Guaranteed Scheduling For Repetitive Hard Real-Time Tasks Under The Maximal Temperature Constraint

Gang Quan Yan Zhang William Wiles Pei Pei
Department of Computer Science and Engineering
University of South Carolina
Columbia, SC 29208
(gquan,zhangy,wilesw,pei)@engr.sc.edu

ABSTRACT

We study the problem of scheduling *repetitive* real-time tasks with the Earliest Deadline First (EDF) policy that can guarantee the given maximal temperature constraint. We show that the traditional scheduling approach, i.e., to repeat the schedule that is feasible through the range of one hyper-period, does not apply any more. Then, we present necessary and sufficient conditions for real-time schedules to guarantee the maximal temperature constraint. Based on these conditions, a novel scheduling algorithm is proposed for developing the appropriate schedule that can ensure the maximal temperature guarantee. Finally, we use experiments to evaluate the performance of our approach.

Categories and Subject Descriptors

D.4.1 [**Algorithms**]: Software—*Operating Systems, Process Management, Scheduling*

General Terms

Algorithms, Reliability, Performance

Keywords

Real-time scheduling, thermal aware, energy consumption, maximal temperature

1. INTRODUCTION

For the past several decades, the processor's performance has increased exponentially. Catering to society's rapidly growing appetite of computing power, the processor's performance is expected to continuously grow dramatically in the future [6]. However, the soaring power consumption of these processors has posed significant challenges in the design of computing applications and systems. One such challenge is to reduce the energy consumption in order to extend the battery life for mobile devices. Another challenge concerns the management of the tremendous heat generated from the system components, especially processors themselves.

CODES+ISSS'08, October 19–24, 2008, Atlanta, Georgia, USA.

The escalating heat has directly led to high packaging and cooling costs. It is estimated that [13] the thermal packaging increases total package cost at 1-3 dollar per watt. With estimated peak power of future processors well over 300 watts in next decade [6], this makes it extremely expensive for industry to develop new generations of computing systems. Moreover, the elevated temperature due to large heat generation also has huge adverse impacts on the reliability of the computing systems or can even cause the system to fail. According to Yeh and Chu [19], even a small increase in the temperature (10^oC) can result in 50% reduction in the component life span. The severity of the problem is further highlighted by Intel's acknowledgement that it has hit a "thermal wall" [8].

While several impressive and novel cooling techniques and thermal materials have been developed recently (e.g. [20]), there has been increasing interest in both academy and industry to address the thermal challenges at the computer architecture and higher design levels [10, 12]. Several novel thermal aware computer architectures are proposed [13], which enable *Dynamic Thermal Management*(DTM) [11, 17], i.e., to control the temperature by adjusting the performance of the processor via mechanisms such as clock throttling and dynamic voltage scaling (DVS).

As a closely related problem, employing scheduling schemes to minimize the energy consumption (e.g. [18]) has attracted great research interests for the past decade. At first sight, since the higher power consumption usually compounds with the higher energy consumption and high temperature, it seems intuitive that power-aware scheduling techniques can be readily applied for the purpose of thermal-aware computing. Unfortunately, an optimal schedule that can minimize total energy consumption is not necessarily the optimal solution under maximal temperature constraints [1, 7].

We are interested in developing real-time scheduling techniques under thermal or temperature constraints. For traditional processors, when the maximal temperature constraint is violated, the odds are that they can still function albeit with degraded reliability and dependability. For some recent processors (e.g. [5]), however, the self-protection reactive controls may be invoked when the temperature exceeds a maximal limit, and thus cause tasks to miss their deadlines. It is therefore critical to guarantee the feasibility under the maximal temperature constraint to ensure the predicability of the real-time system during the design space exploration.

A number of thermal-aware scheduling algorithms have been published that offers different degrees of guarantee. Temperature-aware scheduling techniques such as those in [7, 4] intend to minimize the peak temperature but with no guarantee of the maximal temperature. The on-line scheduling algorithm proposed by Bansal et al. [1], as well as that proposed by Chen et al. [2], sets up the upper bound for the maximal temperature. However, they cannot guarantee the schedulability if the pre-defined maximal temperature is lower. There are also a number of other approaches, such as the off-line algorithm proposed in [1] and the algorithm described in [9], can guarantee the maximal temperature constraints when executing a specific job set or one copy of a task graph. The problem is that they cannot guarantee that the maximal temperature constraints can still be met if the job set or task graph is repetitively executed. As shown later, a speed schedule for a repetitively task set that can satisfy both the timing and the maximal temperature constraint through the range of the least common multiple of task periods is not necessarily feasible later in the schedule. Wang et al. [16, 15] considered using two processor speeds to schedule a hard real-time task set with maximal temperature constraints. To this effort, a processor runs at the highest possible speed until the temperature reaches the temperature threshold. Then the processor runs at a lower constant speed to maintain the temperature. By properly choosing the lower processor speed, they can guarantee that the temperature is always within the given maximal value. This approach does not take advantage of fact that reducing power consumption *helps* to reduce the temperature. Further, they only consider scheduling tasks according to the FIFO and the fixed priority policy. How to extend their results for EDF scheduling policy remains a problem.

In this paper, we study the problem to guarantee both the timing and temperature constraints for a periodic hard real-time task set scheduled according to EDF. We first present the system models and establish our problem formally (Section 2). We then examine the new characteristics of the scheduling problem with maximal temperature constraints (Section 3). Next we propose a novel algorithm under the maximal temperature constraint (Section 4) based on the properties developed in Section 3. We then use experiments to evaluate our proposed algorithm (Section 5) and draw our conclusions(Section 6).

2. SYSTEM MODELS

The real-time system model considered in this paper contains n independent periodic tasks, $\mathcal{T}$, with maximum temperature constraint T_{max}, scheduled according to the EDF policy. A task, τ_i, is characterized using three parameters,*i.e.*, $\tau_i = (p_i, d_i, c_i)$. p_i, d_i ($d_i \le p_i$), and c_i represent the period, the deadline and the worst case execution time for τ_i, respectively. Each task contains an infinite sequence of periodically arriving instances called *jobs*. We use τ_{ij} to represent the jth job from task i. For the DVS processor, we assume that power consumption is a convex function of speed, $P(s) \approx s^\alpha$ with $\alpha > 1$ and the processor speed, s, is proportional to the voltage (i.e., $s \propto V$). For the sake of simplicity, we assume $\alpha = 3$. The results from this paper can be easily extended for the cases when $\alpha \neq 3$.

The thermal model used in this paper is the same used by other researchers [1, 2, 16, 15]. Specifically, assuming a fixed ambient temperature, let $T(t)$ be the temperature at time t. Then we have

$$\frac{dT(t)}{dt} = aP(t) - bT(t), \tag{1}$$

where a, b are constants related to the efficiency of cooling mechanism for the processor, and $P(t)$ is the power consumption at time t. $T(t)$ is scaled so that the ambient temperature is fixed at zero. For details on the derivation of the parameters for this model, we refer the reader to [14]. Based on equation (1), two properties are presented [15] regarding to the processor speed and its temperature:

- Running the processor at constant speed s_C during a sufficiently large interval, the temperature will exponentially converge to T_C if

$$s_C = (\frac{bT_C}{a})^{\frac{1}{3}}, \tag{2}$$

- Let the temperature at t_0 be T_0. If we run the processor at a constant speed s_0 during the interval $[t_0, t]$, then the temperature at t, i.e., $T(t)$ is

$$T(t) = \frac{as_0^3}{b} + (T(t_0) - \frac{as_0^3}{b})e^{-b(t-t_0)}. \tag{3}$$

Based on these models, we formulate our problem as follows:

PROBLEM 1. *Given the hard real-time task set $\mathcal{T} = \{\tau_0, \tau_1, \cdots, \tau_{n-1}\}$, scheduled according to EDF on a variable voltage processor, let T_{max} represent the temperature threshold, determine the appropriate processor speed settings such that $\mathcal{T}$ can meet the required deadlines while keeping the temperature below T_{max} all the time.*

3. THE FEASIBILITY ANALYSIS

One common practice to ensure the feasibility of a periodic real-time task set is to construct a feasible schedule with interval $[0, L]$, where L represents the hyper-period, i.e. least common multiple (LCM) of the task periods. As long as the tasks are feasible in $[0, L]$, by replicating the schedule, the timing feasibility of the real-time system is guaranteed. However, when the execution of the real-time tasks are further constrained by a maximal temperature, is this approach still feasible? We first introduce Theorem 1 to address this question.

THEOREM 1. *Given periodic task set $\mathcal{T}$, let L be the hyper-period of $P_0, P_1, ..., P_{n-1}$, $\hat{S}(t)$ be the speed schedule within interval $[0, L]$ that can guarantee the deadlines of $\mathcal{T}$ under the maximal temperature constraints T_{max} with the initial temperature $T(0)$. Then, when repeating $\hat{S}(t)$ later in the schedule, all task deadlines can be guaranteed if $T(L) \le T(0)$.*

In Theorem 1, since the second hyperperiod will start with a lower (more favorable) initial temperature than the first one, repeating the schedule that is feasible during the first hyper-period is safe to guarantee the temperature and deadlines. The question that remains is: what if $T(L) > T(0)$? We present another theorem for this case.

THEOREM 2. *If $T(L) > T(0)$, when repeating $\hat{S}(t)$, all task deadlines can be guaranteed if and only if both the following conditions hold: (*i*) *Condition 1: $T(L) \le (T_{max} - T(0))(1 - e^{-bL}) + T(0)$; *and (*ii*)* Condition 2: $T(t_m) \le T_{max} - \frac{T(L)-T(0)}{1-e^{-bL}} e^{-bt_m}$ *for all $t_m \in [0, L]$ such that $T(t_m) \ge T(t), t \in [0, L]$.*

PROOF. For interval $[t_0, t_1]$, let the temperature at $t = t_0$ be $T(t_0)$. Then by solving equation (1), we have

$$T(t_1) = \int_{t_0}^{t_1} as^3(\tau)e^{-b(\tau - t_0)}d\tau + T(t_0)e^{-b(t_1-t_0)}. \tag{4}$$

So we have

$$T(L) = \int_0^L as^3(\tau)e^{-b\tau}d\tau + T(0)e^{-bL}. \quad (5)$$

If we repeat $\hat{S}(t)$ for interval $[L, 2L]$, we have

$$T(2L) = \int_L^{2L} as^3(\tau)e^{-b(\tau-L)}d\tau + T(L)e^{-bL} \quad (6)$$

$$= \int_0^L as^3(\tau-L)e^{-b\tau}d\tau + T(L)e^{-bL} \quad (7)$$

Note that $s^3(\tau) = s^3(\tau - L)$. Thus we have

$$T(2L) - T(L) = (T(L) - T(0))e^{-bL}. \quad (8)$$

So, for the $(k+1)$th LCM interval, we have

$$T((k+1)L) - T(kL) = (T(L) - T(0))e^{-kbL}. \quad (9)$$

When $T(L) > T(0)$, the temperatures at $t = 0, L, 2L, ...$ will be monotonically increasing. Also $T(L) - T(0), T(2L) - T(L), T(3L) - T(2L), ..., T((k+1)L, kL)$ forms a geometric series and we have

$$T((k+1)L) = T(0) + \frac{(T(L) - T(0))(1 - e^{-kbL})}{1 - e^{-bL}}. \quad (10)$$

As $k \to \infty$, we have

$$\lim_{k\to\infty} T(kL) = T(0) + \frac{(T(L) - T(0))}{1 - e^{-bL}}. \quad (11)$$

So, $T(kL) \leq T_{max}$ if and only if

$$T(L) \leq (T_{max} - T(0))(1 - e^{-bL}) + T(0). \quad (12)$$

In addition to the ending point of a hyper-period, we will need to ensure that the maximal temperature constraint is not violated within the hyperperiod. Let $t_m \in [0, L]$ such that $T(m) \geq T(t)$ for any $t \in [0, L]$. Let $t_{m'} \in [kL, (k+1)L]$ and $t'_m = t_m + kL$. We want to show that $T(t_{m'}) \leq T_{max}$ if and only if Condition 2 holds. Based on equation (4), similarly, we have

$$T(t_{m'}) = \int_{kL}^{t_{m'}} as^3(\tau)e^{-b(\tau-kL)}d\tau + T(kL)e^{-b(t_{m'}-kL)}, \quad (13)$$

and

$$T(t_m) = \int_0^{t_m} as^3(\tau)e^{-b\tau}d\tau + T(0)e^{-bt_m}. \quad (14)$$

Since $t_m = t_{m'} - kL$, so we have,

$$T(t_{m'}) = T(t_m) + \frac{(T(kL) - T(0))}{e^{bt_m}}. \quad (15)$$

Hence, from equation (11), we have $T(t_{m'}) \leq T_{max}$ if and only if

$$T(t_m) \leq T_{max} - \frac{T(L) - T(0)}{1 - e^{-bL}}e^{-bt_m}. \quad (16)$$

Theorem 1 and Theorem 2 provide the necessary and sufficient condition to predict if a schedule feasible within the first hyper-period is globally feasible. On the other hand, Theorem 2 also implies that not all schedules are feasible under the maximal temperature constraint even if they can guarantee the deadlines and maintain the maximal temperature below T_{max} during the first hyper-period, i.e., $[0, L]$. We believe that this is one of the most unique characteristics of temperature-constrained real-time scheduling problems, compared with others such as the power-aware scheduling problem.

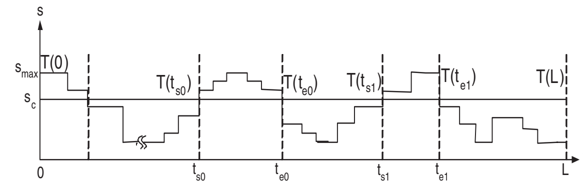

Figure 1: An example LPEDF schedule. Interval $[t_{s0}, t_{e0}]$ and $[t_{s1}, t_{e1}]$ are *island intervals.* Interval $[t_{e0}, t_{s1}]$ and $[t_{e1}, 0]$ are *non-island intervals*

4. SCHEDULING UNDER THE MAXIMAL TEMPERATURE CONSTRAINT

In this section, we develop two constructive scheduling algorithms in accordance with Theorem 1 and Theorem 2.As the lower energy consumption tends to lead to the lower temperature, our algorithms are developed based on the optimal energy-saving scheduling algorithm presented by Yao et al. [18] (We refer to it as Lowest Power EDF (LPEDF).) and update the speed schedule based on the temperature constraints. Before we present the algorithms, we first introduce the following concept.

DEFINITION 1. *For task set $\mathcal{T}$ and temperature threshold T_{max}, let $\tilde{S}(t)$ be the speed schedule for $\mathcal{T}$ based on LPEDF, and let s_C be the processor speed that maintains the processor temperature at T_{max} (according to equation (2)). An interval $[t_i, t_j]$ is called an* island interval *if for any $t \in [t_i, t_j]$, $\tilde{S}(t) \geq s_C$, or a* non-island interval *otherwise.*

For an island intervals, such as $[t_{s0}, t_{e0}]$ in Figure 1, reducing the processor speeds is not an option since it will cause the jobs within to miss deadlines. Therefore, we keep the speed schedule unchanged for each island interval. Note that it is possible to use multiple speeds in an island interval without violating the timing and temperature constraints. This, however, would require the processor run at an even higher processor speed which may not always be available. In addition, it would result in larger energy consumption, which *may* have negative impacts with regard to the given maximal temperature constraints. Nevertheless, how to judiciously update the processor speed for the island interval when possible without compromising the timing and maximal temperature constraints is an interesting problem that is worthy of further study.

For a *non-island interval*, such as $[t_{e0}, t_{s1}]$, from equation (2) and (3), as long as the starting temperature of the interval, e.g. $T(t_{e0})$, is not higher than T_{max}, the temperature will never exceed T_{max} in this interval. It is thus important to determining whether the maximal temperature within an island interval will exceed T_{max}. The following lemma indicates that, as long as the maximal temperature constraint is not violated, the highest temperature always occurs at the end of an island interval. This conclusion is useful in design of our temperature-constrained scheduling techniques. (The proof is omitted due to page limit.)

LEMMA 1. *For task set $\mathcal{T}$ and temperature threshold T_{max}, let $[t_1, t_2]$ be an island interval. If for any $t \in [t_1, t_2]$, we have $T(t) \leq T_{max}$, then $T(t) \leq T(t_2)$.*

4.1 $T(0) \geq T(L)$

When constructing the schedule, one question is how to determine the initial temperature, i.e. $T(0)$. One simple choice for $T(0)$ is to set $T(0) = T(L) = T_{max}$. For periodic task sets, this is not a good choice due to the following lemma.

LEMMA 2. *Given task set $\mathcal{T}$, the maximal temperature constraint T_{max}, and LPEDF schedule $\tilde{S}(t)$, the first island interval always starts at $t = 0$ and has the highest speed assignment.*

Algorithm 1 Function $TInit(I, T_m)$: Determine the maximal starting temperature of the island interval $I = [t_s, t_e]$ with the highest $T(t_e) \leq T_m$.

1: **Input:** $\tilde{S}(t), T_m, a, b, I$ where $\tilde{S}(t)$ is the LPEDF schedule, T_m is the maximal temperature, a,b are the thermal constants, and $I = [t_s, t_e]$ is the island interval;
2: **Output:** The maximal acceptable temperature at t_s, i.e., $T(t_s)$;
3: Let $l_1, l_2, ..., l_p$ be all the schedule intervals in $[t_s, t_e]$ from $\tilde{S}(t)$;
4: $T(t_e) = T_m$;
5: **for** l_i from l_p down to l_1 **do**
6: $\triangle t =$ the length of l_i;
7: $s =$ the speed of l_i;
8: $T(t_s) = \frac{T(t_e) - \frac{as^3}{b}}{e^{-b\triangle t}} + \frac{as^3}{b}$;
9: **if** $T(t_s) < 0$ **then**
10: return; //Temp. constraints cannot be satisfied
11: **else**
12: $T(t_e) = T(t_s)$;
13: **end if**
14: **end for**

This lemma can be proved based on Lemma 1 and the observation that, for a periodic task set (assuming all tasks are ready at $t = 0$), the first critical interval based on LPEDF always starts at $t = 0$ with the highest speed requirement. As such, if the maximal processor speed is higher than s_C, the maximal temperature constraint will always be violated if we set $T(0) = T_{max}$. On the other hand, if we choose $T(0) = 0$, this will limit the applicability of the result schedule, since the schedule will not be guaranteed as feasible whenever $T(0) > 0$. In what follows, we develop an algorithm (Algorithm 1) to determine the initial temperature.

Algorithm 1 starts from the end of the island interval and traverses in a backward fashion (line 5). In each sub interval with constant processor speed, the temperature at its starting point is computed based on a simple transformation of equation (3) (line 8). The result temperature $T(t_s)$ determines the highest acceptable temperature. If $T(t_s) < 0$, this implies that the temperature and timing constraints cannot be satisfied simultaneously. $T(0)$ is therefore is can be determined by applying Algorithm 1 on the first critical interval.

With $T(0)$, we now need to guarantee that the temperature does not exceed T_{max} in the first hyper-period. Given Lemma 1, we fix the temperature at the end of each island interval to be T_{max}. By doing so, we also fix the maximal temperature at the start of each island interval (based on Algorithm 1). After $T(0)$ and $T(L)$, as well as the temperatures at both ends of their island intervals are determined, the interval $[0, L]$ is divided into a series of consecutive intervals with junction temperatures defined. For island intervals, the speed schedules will remain the same as stated before. For a non-island interval $[t_s, t_e]$, we discuss our solutions in two cases: (1) $T(t_s) \leq T(t_e)$; (2) $T(t_s) > T(t_e)$.

Algorithm 2 The overall algorithm with $T(0) \geq T(L)$.

Input: $\tilde{S}, T_{max}, S_{max}, a, b, L$ where $\tilde{S}$ is the LPEDF schedule, T_{max} is the maximal temperature, S_{max} is the maximal processor speed, a,b are the thermal constants, and L is the hyper-period of the task periods;
Output: The new speed schedule $\hat{S}$;
$s_C = (\frac{bT_{max}}{a})^{\frac{1}{3}}$;
Identify all the island intervals $I_1, I_2, ..., I_k$;
Identify all the non-island interval $NI_1, NI_2, ..., NI_p$;
$T(0) = TInit(I_1, T_{max})$;
$T(L) = T(0)$;
for I from $I_2, ..., I_k$ **do**
$I = [t_s, t_e]$;
if $t_e = L$ **then**
$T(t_e) = T(L)$;
else
$T(t_e) = T_{max}$;
end if
$T(start(I)) = TInit(I, T(t_e))$;
$\hat{S}(I) = \tilde{S}(t)$;
end for
for $I = [t_s, t_e]$ from $NI_1, ..., NI_p$ **do**
if $T(t_s) \leq T(t_e)$ **then**
Set $\hat{S}(t) = S_{max}$ until $T(t) = T(t_e)$ and then $\hat{S}(t) = s_{t_e}$;
else
Determine the schedule based on Lemma 3;
end if
end for

If $T(t_s) \leq T(t_e)$, we can run the processor at the maximum speed until $T = T(t_e)$, at which point the speed will be reduced to s_{t_e}, the equilibrium speed of temperature constraint $T(t_e)$. If $T(t_s) > T(t_e)$, an optimal schedule that can maximize the workload within the interval is available [3]. However, this would require the processor speed to follow an exponentially variable curve and hard to be implemented in the practical processors. Our solution is thereby based on Lemma 3, to use a constant speed that can meet both $T(t_s)$ and $T(t_e)$ at $t = t_s$ and $t = t_e$, respectively, without exceeding T_{max} in the middle. The overall algorithm to guarantee the maximal temperature constraint is illustrated in Algorithm 2.

LEMMA 3. *For a non-island interval $[t_s, t_e]$ with $T(t_s) > T(t_e)$, let $\overline{s}$ be the constant speed during $[t_s, t_e]$. Then*

$$\overline{s} = (\frac{b(T(t_e) - T(t_s)e^{-b(t_e - t_s)})}{a(1 - e^{-b(t_e - t_s)})})^{\frac{1}{3}}. \qquad (17)$$

The schedule that uses $\hat{s} = min(\overline{s}, s_C)$ within interval $[t_s, t_e]$ guarantees $T(t_s)$, $T(t_e)$, and T_{max}.

4.2 $T(0) < T(L)$

The framework for the second algorithm is very similar to the first one. The difference is in the way we determine $T(0)$, $T(L)$, and the ending temperature for each island interval.

Consider $T(0)$, $T(L)$, and the first island interval $I_0 = [0, t_1]$. Based on Theorem 2, $T(0)$, $T(L)$, and $T(t_1)$ must satisfy the following equations:

$$T(L) \leq (T_{max} - T(0))(1 - e^{-bL}) + T(0), \qquad (18)$$

$$T(t_1) \leq T_{max} - \frac{T(L) - T(0)}{1 - e^{-bL}} e^{-bt_1}, \qquad (19)$$

Algorithm 3 Determine $T(0)$ and $T(L)$ with $T(0) < T(L)$.

1: **Input:** $\tilde{S}, T_{max}, a, b, L, I_0 = [0, t_1]$, and ϵ where $\tilde{S}$ is the LPEDF schedule, T_{max} is the maximal temperature, a,b are the thermal constants, L is the hyper-period of the task periods, $I_0 = [0, t_1]$ is the first island interval, and ϵ is the predefined error tolerance;
2: **Output:** $T(0)$, $T(L)$;
3: $T_L(0) = 0$;
4: $T_H(0) = TInit(I_0, T_{max})$;
5: **while** $T_H(0) - T_L(0) > \epsilon$ **do**
6: $T(0) = (T_H(0) - T_L(0))/2$;
7: $T(t_1) = \frac{as_0^3}{b} + (T(0) - \frac{as_0^3}{b})e^{-b(t)}$; //eq (3)
8: $T(L)_1 = (T_{max} - T(0))(1 - e^{-bL}) + T(0)$; //eq (18)
9: $T(L)_2 = (T_{max} - T(t_1))\frac{1-e^{-bL}}{e^{-bt_1}} + T(0)$; //eq (20)
10: $T(L) = min(T(L)_1, T(L)_2)$;
11: **if** $T(t_1) > T_{max}$ or $T(L) > T_{max}$ **then**
12: $T_H(0) = T(0)$;
13: **else**
14: $T_L(0) = T(0)$;
15: **end if**
16: **end while**
17: $T(0) = T_L(0)$;
18: $T(t_1) = \frac{as_0^3}{b} + (T(0) - \frac{as_0^3}{b})e^{-b(t)}$; //eq (3)
19: $T(L)_1 = (T_{max} - T(0))(1 - e^{-bL}) + T(0)$; //eq (18)
20: $T(L)_2 = (T_{max} - T(t_1))\frac{1-e^{-bL}}{e^{-bt_1}} + T(0)$; //eq (20)
21: $T(L) = min(T(L)_1, T(L)_2)$;

or, with a simple transformation,

$$T(L) \leq (T_{max} - T(t_1))\frac{1 - e^{-bL}}{e^{-bt_1}} + T(0) \qquad (20)$$

From equations (18)-(20), we developed Algorithm 3 to find the highest acceptable $T(0)$ and $T(L)$. Algorithm 3 uses a binary search method (line 5-18) to find the appropriate $T(0)$ and hence $T(L)$. With given initial temperature $T(0)$, $T(L)$ is constrained by two conditions, i.e., equation(18) and (20). The smaller one is chosen (line 12) to satisfy both conditions. Furthermore, when $T(0) < T(L)$, fixing the ending temperature at T_{max} is not enough. From LPEDF, since the position of each island interval is known already, after $T(0)$ and $T(L)$ are fixed, we then can set the ending temperature of each island interval based on Condition 2 in Theorem 2. After that, we can employ the similar strategies to determine the speed schedule for the non-island intervals. The overall algorithm is quite similar to Algorithm 2 and omitted.

5. EXPERIMENTAL RESULTS

In this section, we use experiments to investigate the performance of our proposed approach (we refer to it as **TCEDF** (Temperature-Constrained EDF scheduling).) As explained in section 1, no other existing approach (e.g. [1, 2, 9, 16]) can guarantee the maximal temperature constraint for the periodic EDF task set and thus be used for a fair comparison. After a close consideration, we chose LPEDF as the reference approach as LPEDF is the optimal algorithm for energy savings and therefore represents an excellent boundary test case. The original LPEDF cannot guarantee the feasibility for a periodic task set under the maximal temperature constraint. However, according to Theorem 1, as long as the temperature at $T(L) \leq T(0)$, the LPEDF approach is feasible.Note that not every speed schedule constructed based on TCEDF are feasible. To guarantee its feasibility, by Theorem 1 and 2, we only need to make sure that all deadlines

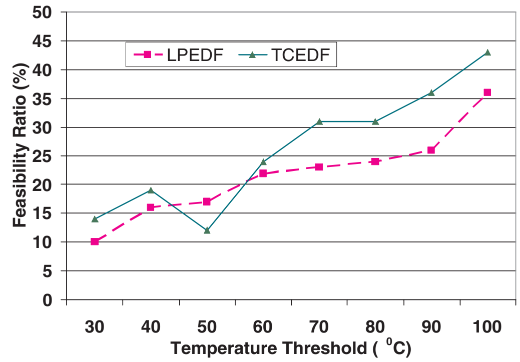

Figure 2: Feasibility ratios of LPEDF and TCEDF for two-task sets (r=0.3).

within the first hyperperiod can be satisfied, which can be easily tested using simulation.

In our experiments, we set $a = 0.03$, $b = 0.02$. The processor's voltage can vary between $[0, 1]V$ continuously. We also set the ambient temperature to be zero. The periodic task sets were randomly generated with periods uniformly distributed within $[1, 30]$ second. The deadlines are set to be less than the periods, with a constant ratio, i.e., $r(r < 1)$. The worst-case execution time for each task was also randomly generated. We varied the maximal temperature from 30^oC to 100^oC with interval of 10^oC. With given maximal temperature and a,b, we can immediately determine s_C based on equation (2). If the maximal processor speed for a generated task set is less than s_C, it is always feasible under the maximal thermal constraint (see Section 4). Therefore we only consider the case when the maximal speed is higher than s_C.

For each given temperature threshold, at least 100 task sets with maximal speed over s_C in their LPEDF schedules were generated. For each task set, TCEDF was first applied for the task set and used to determine the highest acceptable initial temperature. We then simulate the LPEDF using this initial temperature and check if it is feasible based on Theorem 1. In the case when TCEDF failed, the initial temperature was set to zero. The number of successful results by both TCEDF and LPEDF were recorded and compared. Figure 2 shows the results when each task set consists of two periodic tasks, and the deadline-period ratio, i.e. r, was set to 0.3. In Figure 3, each task set consists of three periodic tasks, with $r = 0.3$ and 0.8, respectively. From Figure 2 and Figure 3, we can see that, while LPEDF is the optimal schedule to minimize the energy consumption, it is not necessarily the optimal one under the maximal temperature constraints. In fact, as shown in Figure 2, TCEDF outperforms LPEDF in many cases. The reasons are two fold: First, TCEDF allows $T(L) > T(0)$ (section 4.2), which would be infeasible for LPEDF; Second, TCEDF can better manage non-island interval speeds than LPEDF. This is due to the fact that, when an island interval tends to build up a high temperature, TCEDF reduces the processor speed before the island interval and helps to reduce the temperature. The workload could still be finished in time if a higher speed than that by LPEDF can be applied after the island interval.

Also, from Figure 2 and Figure 3 when $r = 0.3$, we can see that as the temperature threshold increases, the LPEDF and

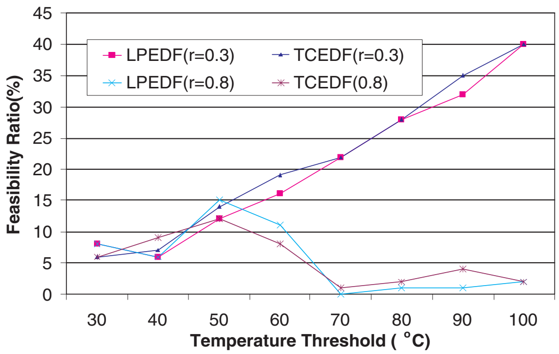

Figure 3: Feasibility ratios of LPEDF and TCEDF for three-task sets.

TCEDF success ratios increase in a similar fashion. This is expected because the initial starting temperature is more relaxed as the maximal temperature is set at a higher level. On the other hand, it is interesting to note the drop in success ratios by both LPEDF and TCEDF in Figure 3 when $r = 0.8$ as the maximal temperature increases. This is because as the maximal temperature goes higher, the maximal speed requirement for the eligible task set also increases since s_C becomes larger. Together with longer island intervals, this makes the temperature less likely to be controlled under the maximal temperature limit. Through these experiments, we can see there are some profound relationship between the schedulability, the initial temperature, the maximal temperature, and other factors in the temperature-constrained real-time scheduling problem. To study these relations and further refine TCEDF is an interesting problem and will be our future research.

6. SUMMARY

Temperature-aware computing plays an increasingly critical role in real-time system design, and the power-aware design techniques alone are inadequate to effectively address the temperature issues. We have made a number of major contributions in this paper: (1) new and interesting findings regarding to scheduling repetitive real-time tasks under the maximal temperature constraints and, correspondingly, the necessary and sufficient conditions to validate the feasibility of real-time schedules; (2) a novel scheduling approach to develop appropriate schedules that can ensure the temperature constraints; (3) experimental results that validate our approach and, at the same time, reveal the profound complexity in real-time scheduling with maximal temperature guarantee. While this paper is based on the traditional periodic task model and EDF scheme, the underlying observations and theories are rather general and can be readily extended to other scheduling models and scenarios, which will be our future work.

7. ACKNOWLEDGEMENT

This work is supported in part by NSF under Career Award CNS-0545913.

8. REFERENCES

[1] N. Bansal, T. Kimbrel, and K. Pruhs. Speed scaling to manage energy and temperature. *Journal of the ACM*, 54(1):1–39, 2007.

[2] J. Chen, C. Hung, and T. Kuo. On the minimization of the instantaneous temperature for periodic real-time tasks. *RTAS*, pages 236–248, 2007.

[3] A. Cohen, F. Finkelstein, A. Mendelson, R. Ronen, and D. Rudoy. On estimating optimal performance of cpu dynamic thermal management. *IEEE Computer Architecture Letter*, 2(1):6–9, 2003.

[4] A. K. Coskun, T. Rosing, and K. Whisnant. Temperature aware task scheduling in mpsocs. *DATE*, pages 1659–1664, 2007.

[5] Intel Corporation. Inel core 2 extreme quad-core processor qx6800 datasheet. 2007.

[6] ITRS. *International Technology Roadmap for Semiconductors (2005 Edition)*. International SEMATECH, Austin, TX., http://public.itrs.net/.

[7] Y. Liu, H. Yang, R. P.Dick, H. Wang, and L. Shang. Thermal vs energy optimization for dvfs-enabled processors in embedded systems. *ISQED*, 2007.

[8] J. Markoff. Intel's big shift after hitting technical wall. *New York Times*, 2004.

[9] S. Murali, A. Mutapcic, D. Atienza, R. Gupta, S. Boyd, and G. D. Micheli. Temperature-aware processor frequency assignment for mpsocs using convex optimization. *CODES+ISSS*, pages 111–116, 2007.

[10] L. Shang, L. Peh, A. Kumar, and N. Jha. Thermal modeling, characterization and management of on-chip networks. *ISM*, pages 67–78, 2004.

[11] K. Skadron. Hybrid architectural dynamic thermal management. *DATE*, pages 10–15, 2004.

[12] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware computer systems: opportunities and challenges. *IEEE Micro*, 23(6):52–61, 2003.

[13] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. *ICSA*, pages 2–13, 2003.

[14] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan. Temperature-aware microarchitecture: Modeling and implementation. *ACM Trans. Archit. Code Optim.*, 1(1):94–125, 2004.

[15] S. Wang and R. Bettati. Delay analysis in temperature-constrained hard real-time systems with general task arrivals. *RTSS*, pages 323–334, 2006.

[16] S. Wang and R. Bettati. Reactive speed control in temperature-constrained real-time systems. *ECRTS*, pages 161–170, 2006.

[17] A. Weissel and F. Bellosa. Dynamic thermal management for distributed systems. *TACS*, 2004.

[18] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced cpu energy. In *AFCS*, pages 374–382, 1995.

[19] L.-T. Yeh and R. C. Chu. *Thermal Management of Microelectronic Equipment: Heat Transfer Theory, Analysis Methods, and Design Practices*. ASME Press, New York, NY, 2002.

[20] C. Zweben. Ultrahigh-thermal-conductivity packaging materials. *SEMI-THERM*, pages 168–174, 2005.