

# Energy Saving Exploiting the Limited Preemption Task Model

Mario Bambagini, Giorgio Buttazzo  
{mario.bambagini, giorgio.buttazzo}@sssup.it  
Scuola Superiore Sant'Anna, Pisa, Italy

Marko Bertogna  
marko.bertogna@unimore.it  
University of Modena and Reggio Emilia, Italy

**Abstract**—Limited preemptive scheduling has been shown to dominate both non-preemptive and fully preemptive scheduling under fixed priority systems, as far as schedulability is concerned. This paper suggests the use of DVS and DMP techniques under limited preemptive scheduling to further reduce energy consumption with respect to a fully preemptive or non-preemptive approach.

## I. INTRODUCTION

Two widely used techniques to save energy are *Dynamic Voltage and Frequency Scaling* (DVFS) and *Dynamic Power Management* (DPM). DVFS approaches decrease the voltage and/or frequency of the processor to reduce energy consumption. On the other hand, DPM techniques aim at switching the processor in a low-power inactive state as long as possible, but still guaranteeing the task real-time constraints.

Most of the approaches proposed in the last years consider a fully preemptive task model. Bini et al. [1] considered a realistic model with discrete frequencies and non-negligible overhead, and proposed to achieve the optimal speed by selecting and modulating between two available frequencies. The modulation between active and sleep state was investigated by Huang et al. [2] focusing mostly on a DPM approach. Awan and Petters [3] proposed to accumulate the task execution slack to switch the processor off during such intervals. Rowe et al. [4] presented a technique that harmonizes task periods to clusters task execution such that processor idle times are lumped together. In these papers, a limited preemptive approach was never explored to improve energy saving.

An approach to limit preemptions consists in dividing each task into a set of non-preemptive chunks by inserting a number of fixed preemption points in specific parts of the code.

As Bertogna et al. [5] have shown, limited preemptive methods increase schedulability with respect to fully preemptive and non-preemptive models, even when preemption cost is negligible. Moreover, Bril et al. [6] presented an exact schedulability analysis for fixed priority scheduling with deferred preemption.

To the best of our knowledge, only Maxim et al. [7] have addressed the problem of exploiting the limited preemption model to further reduce the energy consumption and the number of preemptions by merging the last two chunks of a task and adjusting the speed of the previous ones.

This paper suggests to combine limited preemptive scheduling with DVS and DMP techniques to further reduce en-

ergy consumption with respect to fully preemptive or non-preemptive approaches.

## II. MODEL

We consider a set  $\Gamma$  of  $n$  sporadic tasks [8]  $\tau_1, \tau_2, \dots, \tau_n$  executing upon a single processor platform with preemption support. The processor can vary the running speed  $s$ , defined as the normalized frequency with respect to the nominal frequency,  $s = \frac{f}{f_{nom}}$ . The speed set is assumed to be finite and composed by  $m$  different speeds  $s_1, s_2, \dots, s_m$  sorted in ascending order, thus  $s_{min} = s_1$ ,  $s_{nom} = 1$  and  $s_{max} = s_m$ . The selected speed is set at the system start and is never changed.

Each sporadic task  $\tau_i$  ( $1 \leq i \leq n$ ) is characterized by a worst-case execution time (WCET)  $C_i(s)$ , which is function of the speed, a relative deadline  $D_i$ , and a minimum inter-arrival time  $T_i$ , also referred to as the period. The WCET value of  $\tau_i$  depends on the actual speed of the processor and is computed as  $C_i(s) = \frac{C_i^{nom}}{s}$ , where  $C_i^{nom}$  denotes the amount of required time to execute  $\tau_i$  at the nominal speed (under this assumption  $C_i(s_{nom}) = C_i^{nom}$ ). Each task generates an infinite sequence of jobs, with the first job arriving at any time and subsequent arrivals separated by at least  $T_i$  units of time.

When a task  $\tau_i$  is executed with deferred preemptions,  $q_i^{max}$  and  $q_i^{last}$  denote the length of the largest and the last non-preemptive region of  $\tau_i$ , respectively.

Note that, under non-preemptive scheduling, tasks may share mutually exclusive resources without introducing additional blocking, as long as critical sections are entirely included inside the non-preemptive chunks.

## III. MOTIVATIONAL EXAMPLE

In order to show the benefit of the limited preemption to save energy, let us consider an example with two speeds  $s_{min} = 0.5$  and  $s_{max} = 1$  and two tasks,  $\tau_1$  and  $\tau_2$ , with the following parameters:  $C_1 = 30$ ,  $T_1 = D_1 = 80$ ,  $C_2 = 25$  and  $T_2 = D_2 = 200$  (computation times are referred to  $s_{nom} = s_{max}$ ). Tasks are scheduled using Deadline Monotonic and, for the sake of simplicity, preemption costs are considered negligible. The utilization factor at  $s_{max}$  is 0.5 and the task set results feasible with fully-preemptive, non-preemptive and limited preemptive models. Switching to  $s_{min}$ , the computation times become  $C_1 = 60$  and  $C_2 = 50$  causing a global utilization  $U = 1$ . Although using both the

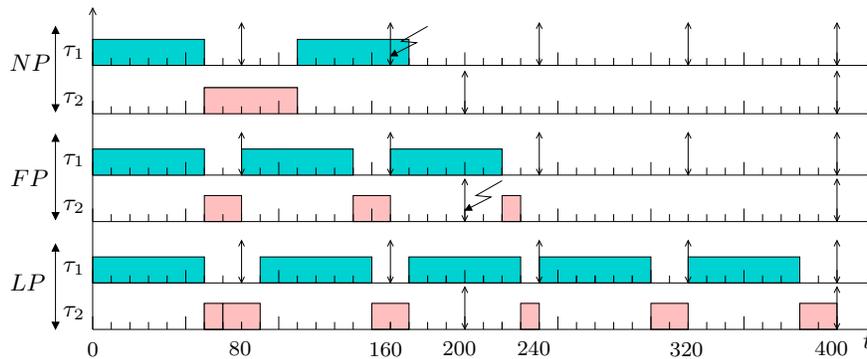


Fig. 1. Schedules at  $s = 0.5$  using Non-Preemptive (NP), Fully-Preemptive (FP) and Limited Preemptive (LP) task models.

fully-preemptive and non-preemptive models the task set results unfeasible, a scheduler exploiting the limited preemptive model with deferred preemptions is able to guarantee the task set feasibility (Figure 1). More precisely, with the limited preemptive model,  $\tau_1$  consists in a single chunk and  $\tau_2$  is split in three chunks of length 10, 20 and 20, respectively.

As a result, using the limited preemption model, the task set can run at  $s_{min}$  instead of  $s_{max}$  as required by both fully-preemptive and non-preemptive models.

#### IV. OPEN QUESTIONS

Given the effectiveness of the limited preemption approach to increase schedulability and reduce the processor speed (with respect to fully preemptive algorithms), the open problem to be investigated is then to find good scheduling strategies that leverage limited preemption models to further reduce energy consumption while guaranteeing real-time constraints.

Three possible areas of investigations have been identified, which reflect the main research approaches adopted for energy saving. They are described below.

- 1) A first step is to develop an algorithm to efficiently compute the slowest processor speed that guarantees real-time constraints. A promising possibility is to extend the algorithm proposed by Bertogna et al. [5] to return not only the set of preemption points, but also the optimal speed.
- 2) From a DPM point of view, a periodic server could be developed for collecting the idle times together and switch the system into a low-power state during its execution. Since several parameters are involved, the configuration of the server is not trivial. Note that a server running at the highest priority would lead to long continuous intervals spent in a low-power state, but it would increase the interference in lower priority tasks (thus reducing their blocking tolerance). On the other hand, a server running at the lowest priority, would fragment the idle intervals into several slices, preventing an efficient use of DPM techniques.
- 3) A combination between DPM and DVS techniques could be investigated to find a trade off between the two approaches described above.

#### REFERENCES

- [1] E. Bini, G. Buttazzo, and G. Lipari, "Speed modulation in energy-aware real-time systems," in *Proceedings of the 17th Euromicro Conference on Real-Time Systems*, ser. ECRTS '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 3–10. [Online]. Available: <http://dx.doi.org/10.1109/ECRTS.2005.29>
- [2] K. Huang, L. Santinelli, J.-J. Chen, L. Thiele, and G. C. Buttazzo, "Adaptive dynamic power management for hard real-time systems," in *Proceedings of the 2009 30th IEEE Real-Time Systems Symposium*, ser. RTSS '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 23–32. [Online]. Available: <http://dx.doi.org/10.1109/RTSS.2009.25>
- [3] M. A. Awan and S. M. Petters, "Enhanced race-to-halt: A leakage-aware energy management approach for dynamic priority systems," in *Proceedings of the 2011 23rd Euromicro Conference on Real-Time Systems*, ser. ECRTS '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 92–101. [Online]. Available: <http://dx.doi.org/10.1109/ECRTS.2011.17>
- [4] A. Rowe, K. Lakshmanan, H. Zhu, and R. Rajkumar, "Rate-harmonized scheduling and its applicability to energy management," *IEEE Trans. Industrial Informatics*, vol. 6, no. 3, pp. 265–275, 2010.
- [5] M. Bertogna, G. C. Buttazzo, and G. Yao, "Improving feasibility of fixed priority tasks using non-preemptive regions," in *RTSS, 2011*, pp. 251–260.
- [6] R. J. Bril, J. J. Lukkien, and W. F. Verhaegh, "Worst-case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption revisited," *Real-Time Systems, Euromicro Conference on*, vol. 0, pp. 269–279, 2007.
- [7] C. Maxim, L. Cucu-Grosjean, and O. Zendra, "Towards reducing preemptions to save energy," in *the 5th Junior Researcher Workshop on Real-Time Computing (JRRTC 2011)*, Nantes, France, Sep. 2011. [Online]. Available: <http://hal.inria.fr/hal-00646997>
- [8] S. Baruah, A. K. Mok, and L. E. Rosier, "Preemptively scheduling hard-real-time sporadic tasks on one processor," in *Proceedings of the 11th Real-Time Systems Symposium (RTSS'90)*, Orlando, Florida, 1990, pp. 182–190.