can vary significantly based on the task which is being executed. In this work, we assume that the power consumption is a function of both, processor speed and the task being executed. We also propose a scheme for assigning speeds to individual periodic tasks to minimize/reduce system temperature/energy consumption.

Organization: This paper is organized as follows: Section II covers the system and task model used in this work. Section III covers the concept of Thermal Impact which is the proposed characterization of this work. This section also introduces the concept of Thermal Utilization for periodic real-time tasks. This is followed by the effect of speed scaling on Thermal Utilization and how speed scaling can be used to reduce/minimize Thermal Utilization, in section IV. Simulation results are presented in section V followed by conclusion and future work in section VI.

II. System and Task Model

1. *Processor Model*

We consider a uni-processor system in this paper. We assume that, for proper functioning of the system, the temperature of the processor must not exceed $\widehat{\Delta}$. The power consumed by the processor ($P(t)$) has two components: (i) dynamic power due to switching of gates, (ii) static power due to leakage currents. In this paper, total power consumed by the processor is written as:

$$P(t, \Theta(t)) = s(t)^{\gamma} a(t) + \delta\Theta(t) + \rho\,, \tag{1}$$

where $s(t)$ is the speed of the processor, $a(t)$ is the switching activity, $\Theta(t)$ is the processor temperature and γ, ρ and δ are constants. The first term models the dynamic power ($P_{dyn}(t)$) while the sum of second and third terms models the static power. Although leakage current increases rapidly with temperature, it is shown in [19] that within typical operating temperature range of processor, static power can be modeled as a linear function of temperature as in equation (1).

2. *Thermal Model*

The thermal effect of executing tasks on the processor is modeled using a simple RC circuit [20] shown in Figure 1. In this model, the flow of heat is modeled by current (*P(t)*) passing through a thermal resistance (*R*). Delay in temperature increase/decrease of the processor is modeled by a thermal capacitance (*C*) and temperature is modeled by potential difference. Θ_a is the ambient temperature.

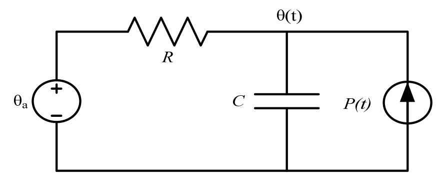

Figure 1: RC Thermal Model

Using this model, the processor temperature at any time *t* satisfies the following first order differential equation:

$$\Theta'(t) = \frac{P(t,\Theta(t))}{C} - \frac{\Theta(t) - \Theta_a}{RC}$$

$$= \frac{s(t)^{\gamma} a(t)}{C} - \left(\frac{1}{RC} - \frac{\delta}{C}\right)\Theta(t) + \left(\frac{\rho}{C} + \frac{\Theta_a}{RC}\right) \tag{2}$$

For notational simplicity, let $\beta = \frac{1}{RC} - \frac{\delta}{C}$ and $\theta(t) = C\,\Theta(t) - \frac{C(R\,\rho + \Theta_a)}{1 - R\,\delta}$. With some algebraic simplification, equation (2) can be written as:

$$\theta'(t) = s(t)^{\gamma} a(t) - \beta\theta(t) \tag{3}$$

This is similar to the variable transformation performed in [6]. Due to the analogy between equation (2) and (3), $\theta(t)$ is referred to as *adjusted temperature,* although its unit is joules. Solving this differential equation,

$$\theta(t) = \int_0^t s(u)^{\gamma} a(u)\, e^{\beta(u-t)} du \; + \; \theta(0)\, e^{-\beta t} \tag{4}$$

If dynamic power is constant, say *P*, over $[0, t)$, then equation (4) simplifies to

$$\theta(t) = \frac{P}{\beta}\left(1 - e^{-\beta t}\right) + \; \theta(0) e^{-\beta t} \tag{5}$$

In the remainder of this paper, the word "temperature" will refer to adjusted temperature. The unadjusted temperature value Θ will be referred to as "actual temperature". The adjusted temperature value corresponding to the actual value of threshold temperature ($\hat{\Delta}$) will be denoted as Δ.

3. *Task Model*

This work considers periodic preemptible real-time tasks. Each periodic task τ is characterized by computation time C_τ and period T_τ. This work considers synchronous periodic tasks (Deadline of a given periodic task instance coincides with the arrival of its next instance). The dynamic power consumption of the periodic task τ at speed s_τ is $P_{\tau,s_\tau} = A_\tau s_\tau^3$ where A_τ is the upper bound on the switching activity of τ. Tasks which are computationally extensive are expected to have higher activity (A_τ) compared to tasks which are computationally non-extensive. Note that, it is assumed that dynamic power consumption for a given task is constant. Dynamic power consumption can, however, vary across different tasks due to difference in activity. The thermal profile of periodic task τ at speed s_τ ($\eta_{\tau,s_\tau}(u)$) is defined as the processor temperature after u units of execution of task τ at speed s_τ, assuming that:

1. Task instance starts to execute when processor temperature is 0.
2. Dynamic power consumed by the task is P_{τ,s_τ} until time C_τ/s_τ.
3. Dynamic power consumed by the task is 0 after time C_τ/s_τ. From Equation (5), thermal profile of τ is:

$$\eta_{\tau,s_\tau}(t) = \begin{cases} \frac{A_\tau s_\tau^3}{\beta}\left(1-e^{-\beta t}\right) & 0 \le t < C_\tau/s_\tau \\ \frac{A_\tau s_\tau^3}{\beta}\left(1-e^{-\beta C_\tau/s_\tau}\right)e^{-\beta(t-C_\tau/s_\tau)} & t \ge C_\tau/s_\tau \end{cases} \tag{6}$$

Periodic task set is denoted by Γ. $\Pi(t)$ represents cyclic schedule of the tasks in periodic task set Γ. The value of $\Pi(t)$ is a tuple which identifies one of the n periodic tasks executing at time t at speed s, or $\emptyset$ if the system is idle at time t. The schedule is assumed to be cyclic i.e., $\Pi(t) = \Pi(t+kL)$ where $k = 0,1,2,..$ and L is the *hyperperiod* of the periodic task set Γ. L is equal to the the least common multiple of the periods of all periodic tasks in Γ. $\eta_\Pi(t)$ represents the thermal profile of periodic schedule $\Pi(t)$.

III. Thermal Impact/Load

We now introduce the proposed characterization. We first introduce the concept of *Accumulated Thermal Impact* (ATI) which is the cumulative thermal effect, *t* seconds after the start of execution of a given task. It is the integral of the thermal profile of a given task until time *t*. *Total Thermal Impact* (TTI) is also defined which characterizes the "net" thermal effect of a given task execution on temperature of the system. It is the integral of thermal profile of a given task until ∞. We first formally define ATI and TTI. These two concepts are used to define a concept called "Thermal Utilization" which is analogous to more familiar computational utilization in many respects. For instance, we show that the thermal utilization of a given periodic task set is independent of the scheduling scheme used to schedule periodic tasks. Based on this result, we formulate a necessary condition for thermal feasibility of periodic task set at the end of this section.

Accumulated Thermal Impact: Accumulated Thermal Impact (ATI) of task τ at speed s_τ is defined as:

$$\omega_{\tau,s_\tau}(t) = \int_0^t \eta_{\tau,s_\tau}(u)du.$$

From Equation (6):

$$\omega_{\tau,s_\tau}(t) = \begin{cases} \frac{A_\tau s_\tau^3}{\beta}\int_0^t\left(1-e^{-\beta u}\right)du & 0 \le t < \frac{C_\tau}{s_\tau} \\ \frac{A_\tau s_\tau^3}{\beta}\left(\begin{array}{c} \int_0^{\frac{C_\tau}{s_\tau}}\left(1-e^{-\beta u}\right)du + \\ \left(1-e^{-\beta\frac{C_\tau}{s_\tau}}\right)\int_0^{t-\frac{C_\tau}{s_\tau}} e^{-\beta u}du \end{array}\right) & t \ge \frac{C_\tau}{s_\tau} \end{cases}$$

Solving these integrals:

$$\omega_{\tau,s_\tau}(t) = \begin{cases} \frac{A_\tau s_\tau^3}{\beta}\left(t+\frac{e^{-\beta t}}{\beta}-\frac{1}{\beta}\right) & 0 \le t < \frac{C_\tau}{s_\tau} \\ \frac{A_\tau s_\tau^2\, C_\tau}{\beta}+\frac{A_\tau s_\tau^3 e^{-\beta t}}{\beta^2}\left(1-e^{\beta\frac{C_\tau}{s_\tau}}\right) & t \ge \frac{C_\tau}{s_\tau} \end{cases} \tag{7}$$

Total Thermal Impact: Note that, the value of ATI as $t \to \infty$ is bounded. We can thus define another term called Total Thermal Impact (TTI) of task τ executing at speed s_τ as the value of ATI of τ at $t \to \infty$. Using Equation (7), TTI of τ executing at speed s_τ can be computed by the following expression:

$$\omega_{\tau,s_\tau}(\infty) = \frac{A_\tau s_\tau^2\, C_\tau}{\beta} \tag{8}$$

Theoretical Results:

Using the definitions, of ATI and TTI, we can now characterize the thermal impact/utilization of a periodic schedule. For all theoretical results presented in this section, it is assumed that all instances of periodic task i are executed at speed s_i

Lemma 1: The integral of temperature within the hyperperiod when initial temperature is $\theta(0)$ is:

$$\int_0^L \left(\theta(0)e^{-\beta u} + \eta_\Pi(u)\right)du = \frac{\left(\theta(0)-\theta(L)\right)}{\beta} + \sum_{i=0}^{n-1}\frac{L}{T_i}\,\omega_{i,s_i}(\infty)$$

Proof: Given a periodic task i with period T_i, the hyperpeiod will have L/T_i instances of the periodic task. Let the start time of j^{th} instance of periodic task i in periodic schedule Π be $st_{i,j}$. Then, by superposition [21]:

$$\int_0^L \left(\theta(0)e^{-\beta u} + \eta_\Pi(u)\right)du$$

$$= \int_0^L \theta(0)e^{-\beta u}du + \sum_{i=0}^{n-1}\sum_{j=0}^{\frac{L}{T_i}-1}\int_0^{L-st_{i,j}} \eta_{i,s_i}(u)du$$

$$= \int_0^\infty \theta(0)e^{-\beta u}\,du - \int_L^\infty \theta(0)e^{-\beta u}\,du + \frac{L}{T_i}\sum_{i=0}^{n-1}\omega_{i,s_i}(\infty)$$

$$- \sum_{i=0}^{n-1}\sum_{j=0}^{\frac{L}{T_i}-1}\int_{L-st_{i,j}}^\infty \eta_{i,s_i}(u)du$$

$$= \int_0^\infty \theta(0)e^{-\beta u}du + \frac{L}{T_i}\sum_{i=0}^{n-1}\omega_{i,s_i}(\infty) - \left(\theta(0)e^{-\beta L} + \sum_{i=0}^{n-1}\sum_{j=0}^{\frac{L}{T_i}-1}\eta_{i,s_i}(L - st_{i,j})\right)\int_0^\infty e^{-\beta u}du$$

Through superposition, the constant term: $\theta(0)e^{-\beta L} + \sum_{i=0}^{n-1}\sum_{j=0}^{\frac{L}{T_i}-1}\eta_{i,s_i}(L - st_{i,j})$ is equal to the temperature at the end of hyperperiod. This is because by definition of thermal profile, the thermal effect of execution of instance j of periodic task i at time L is $\eta_{i,s_i}(L - st_{i,j})$. Therefore the thermal effects of all periodic instances in the hyperperiod can be superposed $\left(\sum_{i=0}^{n-1}\sum_{j=0}^{\frac{L}{T_i}-1}\eta_{i,s_i}(L - st_{i,j})\right)$ at time L along with the effect of cooling from temperature at the start of hyperperiod $(\theta(0)e^{-\beta L})$ to get temperature at the end of hyperperiod. Therefore, the equation above can be rewritten as:

$$\int_0^\infty (\theta(L) - \theta(0))e^{-\beta u}\,du + \frac{L}{T_i}\sum_{i=0}^{n-1}\omega_{i,s_i}(\infty) = \frac{\theta(0) - \theta(L)}{\beta} + \frac{L}{T_i}\sum_{i=0}^{n-1}\omega_{i,s_i}(\infty)$$

■

Lemma 2 (Quan et al. [12]): Assuming that periodic schedule Π is cyclic in nature (i.e. $\Pi(t) = \Pi(t + L)$), the temperature at the end of the hyperperiod, after repeated executions of periodic schedule, converges to the following value [12]:

$$\lim_{k\to\infty}(\theta(kL)) = \frac{\eta_\Pi(L)}{1 - e^{-\beta L}} \tag{9}$$

■

Definition 1: For a given cyclic schedule $\Pi(t)$, *Thermal Steady State* is defined as the scenario where temperature at the end of a given hyperperiod is equal to the temperature at the start of that hyperperiod:

$$\theta(mL) \approx \theta((m+1)L) = \lim_{k\to\infty}(\theta(kL)) = \frac{\eta_\Pi(L)}{1-e^{-\beta L}}$$

For a given periodic schedule $\Pi(t)$, let $\theta_\Pi = \frac{\eta_\Pi(L)}{1-e^{-\beta L}}$.

Theorem 1: If $\theta(0) = \theta_\Pi$, then

$$\int_0^L \left(\frac{\eta_\Pi(L)}{1 - e^{-\beta L}}\, e^{-\beta u} + \eta_\Pi(u)\right)du = \sum_{i=0}^{n-1}\frac{L}{T_i}\omega_{i,s_i}(\infty) \tag{10}$$

Proof: This theorem follows from Lemma 1 and 2. At thermal steady state, the temperature at the start of hyperperiod is $\frac{\eta_\Pi(L)}{1-e^{-\beta L}}$. Furthermore, by definition of thermal steady state, the temperature at the end of hyperperiod also approaches this value. Therefore, by Lemma 1:

$$\int_0^L \left(\theta(0)e^{-\beta u} + \eta_\Pi(u)\right)du = \frac{(\theta(0) - \theta(L))}{\beta} + \sum_{i=0}^{n-1}\frac{L}{T_i}\,\omega_{i,s_i}(\infty) = \sum_{i=0}^{n-1}\frac{L}{T_i}\,\omega_{i,s_i}(\infty)$$

■

It is important to note here that the integral of processor temperature at thermal steady state is independent of the periodic schedule. It is only a function of the periodic task set i.e., TTI and periods of individual periodic tasks.

Theorem 2: The maximum system temperature due to repeated executions of periodic schedule is equal to:

$$\max_{0\le t\le L}\left(max\left(\theta(0), \frac{\eta_\Pi(L)}{1 - e^{-\beta L}}\right)e^{-\beta t} + \eta_\Pi(t)\right), \tag{11}$$

where $\theta(0)$ is the initial temperature of the processor.

Proof: The proof of this theorem is similar to the analysis performed in [12].

■

For the remaining part of this paper, we assume that the initial processor temperature is less than the temperature at the start of hyperperiod at thermal steady state. Therefore, to determine thermal feasibility of a periodic task set/schedule, we have to ensure that there are no thermal violations at thermal steady state. Without this assumption, the thermal feasibility of the periodic task set is also a function of initial temperature.

Theorem 3: Let $\phi(\Pi)$ be the maximum temperature during the periodic schedule when initial temperature $\theta(0) = \theta_\Pi$, the steady state temperature. i.e.,

$\phi(\Pi) = \max_{0\le t\le L}\left(\theta_\Pi e^{-\beta t} + \eta_\Pi(t)\right)$ then,

$$\phi(\Pi) \ge \sum_{i=0}^{n-1}\frac{\omega_{i,s_i}(\infty)}{T_i} \tag{12}$$

i.e., no schedule of periodic tasks in Γ can have maximum temperature less than $\sum_{i=0}^{n-1}\frac{\omega_i(\infty)}{T_i}$ at thermal steady state.

Proof: From Theorem 1, we know that at thermal steady state, integral of temperature during the hyperperiod is $\sum_{i=0}^{n-1}\frac{L}{T_i}\,\omega_{i,s_i}(\infty)$. The average temperature within the hyperperiod is therefore given by: $\frac{\sum_{i=0}^{n-1}\frac{L}{T_i}\omega_{i,s_i}(\infty)}{L} = \sum_{i=0}^{n-1}\frac{\omega_{i,s_i}(\infty)}{T_i}$. The maximum temperature within the hyperperiod at thermal steady state has to be at least as high as the average temperature across the hyperperiod. Therefore, $\phi(\Pi) \ge \sum_{i=0}^{n-1}\frac{\omega_{i,s_i}(\infty)}{T_i}$

■

Now that we have established concept of Thermal Impact and how it relates to periodic schedules/tasks, we can introduce the concept of thermal utilization of a periodic task/schedule and how it relates to thermal feasibility of a periodic task set. This is analogous to computation utilization and how that relates to timing feasibility of a periodic task set.

Definition 2: Given a periodic task i, the *Thermal Utilization* of i at speed s_i is defined as:

$$\Upsilon_{\mathrm{i,s_i}} = \frac{\omega_{i,s_i}(\infty)}{T_i\Delta}$$

Where Δ is the thermal constraint of the processor as defined in section 2.1.

Lemma 3: Integral of hyperperiod thermal profile at thermal steady state can be computed using the following expression:

$$\int_0^L \left(\frac{\eta_\Pi(L)}{1-e^{-\beta L}}\, e^{-\beta u} + \eta_\Pi(u) \right) du = L\Delta. \sum_{i\epsilon\Gamma} \Upsilon_{i,s_i} \qquad (13)$$

Proof: The proof of this lemma follows from Theorem 1 and definition of thermal utilization.

■

Theorem 4: If $\sum_{i\epsilon\Gamma}\Upsilon_{i,s_i} > 1$ and all instances of periodic task i are executed at speed s_i, then there is no feasible schedule for Γ which meets the thermal constraint.

Proof: From Theorem 1 and Theorem 3, we know that, $\phi(\Pi) \geq \frac{\int_0^L \left(\frac{\eta_\Pi(L)}{1-e^{-\beta L}} e^{-\beta u} + \eta_\Pi(u)\right) du}{L}$. Furthermore, in order for a given periodic schedule to be thermally feasible, $\phi(\Pi) \leq \Delta$. From Lemma 3, we know that if thermal utilization is greater than 1: $\int_0^L \left(\frac{\eta_\Pi(L)}{1-e^{-\beta L}}\, e^{-\beta u} + \eta_\Pi(u)\right) du > L\Delta$. Therefore, if thermal utilization is greater than 1, $\phi(\eta_\Pi) > \Delta$, which violates the processor thermal constraint.

■

IV. THERMAL UTILIZATION AND SPEED SCALING

To decrease the thermal utilization of a given periodic task without changing its functionality (periods/deadlines and computations performed cannot be changed), we have to change its TTI. From Equation (8), the TTI of a task i is $\omega_{i,s}(\infty) = \frac{A_i s^2\, C_\tau}{\beta}$. In this equation the scheduler has control only over processor speed s. TTI of a task is clearly proportional to square of processor speed, whereas, its computation utilization is inversely proportional to processor speed. If TTI is smaller, processor temperature is lowered and thermal violation is less likely. Lowering processor temperature also has the added benefit of reducing leakage currents. However, since computation utilization increases when processor speed is reduced, it may be more difficult to meet timing constraints of the periodic task set. It is, in general, highly desirable to lower thermal utilization of the task set without jeopardizing the timing constraints. We now present algorithms to optimize thermal utilization subject to the timing constraints. Proof of optimality of some of these algorithms and the theoretical rationale for certain heuristics are also presented in the rest of this paper.

For notational brevity, we define the following term:

Definition 3: For $M \subseteq \Gamma$, Let:

$$G(M) = \sum_{i\in \mathrm{M}} \frac{C_i}{T_i}(A_i)^{1/3}$$

Theorem 5: If all instances of periodic task i can be executed at speed:

$$s_i^* = A_i^{-1/3} G(\Gamma)\ \forall\, i \in \Gamma \qquad (14)$$

then, the thermal utilization of periodic task set Γ is minimized.

Proof: The energy consumed by a given periodic task i executing at speed s_i during the hyperperiod is equal to:

$$\mathrm{E}_{\mathrm{i,s_i}} = \frac{C_i\, P_{i,s_i}}{s_i} . \frac{L}{T_i}$$

Therefore, by Definition 2, the thermal utilization of Γ can be written as:

$$\sum_{i\epsilon\Gamma} \Upsilon_{\mathrm{i,s_i}} = \frac{1}{L\Delta\beta} \sum_{i\epsilon\Gamma} \mathrm{E}_{\mathrm{i,s_i}}$$

Therefore, if we minimize energy of a periodic task set, its thermal utilization is also minimized. We know from [22] that energy is minimized if speed scaling is performed such that all tasks have same power consumption and the periodic task utilization is equal to 1, i.e.,

$\sum_{i\in\Gamma} \frac{C_i}{T_i\, s_i^*} = 1$ and $A_i s_i^{*3} = A_j s_j^{*3}\ \ \forall\, i,j \in \Gamma$

Hence, the expression for s_i^* in equation (14) follows.

■

Unfortunately, it may not be possible to execute all tasks at speeds given by Theorem 5 because $s_i^* \notin [s_{min}, s_{max}]$ for some task i, where s_{min} and s_{max} are the processor speed constraints.

We now present an algorithm to identify the speed solution which minimizes thermal utilization of the periodic task set Γ for the special case when there is no constraint on the minimum speed i.e., $s_{min} = 0$. Without loss of generality, s_{max} can be assumed to be 1.

```
1. Let $\Gamma_X = \emptyset, \Gamma_{UX} = \Gamma$
2. $M_X = \left\{i: A_i^{-1/3}\, G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) > 1, i \in \Gamma_{UX}\right\}$
3. While $M_X \neq \emptyset$ do
4.     $s_i = 1\ \forall\, i \in M_X$
5.     $\Gamma_X = \Gamma_X \cup M_X$
6.     $\Gamma_{UX} = \Gamma_{UX} \backslash M_X$
7.     $M_X = \left\{i: A_i^{-1/3}\, G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) > 1, i \in \Gamma_{UX}\right\}$
8. Endwhile
9. $s_i = A_i^{-1/3}\, G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) \forall i \in \Gamma_{UX}$
```

Algorithm NoMinSpeed: Assigning speeds to periodic tasks when $s_{min} = 0$ and $s_{max} = 1$

Algorithm NoMinSpeed is iterative. At start of an iteration, Γ_X is a set of periodic tasks for which speed decision has been made. Initially, this set is empty. Conversely, Γ_{UX} is a set of periodic tasks for which speed decision has not been made. This set initially consists of all periodic tasks in Γ. In line 2, M_X is the set of periodic tasks which violate the maximum speed constraint. These tasks are then assigned speed $1(s_{max})$ in line 4. M_X is then recomputed from the set of periodic tasks for which speeds have not been assigned in line 7. This process of computing M_X and assigning speed 1 to all tasks in M_X is repeated until the recomputation of M_X in line 7 yields an empty set. After the termination of while loop, the remaining tasks are assigned speeds in line 9.

Theorem 6: Algorithm NoMinSpeed is optimal with respect to minimizing thermal utilization when $s_{min} = 0$.

Proof: Proof given in Appendix A.

■

If $s_{min} \neq 0$, then it is possible that the speed assignment from Algorithm NoMinSpeed is infeasible for some task i.e., $s_i < s_{min}$ for some task i. Algorithm SeCTUM (Speed

```
1.  Let $\Gamma_X = \emptyset, \Gamma_{UX} = \Gamma$
2.  $M_X = \left\{i: A_i^{-1/3} G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) > 1\right\} \ \forall i \in \Gamma_{UX}$
3.  While $M_X \neq \emptyset$ do
4.      $s_i = 1 \ \forall \ i \in M_X$
5.      $\Gamma_X = \Gamma_X \cup M_X$
6.      $\Gamma_{UX} = \Gamma_{UX} \backslash M_X$
7.      $M_X = \left\{i: A_i^{-1/3} G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) > 1, i \in \Gamma_{UX}\right\}$
8.  Endwhile
9.  $M_N = \left\{i: A_i^{-1/3} G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) < s_{min}, i \in \Gamma_{UX}\right\}$
10. While $M_N \neq \emptyset$ do
11.     $s_i = s_{min} \ \forall \ i \in M_N$
12.     $\Gamma_X = \Gamma_X \cup M_N$
13.     $\Gamma_{UX} = \Gamma_{UX} \backslash M_N$
14.     $M_N = \left\{i: A_i^{-1/3} G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) < s_{min}, i \in \Gamma_{UX}\right\}$
15. Endwhile
16. $s_i = A_i^{1/3} G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) \ \forall i \in \Gamma_{UX}$
```

Algorithm 1: SeCTUM. Speed Constrained Thermal Utilization Minimization when $s_{min} \neq 0$ and $s_{max} = 1$

Constrained Thermal Utilization Minimization) strives to find the optimal speed assignment when $s_{min} \neq 0$. Note that, without loss of generality, s_{max} can be assumed to be 1.

Lines 1-8 of SeCTUM are exactly as lines 1-8 of Algorithm NoMinSpeed. In line 9, M_N is the set of tasks that violate the minimum speed constraint at the end of Algorithm NoMinSpeed. These tasks are then assigned speed s_{min} and set M_N is recomputed out of the set of periodic tasks for which speed assignments have not been done. This process of computing M_N and setting the corresponding speeds of tasks to s_{min} is repeated in the second while loop (lines 10-15) until the computation of M_N yields an empty set. After the termination of second while loop, the remaining periodic tasks are assigned speeds as specified in line 16.

Theorem 7: If $s_i^* \leq s_{max} \ \forall i \in \Gamma$ or $s_i^* \geq s_{min} \ \forall i \in \Gamma$ where s_i^* is as defined in equation (14), then the speed solution given by SeCTUM will be optimal.

Proof: See Appendix B.

∎

If the conditions in Theorem 7 are not satisfied, the solution from Algorithm SeCTUM may be suboptimal. In these instances, the solution from Algorithm SeCTUM can be further improved using the following simple modification.

1. Let S_{SC1} be the speed solution given by SeCTUM
2. Let S_{SC2} be the speed solution given by an altered version of SeCTUM where the assignment of set M_N and second while loop (lines 9-15) are executed before the assignment of set M_X and first while loop (lines 3-8).
3. If speed solution S_{SC2} is computationally feasible, then pick speed solution $S \in \{S_{SC1}, S_{SC2}\}$ which gives the minimum thermal utilization. Otherwise, $S = S_{SC1}$.
4. We call this improved version I-SeCTUM.

We now illustrate the working of I-SeCTUM. Suppose that a periodic task set has the following three periodic tasks:

	Period (ms)	Computation Time (ms)	$Activity/\beta$
Task 1	60	15	30
Task 2	50	20	80
Task 3	100	30	40

R	0.36
C	0.8
ρ	0.1
δ	0.001
Θ_a	40^oC
$\widehat{\Delta}$ (Δ)	100^oC (47.96)

Table 1: Processor Thermal Parameters

Further suppose that the thermal parameters of the processor are as given in Table 1. Based on these parameters, $\beta = 3.4735\ s^{-1}$. Assume that the processor can only vary speed between [0.9, 1.0].

From definition 2, the thermal utilization of three tasks are:
$\Upsilon_{\text{Task1,1}} = \frac{15\times30}{60\times47.96} = 0.156$, $\Upsilon_{\text{Task2,1}} = \frac{20\times80}{50\times47.96} = 0.6672$, $\Upsilon_{\text{Task3,1}} = \frac{30\times40}{100\times47.96} = 0.2502$

Hence, the total thermal utilization of the task set is:

$$\Upsilon_{\text{Task1,1}} + \Upsilon_{\text{Task2,1}} + \Upsilon_{\text{Task3,1}} = 1.0738$$

Since the thermal utilization of this task set is greater than 1, it is not thermally feasible (See Theorem 4). However, this thermal utilization can be reduced using speed scaling. We will now compute speed solution S_{SC1}(first step if I-SeCTUM). For notational brevity, we will call the term $A_i^{-1/3} G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right)$, the target speed ($\bar{s}_i$) of a given task i. The value of $G(\Gamma_{UX})$ in line 2 of SeCTUM is:

$$G(\Gamma_{UX}) = \left(\frac{15}{60}\sqrt[3]{30\beta} + \frac{20}{50}\sqrt[3]{80\beta} + \frac{30}{100}\sqrt[3]{40\beta}\right) = 5.3405$$

The target speeds from line 2 are:

$\bar{s}_{Task1} = (30\beta)^{-1/3} \times 5.3405 = 1.1349$
$\bar{s}_{Task2} = (80\beta)^{-1/3} \times 5.3405 = 0.8184$
$\bar{s}_{Task3} = (40\beta)^{-1/3} \times 5.3405 = 1.0311$

The target speeds for Task 1 and Task 3 are greater than 1 (s_{max}). Now we will set the speeds of Task 1 and Task 3 to 1 (s_{max}) in line 4 of SeCTUM. After this assignment, the only task in Γ_{UX} is Task 2. Therefore, the target speed of Task 2 from line 9 of SeCTUM is:

$$\bar{s}_{Task2} = (80\beta)^{-\frac{1}{3}} \times \frac{20}{50}\sqrt[3]{80\beta} \Big/ \left(1 - \frac{15}{60\times1} - \frac{40}{100\times1}\right) = 0.8889$$

Since, the target speed of Task 2 is less than 0.9 (s_{min}), Task 2 is added to set M_N in line 9 of SeCTUM. Task 2 is then assigned speed 0.9 in line 11 of SeCTUM. After this assignment, there are no tasks remaining in Γ_{UX}. Therefore, the first speed solution $S_{SC1} = \{1, 0.9, 1\}$. The thermal utilization of periodic task set with this speed solution is 0.947. With this thermal utilization, the periodic task set may be thermally feasible.

We now try to improve this speed solution by evaluating speed solution S_{SC2} (Step 2 of I-SeCTUM). For this speed solution, line 9-15 of SeCTUM is executed before line 2-8. Again, the target speeds from line 9 are:

$\bar{s}_{Task1} = (30\beta)^{-1/3} \times 5.3405 = 1.1349$

$\bar{s}_{Task2} = (80\beta)^{-1/3} \times 5.3405 = 0.8184$

$\bar{s}_{Task3} = (40\beta)^{-1/3} \times 5.3405 = 1.0311$

Task 2 is now added to set M_N because $\bar{s}_{Task2} < 0.9$. Task 2 is then assigned speed of 0.9 in line 11 of SeCTUM and the new target speeds are computed in line 14 as:

$\bar{s}_{Task1} = 1.04433, \qquad \bar{s}_{Task3} = 0.9489$

The recomputation of set M_N yields an empty because the target speeds of both unassigned tasks are greater than 0.9 (s_{min}). Next, we execute lines 1-8 of SeCTUM. The target speed for Task 1 (1.04433) is greater than $1(s_{max})$. Therefore, Task 1 is added to M_X. The speed of Task 1 is then set to 1 in line 4 and new target speed for Task 3 is computed in line 7 as: $\bar{s}_{Task3} = 0.9818$. The recomputation of M_X yields an empty set and task 3 is assigned speed 0.9818 in line 16. Therefore speed solution $S_{SC2} = \{1, 0.9, 0.9818\}$. Note that if the target speed of task 3 in line 7 had been greater than 1, than the resulting speed solution S_{SC2} would have been computationally infeasible. This is because any speed assignment for task 3 less than $\bar{s}_{Task3}$ would have made the computation utilization of the periodic task set greater than 1.

The thermal utilization for periodic task set with speed solution S_{SC2} is 0.938. This is less than the corresponding thermal utilization for speed solution S_{SC1} (0.947). Therefore, in accordance with step 3 of I-SeCTUM, speed solution S_{SC2} is selected. In this case, even though the conditions for optimality in Theorem 7 are not satisfied, S_{SC2} is also the optimal speed assignment obtained by solving the non-linear optimization problem.

V. RESULTS

In this section, we illustrate through a series of experiments how the concept of thermal impact/utilization affects the thermal schedulability of periodic tasks; and how speed scaling can be employed to reduce thermal utilization of a periodic task set.

System Model Parameters: The system considered in this work is a uni-processor. It is assumed that the system is DVFS capable with speed varying between [0.625, 1.0]. This is consistent with the speed range of Intel Turbo Boost technology for state of the art processors. The thermal parameters of the processor are given in Table 1.

Task Model Parameters: Experiments are conducted for periodic real-time tasks with varying computation and thermal utilization. Periodic task sets are constructed with the following computation utilizations [0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.93, 0.95, 0.97, and 0.99]. For each computation utilization, 1000 periodic task sets are simulated with thermal utilization varying between 0.3 to 1.4. The task periods vary between 1ms to 1s. Hyperperiod length for all periodic task sets is set to 1s.

In the first set of experiments, we analyze the impact of thermal utilization on thermal feasibility of periodic task sets. In Theorem 4, we showed that having thermal utilization ≤ 1 is necessary for thermal feasibility. In the first set of experiments, we empirically evaluate the sufficiency of this condition. If the condition was both necessary and sufficient,

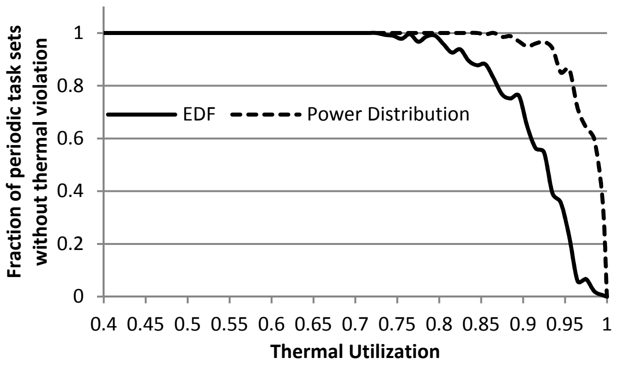

Figure 2: Thermal feasibility analysis of periodic tasks wrt Thermal Utilization

then every task set with thermal utilization ≤ 1 will be thermally feasible. Figure 2 shows how close we get to this ideal case. In Figure 2, we schedule a given task set using two different scheduling algorithms: (a) EDF (b) Power Distribution.

In the EDF algorithm, the task instance with earliest deadline is always given the highest priority and is executed ahead of lower priority task instances. The EDF algorithm does not consider the thermal impact of executing a task instance while scheduling. Once the task set is scheduled, we use the thermal model given in section II to estimate the peak processor temperature. Furthermore, the processor temperature is estimated at thermal steady state as defined in Definition 1. If the peak processor temperature is less than the acceptable threshold (Δ), then the task set is thermally feasible. Otherwise, we conservatively assume the task set to be thermally infeasible (Note that there may be other scheduling algorithms which may successfully schedule the task set without violation of system thermal constraint). The solid line in Figure 2 plots the fraction of thermally feasible task sets as a function of thermal utilization. Observe that when thermal utilization is less than 0.7, almost all task sets are thermally schedulable using EDF.

Figure 2 also plots a similar curve for a different scheduling algorithm called Power Distribution. This scheme is a thermal-aware scheduling heuristic developed by the authors of this paper. In this heuristic, the task instances for execution are selected in such a way that the processor temperature does not vary significantly throughout the hyperperiod; subject to all task deadlines being met. Since this heuristic is thermally aware, it is, in general, able to schedule more task sets than EDF. Hence, the fraction of thermally feasible task sets is higher for Power Distribution than EDF. For this heuristic, almost all task sets with thermal utilization less than 0.85 can be feasibly scheduled.

Other scheduling strategies may further schedule task sets not successfully scheduled by Power Distribution scheme. However, even, just based on Power Distribution scheme, one can conclude that the gap between necessary and sufficient conditions for thermal feasibility is not large.

The next set of results evaluate the impact of speed scaling on thermal utilization. The following three schemes are compared

1. Optimal: Results corresponding to Optimal speed assignment solution.

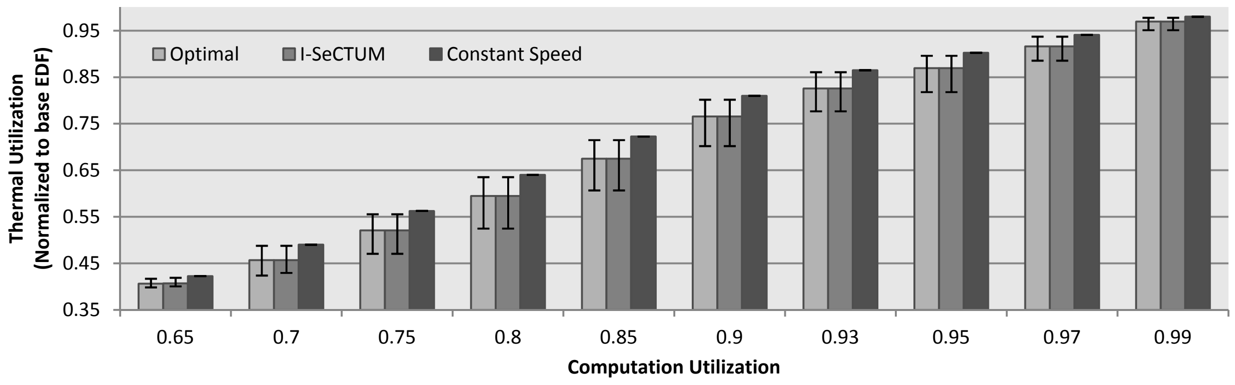

Figure 3: Thermal Utilization of Optimal, I-SeCTUM and Constant Speed schemes normalized to thermal utilization of periodic taskset without speed scaling

2. I-SeCTUM: Results corresponding to speed assignment solutions obtained from heuristic I-SeCTUM.
3. Constant Speed: In this speed assignment approach, each periodic task is executed at the same constant speed such that the computation utilization of the periodic task set becomes 1; subject to the minimum speed constraint. This is similar to the approach presented in [22] for energy minimization.

Figure 3 plots the resulting thermal utilization after speed scaling for the three algorithms, for different groups of task sets categorized based on their computation utilization. For each group (corresponding to a particular computation utilization), 1000 periodic task sets are simulated. All results are normalized to the thermal utilization without speed scaling, and the average of normalized thermal utilizations for each group is plotted in Figure 3. Three key observations can be made from this figure:

1. When computation utilization is small, speed scaling can significantly reduce thermal utilization. For instance, when computation utilization is 0.65, the thermal utilization after speed scaling is less than 45% of the thermal utilization without speed scaling. This result is expected because the processor speed can be more easily reduced, without violating timing constraints, when computation utilization is low.
2. Algorithm I-SeCTUM is better than constant speed scheme. In other words, unlike [22], it is better to run each task at an individual selected speed rather than a single speed for all tasks. This qualitative difference in result arises because Algorithm I-SeCTUM is more general in the sense that it does not assume that all tasks consume the same power when executed. Instead, I-SeCTUM accounts for differences in activity (hence power consumption) between the tasks.
3. Algorithm I-SeCTUM is almost as effective as optimal scheme irrespective of the computation utilization. This is highly desirable because the runtime complexity of I-SeCTUM is much less than the optimal scheme which requires solving a non-linear optimization problem. In fact, I-SeCTUM can be used at runtime to reclaim processing times unused by other task instances. This is a significant factor because real-time tasks are often scheduled based on worst case execution times which are usually much larger than the actual execution times. The results with resource reclaiming are not included in this paper. An outline of the resource reclaiming scheme is given in section VI. The effectiveness of I-SeCTUM as compared to Optimal is further demonstrated in Figure 4, where a histogram of difference between I-SeCTUM's and Optimal's solutions is plotted. Observe that in more than 95% of the case, I-SeCTUM's solution exactly matches the optimal solution. Almost all of the remaining suboptimal solutions have difference in thermal utilization of less than 0.01. Constant speed, on the other hand, is suboptimal in considerably more cases.

Finally, Figure 5 shows the effect of reductions in thermal utilization through speed scaling on processor temperature. To illustrate this, we plot the maximum temperature achieved by the speed assignments of all three schemes for various computation utilizations. The maximum temperature result for periodic task set without speed scaling is also given. All tasks are scheduled using EDF scheduling strategy and the maximum temperature is computed at thermal steady state (Definition 1). These results are similar to the results shown in Figure 3, showing strong correlation between thermal utilization and maximum temperature.

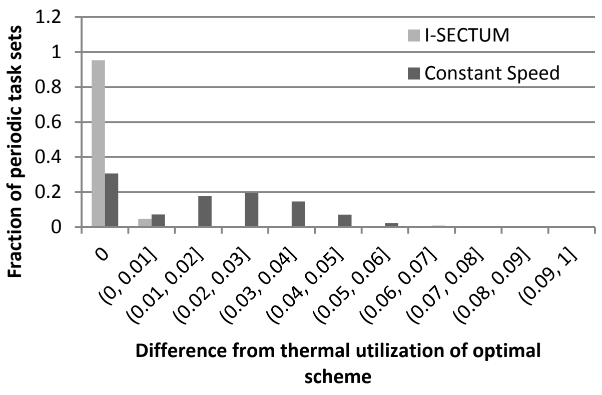

Figure 4: Difference in thermal utilizations of I-SeCTUM and Constant Speed schemes from Optimal scheme

VI. CONCLUSION AND FUTURE WORK

This work presents a novel characterization for periodic tasks which quantifies their thermal load/utilization. We also

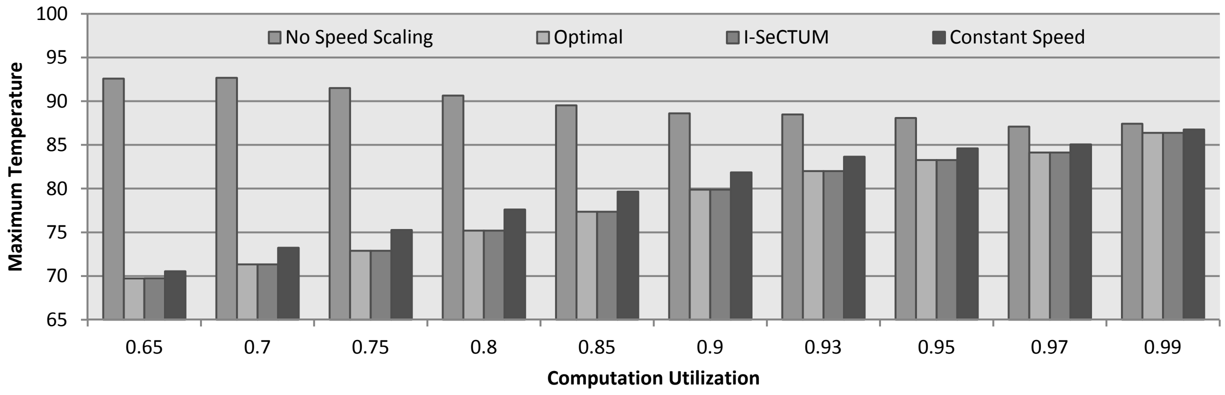

Figure 5: Maximum Temperature of No Speed Scaling, Optimal, I-SeCTUM, and Constant Speed schemes

present a necessary condition for thermal feasibility of periodic real-time tasks based on their thermal utilization. A scheme to reduce temperature/energy consumption of periodic task sets using speed scaling is also presented. The results show that the proposed characterization is beneficial for performing thermal feasibility analysis of periodic real-time tasks. The proposed speed scaling scheme is also shown to perform better than existing schemes in literature when there are differences in power consumption of individual real-time tasks. Following extensions to this work are possible.

Multi-Processor Extension: The proposed characterization of thermal utilization needs to be extended to modern processing platforms. These platforms primarily consist of Many/Multi processing cores. The challenge in this extension lies in the thermal interaction between cores and how that alters the TTI and thermal utilization of periodic tasks. Depending on how cores interact thermally, the thermal utilization of periodic tasks will also be a function of the core on which the task is executing.

Aperiodic and Sporadic Tasks: The proposed concept/ characterization may also be used to form algorithms/ heuristics for scheduling of aperiodic and sporadic tasks in thermally-constrained environments.

Dynamic Reclaiming Algorithm: I-SeCTUM can be extended to further minimize energy/temperature when tasks require less computation time then the worst case estimate. Suppose that at time t, additional slack of u time units is available due to early completion of periodic task instance τ. This additional slack may be used by I-SeCTUM to speed scale all ready task instances with priority lower than τ. This can further reduce thermal utilization/energy consumption of periodic task set at run-time, after I-SeCTUM is already applied on the periodic task set to minimize/reduce its thermal utilization as explained in section IV of this paper.

REFERENCES

[1] Liu, Chung Laung, and James W. Layland. "Scheduling algorithms for multiprogramming in a hard-real-time environment." *Journal of the ACM (JACM)*20.1 (1973): 46-61.

[2] "International technology roadmap for semiconductors," 2010. [Online].Available:http://www.itrs.net/links/2010itrs/home2010.htm

[3] G. Link and N. Vijaykrishnan, "Thermal trends in emerging technologies," in International Symposium on Quality Electronic Design, march 2006, pp. 8 pp. –632.

[4] 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.

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

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

[7] 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, December 2006, pp. 323–334.

[8] V. Chaturvedi, H. Huang, and G. Quan, "Leakage aware scheduling on maximum temperature minimization for periodic hard real-time systems," International Conference on Computer and Information Technology, vol. 0, pp. 1802–1809, 2010.

[9] 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.

[10] J.-J. Chen, C.-M. Hung, and T.-W. Kuo, "On the minimization of the instantaneous temperature for periodic real-time tasks," in Proceedings of the Real-Time and Embedded Technology and Applications Symposium, April 2007, pp. 236–248.

[11] S. Wang, J. Chen, Z. Shi, and L. Thiele, "Energy-efficient speed scheduling for real-time tasks under thermal constraints," in 2009 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications. IEEE, 2009, pp. 201–209.

[12] G. Quan, Y. Zhang, W. Wiles, and P. Pei, "Guaranteed scheduling for repetitive hard real-time tasks under the maximal temperature constraint," in Proceedings of the 6th IEEE/ACM/IFIP international conference on Hardware/Software codesign and system synthesis, October 2008, pp. 267–272.

[13] G. Quan and Y. Zhang, "Leakage aware feasibility analysis for temperature-constrained hard real-time periodic tasks," in Proceedings of the Euromicro Conference on Real-time Systems, July 2009, pp. 207–216.

[14] Lars Schor, Hoeseok Yang, Iuliana Bacivarov and Lothar Thiele, "Worst-Case Temperature Analysis for Different Resource Models", IET Circuits, Devices and Systems 2012.

[15] Lars Schor, I. Bacivarov.; Hoeseok Yang; L Thiele, "Worst-Case Temperature Guarantees for Real-Time Applications on Multi-core Systems," Real-Time and Embedded Technology and Applications Symposium (RTAS), 2012.

[16] L Thiele, S. Chakraborty; M. Naedele, "Real-time calculus for scheduling hard real-time systems,". Proceedings of IEEE International Symposium on Circuits and Systems, ISCAS 2000 Geneva.

[17] Lars Schor, Hoeseok Yang, Iuliana Bacivarov, and Lothar Thiele. "Thermal-Aware Task Assignment for Real-Time Applications on Multi-Core Systems.", International Symposium on Formal Methods for Components and Objects (FMCO) 2011.

[18] P. Kumar and L. Thiele, "Timing Analysis on a Processor with Temperature-Controlled Speed Scaling," Real-Time and Embedded Technology and Applications Symposium, April 2012, pp. 77-86.

[19] Yongpan Liu; Dick, R.P.; Li Shang; Huazhong Yang, "Accurate Temperature-Dependent Integrated Circuit Leakage Power Estimation is

Easy," *Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE '07* , vol., no., pp.1-6, 16-20 April 2007

[20] 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, March 2004.

[21] R. Ahmed, P. Ramanathan, K.K. Saluja, C. Yao, "Scheduling Aperiodic Tasks in Next Generation Embedded RealTime Systems", International Conference on VLSI Design 2013.

[22] Aydin, H., Melhem, R., Mosse, D., Mejia-Alvarez, P., "Dynamic and aggressive scheduling techniques for power-aware real-time systems," Proceedings of Real-Time Systems Symposium, 2001.

VII. Appendix A

Proposition 1: Let $\Gamma_X \subset \Gamma$ be a set of all periodic tasks for which the speed decision has been made with speeds equal to $s_{i,X}\ \forall\ i\ \in\ \Gamma_X$. Then, the energy consumption of the remaining periodic tasks in set $\Gamma_{UX} \subset \Gamma - \Gamma_X$ is minimized if they can be executed at speed:

$$s_i = s_{i,X} \qquad \forall\ i\ \in \Gamma_X$$
$$s_i = A_i^{-1/3}\, G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) \qquad \forall i \in \Gamma_{UX} \tag{15}$$

Proof: Given that the speed decision for all periodic tasks in Γ_X has been made and their speed cannot be changed, the computation utilization of all periodic tasks in Γ_X is equal to $\sum_{\forall i\in\Gamma_X} \frac{C_i}{T_i s_{i,X}}$. Therefore, the maximum computation utilization of periodic tasks in Γ_{UX} is $\left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right)$. Therefore, by theorem 5, the energy consumption of the periodic tasks in set Γ_{UX} is minimized if:

$$s_i = A_i^{-1/3}\, G(\Gamma_{UX}) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) \qquad \forall i \in \Gamma_{UX}$$

■

However, if we now consider the minimum and maximum speed constraints, we may not be able to execute all periodic tasks at their optimal target speed. To solve the problem with speed constraints, we first enforce the maximum speed constraint without enforcing the minimum speed constraint.

Proposition 2: Let $M_X = \{i: s_i^* > s_{max}\}$ where s_i^* is as defined in equation (14), then for the special case where $s_{min} = 0$, the thermal utilization of the periodic task set is minimized only if: $s_i = s_{max}\ \forall i \in M_X$

Proof: Suppose that we start solving the optimization problem by setting the speed of a single periodic task $\tau \in Mx$ to s_{max}. Then by proposition 1, we know that the optimal solution for remaining periodic tasks would try to do speed assignment such that

$$\bar{s}_i = A_i^{-1/3}\, G(\Gamma\backslash\tau) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) \ \forall i \in \{\Gamma\backslash\tau\}$$

Since periodic task τ was executed at a speed $s_\tau = s_{max} < s_\tau^*$, $\bar{s}_i > s_i^*\ \ \forall i \in \{\Gamma\backslash\tau\}$. Therefore, if we now compute set $\widehat{M}_X = \{i: \bar{s}_i > s_{max}\}$, we know that $\widehat{M}_X \supseteq M_X$.

Therefore, the order in which the speeds of periodic tasks in M_X are assigned does not influence the solution. Furthermore, due to the convexity of the objective function $\left(\frac{1}{\Delta}\sum_{i=0}^{n-1}\frac{C_i P_i s_i^2}{T_i}\right)$ in terms of variable s, if $s_i^* \geq s_{max}$ for a given periodic task i, the optimal speed assignment would set the speed of periodic task i equal to s_{max}.

■

In Algorithm, we first set the speeds of tasks in M_X to $s_{max} = 1$. After assignment M_X is recomputed and assigned. This process is repeated until the recomputation of M_X yields an empty set. By proposition 2, the optimal solution would assign all tasks in M_X to $s_{max} = 1$ for all iterations of the while loop. After the completion of while loop, the remaining tasks are assigned optimal speed based on proposition 1. Therefore, the solution given by algorithm NoMinSpeed is optimal if $s_{min} = 0$.

VIII. Appendix B

Proposition 3: Let $M_X = \{i: s_i^* > s_{max}\}$ and $M_N = \{i: s_i^* < s_{min}\}$ where s_i^* is as defined in equation (14), then for the special case where $M_X = \emptyset$, the thermal utilization of the periodic task set is minimized only if:

$s_i = s_{min}\ \forall i \in M_N$

Proof: Suppose that we start solving the optimization problem by setting the speed of a single periodic task $\tau \in M_N$ to s_{min}. Then by proposition 1, we know that the optimal solution for remaining periodic tasks would try to do speed assignment such that

$$\bar{s}_i = A_i^{-1/3}\, G(\Gamma\backslash\tau) \Big/ \left(1 - \sum_{j\in\Gamma_X} \frac{C_j}{T_j s_j}\right) \ \forall i \in \{\Gamma\backslash\tau\}$$

Since periodic task τ was executed at a speed $s_\tau = s_{min} > s_\tau^*$, $\bar{s}_i < s_i^*\ \ \forall i \in \{\Gamma\backslash\tau\}$. Therefore, if we now compute set $\widehat{M}_N = \{i: \hat{s}_i^* < s_{min}\}$, we know that $\widehat{M}_N \supseteq M_N$. Furthermore, if we now compute set $\widehat{M}_X = \{i: \bar{s}_i > s_{max}\}$, we know that $\widehat{M}_X \subseteq M_X$. $\widehat{M}_X = \emptyset$ since $M_X = \emptyset$.

Therefore, the order in which the speeds of periodic tasks in M_N are assigned does not influence the solution. Furthermore, due to the convexity of the objective function $\left(\frac{1}{\Delta}\sum_{i=0}^{n-1}\frac{C_i P_i s_i^2}{T_i}\right)$ in terms of variable s, if $s_i^* \leq s_{min}$ for a given periodic task i, the optimal speed assignment would set the speed of periodic task i equal to s_{min}.

■

Proposition 4: Let $M_X = \{i: s_i^* > s_{max}\}$ and $M_N = \{i: s_i^* < s_{min}\}$ where s_i^* is as defined in equation (14), then for the special case where $M_N = \emptyset$, the thermal utilization of the periodic task set is minimized only if:

$s_i = s_{max}\ \forall i \in M_X$

Proof: Similar to proposition 3

■

Theorem 7 follows directly from proposition 3 and proposition 4. Furthermore, by proposition 3, $s_{min} < s_i < s_{max} \forall i \in \Gamma_{UX}$ in line 16 of SeCTUM. Therefore, proposition 1 may be used to assign speeds to all tasks in Γ_{UX}.