Hyper-periodic Thermal Management for Hard Real-time Systems

Long Cheng†, Zhihao Zhao§, Kai Huang§ and Alois Knoll†
†Chair of Robotics and Embedded Systems, Technical University Munich, Germany
§School of Data and Computer Science, Sun Yat-Sen University
Email: †{chengl, knoll}@in.tum.de §zackzhao0@gmail.com, huangk36@mail.sysu.edu.cn

***Abstract*—This paper proposes hyper-period thermal management to minimize the peak temperature while guaranteeing hard real-time constraints for real-time systems. To establish the criterion of the optimal solution, a closed-form representation of the peak temperature is derived. With this formulation and the linear convex hull of an arrival curve, an approach which derives hyper-periodic thermal management schemes is proposed to minimize the peak temperature. Case studies show that our approach achieves lower peak temperature compared to previous work.**

I. Introduction

Nowadays, the exponentially increasing power density in current electronic processors has caused increasingly high temperature in embedded systems. The die temperature on current processors can easily reach 120 °C [1]. This high temperature of the circuits severely reduces the reliability of a system. Therefore, thermal management is essential in modern embedded system design.

To manage the temperature, a common method is using hardware cooling devices to remove heat from the processor. However, the packaging cost increases exponentially as the power dissipation of electronic devices increases. It is estimated that the packaging technology costs $3/watt of dissipated heat [2]. Besides, the increasing size of cooling devices limits the usability for hand-held devices. Therefore, alternative technologies which manage on-chip temperature by dynamically detecting and alleviating thermal violations have been adopted to improve the reliability and performance. Such technologies can be generally termed as Dynamic Thermal Management (DTM) techniques [3].

DTM techniques control the temperature by altering the power consumption of processors, as power consumption directly contributes to heat generation [4]. Power consumption of processors is sourced mostly from dynamic and leakage power consumption. To reduce dynamic and leakage power consumption, two main mechanisms have been employed, i.e., Dynamic Voltage Frequency Scaling (DVFS) and Dynamic Power Management (DPM), respectively [5]. The DVFS adjusts the supply voltage or frequency of the processor to reduce the dynamic power consumption such that the on-chip temperature does not go beyond the constraint. On the other hand, in DPM, the leakage current is diminished by transiting the processor to sleep state such that the temperature can be reduced. Since dynamic power dominated the power consumption in early processors [6], DVFS has been adopted widely in temperature management researches [1], [7]–[10].

The DPM technologies, however, is adopted in this paper. The reason is that the leakage power increases exponentially and becomes comparable or even greater than dynamic power [6] in recent processors since transistor technology is shifting toward sub-micron domains. It is reported that leakage power dominates the total power consumption in 32 nm or more advanced processors [11]. This indicates that DPM technologies can achieve a better efficiency in optimizing the temperature on modern processors. Therefore, this paper investigates the DPM technology which switches a processor between active and sleep modes in certain paces to minimize the peak temperature.

The subject of designing a DPM approach is to determine when and how long the processor should be switched to sleep mode. In [5] a thermally optimal stop-go scheduling, which is named JUst Sufficient Throttling (JUST), is developed to minimize the peak temperature within given makespan constraints. This scheduling can only handle static order tasks and is not intended for universal event arrivals. For irregularly arriving events, one option is transiting the processor to the sleep state according to the event arrivals. Kumar and et al. [2], presented an online/offline-combined approach to minimize the peak temperature for non-deterministic tasks and named it Cool Shaper. In order to reduce the peak temperature, this approach dynamically delays the execution of workload in runtime with a pre-computed shaper, whose parameters are offline computed. Since erratically transiting the processor makes the prediction of the thermal behavior almost impossible, the peak temperature is obtained by offline simulation, which requires considerable calculation effort. In addition, since the performance of the processor is reduced when DPM technologies are deployed, the execution time penalty should be considered carefully to ensure that all events finish within their deadlines. This evaluation is also difficult to complete due to the unpredictability in power state transition. To simplify the predications towards temperature evolution and execution time penalty, one can put the processor to sleep state in a pre-designed way. Masud Ahmed et al. [4] presented an offline algorithm which utilizes thermal-aware periodic resources to minimize the peak temperature for sporadic tasks. Moreover, Cheng et al. [12] proposed periodic thermal management to reduce peak temperature for general event arrivals. Although periodically putting the processor to a lower power state allows us forecast the processor behaviours offline, the processor behavior is restricted by the rigid periodic pattern which cannot fully exploit the idle time when events arrives at a lower rate.

In this paper, we explore how to apply hyper-periodic DPM to optimize the peak temperature for general event arrivals while the deadlines constraints are guaranteed. Compared to periodic thermal management, hyper-period thermal management can relax the strictness in periodic policies and offer a tighter approximation to the required bound such that a lower peak temperature can be achieved. The hyper-period is derived offline in order to achieve minimum runtime overhead. A single core processor that has two power dissipation modes, ‘active’ and ‘sleep’ mode, is considered. For the purpose of modeling the non-deterministic event

978-1-5386-3166-9/17/$31.00

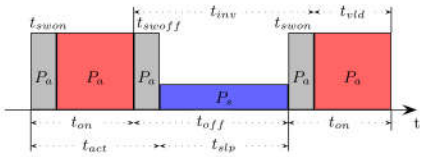

Fig. 1: Hardware model

arrivals and service provided by the processors, the arrival curve model [13] is employed. The detailed contributions of this paper are as follows:

- The closed-form solution of the peak temperature w.r.t. the hyper-periodic thermal management is developed.
- Based on the closed form and a linear convex-hull approximation of the service bound, an approach is developed to derive hyper-periodic thermal management. The generated scheme contains multiple on/off periods within one hyper-period.
- Simulations are made to study the effectiveness of our approach and the results are compared to that of related work in the literature. Our approach can achieve lower peak temperature compared to the periodic thermal management.

The rest of this paper is organized as follows. Section II presents system models. Section III defines the problem. Section IV derives the closed-form solution of the peak temperature. Section V presents our HPTM algorithms. Several cases are studied in Section VI and Section VII concludes.

II. System Model

A. Hardware Model

In this paper, we consider a single core processor which has two power dissipation modes, i.e., 'active' and 'sleep' mode. The workload is handled with a constant speed only when the processor is in 'active' mode with power consumption P_a. When there is no event to serve, the processor can be turned to 'sleep' mode with a lower power consumption P_s. The time and power overheads during mode-switching are also considered in this paper. t_{swoff} and t_{swon} units of time are required to switch the processor from 'active' model to 'sleep' model and back. Fig. 1 graphically illustrates the hardware model. The processor consumes power P_a during mode-switching but offers no service. Due to the limit of mode-switching, t_{off} and t_{on}, which denote the time length for which the processor is turned to 'sleep' and 'active' mode, must be larger than t_{swoff} and t_{swon} to cover the mode-switching overhead, this yields:

$$t_{off} > t_{swoff} \quad , \quad t_{on} > t_{swon} \tag{1}$$

B. Computational Model

The arrival curve $\bar{\alpha}(\Delta) = [\bar{\alpha}^u(\Delta), \bar{\alpha}^l(\Delta)]$ is introduced to model general event arrivals [13], [14]. The upper arrival curve $\bar{\alpha}^u(\Delta)$ and the lower arrival curve $\bar{\alpha}^l(\Delta)$ are the upper and lower bound of $R(t)$:

$$\bar{\alpha}^l(\Delta) \leq R(t) - R(s) \leq \bar{\alpha}^u(\Delta), \forall t - s = \Delta \tag{2}$$

where $R(t)$ represents the number of events that arrives at the processor in time interval $[0,t)$. The concept of arrival curve unifies many other common timing models of event streams.

Similarly, the service curve $\beta(\Delta) = [\beta^u(\Delta), \beta^l(\Delta)]$ is adopted to model the resource providing capability of the processor [14]. Analogously, the upper service curve $\beta^u(\Delta)$ and lower service curve $\beta^l(\Delta)$ satisfy:

$$\beta^l(\Delta) \leq C(t) - C(s) \leq \beta^u(\Delta), \forall t - s = \Delta \tag{3}$$

where $C(t)$ is the number of total time slots that the processor provides to handle the workload in time interval $[0,t)$.

$\bar{\alpha}(\Delta)$ is event-based and specifies the bounds of the input events number in any time interval Δ, while $\beta(\Delta)$ is time-based and specifies the bounds of the amount of available execution time in any time interval Δ. Thus, $\bar{\alpha}(\Delta)$ should be transformed to time-based arrival curve $\alpha(\Delta)$. Suppose that the worst-case execution time of one event in the arrival stream is c, then the arrival curve transformation can be performed as $\alpha^u(\Delta) = c \cdot \bar{\alpha}^u(\Delta)$ and $\alpha^l(\Delta) = c \cdot \bar{\alpha}^l(\Delta)$ [15].

Now, for N-event streams, where $N \geq 2$, the event streams $S_1, S_2, \cdots, S_N$ are ordered according to their deadlines such that D_i, the relative deadline of event stream S_i, is smaller than that of S_j when $i < j$. For simplicity, we depict the input by tuple $\mathbf{EM}(N) = (\bar{\alpha}_1(\Delta), c_1, D_1, \cdots, \bar{\alpha}_N(\Delta), c_N, D_N)$. Now, for the workload modeled by $\mathbf{EM}(N)$, the worse-case deadline constraints can be satisfied by a processor with service curve $\beta(\Delta)$ if the following condition holds:

$$\beta^l(\Delta) \geq \beta_B(\Delta), \forall \Delta \geq 0. \tag{4}$$

where $\beta_B(\Delta)$ is the service bound and is calculated based on the input and scheduling policy. Suppose the scheduling policy is earliest deadline first (EDF), then the service bound for the N-event streams can be formulated as [15]:

$$\beta_B(\Delta) = \sum_{i=1}^{N} \alpha_i^u(\Delta - D_i). \tag{5}$$

Note that EDF isn't necessarily the only scheduling policy can be adopted here. For example, when fixed priority (FP) scheduling is employed, the service bound can be formulated with the backward approach used in [16] and fits in with our approach as well as EDF.

To reduce the complexity, the linear convex hull is employed to approximate the service bound. A linear convex hull of a curve is basically the tightest convex approximation of the curve by linear segments. The linear convex hull is defined in the following.

Def. 1: In the Cartesian coordinate system, we denote a straight line which passes through point $\nu = (x_\nu, y_\nu)$ with slope ρ as a 2-tuple $\langle \nu, \rho \rangle$ where $x_\nu, y_\nu, \rho \in \mathbb{R}_{\geq 0}$

Def. 2 (Linear convex hull): A linear convex hull for a service bound is defined as segments

$$\tilde{\beta}(\Delta) = \min_{1 \leq i \leq m} \{\langle \nu_i, \rho_i \rangle\} \tag{6}$$

where $\rho_i > \rho_{i+1}$, $x_{\nu_i} < x_{\nu_{i+1}}$ and $y_{\nu_i} \leq y_{\nu_{i+1}}$ for all $1 \leq i < m$.

An example can be found in Fig. 2, in which the service bound $\beta_B(\Delta)$ is plotted as dash lines and the linear convex hull is the black solid segments. For a given $\beta_B(\Delta)$, the linear convex-hull can be derived by the algorithm presented in [17].

C. Thermal Model

The classic temperature model, which is based on the Fourier law of heating [18], is adopted in this paper.

$$C\frac{dT}{dt} = P - G(T - T_{amb}) \tag{7}$$

where T, C, G and P denote the temperature, thermal capacitance, thermal conductance, and power dissipation of the processor, respectively. The ambient temperature is denoted as T_{amb}. Moreover, the unit of all temperature variables is set as the absolute temperature (Kelvin, K). We assume the power dissipation has a linear relationship with respect to the temperature, similar to [2], [18]. Then, the power dissipation is formulated as $P = \varphi T + \theta$, where φ and θ are constants. Rewriting (7) yields $\frac{dT}{dt} = -mT + n$, where $m = \frac{G-\varphi}{C}, n = \frac{\theta + GT_{amb}}{C}$. Then, we can have the steady-state temperature of the currently power state as $T^{\infty} = n/m$ with $\frac{dT}{dt} = 0$. Since m and n are constants, the closed-form solution of the temperature is formulated as:

$$T(t) = T^{\infty} + (T_{init} - T^{\infty}) \cdot e^{-m \cdot t} \tag{8}$$

where T_{init} indicates the initial temperature of the processor.

Now we take the two power consumption modes into thermal model. Since the power dissipation during mode-switching equals the consumption in 'active' mode, the power dissipation is $P_a = \varphi_a T(t) + \theta_a$ when the processor is in 'active' mode or mode-switching, otherwise the power consumption is $P_s = \varphi_s T(t) + \theta_s$. Finally, the coefficients in Eqn. 8 are formulated as:

$$T_a^{\infty} = \frac{\theta_a + GT_{amb}}{G - \varphi_a}, m_a = \frac{G - \varphi_a}{C}, T_s^{\infty} = \frac{\theta_s + GT_{amb}}{G - \varphi_s} \quad , \quad m_s = \frac{G - \varphi_s}{C} \tag{9}$$

Moreover, we also assume the following properties hold for the thermal model.

- $m_a > 0$ and $m_s > 0$.
- The steady-state temperature in 'active' mode is not smaller than that in 'sleep' mode, that is, $T_a^{\infty} \geq T_s^{\infty}$.
- The initial temperature $T_{init} = T_{amb} \leq T_s^{\infty}$.

For brevity, we specify the thermal mode of the processor by the tuple $\mathbf{TM} = (T_a^{\infty}, m_a, T_s^{\infty}, m_s)$.

III. Problem Statement

Originated from periodic thermal management (PTM), HPTM combines several periods of different patterns of PTM together to form a hyper-period and controls the processor by repeating those PTM schemes in one hyper-period. Due to the problem complexity, we first consider the hyper-period formed based on two different patterns of PTM in this paper and plan to extend our HPTM schemes with more patterns in future researches. The following example illustrates the intuition of HPTM.

In this example, the thermal model parameters are depicted in Tab. I. The periodic management policy also adopts the linear convex hull. The linear convex hull, $\tilde{\beta}_B(\Delta)$, is set as the black solid line in Fig.2. To satisfy the deadline constraints, the lower service curve derived should be no less than $\tilde{\beta}_B(\Delta)$. For a given $t_{off} = 3ms$, Fig.2 demonstrates the corresponding PTM and HPTM schemes. As shown in the figure, although derived based on the same bound $\tilde{\beta}(\Delta)$, the PTM and the HPTM offer different lower service curves. Due to the strictness of periodic pattern, PTM cannot fully exploit the lower events coming rate after a burst happens, thus causing the gap between the service curve and $\tilde{\beta}_B(\Delta)$ widen as Δ increases. On the other hand, the HPTM, which includes two different patterns of PTM in the hyper-period of itself, can better utilize the lower events coming rate and narrow the gap, thus offering a tighter lower service curve to the service bound. Based on the results in section IV, the peak temperature for the PTM and HPTM schemes can be calculated as 373.4 K, 367.6K, which proves our observation.

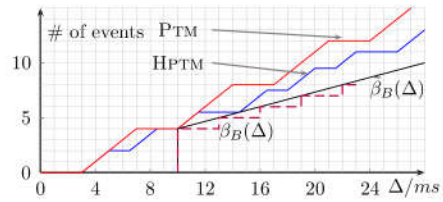

Fig. 2: The lower service curves derived by PTM and HPTM schemes.

A. Formation of HPTM

We first define some notations. A PTM is specified by a tuple $< t_{on}, t_{off} >$, where t_{on} and t_{off} denote the time interval for which the processor is switched to 'active' and 'sleep' mode, respectively. In this paper, a HPTM scheme is generated from two patterns of PTM. For brevity, we denote the certain PTM which has the larger t_{off} as PTM_2 and the other as PTM_1. That is,

$$t_{off}^1 \leq t_{off}^2 \tag{10}$$

The corresponding numbers of PTM_2 and PTM_1 in one hyper-period are denoted as N_2 and N_1, respectively. Moreover, since the processor should be turned on first from the initial state, we assume that PTM_1 and PTM_2 are combined together in a style, which is denoted as $\wedge$, so that the processor is firstly turned on and then off in each interval period of the hyper-period. Therefore, a HPTM can be specified by the tuple HPTM (N_1, PTM_1, N_2, PTM_2, $\wedge$).

Now, one should further determine if $t_{on}^1 \geq t_{on}^2$ or $t_{on}^1 \leq t_{on}^2$ to derive the lower service curve. For simplicity, we assume that t_{on}^1 is always not bigger than t_{on}^2 or not smaller than t_{on}^2. Otherwise, our problem will be too complicated to analyze. First, suppose $t_{on}^1 \geq t_{on}^2$. Since $t_{off}^1 \leq t_{off}^2$, then PTM_2 has a lower long term slope and the lower service curve begins with N_2 periods of PTM_2 and then N_1 periods of PTM_1. This kind of lower service curve doesn't match the shape of general service bound very well, due to the fact that the lower slope section of a service bound usually comes after a burst, which demands a higher slope PTM. Therefore, we consider $t_{on}^1 \leq t_{on}^2$.

Finally, we discuss the value of N_2. For different values of N_2, the scheme to derive the lower service curve is also different. In addition, the complexity of the scheme increases when N_2 grows. From experimental results, we also notice that the result improvement from a bigger N_2 is slight. Therefore, we restrict ourselves to the scenario of $N_2 = 1$ in this paper for simplicity of analysis. In conclusion, we study how to minimize the peak temperature under deadline constraints with the HPTM schemes holding these regulations:

$$t_{off}^1 \leq t_{off}^2, t_{on}^1 \leq t_{on}^2, N_2 = 1 \tag{11}$$

For this kind of HPTM schemes, after point $(t_{off}^2, 0)$, the lower service curve repeats the hyper-period which is N_1 consecutive periods of PTM_1 followed by one period of PTM_2, as shown in Fig. 2 and Fig. 4.

B. problem statement

We consider the temperature varying in a time interval L, where $L >> t_h$ and L/t_h is an integer, t_h indicates the length of the hyper-period. Due to the time overhead of mode-switching, the (t_{on}^i, t_{off}^i) cannot be directly utilized in thermal and service curve analysis, where $i = 1, 2$. Therefore we define that: t_{act}^i and t_{slp}^i denote the time intervals in which the processor consumes

power P_a and P_s in a period of PTM$_i$. Analogously, t^i_{vld} and t^i_{inv} denote the time intervals in which the processor can serve and not serve coming events in the period. Based on hardware model, they are formulated as:

$$t^i_{act} = t^i_{on} + t_{swoff} \quad , \quad t^i_{slp} = t^i_{off} - t_{swoff} \tag{12}$$

$$t^i_{vld} = t^i_{on} - t_{swon} \quad , \quad t^i_{inv} = t^i_{off} + t_{swon} \tag{13}$$

where $i = 1, 2$. Now our problem is defined as:
*Given a system characterized by the hardware model and the thermal model **TM** described above, task streams that are modeled by **EM**(N), our goal is to derive a hyper-periodic thermal management* HPTM *(N_1,* PTM$_1$*, 1,* PTM$_2$*, $\wedge$) where $t^2_{off} \geq t^1_{off}$ and $t^2_{on} \geq t^1_{on}$, such that the peak temperature is minimized while all the deadline constraints can be met.*

IV. Peak Temperature Analysis

Before presenting the formula for calculating the peak temperature of HPTM, we show some basic lemmas that can help us understand how the temperature evolves with a HPTM. First, some results of PTM thermal analysis in [12] are presented.

- Given a periodic thermal management PTM (t_{on}, t_{off}), the peak temperature of the processor is:

$$T^\star = \lambda T_a^\infty + (1-\lambda) T_s^\infty, \tag{14}$$

  where $\lambda = \frac{1-e^{-m_a t_{act}}}{1-e^{-m_a t_{act} - m_s t_{slp}}}$.

- Denoting the local maximal temperature in the jth period of PTM as T_j, the temperature difference between two consecutive local maximums, $T_{j+1} - T_j$, is formulated as:

$$T_{j+1} - T_j = (T^\star - T_j)(1 - e^{-m_a t_{act} - m_s t_{slp}}) \tag{15}$$

With these results we state the following lemma.

Lem. 1: With a hyper-period thermal management HPTM (N_1, PTM$_1$, 1, PTM$_2$, $\wedge$) where $t^2_{off} \geq t^1_{off}$ and $t^2_{on} \geq t^1_{on}$, when the temperature varies in a steady range, the temperature T in one hyper-period reaches the local peak temperature at the end of t^2_{act}.

Proof: First, some notations are introduced for brevity. $T_1^\star$ and $T_2^\star$ denote the peak temperature of PTM$_1$ and PTM$_2$, respectively. To offer bounds to the temperature, we construct other two PTM schemes, i.e., PTM$_3$(t^1_{on},t^2_{off}) and PTM$_4$(t^2_{on},t^1_{off}). Their peak temperatures are denoted as $T_3^\star$ and $T_4^\star$, respectively. As $t^2_{off} \geq t^1_{off}$ and $t^2_{on} \geq t^1_{on}$, we can easily get that

$$\begin{aligned} T_3^\star &\leq T_1^\star \leq T_4^\star \\ T_3^\star &\leq T_2^\star \leq T_4^\star \end{aligned} \tag{16}$$

Since $T_3^\star$ or $T_4^\star$ can only be reached by the purely periodic PTM (t^1_{on},t^2_{off}) or PTM (t^2_{on},t^1_{off}), we directly know when the temperature varies in a steady range, the range can be bounded by: $T_3^\star < T < T_4^\star$.

Fig.3 demonstrates the temperature evolution with a HPTM. We mark seven key points in the figure with capital letters 'A' to 'G', respectively. The corresponding temperatures at these seven points are denoted as T_a to T_g, respectively.

To find the maximum temperature in one hyper-period, we analyze the temperature evolution by three time intervals: 'A' to 'C', 'C' to 'E' and 'E' to 'G'. From Eqn. 15, two equations can immediately be obtained:

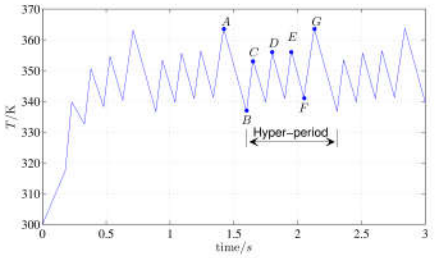

Fig. 3: The temperature evolution of the processor with the parameters described in Tab.1 for HPTM (3,PTM ($0.05s$,$0.1s$),1,PTM ($0.08s$,$0.18s$),$\wedge$)

$$\begin{aligned} T_c - T_a &= (T_3^\star - T_a)(1 - e^{-m_a t^2_{act} - m_s t^1_{slp}}) \\ T_g - T_e &= (T_4^\star - T_e)(1 - e^{-m_a t^1_{act} - m_s t^2_{slp}}). \end{aligned} \tag{17}$$

Since $T_3^\star \leq T \leq T_4^\star$ and $e^x < 1$ for all $x < 0$, the following inequalities can be derived consequently:

$$T_c \leq T_a \quad , \quad T_e \leq T_g. \tag{18}$$

As we mentioned earlier in this proof, the temperature varies in a steady range, which is tightly upper bounded by the maximum temperatures in every hyper-period. Since the range is steady, the maximum temperatures in different hyper-periods should be converged to the same value. Therefore, $T_a = T_g$ and the inequality presented below yields.

$$\max(T_c, T_e) \leq T_g. \tag{19}$$

For the time interval from 'C' to 'E', the local temperature maxima within it, such as T_c, T_d, T_e in the figure, will either be monotonously non-decreasing if $T_c \leq T_1^\star$ or otherwise monotonously non-increasing. In either case, all the local temperature maxima are not more than $max(T_c, T_e)$. From Eqn. 19, one can consequently see that they are also not more than T_g. Therefore we conclude that all the local temperature maxima in one hyper-period are not more than T_g, which is the temperature at the end of t^2_{act}. This proof is suitable for an arbitrary HPTM scheme. Thus the lemma is proved. ■

Denoting the local peak temperature in the jth hyper-period as T^g_j, then based on Lem. 1, the peak temperature $T^\star$ can be defined as the maximum of all the T^g_j:

$$T^\star = \max(T^g_1, \cdots, T^g_{\frac{L}{t_h}}) \tag{20}$$

To derive the closed form of $T^\star$, we first present another basic result for Periodic Thermal Management to give an insight into HPTM peak temperature derivation.

Lem. 2: Denote the temperature at the beginning of t_{act} in the jth period of a PTM scheme as T^l_j , then T^l_{j+N} can be represented by T^l_j as the following equation:

$$T^l_{j+N} = \widetilde{T^\star}[1 - e^{(-m_a t_{act} - m_s t_{slp})N}] + T_j e^{(-m_a t_{act} - m_s t_{slp})N}. \tag{21}$$

where $\widetilde{T^\star} = \widetilde{\lambda} T_s^\infty + (1-\widetilde{\lambda}) T_a^\infty$ and $\widetilde{\lambda} = \frac{1-e^{-m_s t_{slp}}}{1-e^{-m_a t_{act} - m_s t_{slp}}}$.

Proof: Similar with the proof of Eqn. 15 in [12], we can obtain the same result for T^l_j:

$$T^l_{j+1} - T^l_j = (\widetilde{T^\star} - T^l_j)(1 - e^{-m_a t_{act} - m_s t_{slp}}). \tag{22}$$

Then we create a sequence Y_j and let $Y_j = T^l_{j+1} - T^l_j$. Thence, based on Eqn.22, we have $Y_{j+1} - Y_j = (1-e^{-m_a t_{act} - m_s t_{slp}})(T^l_j - T^l_{j+1}) = (e^{-m_a t_{act} - m_s t_{slp}} - 1)Y_j$, which yields:

$$Y_{j+1} = e^{-m_a t_{act} - m_s t_{slp}} Y_j \tag{23}$$

Thus, Y_j is a geometric sequence. Therefore, we have:

$$\begin{aligned} T^l_{j+N} - T^l_j &= Y_{j+N-1} + Y_{j+N-2} + \cdots + Y_j \\ &= \frac{Y_j(1-e^{(-m_a t_{act} - m_s t_{slp})N})}{1-e^{-m_a t_{act} - m_s t_{slp}}} \\ &= \widetilde{T^\star}[1-e^{(-m_a t_{act} - m_s t_{slp})N}] - T^l_j[1-e^{(-m_a t_{act} - m_s t_{slp})N}] \end{aligned} \tag{24}$$

Adding T^l_j at both sides of Eqn.24 yields $T^l_{j+N} = \widetilde{T^\star}[1 - e^{(-m_a t_{act} - m_s t_{slp})N}] + T_j e^{(-m_a t_{act} - m_s t_{slp})N}$. ■

With above lemmas, the first main result of this paper is presented as the theorem below.

Thm. 1: Given a processor with the hardware model and thermal model described in this paper and a hyper-period thermal management HPTM (N_1,PTM$_1$,1,PTM$_2$,$\wedge$) where $t^2_{off} \geq t^1_{off}$ and $t^2_{on} \geq t^1_{on}$. The peak temperature of the processor during the time interval L is:

$$T^\star = T^\infty_a + [\varpi \widetilde{T^\star_1} + (1-\varpi)\widetilde{T^\star_2} - T^\infty_a]e^{-m_a t^2_{act}} \tag{25}$$

where $\varpi = \frac{1-e^{-S_1 N_1}}{1-e^{-S_1 N_1 - S_2}}$, $S_1 = m_a t^1_{act} + m_s t^1_{slp}$, $S_2 = m_a t^2_{act} + m_s t^2_{slp}$.

Proof: Denote the temperature at the beginning of the N_1 periods of PTM$_1$ and the beginning of t^2_{act} in the jth hyper-period as T^b_j and T^f_j, respectively. For example, the T_b and T_f in Fig. 3 are these kinds of temperatures, respectively. Then based on Lem. 2, we have:

$$T^f_j = \widetilde{T^\star_1}(1-e^{-S_1 N_1}) + T^b_j e^{-S_1 N_1} \tag{26}$$

$$T^b_j = \widetilde{T^\star_2}(1-e^{-S_2}) + T^f_{j-1} e^{-S_2} \tag{27}$$

Substitute the T^b_j in Eqn. 27 for that in Eqn. 26, the next equation yields.

$$\begin{aligned} T^f_j - T^f_{j-1} &= \widetilde{T^\star_1}(1-e^{-S_1 N_1}) + e^{-S_1 N_1}(1-e^{-S_2})\widetilde{T^\star_2} \\ &\quad - T^f_{j-1}(1-e^{-S_1 N_1 - S_2}) \\ &= (1-e^{-S_1 N_1 - S_2})[\varpi \widetilde{T^\star_1} + (1-\varpi)\widetilde{T^\star_2} - T^f_{j-1}] \end{aligned} \tag{28}$$

Since the peak temperature that we want to formulate is the local peak temperature when the overall temperature evolution is steady, that is, the temperatures at the points that in different hyper-periods but have the same location in its own hyper-period have the same value. Therefore $T^f_j - T^f_{j-1} = 0$, we can gain that the steady temperature for T^f_js is $T^f_{sty} = \varpi \widetilde{T^\star_1} + (1-\varpi)\widetilde{T^\star_2}$. From Eqn. 20, the peak temperature is the maximum of T^g_j for $j = \{1,2,\cdots,\frac{L}{t_h}\}$. When the overall temperature evolution is steady, the T^g_js also have a same value, T^g_{sty}, which is exactly the maximum of T^g_j, or in other words,

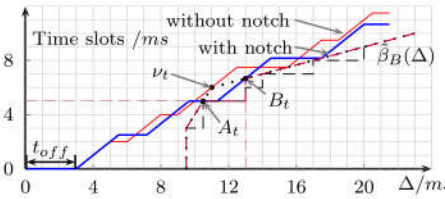

Fig. 4: The linear convex hull with a notch of $\beta_B(\Delta)$ when $t_{off} = 3ms$

the overall peak temperature $T^\star$. Based on T^f_{sty} and Eqn. 8, the peak temperature $T^\star$, i.e., T^g_{sty} , can be formulated as:

$$\begin{aligned} T^\star &= T^\infty_a + [T^f_{sty} - T^\infty_a]e^{-m_a t^2_{act}} \\ &= T^\infty_a + [\varpi \widetilde{T^\star_1} + (1-\varpi)\widetilde{T^\star_2} - T^\infty_a]e^{-m_a t^2_{act}} \end{aligned} \tag{29}$$

■

V. Peak Temperature Optimization

In this section, we present a series of analysis on how to generate the HPTM schemes while guaranteeing real-time constraints. We first present the linear convex hull with a notch.

A. Compensated $\tilde{\beta}(\Delta)$ with a notch

To improve the accuracy of our approach, we compensate the linear convex hull, $\tilde{\beta}(\Delta)$ in this section. We first introduce the tangent point, which is crucial to the derivation of HPTM. For a given $\tilde{\beta}(\Delta)$ and a given t^2_{off}, we can first compute a tangent line which originates from (t^2_{off}, 0) to $\tilde{\beta}(\Delta)$. Then the tangent point $\nu_t = (x_{\nu_t}, y_{\nu_t})$, where this tangent line and the linear convex hull meet, should be computed to generate the HPTM. The detailed definition of ν_t is presented in [17]. As shown in Fig. 4, point ν_t may cause a large deviation from $\tilde{\beta}(\Delta)$. To narrow this deviation, we compensate $\tilde{\beta}(\Delta)$ with a notch to eliminate ν_t. It is worth noting that the notch just introduces a slight increase in computation expense while the results can be improved considerably. As Fig. 4 shows, the notch of $\tilde{\beta}(\Delta)$ is generated by a horizontal line passing through a point $A_t = (x_{A_t}, y_{A_t})$ and a vertical line passing through a point $B_t = (x_{B_t}, y_{B_t})$ with $x_{B_t} > x_{A_t}$. The other parts of the linear convex hull is untouched. Then the linear convex hull compensated with a notch is obtained and can be defined as:

Def. 3 (Linear convex hull with a notch): Given $\tilde{\beta}(\Delta)$ and ν_t, a linear convex hull with a notch is defined as:

$$\check{\beta}(\Delta) = \begin{cases} \tilde{\beta}(\Delta) & \text{if } 0 \leq \Delta < x_{A_t} \text{ or } \Delta \geq x_{B_t} \\ y_{A_t} & \text{if } x_{A_t} \leq \Delta < x_{B_t} \end{cases} \tag{30}$$

where

$$\begin{aligned} y_{A_t} = \beta_B(x_{\nu_t}) \quad , \quad x_{A_t} = \{x \mid \tilde{\beta}(x) = y_{A_t}\} \\ x_{B_t} = \sup\{x \mid \beta_B(x) = y_{A_t}\} \quad , \quad y_{B_t} = \tilde{\beta}(x_{B_t}) \end{aligned} \tag{31}$$

Moreover, according to this definition, we have the following inequality:

$$x_{A_t} \leq x_{\nu_t} \leq x_{B_t} \tag{32}$$

B. HPTM Algorithms

In this section, the feasible region of t^2_{off} is first determined. Then, we show how to get the optimal solution for a given t^2_{off}. Finally, an algorithm is given to offer the best t^2_{off}, i.e., the optimal solution.

1) The feasible region of t^2_{off}: The feasible region of t^2_{off} should be determined first of all to guarantee that the solution to $< N_1, \text{PTM}_1, t^2_{on} >$ exists. For example, when $t^2_{off} = D$, worst-case workload will miss their deadlines even if t^2_{on} equals infinity. Based on Eqn. 1, we have $t^2_{off} \geq t_{swoff}$. To avoid situations as the example, t^2_{off} must be upper bounded by t^{max}_{off}, which is calculated as:

$$t^{max}_{off} = \max\left\{t_{off} : \beta^{\top}_R(\Delta) \geq \beta_B(\Delta), \forall \Delta \geq 0\right\}, \qquad (33)$$

where $\beta^{\top}_R(\Delta) = \max\{0, \Delta - t^2_{off} - t_{swon}\}$. Therefore, we have the feasible region of t^2_{off} as $t^2_{off} \in [t_{swoff}, t^{max}_{off}]$.

2) Deriving $< N_1, \text{PTM}_1, t^2_{on} >$ with a given t^2_{off}: Due to the mode switch overhead, the given t^2_{off} should be revised as t^2_{inv} from Eqn. 13. In the analysis, we onely use $< t^i_{vld}, t^i_{inv} >$ and $< t^i_{act}, t^i_{slp} >$. When they are finally obtained, we retrieve $< t^i_{on}, t^i_{off} >$ back.

For the sake of clarity, we introduce a denotation. we denote the difference in length between the horizontal part and the vertical part of the notch as $\bar{f}$, which can be calculated as $\bar{f} = x_{B_t} - y_{B_t} + y_{A_t} - x_{A_t}$ from the geometrical property. Then our problem can be divided into two cases: $\bar{f} < 0$ and $\bar{f} > 0$. Due to space limit, we only discuss $\bar{f} < 0$ in this paper. When $\bar{f} > 0$, the intuition is quite similar. For the details, we refer to the technical report [19].

We first show a useful property of the linear convex hull, which will be utilized in our analysis.

Prop. 1: When $\bar{f} < 0$, the slope of the segment that precedes the notch is larger than 1.

Proof: We denote the slope of the line connecting point 'A' and 'B' as $\rho(A,B)$. Then $\bar{f} < 0$ indicates that point C_t locates at the left side of point A_t. Therefore, we see that $\rho(C_t, B_t) < \rho(A_t, B_t)$. Since point B_t locates on segment $< \nu_t, \rho_t >$, which has a smaller slope than segment $< \nu_{t-1}, \rho_{t-1} >$, and $x_{B_t} \geq x_{\nu_t}$ (from Eqn. 32), one can easily prove that $\rho(A_t, B_t) < \nu_{t-1}$, which yields $\nu_{t-1} > 1 = \rho(C_t, B_t)$. From the definition of linear convex hull, we have $\nu_i > \nu_j$ for $i < j$. Then $\rho_i > 1$ holds for $i = 1, 2, \cdots, t-1$. ■

In this case, the end of the second t^2_{inv} should touch $\check{\beta}(\Delta)$ and the first t^2_{vld} should pass through point B_t such that the long-term slope is the minimum, as shown Fig. 5. Based on these two prerequisites, we can determine when the second t^2_{inv} starts, which means the length of the hyper-period, t_h, is determined. Therefore, the blue lines shown in Fig. 5 is fixed.

Now, suppose we already knew t^2_{vld}, then $< N_1, \text{PTM}_1 >$ can be computed directly. The reason is that since t^2_{vld}, t^2_{inv} and t_h are given, one can easily derive t_{p1} and ρ_a, which are the length and the long-term slope of N_1 periods of PTM_1, respectively. To lessen the transition overhead, N_1 should be its lower bound, whose formulation is detailed later. Then from N_1, t_{p1} and ρ_a, PTM_1 can be calculated.

From Prop. 1, one can see that the lower service curve is always no less than $\check{\beta}(\Delta)$ in the first hyper-period as t^2_{vld} varies. Therefore, the main intuition of our approach is varying t^2_{vld} step by step in its feasible region, which can guarantee the lower service curve is no less than $\check{\beta}(\Delta)$ in the following periods, and then determine the corresponding $< N_1, \text{PTM}_1 >$. Finally the peak temperature can be calculated from Thm. 1

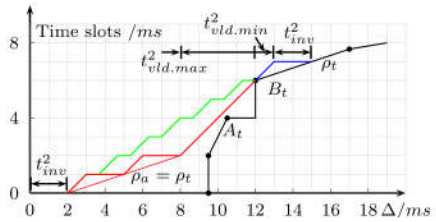

Fig. 5: The derivation of the HPTM scheme when $\bar{f} < 0$.

Algorithm 1 Completing HPTM for $\bar{f} < 0$

Input: **TM**, t_{swon}, t_{swoff}, t^2_{inv}, ε, $\tilde{\beta}(\Delta) = \{\beta(\Delta), \nu_t\}$
Output: **Result**$=< N_1, \text{PTM}_1, t^2_{on} >$, T_{op}

1: $T_{op} = T^{\infty}_a$, $t^2_{vb} = 0$, $t^2_{va} \leftarrow \frac{\rho_t}{1-\rho_t} t^2_{inv}$
2: **while** $t^2_{vb} \leq \frac{y_{B_t} - \rho_t x_{B_t} + \rho_t t^2_{inv}}{1-\rho_t}$ **do**
3: $\quad \rho_a \leftarrow \frac{y_{B_t} - t^2_{vb}}{x_{B_t} - t^2_{inv} - t^2_{vb}}$, $t^2_{vld} \leftarrow t^2_{va} + t^2_{vb}$, $t^2_{on} \leftarrow t^2_{vld} + t_{swon}$
4: $\quad t^1_{vld.max} \leftarrow \min(t^2_{vld}, \frac{\rho_a t^2_{inv}}{1-\rho_a})$, $N_1 \leftarrow \lceil \frac{y_{B_t} - t^2_{vb}}{t^1_{vld.max}} \rceil$
5: $\quad t^1_{on} \leftarrow \frac{y_{B_t} - t^2_{vb}}{N_1} + t_{swon}$, $t^1_{off} \leftarrow \frac{x_{B_t} - t^2_{vb} - t^2_{inv}}{N_1} - t^1_{on}$
6: $\quad$ calculate the peak temperature T_p based on Eqn. 25
7: $\quad$ **if** $T_p \leq T_{op}$ **then**
8: $\quad\quad T_{op} \leftarrow T_p$, **Result**$\leftarrow < N_1, \text{PTM}_1 = (t^1_{on}, t^1_{off}), t^2_{on} >$
9: $\quad$ **end if**
10: $\quad t^2_{vb} \leftarrow t^2_{vb} + \varepsilon$
11: **end while**

and consequently the optimal t^2_{vld} which leads to the minimal peak temperature can be found.

First, the feasible region of t^2_{vld} is determined. The lower bound, $t^2_{vld.min}$, can be calculated from the blue lines shown in Fig. 5 as $t^2_{vld.min} = \frac{\rho_t}{1-\rho_t} t^2_{inv}$, where ρ_t is the slope of the segment immediately following the notch. To guarantee the lower service curve of HPTM is no less than $\check{\beta}(\Delta)$ in the following hyper-periods, ρ_a should be no less than ρ_t, as the red line in Fig. 5. Therefore the upper bound of t^2_{vld} is calculated as $t^2_{vld.max} = \frac{y_{B_t} - \rho_t x_{B_t} + \rho_t t^2_{inv}}{1-\rho_t} + t^2_{vld.min}$.

Now, we discuss the lower bound of N_1 for given t^2_{vld} and t^2_{inv}. First, from geometrical property in Fig. 5, we have $t_{p1} = t_h - t^2_{vld} - t^2_{inv}$ and the long-term slope $\rho_a = \frac{y_{B_t} - (t^2_{vld} - t^2_{vld.min})}{t_{p1}}$. Then according to Eqn. 11 and Eqn. 13, we have $t^1_{inv} \leq t^2_{inv}$, $t^1_{vld} \leq t^2_{vld}$. With $t^1_{inv} = \frac{(1-\rho_a) t^1_{vld}}{\rho_a}$ and the above two inequalities, we can get $t^1_{vld} \leq \min(t^2_{vld}, \frac{\rho_a t^2_{inv}}{1-\rho_a})$. Since we have $N_1 \times t^1_{vld} = t_{p1} \times \rho_a$, the lower bound of N_1 can be determined as $\frac{t_{p1} * \rho_a}{\min(t^2_{vld}, \frac{\rho_a t^2_{inv}}{1-\rho_a})} = \frac{y_{B_t} - (t^2_{vld} - t^2_{vld.min})}{\min(t^2_{vld}, \frac{\rho_a t^2_{inv}}{1-\rho_a})}$.

Algorithm. 1 outlines the pseudo-code which describes the derivation of $< N_1, \text{PTM}_1, t^2_{on} >$ for a given t^2_{inv} when $\bar{f} < 0$. It takes as input the thermal model **TM**, the time overheads of mode-switching, the given t^2_{inv}, searching step ε, and the linear convex hull with a notch $\check{\beta}(\Delta)$. We first initialize the variables and calculate t^2_{va} (line 1). Then, the optimal t^2_{vb} is found by searching its feasible region with a step ε (line 2). After obtaining N_1 at line 4, the algorithm gets the PTM_1 at line 5. Finally, the peak temperature is calculated by the close-

Algorithm 2 Peak temperature minimizing

Input: **TM**,**EM**(N), t_{swon}, t_{swoff}, ξ
Output: **OpHPTM**=$< N_1, \text{PTM}_1, 1, \text{PTM}_2, \wedge >$, T_{op}

1: calculate the convex hull with a notch $\check{\beta}(\Delta)$ based on **EM**(N) and the scheduling policy
2: get t_{off}^{max} from Eqn. 33, $T_{min}^{\star} \leftarrow T_a^{\infty}$, **OpHPTM** $\leftarrow \vec{0}$
3: **for** $t_{off}^2 = t_{swoff}$ to t_{off}^{max} with step ξ **do**
4: get the tangent point ν_t and calculate $< A_t, B_t >$ by (31)
5: calculate $\bar{f}$, then according to if it is positive choose the corresponding algorithm to compute $< N_1, \text{PTM}_1, t_{on}^2 >$.
6: calculate the peak temperature $T(t_{off}^2)$ by Thm. 1
7: **if** $T(t_{off}^2) \leq T_{min}^{\star}$ **then**
8: $T_{min}^{\star} \leftarrow T(t_{off}^2)$
9: **OpHPTM**$\leftarrow < N_1, \text{PTM}_1, 1, \text{PTM}_2 = (t_{on}^2, t_{off}^2), \wedge >$
10: **end if**
11: **end for**

TABLE I: Thermal and Hardware Model Parameters

G	C	$\varphi_i = \varphi_a$	θ_i	θ_a	T_{amb}	$t_{swon} = t_{swoff}$
0.3 $\frac{W}{K}$	0.03 $\frac{J}{K}$	0.1 $\frac{W}{K}$	-25 W	-11 W	300 K	0.1 ms

form solution given by Eqn. 25, and then is compared to the current-lowest temperature to search for the best solution.

3) Searching the optimal t_{off}^2: Based on simulations, we observe that the peak temperature $T^{\star}$ varies irregularly in the domain of t_{off}^2. Therefore, we make a thorough search with a fixed step in the feasible region of t_{off}^2 to find the optimal t_{off}^2. The pseudo code of the algorithm is shown in algorithm 2.

VI. Case Studies

In this section, we study the viability of our approach and compare it with the approaches proposed in previous works. The case studies are implemented in Matlab using RTC-Toolbox. All the results are obtained from a simulation platform with an Intel i7 4770 processor and 16 GB memory.

A. Setup

The thermal and power parameters are set as described in Tab. I [2], [18]. We specify an event stream by the timing model of the *pjd* model, i.e., the period p, jitter j, minimal inter-arrival distance d of the stream, and the worst-case execution time c. Since the performance of our approach is related to the burst, which is affected by the jitter and the worst-case execution time in this model, we adopt jitter factor, χ, and the WCET factor, ω, to vary the volume of burst in the experiments. The jitter is calculated as $j = \chi * p$ and the worst-case execution time is obtained by $c = \omega * p$. The relative deadline is set as the period in all the simulations.

Two multi-event stream applications, which are the video-conferencing system [18] and the task streams set studied in [15], are adopted for simulations. The Earliest Deadline First scheduling policy is adopted for multi-event scenarios. The parameters of the two applications are summarized in Tab. II and Tab. III.

The approach LSC-PTM and TAPR (thermal-aware periodic resources) studied in [4] are adopted for the comparison. LSC-PTM is a single-period pattern thermal management approach. TAPR models its input events by sporadic task (c, D, P), which

TABLE II: Event stream setting for video-conferencing (msec)

	video	audio	network
p	40	30	30
j	5	2	3
d	1	1	1
c	6	3	2

TABLE III: Event stream setting for task streams (msec)

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
p	198	102	283	354	239	194	148	114	313	119
j	387	70	269	387	222	260	91	13	302	187
d	48	45	58	17	65	32	78	-	86	89
c	12	7	7	11	8	5	13	14	5	6

is specified by a worst-case execution time c, a (relative) deadline D and a minimum inter-arrival separation P. The P in this model has to be revised as $\max[(p-j), d]$ to ensure the system safety in worst-case because (c, D, P) does not contain all the information in (p, j, d, c) model. We compare these approaches in single and multiple event-stream scenarios.

B. Results

The effectiveness of our approach for minimizing the peak temperature is evaluated in our simulations. We normalize the results as the Relative Peak Temperature (RPT), which is the difference between the derived minimal peak temperature and the steady temperature of sleep state.

Fig. 6 and Fig. 7 show the results for single event stream and two-event stream scenarios, respectively. Fig. 8 depicts the results for different WCET factors with a fixed $\chi = 1.5$ when S_3 is the input. We also take the three-event application, 'video conferencing', as input in our simulation. The results w.r.t. different jitter factors are displayed in Fig. 9.

From simulation results, we make below observations: (1) In general, approach HPTM produces the best results. This is expected since HPTM utilizes the hyper-period to better approximate the real-time service bound and thus generates lower temperatures. (2) In Fig. 8, the difference between the results of LSC-PTM and HPTM gets larger as the system utilization increases. The reason is that the gap between the lower service curves of the two approaches is magnified when the WCET factor increases, thus broadens the difference in temperature. (3) In Fig. 9, the difference between the results of LSC-PTM and HPTM widens as the jitter factor increases. The gap, however, will not further widen after certain jitter threshold has been reached. This is caused by these facts: one the one hand, when jitter is relatively small, the growing jitter consequently increases the burst in the input. LSC-PTM is restricted by the burst and its lower service curve has to be raised, causing the temperature increase quite quickly. At the same time, HPTM can better utilize the lower slope after the burst to 'cool' the processor. Thus the temperature increases slower than that in LSC-PTM. On the other hand, as jitter increases to be larger than the threshold, the peak temperature in the HPTM is elevated quickly when the huge burst is tackled, resulting in a lower effectiveness to the progress of 'cooling' the processor. Despite of this, our approach still provides lower peak temperature than the purely periodic one. (4) The peak temperature derived by TAPR is always the highest. The RPT even becomes flat at some points when it reaches the upper threshold $T_a^{\infty} - T_a^{\infty}$, which is 70 in this paper. This is due to the limitation of its event model where the non-determinism of pjd pattern cannot be properly modeled and that the modified $P = \max[(p-j), d]$ overestimates incoming

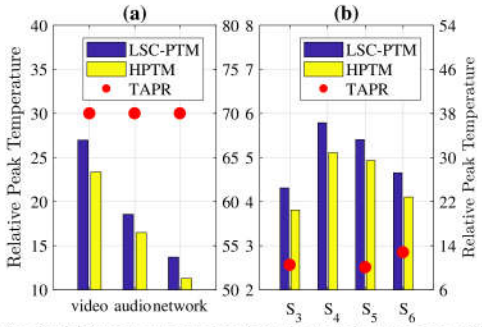

Fig. 6: Relative Peak Temperature produced by the tested approaches with $\chi = 1.2$ for (a) the single event streams in the video-conferencing application, (b) S_3 to S_6. The right Y axes indicate the Relative Peak Temperature of approach TAPR.

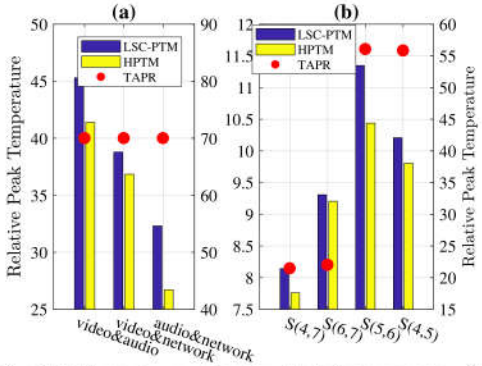

Fig. 7: Relative Peak Temperature produced by the tested approaches with $\chi = 1.5$ for (a) two-event streams in the video-conferencing application, (b) randomly selected two-event streams in $S(3,4,5,6)$. The right Y axes indicate the Relative Peak Temperature of approach TAPR.

workload. As shown in the figures, the RPT of TAPR increases rapidly when the jitter increases.

Finally, we also report the time expenses of our approach. All tested approaches work in offline manner and can give results in less than 0.2 second. HPTM costs more time to finish than other two approaches. Since HPTM is an offline approach, the slightly increase in time expense is not crucial and acceptable.

In summary, our proposed hyper-period based approach offer better results than existing works, reducing the peak temperature by up to 8K for the two-event stream of audio and network. The approach is also efficient and can give results in the order of hundred of milliseconds.

VII. Conclusion

In this paper, we present a new approach to minimize the peak temperature for a hard real-time system in which the input event streams are modeled by arrival curves. Our approach can appropriately combine two patterns of PTM together to form a hyper-period thermal management (HPTM) which can better approximate the service bound so that the peak temperature is minimized while meeting the deadline constraints. Another advantage of our approach is that the HPTM is offline generated and thus requires minimum runtime computation. Simulation results show that our approach generates better results in terms of peak temperature compared to work in the literature, especially for large jitter and system utilization scenarios .

VIII. Acknowledgements

This work has been partly funded by China Scholarship Council (grant number: 201306120019).

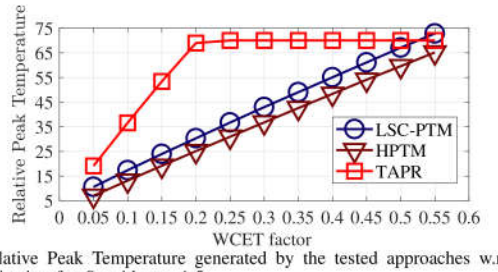

Fig. 8: Relative Peak Temperature generated by the tested approaches w.r.t. different system utilization for S_3 with $\chi = 1.5$.

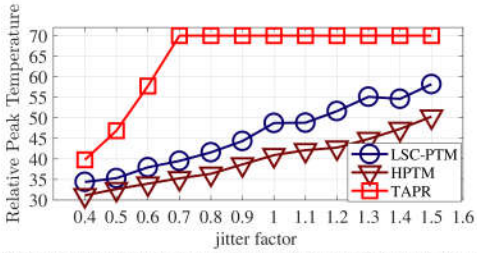

Fig. 9: Relative Peak Temperature generated by the tested approaches w.r.t. different jitter factor for the application of video-conferencing with EDF scheduling.

References

[1] Sushu Zhang and et al. Approximation algorithm for the temperature-aware scheduling problem. In *ICCAD*, pages 281–288. IEEE, 2007.

[2] Pratyush Kumar and et al. Cool shapers: shaping real-time tasks for improved thermal guarantees. In *DAC*, pages 468–473. IEEE, 2011.

[3] David Brooks and et al. Dynamic thermal management for high-performance microprocessors. In *HPCA*, pages 171–182. IEEE, 2001.

[4] Masud Ahmed and et al. Minimizing peak temperature in embedded real-time systems via thermal-aware periodic resources. *Sustainable Computing: Informatics and Systems*, 1(3):226–240, 2011.

[5] Pratyush Kumar and et al. Thermally optimal stop-go scheduling of task graphs with real-time constraints. In *ASP-DAC*, pages 123–128. IEEE Press, 2011.

[6] Gang Chen and et al. Energy optimization for real-time multiprocessor system-on-chip with optimal dvfs and dpm combination. *ACM/TECS*, 13(3s):111, 2014.

[7] Thidapat Chantem and et al. Online work maximization under a peak temperature constraint. In *ISLPED*, pages 105–110. ACM, 2009.

[8] Shengquan Wang and et al. Reactive speed control in temperature-constrained real-time systems. *Real-Time Systems*, 39(1-3):73–95, 2008.

[9] Nikhil Bansal and et al. Speed scaling to manage energy and temperature. *JACM*, 54(1):3, 2007.

[10] Jian-Jia Chen and et al. Proactive speed scheduling for real-time tasks under thermal constraints. In *RTAS*, pages 141–150. IEEE, 2009.

[11] International technology roadmap for semiconductors. http://www.itrs.net/reports.html.

[12] Cheng and et al. Periodic thermal management for hard read-time systems. In *SIES*. IEEE, 2015.

[13] L. Thiele and et al. Real-time Calculus for Scheduling Hard Real-time Systems. *ISCAS*, 4:101–104, 2000.

[14] Jean-Yves Le Boudec and et al. *Network calculus: a theory of deterministic queuing systems for the internet*. Springer, 2001.

[15] Kai Huang and et al. Periodic power management schemes for real-time event streams. In *CDC/CCC*, pages 6224–6231. IEEE, 2009.

[16] Kai Huang and et al. Applying real-time interface and calculus for dynamic power management in hard real-time systems. *Real-Time Systems*, 47(2):163–193, 2011.

[17] Kai Huang and et al. Energy-efficient scheduling algorithms for periodic power management for real-time event streams. In *RTCSA*, volume 1, pages 83–92. IEEE, 2011.

[18] Devendra Rai and et al. Worst-case temperature analysis for real-time systems. In *DATE*, pages 1–6. IEEE, 2011.

[19] Cheng and et al. Hyper-periodic thermal management for hard real-time systems. Technical report, Technical University of Munich, 2017.