Response Time Analysis for Thermal-Aware Real-Time Systems Under Fixed-Priority Scheduling

Younès Chandarli, Nathan Fisher, Damien Masson

Approximate Response Time Analysis for Thermal-Aware Real-Time Systems Under Fixed-Priority Scheduling and Reactive Control

Younès Chandarli, Nathan Fisher and Damien Masson

June 27, 2014

Abstract

This paper investigates schedulability analysis for thermal-aware real-time systems. Thermal constraints are becoming more and more critical in new generation miniaturized embedded systems, e.g. medical implants. As part of this work, we adapt the PFP_{ASAP} algorithm proposed in [1] for energy-harvesting systems to thermal-aware ones. We prove its optimality for non-concrete fixed-priority task sets and propose a response-time analysis based on worst-case upper bounds. We evaluate the efficacy of the proposed bounds via extensive simulation over randomly-generated task systems.

1 Introduction

The main purpose of real-time systems is to guarantee predictable timing behavior for controlled devices. Therefore, the correctness of the results provided by such systems depends not only on the logical correctness of the output but also on the time at which it is yielded. Several formal models of real-time behavior have been proposed (e.g. task models such as sporadic, periodic, aperiodic, DAG, etc.). Prior research in real-time systems have also addressed a wide array of hardware architectures (e.g. monoprocessor, multiprocessors, memory caches, etc). However, for a new generation of real-time systems applications, e.g. medical implants, the physical environment poses additional design challenges.

One such new challenge is the necessity of managing the energy and the thermal behavior of systems. As technology scales, chips power consumption and power density are increasing rapidly. Indeed, the miniaturization of small embedded systems has allowed new real-time applications. Implantable medical devices (IMD) are an example of these new embedded systems where managing the thermal aspect is essential. IMDs are increasingly being used in medical treatments (e.g. pacemakers for heart diseases or neural implants to restore hearing/vision). However, recent studies [20, 13] have shown that the heat generated by IMDs due to the processor activity is non-negligible. Thus, designing thermal aware IMDs becomes critical as medical research has shown that a temperature increase of even $1C^{\circ}$ can damage tissues [12] and may cause death in extreme cases [18].

Therefore, thermal-aware real-time systems must respect not only timing constraints, expressed with deadlines, but also thermal constraints which are expressed as a maximum temperature not to be exceeded. For fixed-priority real-time scheduling on monoprocessor

platforms, considering this constraint requires the schedulers to add cooling periods. This additional idle times must be taken into account by scheduling algorithms and included in schedulability analysis.

Thermal-aware system design presents challenges similar to the design of energy-harvesting systems. The later collects the environmental energy to store it and use it to supply real-time systems. The real-time scheduling of these kind of systems must respect tasks deadlines without running out of energy. The similarities with thermal-aware systems come from the fact that the scheduling for energy-harvesting system has to consider a battery replenishment time (which is analogous to the cooling periods required in thermal-aware systems). In this work we use the PFP_{ASAP} scheduling algorithm proposed in [1], which was proved to be optimal for non-concrete fixed-priority energy-harvesting systems, to build a reactive thermal-aware scheduling approach and an approximate schedulability analysis based on upper and lower bounds of tasks worst-case response-time.

The remainder of this paper is organized as follows. Section 2 gives a brief state of the art about thermal-aware real-time systems. Section 3 specifies and describes the model and the scope of this work. Section 4 presents the PFP_{ASAP} scheduling algorithm and its optimality for non-concrete systems. Section 5 details an approximate response-time analysis based on upper and lower bounds of tasks worst-case response time. Section 6 shows some simulations results to evaluate the effectiveness of the proposed schedulability analysis. Finally, Section 7 concludes this paper.

2 Related Work

In this section we give a brief overview of prior research related to thermal-aware and energy-aware real-time scheduling. Most works addressing this problem consider Dynamic Voltage and Frequency Scaling (DVFS) strategies. DVFS consists of scaling down the CPU speed and thereby lengthening task execution times to reduce energy consumption and lower the peak temperature [10, 21, 22, 23].

Among existing work, the proposed techniques can be divided into *reactive* and *proactive* approaches. The difference between these two approaches is that reactive schemes adapt to the temperature of the system when it reaches the maximum temperature or a specific trigger by switching the CPU speed or by changing scheduling decisions. In this scope, Wang et al. proposed a schedulability analysis for speed scaling scheme for frame-based task model in [23], and they completed this with a worst-case response time analysis for FIFO and fixed priority scheduling in [22, 23]. In contrast, proactive approaches set the configuration of the system judiciously beforehand (CPU speed and scheduling decisions) so that the maximum temperature is never reached [10, 16, 17]. In this scope, Chen et al. proposed in [9, 10] a proactive EDF-based scheduling approach that changes the processor speed proactively by requests issued by the scheduler.

There exists also some works that address this scheduling problem without DVFS schemes by considering processors with only one frequency. In this scope, Ahmed et al. [2] proposed a technique that computes proactively the length of execution and cooling intervals so that a certain temperature is never reached. This idea was extended in [11] to support unpredictable ambient temperature fluctuations. Rehan et al. proposed in [3, 4] a kind of thermal utilization of the system (using a fluid schedule) and leveraged it to obtain a necessary and sufficient conditions for systems thermal feasibility.

All the mentioned work have the following limitations :

1) Except for work in the previous paragraph [2, 3, 4, 11], all the proposed solutions rely on speed scaling to manage energy and temperature. These approaches cannot be

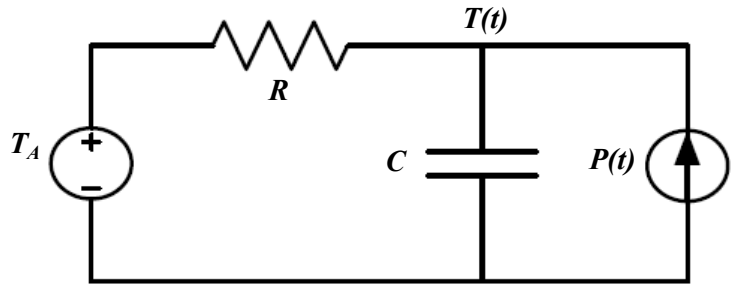


Figure 1: Thermal model

applied to systems without DVFS capabilities.

2) Most of the scheduling solutions proposed in the literature are EDF-based. Knowing that static fixed-priority scheduling is highly used in industry, it deserves more attention and effort to study the fixed-priority real-time scheduling under thermal constraints.

Recently, some results for the scheduling problem of energy-harvesting systems, which seem to be similar to the thermal aware model, were proposed. In [1, 5, 15] some scheduling algorithms and schedulability analysis were proposed. It consists of keeping the battery level enough high to permit task execution. The algorithm proposed in [1] behaves like a reactive approach in the thermal-aware model. In fact, it delays executions until the battery is enough replenished to execute at least one time unit. This behavior is similar to reactive approach that delays executions or switch CPU speed when the maximum temperature is reached. In this paper we use the PFP_{ASAP} algorithm proposed in [1] to build a reactive approach for the thermal-aware and fixed-priority rel-time systems. We propose a schedulability analysis based on worst-case response time approximation techniques.

3 Models

3.1 Task Model

We consider a classical non-concrete real-time task set defined by a set of n sporadic and independent tasks $\{\tau_1, \tau_2, \ldots, \tau_n\}$. Each task τ_i is characterized by its priority P_i, its worst-case execution time C_i, its minimum inter-arrival time T_i, its deadline D_i and its first release time O_i. Deadlines are constrained or implicit, i.e., $\forall\ i,\ D_i \leq T_i$.

3.2 Thermal Model

In our model, the temperature of the system fluctuates due to heat dissipation when real-time tasks are executed on the CPU. The temperature must stay between two thresholds T_A and T_{max} where T_{max} is the maximum tolerated temperature and T_A is the ambient energy. The temperature of the system at time t is denoted as $T(t)$. The only way to cool down the system is to temporarily suspend task execution. Furthermore, we consider that the system may be one of two states at any given time: active (i.e., heating) during which tasks may execute or inactive (i.e., cooling) during which tasks are not permitted to execute.

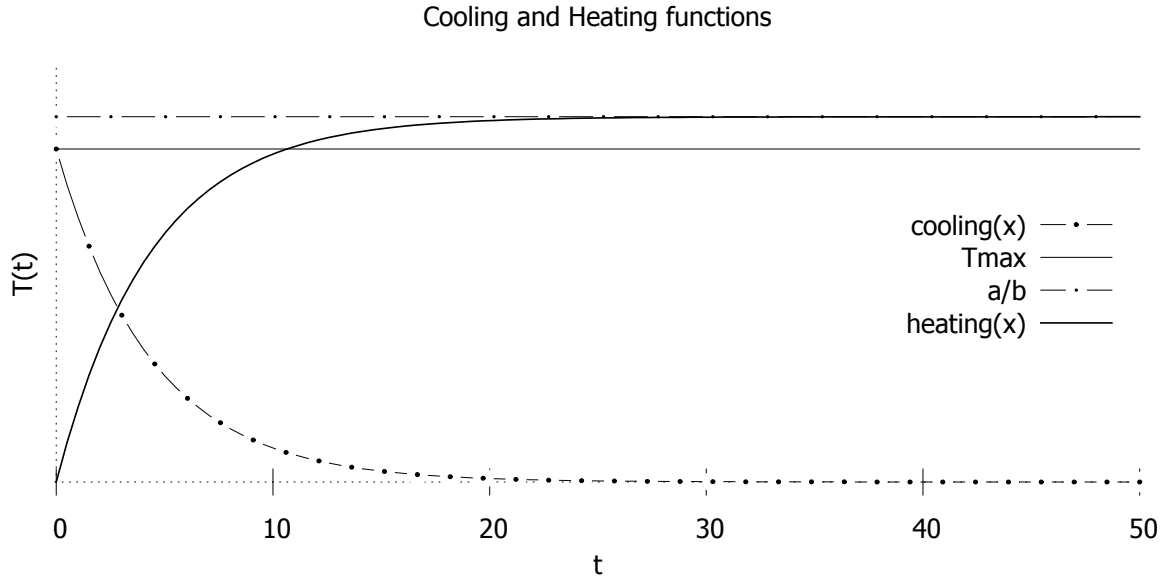


Figure 2: Cooling and Heating Functions

3.2.1 Heating Model

The thermal behavior of a processor can be modeled using an RC circuit [19] shown in Figure 1. In this model, the heating is modeled by the current denoted $P(t)$ passing through a thermal resistance R. The thermal capacitance is denoted C. Using this model, the derivative of the system temperature with respect to time can be calculated with Fourier's law [21] given by Equation 1.

$$T'(t) = \frac{P(t)}{C} - \frac{T(t) - T_A}{R \times C} \tag{1}$$

The current passing through the resistance can be separated into two parts: the dynamic part $P_D(t)$ that evolves linearly with the processor frequency, denoted s, and the part corresponding to the energy leakage $P_L(t)$ which is a function of the temperature.

$$P(t) = P_D(t) + P_L(t) \tag{2}$$
$$P_D(t) = \beta_0 s^\alpha \tag{3}$$
$$P_L(t) = \beta_1 T(t) + \beta_2 \tag{4}$$

Equations 2 to 4 give the formula to compute $P(t)$, where α, β_0, β_1 and β_2 are system specific constants [21]. We consider only a monoprocessor with active/inactive modes; thus, during active periods, $P_D(t)$ is constant. Let us denote $a = \frac{\beta_0 s^\alpha}{C}$, $b = \frac{1}{R \cdot C} - \frac{\beta_1}{C}$ and scale $T(t)$ to be $T(t) - \frac{R\beta_2 - T_A}{R\beta_1 - 1}$ to shift T_A to 0. We can now recognize in Equation 1 a classical linear differential equation:

$$T'(t) = a - b \times T(t) \tag{5}$$

Then, the solution is given by:

$$T(t) = \frac{a}{b} + \left(T(t_0) - \frac{a}{b}\right) \cdot e^{-b(t-t_0)} \tag{6}$$

The heating function only depends on time and constants and is not task-specific.

Recall that the parameters a and b are processor specific constants. Typical settings for these two variables are $b0.228$, and $a > 1$ with $\alpha 3$ (See [2]).

3.2.2 Cooling Model

During the cooling phases, the processor is inactive. In this paper, we assume for simplicity that the frequency s is 0. (However, this can easily be generalized to allow some fixed power dissipation during inactive phases.) Then $a = 0$ and the formula becomes:

$$T(t) = T(t_0) \cdot e^{-b(t-t_0)}. \tag{7}$$

Again, the cooling function only depends on time and is not task specific. Figure 2 shows the curves of cooling and heating functions. We can see that the cooling function slows down rapidly because of the exponential function. This means that cooling for several short intervals is better for temperature and thereby for tasks response time than few and long ones.

4 The PFP_{ASAP} algorithm

In [1], a scheduling algorithm for energy harvesting systems called PFP_{ASAP} was introduced. This algorithm is a fixed-priority one which takes into account the tasks energy cost and the battery capacity during scheduling operations for energy-harvesting systems. Tasks are executed according to their priority when the available energy is enough to execute and only replenishes the battery otherwise, jobs execution can be suspended to replenish energy as much as needed to execute at least one time unit. This algorithm was proved to be optimal for non-concrete fixed priority energy-harvesting systems. In this section we adapt this algorithm to thermal-aware systems and we explore its optimality for the model described in Section 3.

With the thermal constraints, the behavior of PFP_{ASAP} becomes as following: it executes jobs whenever the temperature is enough below T_{max} to execute at least one time unit without exceeding, then, it idles the system to cool down for as long as needed to resume executions (one time unit is sufficient).

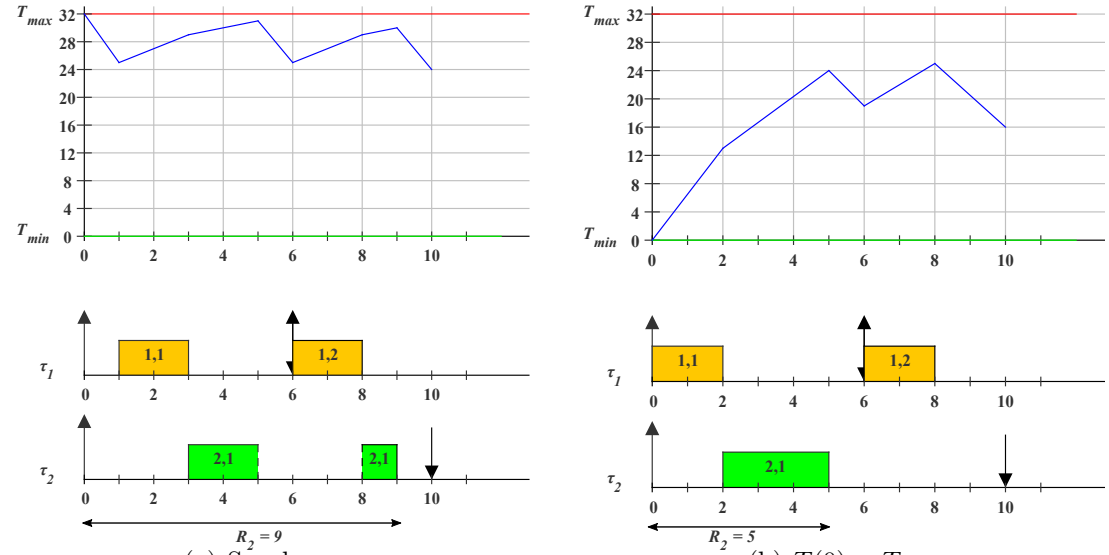
Below, we will first address the PFP_{ASAP} worst-case scenario, then we will discuss its optimality.

4.1 Worst-case scenario

The aim of this section is to prove that the worst-case scenario for non-concrete fixed-priority thermal-aware systems is still the synchronous activation of tasks but with $T(0) = T_{max}$.

Figure 3(a) illustrates the case where all the tasks are requested simultaneously. If at least one higher priority task is requested later, the response time of lower priority tasks decreases as illustrated in Figure 3(c). Then, if higher priority tasks are requested earlier, the response time of lower priority tasks cannot be longer than the one in the synchronous scenario as shown in Figures 3(d). Furthermore, if the initial temperature of the system is less than T_{max}, then, less cooling time is needed which leads to a shorter response time for all tasks.

Theorem 1. *Let Γ denote a non concrete task set composed of n priority-ordered tasks with constraint or implicit deadlines. The PFP_{ASAP} worst-case scenario for any task of Γ occurs whenever this task is requested simultaneously with requests of all higher priority tasks and the system temperature is at the maximum level T_{max}.*



(a) Synchronous (b) $T(0) < T_{max}$

(c) Asynchronous (d) Asynchronous

Figure 3: Worst-case scenario

To prove this theorem we compare jobs response times in the scenario proposed by the theorem with all other possible ones. The response time of a job is the difference between its termination time and its offset or request time.

Proof. Let $\{\tau_1, \tau_2, \ldots, \tau_n\}$ be a set of n priority-ordered tasks where τ_n is the task with the lowest priority. Let S_i^s denote the scenario where task τ_i and all higher priority tasks are requested simultaneously at the lower battery level E_{min}. The worst-case scenario for a task τ_i is the one that maximizes its response time, i.e. the scenario that maximizes the termination date of the first job of the i^{th} priority level.

If S_i^s is not the worst scenario, there must be an other one leading to a greater response time for the i^{th} priority level.

Firstly, we consider the scenario where $T(0) < T_{max}$. In this case the system is not heated at the maximum. Therefore, the system may need less cooling periods than the scenario where $T(0) = T_{max}$, and PFP_{ASAP} introduces shorter or equal cooling periods and leads to a shorter response time for all the tasks. This is in contradiction with our hypothesis, thus, such a scenario cannot lead to longer response times.

Secondly, we consider the scenario with different offsets. Let us denote S_i^a as the scenario where $T(0) = T_{max}$ and all tasks have different offsets. In this case we distinguish two possibilities:
(i) Where at least a task of higher priority than τ_i is requested later: knowing that all the tasks consume energy and heat the system following the same pattern, i.e., by considering that the heating comes only from processor energy consumption and that heating is greater than cooling, task τ_i will undergo less higher priority interferences, and then, it may need less cooling to finish. Therefore, the final response time of τ_i is less than or equal to the one given by scenario S_i^s which is a contradiction. Thus, such a scenario cannot lead to longer response times.
(ii) Where at least a task of higher priority than τ_i is requested earlier: when τ_i is requested later than a higher priority task, it undergoes less interference from this task because, first, a part of it was executed before τ_i request time, and second, the increase of temperature due to the higher priority task execution cannot be higher than T_{max}, and finally, if τ_i is requester much later than the higher priority tasks, we just shift the landscape and will have case (i) like illustrated in Figure 3(d). Therefore, this scenario cannot be worse than S_i^s.

Therefore, in all possible situations, the response-time of a task τ_i is lesser or equal to the one led by a synchronous activation of all higher priority tasks when the temperature is at maximum level.

□

4.2 The optimality of PFP_{ASAP}

The PFP_{ASAP} algorithm was proved to be optimal for the fixed-priority scheduling problem off non-concrete energy-harvesting systems [1] which is close to the same scheduling problem of thermal-aware systems. In this subsection we extend the optimility of PFP_{ASAP} to non-concrete thermal-aware systems.

Theorem 2. *The PFP_{ASAP} scheduling algorithm is optimal for fixed-priority thermal-aware non-concrete task sets with constrained or implicit deadlines.*

Proof. Let Γ denote a non concrete task set. We suppose that Γ is feasible using a fixed-priority assignment, but not schedulable with PFP_{ASAP} using the same priority

assignment. This means that at least one task denoted as τ_k misses its deadline in the worst-case scenario (see Theorem 1). Indeed, it is sufficient to consider only the first job because we are dealing only with constrained or implicit deadlines. According to PFP_{ASAP} rules, a deadline miss can occur in the worst-case scenario only in two cases: 1) the workload is greater than the available time, 2) the workload plus the accumulated cooling time is greater than the available time.

1) if the workload from the critical instant (time 0) to time D_k the first deadline of τ_k is alone greater than time interval $[0, D_i]$, then, it is obviously impossible to schedule the first job of τ_k and higher priority jobs without missing D_k, this is not possible even without thermal constrains because the available time is not sufficient to schedule all the workload within $[0, D_k]$. Then, in this case the task set cannot be feasible with any algorithm and the supposed algorithm cannot exist.

2) if a deadline is missed with PFP_{ASAP} even thought the workload is lesser than the available time, then, this means that the sum of workload of time interval $[0, D_k]$ and the needed cooling time is greater than the available time, i.e D_k time units. Knowing that the cooling periods produced by PFP_{ASAP} are as long as needed to execute at least one time unit which means that they are as short as possible. Furthermore, we know that the cooling function is exponentially decreasing (See Equation 7), then, the shorter cooling periods are, the shorter the total needed cooling time is. This is true because the longer cooling is, the less efficient it is, as we mentioned in Section 3. More formally, cooling x times one time unit is greater than one time cooling period of length x time units; observe that $e^{-bx} \leq xe^{-b}$ for all $x \geq 1$. Thus Equation 7) implies

$$T(0) - T(0) \cdot e^{-b \cdot x} \geq T(0) - x \times T(0) \cdot e^{-b(1-0)} \tag{8}$$

Therefore, any other schedule than PFP_{ASAP}'s one has necessarily cooling periods of same length or longer, then, the response time of τ_k produced by the supposed algorithm is necessarily greater than D_k. Thus, in this case, no other algorithm can schedule this task set.

Then we prove that PFP_{ASAP} is optimal for non concrete fixed-priority thermal-constrained task sets with constrained or implicit deadlines. □

5 Response-Time Analysis

This section provides a response-time analysis for the schedule produced by the optimal algorithm PFP_{asap} in the worst-case scenario, i.e., the synchronous release of all the tasks when $T(0) = T_{max}$. We discuss the difficulty of an exact analysis and then we propose an approximate one.

5.1 Exact Analysis

The exact analysis provides the exact response time of all tasks. Thus, it must estimate accurately the length of all cooling and heating periods.

However, this cannot be done with a generic equation because due to the discrete time, all cooling periods are not of the same length in the actual schedule. Furthermore, without the effective values of parameters, it is hard to estimate the order and the number of long and short cooling periods which have a significant impact on the response time value. Therefore, the only way to get an exact analysis is to simulate the schedule of

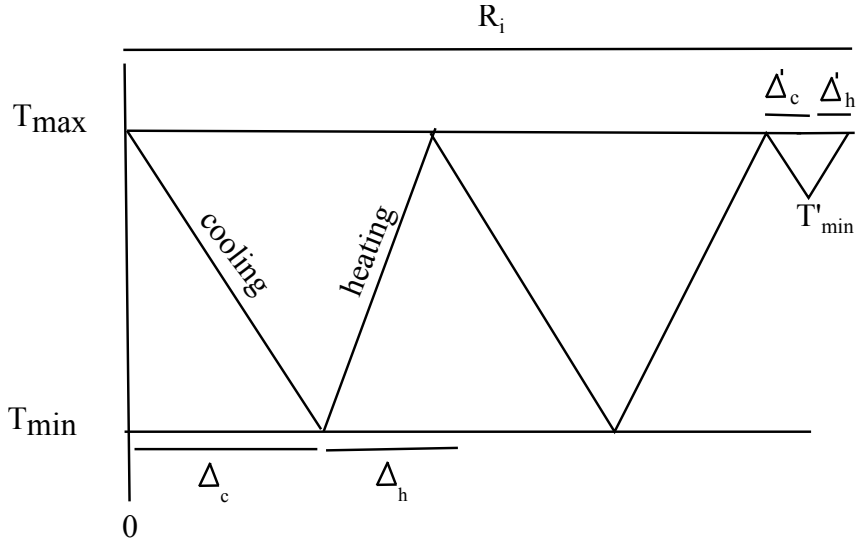


Figure 4: First Upper Bound ($UB_{T_{min}}$)

PFP_{ASAP} in the worst-case scenario and compute the response time of the first job of each task.

5.2 Approximate Analysis

The aim of this work is to propose a schedulability analysis for thermal-aware real-time systems. Such an analysis must consider not only the processor workload but also the additional cooling time needed to respect the thermal constraints. To cope with the difficulty of providing an exact analysis, one can propose an approximate one that can be only sufficient instead of necessary and sufficient. This can be achieved by upper bounding tasks worst-case response time produced by PFP_{ASAP} algorithm.

5.2.1 First Upper bound ($UB_{T_{min}}$):

Knowing that PFP_{ASAP} cools down the system enough to execute at least one time unit, cooling periods are as short as possible. Furthermore, we know also that the cooling function is exponentially decreasing and the heating is asymptotically increasing, then, one can lengthen jobs response times by putting the cooling units together and the heating ones together such that T_{max} is never exceeded. By doing so, the cooling slows down after a while and the system needs more time to cool down, and heating becomes faster which heats up the system in a shorter amount of time.

Description The upper bound of task τ_i worst-case response time according to $UB_{T_{min}}$ is described by Figure 4. It consists of:

- cooling down the system from T_{max} to T_{min}, where $T_{min} > T_A$,[1]

[1]Observe that a low T_{min} value may result in an extremely pessimistic upper bound due to the nature of the cooling function; It decreases asymptotically to T_A, so waiting until T_{min} is too pessimistic because of the nature of the cooling function.

- a ceiling function applied to the time from T_{max} to $T_{min}$to ensure an integer number of time units (this is safe since it only over estimates the time required to reach T_{min}),
- executing jobs and heating up the system until T_{max} is reached or there is no pending workload,
- repeating this cycle of cooling-heating until there is no pending workload,
- the last cycle may be shorter because of the remaining workload which is shorter than a full cycle. The corresponding cooling time is adjusted.

The response time upper bound of task τ_i of priority level-i that is requested simultaneously with higher priority tasks with $T(0) = T_{max}$ is given by Equation 9

$$\begin{cases} w_i^{n+1} & = & N(w_i^n) \times (\Delta_c + \Delta_h) + \Delta'_c + \Delta'_h \\ R_i^{UB_{T_{min}}} & = & w_i^{n+1} = w_i^n \end{cases} \tag{9}$$

where :

- $N(w)$ is the number of full cooling-heating cycles needed to execute the workload w without exceeding T_{max}:

$$N(w) = \left\lfloor \frac{w}{\Delta_h} \right\rfloor \tag{10}$$

- w is the workload of time interval $[0, w_i^n[$:

$$w = \sum_{j \leq i} \left\lceil \frac{w_i^n}{T_j} \right\rceil \times C_j \tag{11}$$

- Δ_h is the time to execute jobs and heat up the system from T_{min} to T_{max}. (Obtained by solving Equation 6):

$$\Delta_h = \left\lfloor \frac{\ln\left(\frac{b \cdot T_{min} - a}{b \cdot T_{max} - a}\right)}{b} \right\rfloor \tag{12}$$

- Δ_c is the time to cool down the system from T_{max} to T_{min}. (Obtained by solving Equation 7):

$$\Delta_c = \left\lceil \frac{\ln(T_{max}) - \ln(T_{min})}{b} \right\rceil \tag{13}$$

- Δ'_h is the remaining execution time of the busy period:

$$\Delta'_h = w - N(w_i^n) \times \Delta_h \tag{14}$$

- Δ'_c is the cooling time needed for Δ'_h:

$$\Delta'_c = \left\lceil \frac{\ln(T_{max}) - \ln(T'_{min})}{b} \right\rceil \tag{15}$$

- T'_{min} is the maximum temperature needed to execute the remaining part of the workload Δ'_h without exceeding T_{max}:

$$T'_{min} = (T_{max} - a/b)e^{b\Delta'_h} + a/b \tag{16}$$

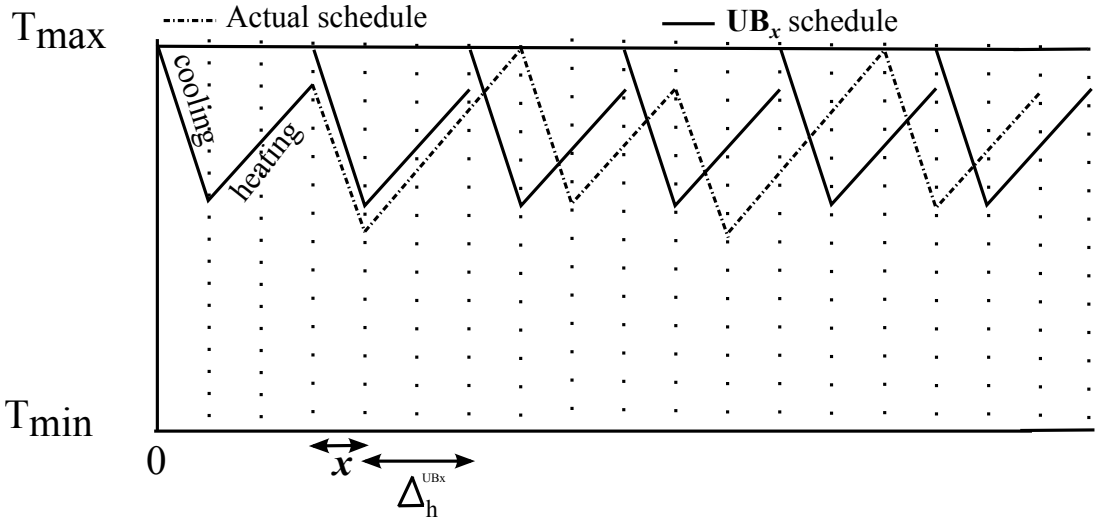


Figure 5: Parametric Upper Bound UB_x

Theorem 3. *An upper bound on the worst-case response time for task τ_i in the worst-case scenario described in Section 4.1 can be obtained from a sequence of execution units of higher priority jobs and the necessary cooling time units where the cooling periods are as long as needed to cool the system from T_{max} to T_{min} ($T_{min} > 0$) and the heating periods are as long as needed to reach T_{max} starting from T_{min}, as described by Formula 9.*

Proof. See appendix A □

5.2.2 Parametric Upper bound (UB_x):

Description The idea of this upper bound is to keep the same behavior of the PFP_{ASAP} algorithm by cooling down for some time units and then executing jobs until reaching T_{max}. The approximation comes from the fact that time is discrete and that cooling periods are of a fixed length x (where $x \in N^*$) instead of the minimum length needed to execute at least one time unit. Then, the execution or the heating periods may not reach T_{max} in an integer number of time units. Thus, we consider only the integer part of heating periods (with floor function) and that T_{max} is exactly or nearly reached at the end of each heating period which adds additional cooling time than the actual schedule. Figure 5 describes the scenario used to obtain UB_x. It consists of:

- Cooling down the system for x time units, where x a is positive integer that must be greater or equal to $\Delta_c^{UB_x}$ the minimum time needed to decrease the temperature such that the system can execute at least one time unit, i.e. $x \geq \Delta_c^{UB_x}$. Equation 17 computes $\Delta_c^{UB_x}$; the ceiling function is used to respect the discrete time and to ensure that the system is enough cold to execute at least one time unit.

$$\Delta_c^{UB_x} = \left\lceil \frac{\ln\left(\frac{bT_{max}}{(bT_{max}-a)e^b+a}\right)}{b} \right\rceil \tag{17}$$

- Then, executing jobs and heating up the system until T_{max} is reached (without exceeding) or there is no pending workload. The length of this period is an integer. The floor function is used to ensure not exceeding T_{max}.

- Repeating the cooling-heating cycles until there is no pending workload.
- The length of the last cooling period is still the same even if the remaining workload is smaller.

The upper bound according to UB_x of task τ_i of priority level-i that is requested simultaneously with higher priority tasks with $T(0) = T_{max}$ is given by Equation 18.

$$\begin{cases} w_i^{n+1} &= N(w_i^n) \times x + w \\ R_i^{UB_x} &= w_i^{n+1} = w_i^n \end{cases} \tag{18}$$

where :

- $N(w)$ is the number of cooling periods needed to execute the workload w without exceeding T_{max}:

$$N(w) = \left\lceil \frac{w}{\Delta_h^{UB_x}} \right\rceil$$

- w is the workload of time interval $[0, w_i^n[$:

$$w = \sum_{j \leq i} \left\lceil \frac{w_i^n}{T_j} \right\rceil \times C_j$$

- $\Delta_h^{UB_x}$ is the time to execute jobs and heat up the system from the temperature reached after x time units of cooling to T_{max}:

$$\Delta_h^{UB_x} = \left\lfloor \frac{\ln\left(\frac{b \cdot T_{max} \cdot e^{-b \cdot x} - a}{b \cdot T_{max} - a}\right)}{b} \right\rfloor \tag{19}$$

We choose cooling periods longer or equal to $\Delta_c^{UB_x}$, i.e. $x \geq \Delta_c^{UB_x}$, because it is sufficient to execute at least one time unit without exceeding T_{max} which is close the behavior of PFP_{ASAP} algorithm.

To prove that UB_x upper bounds the actual response time, we first check the case when $x = \Delta_c^{UB_x}$. We know that in the actual schedule, the accumulation of the temperature gained at the end of each heating period, due to the discrete time, is lesser than T_{max}. We denote the gap δ.Thus, this accumulated temperature can be used at least by one heating period which is supposed to be longer as shown in Figure 6. Then, we compare the length of the new heating period Δ'_h (See Equation 20) to the one of UB_x, i.e. $\Delta_h^{UB_x}$, and the total number of cooling/heating produced by UB_x, i.e., $N^{UB_x}(w)$, and the one produced by the actual schedule denoted $N'(w)$.

$$\begin{cases} T_{max} &= (a + (b \cdot T_2 - a)e^{-b \cdot \Delta'_h})/b \\ T_2 &= (T_{max} - \delta)e^{-b \cdot x} \end{cases}$$

$$\Delta'_h = \left\lfloor \frac{\ln\left(\frac{b \cdot (T_{max} - \delta) \cdot e^{-b \cdot x} - a}{b \cdot T_{max} - a}\right)}{b} \right\rfloor \tag{20}$$

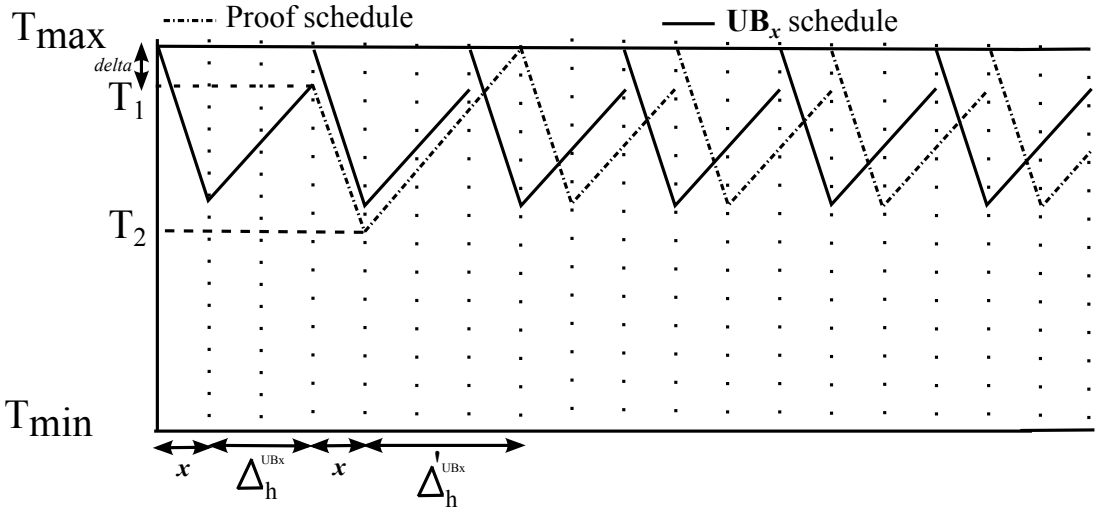


Figure 6: UB_x proof insight

Lemma 1. *For $x = \Delta_c^{UB_x}$, each heating interval of the actual schedule Δ'_h given by Equation 20 is greater or equal than UB_x's ones, i.e. $\Delta_h^{UB_x} \leq \Delta'_h$.*

Proof. Let's suppose that $\Delta_h^{UB_x} > \Delta'_h$, then:

$$\begin{aligned}
&\Delta_h^{UB_x} > \Delta'_h \Rightarrow \\
&\left\lfloor \frac{\ln\left(\frac{b \cdot T_{max} \cdot e^{-bx} - a}{bT_{max} - a}\right)}{b} \right\rfloor > \left\lfloor \frac{\ln\left(\frac{b(T_{max} - \delta)e^{-bx} - a}{bT_{max} - a}\right)}{b} \right\rfloor \\
&\Rightarrow \frac{\ln\left(\frac{bT_{max}e^{-bx} - a}{bT_{max} - a}\right)}{b} > \frac{\ln\left(\frac{b(T_{max} - \delta)e^{-bx} - a}{bT_{max} - a}\right)}{b} \\
&\Rightarrow \frac{bT_{max}e^{-bx} - a}{bT_{max} - a} > \frac{b(T_{max} - \delta)e^{-bx} - a}{bT_{max} - a}
\end{aligned}$$

Knowing that $bT_{max} < a$, then:

$$\begin{aligned}
\Delta_h^{UB_x} > \Delta'_h \quad &\Rightarrow \quad bT_{max}e^{-bx} - a < b(T_{max} - \delta)e^{-bx} - a \\
&\Rightarrow \quad \delta < 0
\end{aligned}$$

Contradiction because $b < 1$, $\delta \geq 0$ and $b \cdot T_{max} < a$.
Therefore, we prove by contradiction that $\Delta_h^{UB_x} \leq \Delta'_h$ □

Lemma 2. *For $x = \Delta_c^{UB_x}$, the number of cooling periods produced by a PFP_{ASAP} actual schedule denoted $N'(w)$ is lesser or equal to the ones produced by UB_x, i.e. $N'(w) \leq N^{UB_x}(w)$.*

Proof. Let's suppose that $N'(w) > N^{UB_x}(w)$. From Lemma 1 we know that $\Delta_h^{UB_x} \leq \Delta'_h$ at least for one time, then:

$$N'(w) = \left\lceil \frac{w - \Delta'_h}{\Delta_h^{UB_x}} \right\rceil + 1 = \left\lceil \frac{w - (\Delta_h^{UB_x} + \delta)}{\Delta_h^{UB_x}} \right\rceil + 1$$

where $\delta \geq 0$. Then, $N'(w)$ can be written as following:

$$N'(w) = \left\lceil \frac{w - \delta}{\Delta_h^{UB_x}} \right\rceil$$

Therefore, if $N'(w) > N^{UB_x}(w)$, then:

$$\begin{aligned} N'(w) > N^{UB_x}(w) &\Rightarrow \left\lceil \frac{w-\delta}{\Delta_h^{UB_x}} \right\rceil > \left\lceil \frac{w}{\Delta_h^{UB_x}} \right\rceil \\ &\Rightarrow \frac{w-\delta}{\Delta_h^{UB_x}} > \frac{w}{\Delta_h^{UB_x}} \\ &\Rightarrow \delta < 0 \end{aligned}$$

Contradiction, because $\Delta'_h \geq \Delta_h^{UB_x}$ and $\delta \geq 0$. Therefore, we prove that $N'(w) \leq N^{UB_x}(w)$ □

Theorem 4. *An upper bound on the worst-case response time for task τ_i in the worst-case scenario described in Section 4.1 can be obtained from a sequence of execution units of τ_i and those of higher priority jobs and the necessary cooling time units where the cooling periods are of x time units and the heating periods are integers and as long as needed to reach T_{max} (without exceeding) after x time units of cooling, as described by Formula 18.*

Proof. To prove this theorem, we have to first the case where $x = \Delta_c^{UB_x}$, and the one where $x > \Delta_c^{UB_x}$.

Case where $x = \Delta_c^{UB_x}$ From Lemma 1 and Lemma 2 we know that $N'(w) \leq N^{UB_x}(w)$, then:

$$\begin{aligned} N'(w) \leq N^{UB_x}(w) &\Rightarrow N'(w_i^n)x + w_i^n \leq N^{UB_x}(w_i^n)x + w_i^n \\ &\Rightarrow w'_i \leq w_i^{UB_x} \Rightarrow R'_i \leq R_i^{UB_x} \\ &\Rightarrow R'_i \leq R_i^{UB_x} \end{aligned}$$

Hence, when $x = \Delta_c^{UB_x}$, UB_x is an upper bound of tasks worst-case response time according to PFP_{ASAP} algorithm.

Case where $x > \Delta_c^{UB_x}$ Lengthening cooling periods by increasing the x parameter is expected to increase the pessimism of UB_x by increasing tasks response time over estimation given by Equation 18. To prove that, one can check if the UB_x's response time computation function is increasing. From Equation 18, the response time function can be written as follows:

$$w_i^{n+1} = \left\lceil \frac{w}{\left\lfloor \frac{\ln\left(\frac{b \cdot T_{max} \cdot e^{-b \cdot x} - a}{b \cdot T_{max} - a}\right)}{b} \right\rfloor} \right\rceil \times x + w$$

Recall that the ceil function is used to ensure a non null integer length for cooling period, and that floor function is used ensure never exceeding T_{max} after a heating period. Even though, these two functions contribute to increase the pessimism of UB_x, removing them does not change the response time function monotonicity. Then, we can study the monotonicity of this new function that we call $f(x)$ by computing its derivative function as follows:

$$f(x) = \frac{w \cdot b \cdot x}{\ln\left(\frac{b \cdot T_{max} \cdot e^{-b \cdot x} - a}{b \cdot T_{max} - a}\right)} + w$$

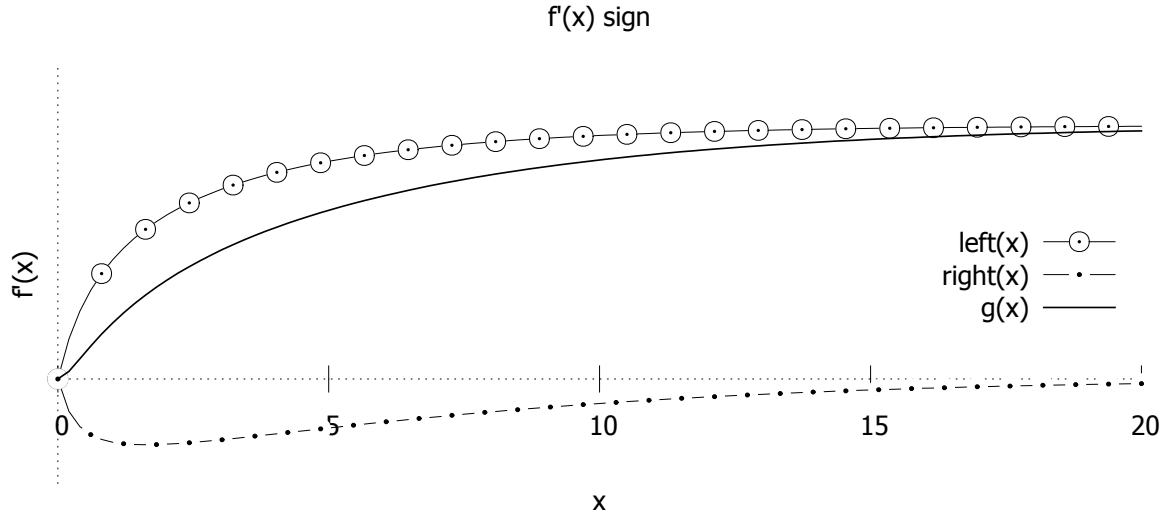


Figure 7: $f'(x)$'s sign

$$f'(x) = \frac{b \cdot \ln\left(\frac{b \cdot T_{max} \cdot e^{-b \cdot x} - a}{b \cdot T_{max} - a}\right) + \frac{b^3 \cdot T_{max} \cdot x \cdot e^{-b \cdot x}}{b \cdot T_{max} \cdot e^{-b \cdot x} - a}}{\left(\ln\left(\frac{b \cdot T_{max} \cdot e^{-b \cdot x} - a}{b \cdot T_{max} - a}\right)\right)^2}$$

Due to the lack of space, we do not show the whole study of $f(x)$'s sign, we do this only with deductions. Thus, the sign of $f'(x)$ depends only on the numerator part of the fraction, we denote this part $g(x)$. Then, knowing that $g(0) = \ln(a/(a - bT_{max})) > 0$, we can say that $g(x)$ is positive in interval $[1, +\infty[$ because first the left part (the logarithm part) is positive, because of the logarithm function, and increasing, due to to the reverse exponential function; and second the right part is negative, because $b \cdot T_{max} < a$, and slightly decreasing from 0 for a while and then it increases asymptotically to 0 as shown in Figure 7. Therefore, $f'(x)$ is positive which means that $f(x)$ is increasing in interval $[1, +\infty[$, and then tasks response time according to UB_x increases when x is increasing.

Therefore we prove that UB_x upper bounds the actual PFP_{ASAP} worst-case response time. □

5.2.3 Lower bound ($LB_{x=1}$):

Knowing that the actual schedule respects the discrete time constraint, we can compute a lower bound of the actual tasks worst-case response time by violating this constraint, i.e., allowing non-discrete execution times (See 8). The following points summarize the behavior of $LB_{x=1}$:

- Cooling down the system for one time unit. This is sufficient because continous time allows executing less than one time unit.
- Then, executing jobs and heating up the system until T_{max} is reached or there is no pending workload; this length of period is not necessarily integer.
- Repeat this cycle cooling-heating until there is not pending workload

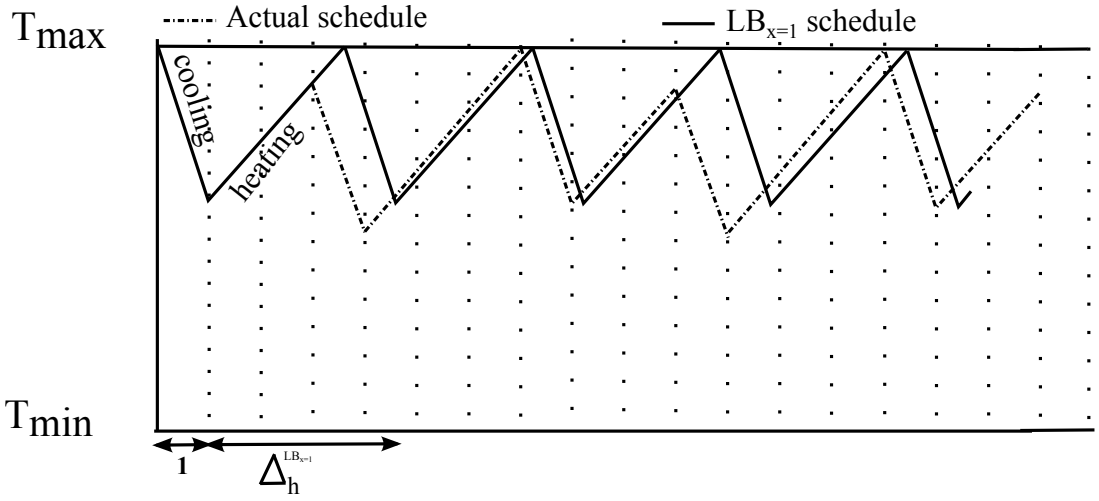


Figure 8: Lower Bound ($LB_{x=1}$)

The lower bound of task τ_i of priority level-i that is requested simultaneously with higher priority tasks with $T(0) = T_{max}$ is given by Equation 21

$$\begin{cases} w_i^{n+1} &= N(w_i^n) + w \\ R_i &= w_i^{n+1} = w_i^n \end{cases} \tag{21}$$

where :

- $N(w)$ is the number of cooling periods needed to execute the workload w without exceeding T_{max}:

$$N(w) = \left\lceil \frac{w}{\Delta_h^{LB_{x=1}}} \right\rceil \tag{22}$$

- w is the workload of time interval $[0, w_i^n[$.
- $\Delta_h^{LB_{x=1}}$ is the time to execute jobs and heat up the system from the temperature reached after one time unit of cooling to exactly T_{max}:

$$\Delta_h^{LB_{x=1}} = \frac{\ln\left(\frac{b \cdot T_{max} \cdot e^{-b} - a}{b \cdot T_{max} - a}\right)}{b} \tag{23}$$

Conjecture 1. *A lower bound on the worst-case response time for task τ_i in the worst-case scenario described in Section 4.1 can be obtained from a sequence of execution units of τ_i and higher priority jobs and the necessary cooling time units where the cooling periods are of one time unit and the heating periods are continuous (not necessarily integers) and as long as needed to reach exactly T_{max} after one time unit of cooling, as described by Formula 21.*

We do not prove the conjecture due to space limitation.

5.2.4 $UB_{T_{min}}$ vs. UB_x

The tightness of the upper bound UB_x relative to $UB_{T_{min}}$ depends on the parameter x. In fact, the greater x is, the more pessimistic UB_x is because of the nature of the

cooling function which is asymptotically decreasing to T_A. Then, for small values of x, UB_x is tighter and for great values $UB_{T_{min}}$ is. The experiments presented in Section 6 demonstrates the differences in practice between UB_x and $UB_{T_{min}}$ in term of tightness and complexity.

5.2.5 Utilization bound

Under thermal constraints, cooling periods are needed to prevent the system to exceed T_{max}. This means that for a certain processor utilization, which determines the time in which the processor is occupied, more time is needed for cooling which means that the processor cannot be used at 100%. One can use this idea to propose a new maximum processor utilization that can respect the thermal constraints. In the following we discuss utilization bounds that consider cooling time.

Maximum utilization Without considering thermal constraints, task sets cannot be feasible with a processor utilization greater than 100% for monoprocessor platforms. Furthermore, knowing that respecting the thermal constraints needs to add some cooling time, then, it is obvious that a task sets with a processor utilization of 100% cannot be feasible with PFP_{ASAP}. To compute the maximum supportable processor utilization that take into account cooling time, one can use the idea of over estimating response times, by over estimating the cooling time needed to execute the workload of one hyper-period. We can use for instance the idea of UB_x to compute an upper bound for the maximum supportable processor utilization.

Lemma 3. *An upper bound of the processor utilization $U = \sum_{1 \leq i \leq n} C_i/T_i$ for thermal-aware fixed-priority task sets can be obtained by Equation 24.*

$$U \leq \frac{\Delta_h^{UB_x}}{\Delta_h^{UB_x} + x} \tag{24}$$

Proof. We first upper bound the workload of one hyper period with UB_x then we compute the corresponding processor utilization, and finally we compute the maximum achievable utilization. The workload of an hyper-period L can be obtained by multiplying L by the processor utilization U. Then, we can replace w by $U \cdot L$ in Equation 18 to compute the time needed (cooling + workload) to satisfy the workload $U \cdot L$. Finally, we can compute the new utilization u^*, that considers cooling time, by dividing the time demand (cooling + workload) by the available time L, Equation 25 shows how to compute U^*.

$$U^* = \frac{\left\lceil \frac{U \cdot L}{\Delta_h^{UB_x}} \right\rceil \cdot x + U \cdot L}{L} \tag{25}$$

For a task set to be feasible, the new utilization U^* must be lesser than 1 because the available time must be greater or equal to the time demand. Then,

$$\begin{aligned} U^* \leq 1 \quad &\Rightarrow \quad \frac{\left\lceil \frac{U \cdot L}{\Delta_h^{UB_x}} \right\rceil \cdot x + U \cdot L}{L} \leq 1 \\ &\Rightarrow \quad \frac{\frac{U \cdot L \cdot x}{\Delta_h^{UB_x}} + U \cdot L}{L} \leq 1 \\ &\Rightarrow \quad U \cdot \left(\frac{x}{\Delta_h^{UB_x}} + 1 \right) \leq 1 \\ &\Rightarrow \quad U \leq \frac{\Delta_h^{UB_x}}{\Delta_h^{UB_x} + x} \end{aligned}$$

□

Liu and Layland bound We can use the same reasoning as the above utilization upper bound to propose a sufficient and pessimistic feasibility test based on Liu and Layland bound. In fact, we can compare the total time utilization (processor and cooling) to Liu and Layland bound.

Lemma 4. *An upper bound of the processor utilization $U = \sum_{1 \leq i \leq n} C_i/T_i$ for thermal-aware fixed-priority task sets with implicit deadlines can be obtained by Equation 26.*

$$U \leq \frac{\Delta_h^{UB_x} \cdot n(\sqrt[n]{2} - 1)}{\Delta_h^{UB_x} + 1} \tag{26}$$

Proof. To prove this Lemma we just have to compare the over estimated utilization U^* given by Equation 25 to Liu and Layland bound.

$$\begin{aligned} U^* \leq n(\sqrt[n]{2} - 1) \quad &\Rightarrow \quad \frac{\left\lceil \frac{U \cdot L}{\Delta_h^{UB_x}} \right\rceil \cdot x + U \cdot L}{L} \leq n(\sqrt[n]{2} - 1) \\ &\Rightarrow \quad \frac{\frac{U \cdot L \cdot x}{\Delta_h^{UB_x}} + U \cdot L}{L} \leq n(\sqrt[n]{2} - 1) \\ &\Rightarrow \quad U \cdot \left(\frac{x}{\Delta_h^{UB_x}} + 1 \right) \leq n(\sqrt[n]{2} - 1) \\ &\Rightarrow \quad U \leq \frac{\Delta_h^{UB_x} \cdot n(\sqrt[n]{2} - 1)}{\Delta_h^{UB_x} + x} \end{aligned}$$

□

6 Performance Evaluation

In this section, we present the results of an empirical investigation, examining the effectiveness of our sufficient schedulability tests.

6.1 Taskset generation

To perform these experiments, we randomly generated 100000 task sets, varying the processor utilization. We varied U in the range $[0.05, 1]$ in steps of 0.05. Hence we obtained 5000 distinct task sets for each U step. Each tasks set comprised 10 tasks. The thermal parameters was set as following, $T_{max} = 32\ C^\circ$, $b = 0.228$, and $a = \beta_0 \cdot S^3 = 8$. These parameters are the ones of the whole system (including an eventual cooling device) and correspond to a classical Intel Pentium processor parameters [11]. The task parameters were randomly generated as follows: task processor utilization ($U_i = C_i/T_i$) using the *U-Unifast Discard* algorithm [7], and periods randomly generated between 2 and 25200 time units with a hyper-period limitation technique [14]. Task deadlines were implicit.

We used YARISS as a simulation environment [8] which respects the following hypotheses: discrete time (all scheduling operations are performed before or after one time unit), the heating behavior follows the Fourier's law (See Equation 6) and temperature values are real numbers.

6.2 Schedulability tests investigated

We investigated the performance of the following schedulability tests.

SIM: is an empirical necessary and sufficient test based on simulating the schedule of PFP_{ASAP} over more than one hyper-period, starting with synchronous release and the

maximum temperature level which corresponds to the worst-case scenario discussed in Section 4.1.

$UB_{T_{min}}$: the sufficient test presented in Section 5.2.1, we consider that $T_{min} = 1\ C°$.

UB_x: the sufficient test presented in Section 5.2.2, the parameter x is varied from 1 to 18.

$LB_{x=1}$: the necessary test presented in Section 5.2.3.

CFP: the exact test for fixed priority ignoring thermal and energy constraints. This was used to provide a schedulability bound, considering only processing time.

UTZ: the necessary condition described in Section 5.2.5.

LnL: the sufficient condition described in Section 5.2.5.

6.3 Experiments

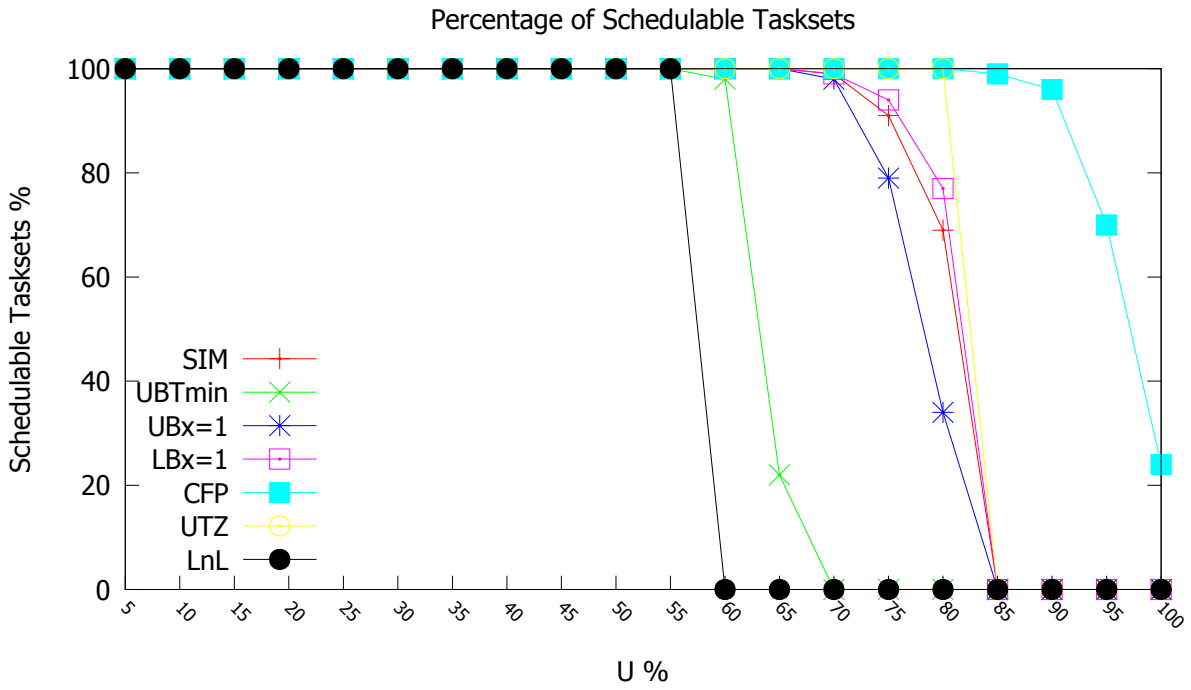


Figure 9: Percentage of Task sets schedulable over U variation

Figure 9 shows how the percentage of task sets that are deemed schedulable by each of the tests varies with processor utilization. The CFP test has notionally the highest performance since it is widely optimistic and ignores thermal considerations. When temperature is considered, UTZ, $LB_{x=1}$ provide necessary tests, upper bounding the number of task sets that are proved to be schedulable by the exact empirical test SIM. We observe that the results confirm that $UB_{T_{min}}$ and UB_x provide sufficient schedulability tests and that for $x = 1$, UB_x is tighter bound than $UB_{T_{min}}$, with a larger improvement at higher utilization levels. Furthermore, this experiment confirms also the validity of the utilization bounds given in Section 5.2.5 and 5.2.5, withe considered thermal parameters, the maximum achievable utilization for 10 tasks is 80% for UTZ and the adapted Liu and Layland bound is 57%.

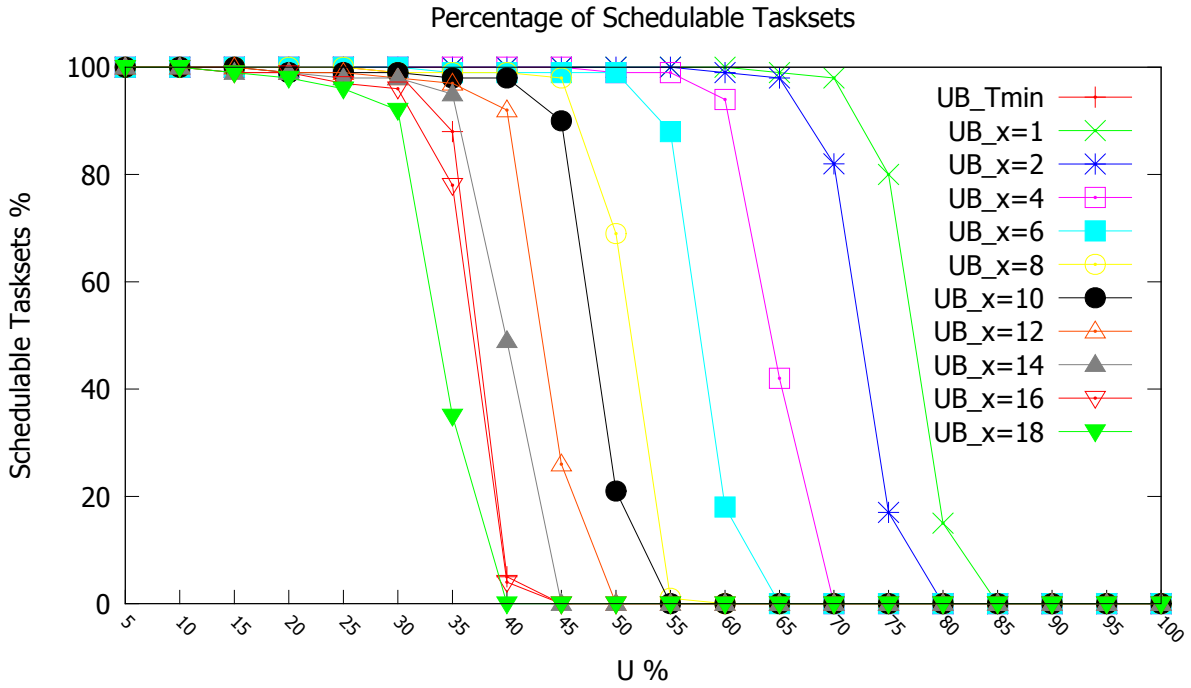


Figure 10: Schedulability by Varying x

Figure 10 compares the pessimism of $UB_{T_{min}}$ based shcedulability test to UB_x's one by varying the x parameter. We observe that UB_x stays less pessimistic than $UB_{T_{min}}$ for small values ($1 \leq x \leq< 14$ in this experiment), however, it becomes more pessimistic starting from $x = 14$. This result is as expected because the loner cooling periods are, the slower temperature decreases and the longer response times are.

Figure 11 shows average deviation of bounds from the exact response time given by simulations over processor utilization. The upper bounds have positive values and lower bounds have negative values (the deviation of SIM is 0 because it it gives the exact response time). We can see that deviation of UB_x, $LB_{x=1}$ and CFP are still stable over utilization variation in contrast of $UB_{T_{min}}$ which behaves badly when utilization goes high. We notice also that when $x = 1$, UB_x and $LB_{x=1}$ are very close to the actual response time which makes them very interesting tools for approximate schedulability analysis. However, increasing x leads UB_x to be less precise, when $x > 14$, UB_x behaves as bad as $UB_{T_{min}}$ or worse.

We also perform further set of experiments showing how schedulability depends on different parameters, including deadlines model and the number of tasks, via the Weighted Schedulability Measure [6]. The conclusions are the following. By varying relatives deadlines, we observe that all of the schedulability tests are influenced by the tightness of deadlines to a similar degree, with heavily constrained deadlines having significant impact on schedulability in all cases. We observe also that when we measure the trade-off between tests rate of schedulability and their overhead, the loss of schedulablity is higher than the gain of overhead. More details about these experiments are available in the Appendix.

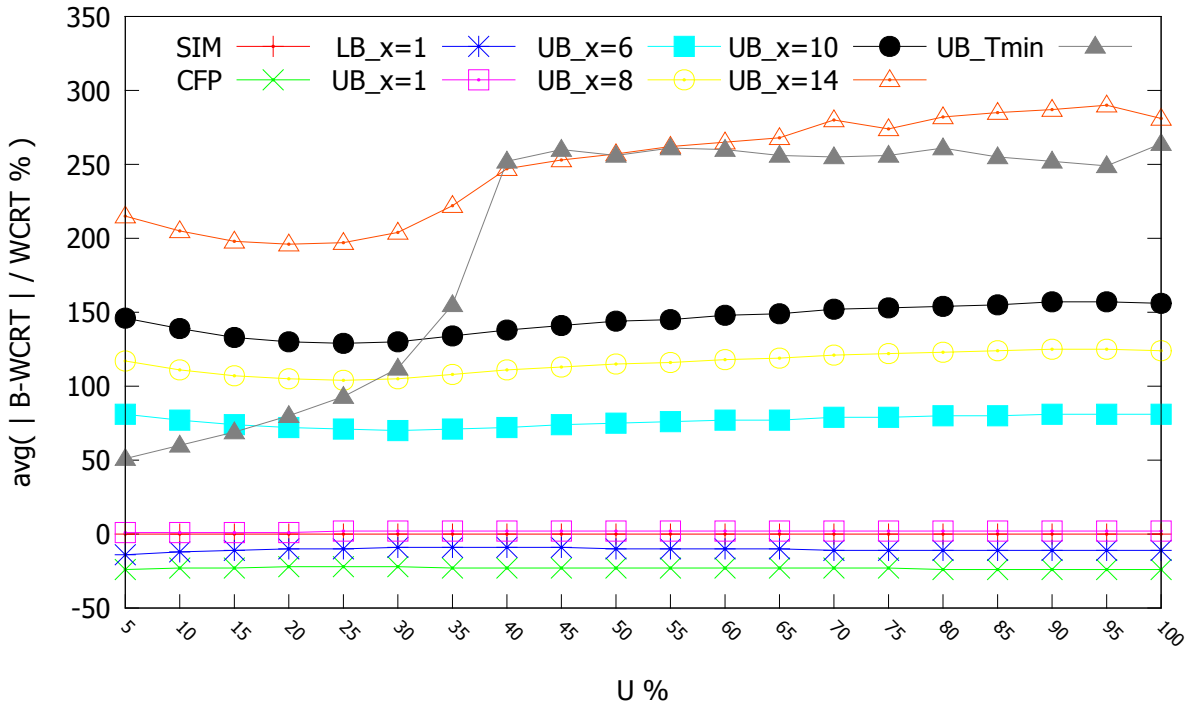


Figure 11: Bounds tightness over U variation

7 Conclusion

In this paper, we addressed the problem of fixed-priority real-time scheduling for thermal-aware systems, where both time and thermal constraints have to be met. Previous research showed that the scheduling policy PFP_{ASAP} is optimal among all fixed-priority scheduling algorithms for non-concrete energy-harvesting systems. The main contributions of this paper are as follows: we adpated PFP_{ASAP} algorithm to the thermal-aware model, we proved its optimality and we proposed two schedulability tests based on response-time upper bounds $UB_{T_{min}}$ and UB_x which is a parametric bound. Finally we performed simulations to validate the theoretical results. As future work, we plan to study deeply the possible similarities between energy-harvesting model and the thermal-aware's one and to explore more adaptable and extensible results of each model.

A Proof of Theorem 3

In the computation of $UB_{T_{min}}$, the cooling periods are as long as needed to cool down the system from T_{max} to T_{min} and the execution periods are as long as needed to heat up the system from T_{min} to T_{max}. Furthermore, we know that in the actual schedule produced by PFP_{ASAP} algorithm for task τ_i in the critical instant, a cooling period is as long as needed to execute at least one time unit. The length of this period is shorter or equal than a cooling period of $UB_{T_{min}}$, and thus, by repeating these short cooling/heating cycles, it cools down the system faster than fewer and longer cooling periods. Then, to prove Theorem 3, we suppose that the actual schedule has at least one short cycle and we compare the two response times by using Fourier's law described by Equation 1.

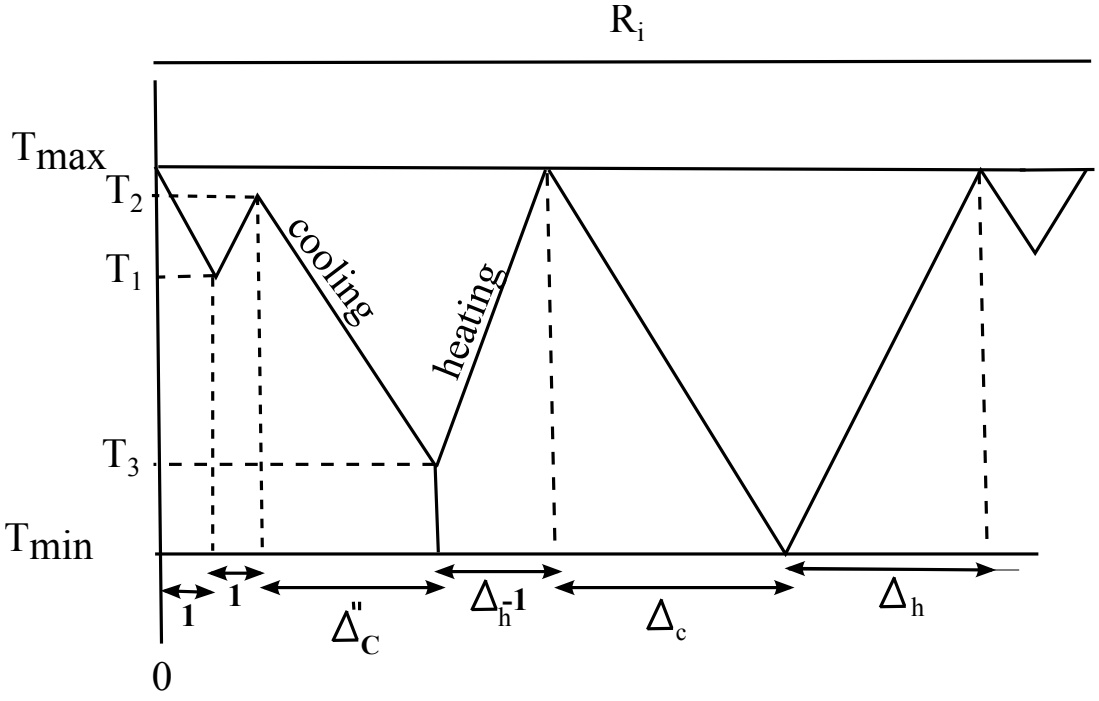


Figure 12: $UB_{T_{min}}$ proof insight

As described in Figure 12, we suppose that the system cools down for one time unit from T_{max} to T_1 (See Equation 27), and then heats up for one time unit reaching temperature T_2 (See Equation 27). After that, we follow the same schedule as $UB_{T_{min}}$ by finishing the cooling period needed to finish the workload $\Delta_h - 1$ and reach T_{max} (See Equation 28). The temperature of the system after this cooling period is denoted T_3 (See Equation 27).

$$\begin{cases} T_1 = T_{max} \cdot e^{-b} \\ T_2 = \dfrac{a}{b} + \left(T_1 - \dfrac{a}{b}\right) e^{-b} \\ T_{max} = \dfrac{a}{b} + \left(T_3 - \dfrac{a}{b}\right) e^{-b \cdot (\Delta_h - 1)} \\ T_3 = \dfrac{a + (b \cdot T_{min} - a) \cdot e^{-b}}{b} \\ T_3 = T_2 \times e^{-b \cdot \Delta''_c} \\ \Delta''_c = \left\lceil \dfrac{\ln(T_2) - \ln(T_3)}{b} \right\rceil \end{cases} \tag{27}$$

$$\Delta''_c = \left\lceil \frac{\ln\left(\frac{a + \left(b \cdot T_{max} \cdot e^{-b} - a\right) \cdot e^{-b}}{a + (b \cdot T_{min} - a) \cdot e^{-b}}\right)}{b} \right\rceil \tag{28}$$

Let's now compare the response time given by this assumption and the one given by $UB_{T_{min}}$. Knowing that the first step of cooling, i.e., the first time unit or between T_{max} and T_1, is the same for both cases, and by supposing that after reaching T_{max} at the end of the first heating cycle, i.e., form temperature T_3 to T_{max}, the actual schedule follows the same pattern as $UB_{T_{min}}$ cycles (See Figure 12), if $UB_{T_{min}}$ does not upper bound the actual response time, then the total cooling time needed for Δ_h workload in the actual

schedule is greater than the one of $UB_{T_{min}}$, i.e., $\Delta_c'' + 1 > \Delta_c$. Knowing that Δ_c can be written as Equation 29

$$\begin{aligned} \Delta_c &= 1 + \left\lceil \frac{\ln(T_1) - \ln(T_{min})}{b} \right\rceil \\ &= 1 + \left\lceil \frac{\ln\left(\frac{T_{max} \cdot e^{-b}}{T_{min}}\right)}{b} \right\rceil \end{aligned} \tag{29}$$

Theretofore, if $\Delta_c'' + 1 > \Delta_c$, then,

$$\begin{aligned} &\Delta_c'' + 1 > \Delta_c \Rightarrow \\ &\quad \left\lceil \frac{\ln\left(\frac{a + \left(b \cdot T_{max} \cdot e^{-b} - a\right) \cdot e^{-b}}{a + (b \cdot T_{min} - a) \cdot e^{-b}}\right)}{b} \right\rceil > \left\lceil \frac{\ln\left(\frac{T_{max} \cdot e^{-b}}{T_{min}}\right)}{b} \right\rceil \\ &\Rightarrow \ln\left(\frac{a + \left(b \cdot T_{max} \cdot e^{-b} - a\right) \cdot e^{-b}}{a + (b \cdot T_{min} - a) \cdot e^{-b}}\right) > \ln\left(\frac{T_{max} \cdot e^{-b}}{T_{min}}\right) \\ &\Rightarrow \frac{a + \left(b \cdot T_{max} \cdot e^{-b} - a\right) \cdot e^{-b}}{a + (b \cdot T_{min} - a) \cdot e^{-b}} > \frac{T_{max} \cdot e^{-b}}{T_{min}} \\ &\Rightarrow \frac{a + (b \cdot T_{min} - a) \cdot e^{-b}}{a + (b \cdot T_{max} \cdot e^{-b} - a) \cdot e^{-b}} < \frac{T_{min}}{T_{max} \cdot e^{-b}} \\ &\Rightarrow 1 - \frac{b \cdot e^{-b}(T_{max}\dot{e}^{-b} - T_{min})}{a + (b \cdot T_{max} \cdot e^{-b} - a) \cdot e^{-b}} < 1 - \frac{(T_{max} \cdot e^{-b} - T_{min})}{T_{max} \cdot e^{-b}} \\ &\Rightarrow b \cdot e^{-2b} \cdot T_{max} > a + (b \cdot T_{max} \cdot e^{-b} - a) \cdot e^{-b} \\ &\Rightarrow b \cdot e^{-2b} \cdot T_{max} - b \cdot e^{-2b} \cdot T_{max} > a(1 - e^{-b}) \\ &\Rightarrow e^{-b} > 1 \; \textit{contradiction}! \end{aligned}$$

Therefore, we prove by contradiction that $\Delta_c'' + 1 \leq \Delta_c$, and then $UB_{T_{min}}$ upper bounds the actual worst-case response time.

B Weighted Schedulability

As well as processor utilization, task set schedulability is dependent on a number of other key parameters, including: tasks deadlines model and the number of tasks. Evaluating all possible combinations of these parameters is not possible, instead, the evaluation in this section varies one parameter at a time, with the results presented in terms of the weighted schedulability measure [6].

The figures in this section show the weighted schedulability measure $W_y(p)$ for each schedulability test y as a function of parameter p. For each value of p, this measure combines results for all of the task sets Γ generated for all of a set of equally spaced utilization levels (5% to 100% in steps of 5%). Deadlines are constrained or implicit.

Let $S_y(\Gamma, p)$ be the binary result (1 or 0) of schedulability test y for a task set Γ with parameter value p:

$$W_y(p) = \left(\sum_{\forall \Gamma} U_\Gamma \times S_y(\Gamma, p) \right) / \sum_{\forall \Gamma} U_\Gamma \tag{30}$$

where U_Γ is the processor utilization of taskset Γ. The weighted schedulability measure reduces what would otherwise be a 3-dimensional plot to 2 dimensions [6]. Weighting the

individual schedulability results by task set utilization reflects the higher value placed on being able to schedule higher utilization task sets.

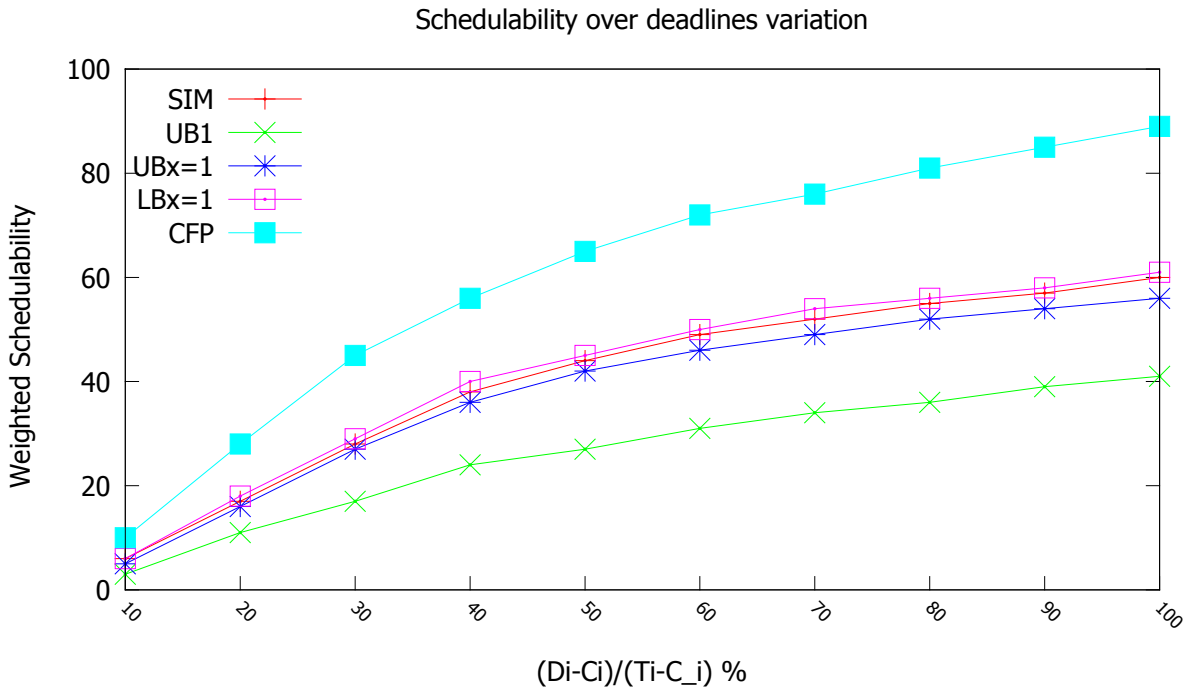


Figure 13: Varying relative deadlines in $[C_i, T_i]$

Figure 13 shows the impact of constrained deadlines on performance. Here we vary the deadlines from heavily constrained where $D_i - C_i$ is 10% of $T_i - C_i$ to 100% of $T_i - C_i$ (i.e. implicit deadlines). We observe that all of the schedulability tests are influenced by the tightness of deadlines to a similar degree, with heavily constrained deadlines having significant impact on schedulability in all cases.

Figure 14 shows the impact of the number of tasks on the effectiveness of the feasibility tests. We can see that they the number of tasks does not alter the feasibility rate of all the bounds.

Figure 15 shows the trade-off of schedulability and overhead between UB_x and $UB_{T_{in}}$ over parameter x variation. The shown percentages represent the percentage of gained or lost schedulability and overhead comparing to $UB_{T_{min}}$. For example, for $x = 1$, The schedulability test based on $UB_{x=1}$ has more than 100% of schedulability and almost 40% more overhead than $UB_{T_{in}}$. Positive values mean better performances than $UB_{T_{in}}$ and negative value mean worse performances. We observe that increasing parameter x decreases the percentage of schedulable task sets, however, it creases the test complexity or overhead. We observe also that the loss of schedulablity is higher than the gain of overhead.

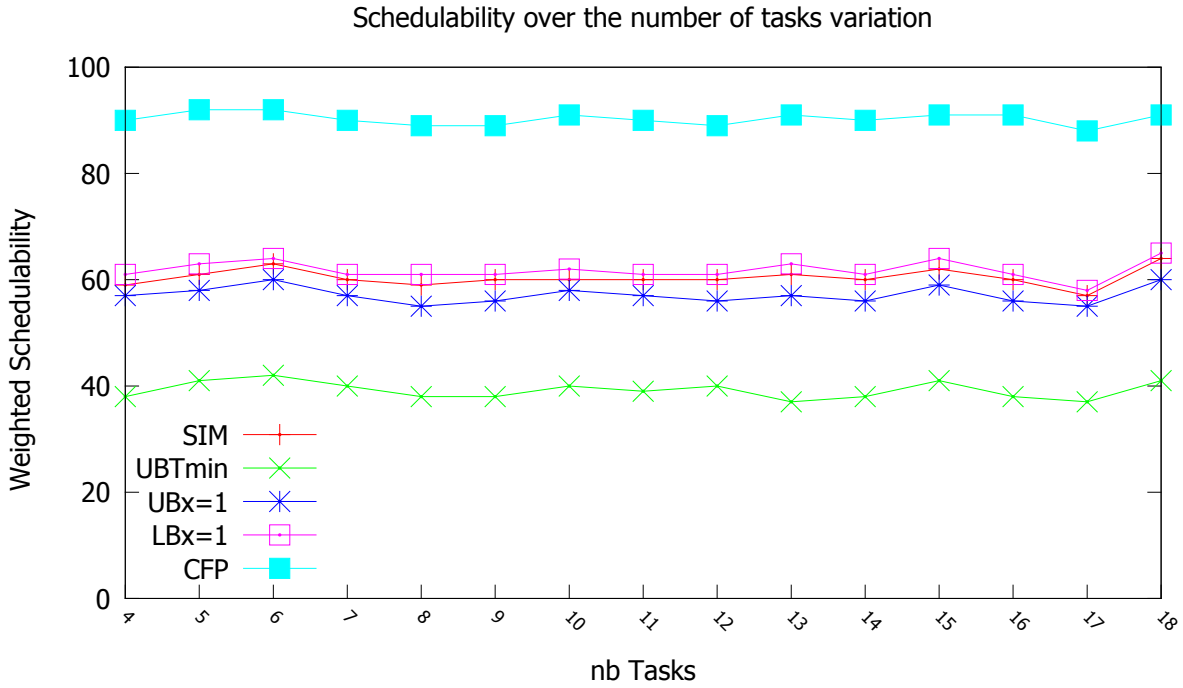


Figure 14: Varying the number of tasks

References

[1] Y. Abdeddaïm, Y. Chandarli, and D. Masson. The Optimality of PFP_{asap} Algorithm for Fixed-Priority Energy-Harvesting Real-Time Systems. In *ECRTS*, July 2013.

[2] M. Ahmed, N. Fisher, S. Wang, and P. Hettiarachchi. Minimizing peak temperature in embedded real-time systems via thermal-aware periodic resources. In *Sustainable Computing: Informatics and Systems*. Elsevier, 2011.

[3] R. Ahmed, P. Ramanathan, and K. Kewal. On thermal utilization of periodic task sets in uni-processor systems. In *RTCSA*. IEEE Computer Society, 2013.

[4] R. Ahmed, P. Ramanathan, and K. Kewal. On thermal utilization of periodic task sets in uni-processor systems. In *ECRTS*. IEEE Computer Society, 2014.

[5] A. Allavena and D. Mossé. Scheduling of frame-based embedded systems with rechargeable batteries. In *WPMRTES (with RTAS)*.

[6] A. Bastoni, B. B. Brandenburg, and J. H. Anderson. Cache-related preemption and migration delays: Empirical approximation and impact on schedulability. In *OSPERTA (with RTAS)*, 2010.

[7] E. Bini and G. C. Buttazzo. Measuring the Performance of Schedulability Tests. *Real-Time Systems*, 30(1-2):129–154, May 2005.

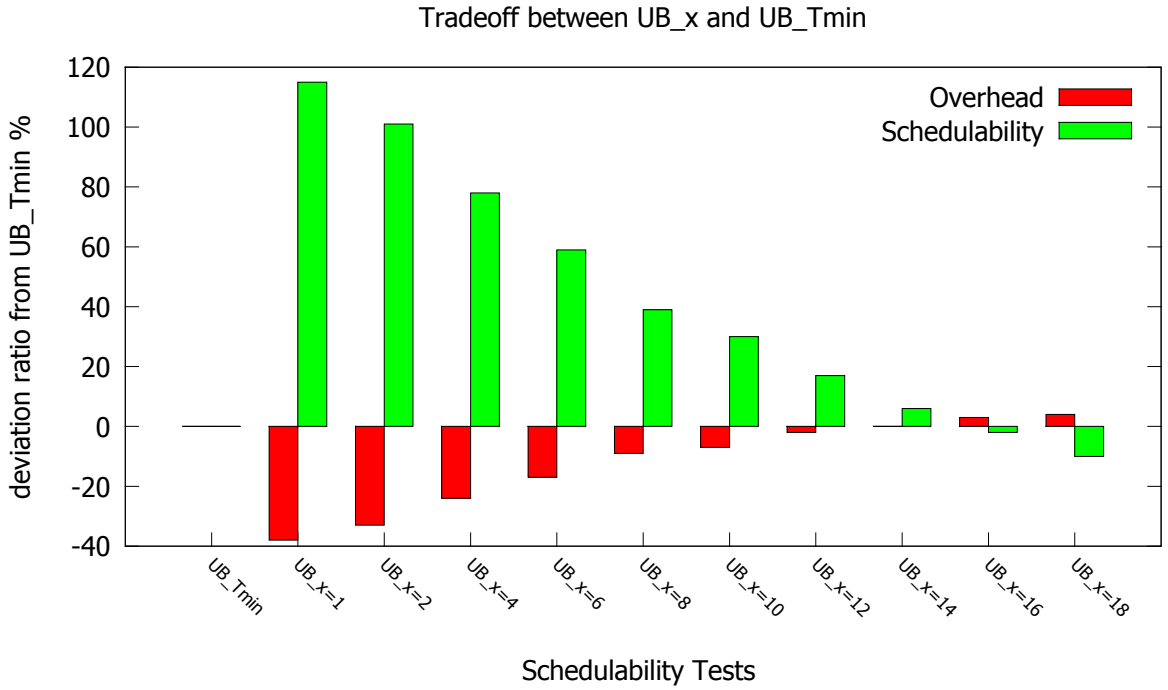


Figure 15: Trade-off of schedulability and overhead between UB_x and $UB_{T_{in}}$

[8] Y. Chandarli, F. Fauberteau, D. Masson, S. Midonnet, and M. Qamhieh. YARTISS : A Tool to Visualize, Test, Compare and Evaluate Real-Time Scheduling Algorithms. In *WATERS*, 2012.

[9] J.-J. Chen, C.-M. Hung, and T.-W. Kuo. On the minimization fo the instantaneous temperature for periodic real-time tasks. In *RTAS*, pages 236–248. IEEE Computer Society, 2007.

[10] J.-J. Chen, S. Wang, and L. Thiele. Proactive speed scheduling for real-time tasks under thermal constraints. In *RTAS*, pages 141–150. IEEE Computer Society, 2009.

[11] P. M. Hettiarachchi, N. Fisher, M. Ahmed, L. Y. Wang, S. Wang, and W. Shi. The design and analysis of thermal-resilient hard-real-time systems. In *RTAS*, pages 67–76, 2012.

[12] L. Joseph C., M. Kimberly A., P. Madhavi, and P. Otto J. Stimulus-activated changes in brain tissue temperature in the anesthetized rat. In *Metabolic Brain Disease*, pages 225–237, 1989.

[13] G. Lazzi. Thermal effects of bioimplants. In *IEEE Engineering in Medicine and Biology Magazine*, pages 75–81. IEEE Computer Society, 2005.

[14] C. Macq and J. Goossens. Limitation of the hyper-period in real-time periodic task set generation. In *International conference on real-time systems*, 2001.

[15] C. Moser, D. Brunelli, L. Thiele, and L. Benini. Real-time scheduling with regenerative energy. In *ECRTS*, 2006.

[16] G. Quan and Y. Zhang. Leakage aware feasibility analysis for temperature-constrained hard real-time periodic tasks. In *ECRTS*, pages 207–216. IEEE Computer Society, 2009.

[17] G. Quan, Y. Zhang, W. Wiles, and P. Pei. Guaranteed scheduling for repetitive hard real-time tasks under the maximal temperature constraint. In *CODES+ISSS*, pages 267–272. ACM, 2008.

[18] P. Ruggera, D. Witters, G. von Maltzahn, and H. Bassen. In vitro assessment of tissue heating near metallic medical implants by exposure to pulsed radio frequency diathermy. In *Physics in Medicine and Biology*, pages 2919–2928, 2003.

[19] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan. Temperature-aware microarchitecture: Modeling and implementation. *ACM Transactions on Architecture and Code Optimization (TACO)*, 1(1):94–125, 2004.

[20] K. Sohee, T. P., N. R.A., and S. F. Thermal impact of an active 3-d microelectrode array implanted in the brain. In *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, pages 493–501. IEEE Computer Society, 2007.

[21] S. Wang, Y. Ahn, and R. Bettati. Schedulability analysis in hard real-time systems under thermal constraints. *Real-Time Syst.*, 46(2):160–188, 2010.

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

[23] S. Wang and R. Bettati. Reactive speed control in temperature-constrained real-time systems. *Real-Time Syst.*, 39(1-3):73–95, 2008.