Energy Aware EDF Scheduling in Distributed Hard Real Time Systems[♣]

M.Angels Moncusí, Alex Arenas
{amoncusi,aarenas}@etse.urv.es
Dpt d'Enginyeria Informàtica i Matemàtiques
Universitat Rovira i Virgili
Campus Sescelades, Av dels Països Catalans, 26
E-43007 Tarragona, Spain

Jesus Labarta
jesus@ac.upc.es
Dpt d'Arquitectura de Computadors
Universitat Politècnica de Catalunya
Jordi Girona, 1-3. D6 Campus Nord
08034 Barcelona, Spain

Abstract

We present an energy aware scheduling algorithm for distributed hard real time systems with end-to-end deadlines. The approach is focused on the general scenario where the end-to-end deadline is not distributed among the tasks in the chain of precedence. The energy saving is obtained by applying DVS to the dynamic scheduling. The complexity of energy saving algorithm in this scenario mainly relies on the forecasting of the arrival of a message from the precedent task in a chain of an end-to-end deadline that could activate a new task in any host. We propose a conservative prediction of this arrival that guarantees the hard real time specifications still saving energy. The results show that heavy loaded systems are suitable for an energy reduction that ranges from 2% to 27% depending on the global load of the system and the portion of WCET consumed by tasks.

Keywords: Distributed real-time system, End-to-End Deadlines, Low power EDF scheduling, Dynamic Voltage Supply.

1. Introduction

Distributed real time systems have an increasing utilization due to low cost network facilities that allow the interconnection of multiple devices and their controllers into a single large system. These real time systems are often located in environments where low energy consumption is crucial from the operability and lifelong point of view. During the last years, the energy saving paradigm at all system levels including the operating system and compilation level has deserved a lot of attention [1,2]. The most interesting power aware technique is based on the use of the DVS (Dynamic Voltage Scheduling). It consist in reducing the clock speed along with the voltage supply or even power down the processor whenever the system does not require its maximum performance.

In CMOS devices, the energy consumed by specific task τ_i can be given as $E_i=C_{ef}*V^2_{dd}*c'_i$, where C_{ef} is the effective switch capacitance, V_{dd} is the supply voltage and c'_i is the number of cycles needed to executed τ_i [3]. We can decrease the energy consumed by a task reducing the processor power consumption quadratically at the expense of linearly decreasing of the processor speed.

The off-line and on-line scheduling techniques with power saving for uniprocessor systems have been largely studied in the last decade [4-6], while there are comparatively less studies for distributed systems [7,8]. In these distributed system studies, they use an off-line scheduler that generates a calendar and calculates static clock speeds based on the worst case execution time of the tasks is used. During run-time, the scheduler follows the marked steps, adapting dynamically the clock speed depending on the exact execution time. To our knowledge, there is no any on-line power low scheduling algorithm for distributed systems with tasks with precedence constraints allocated into a system of heterogeneous hosts.

In the present work, we propose an on-line energy aware algorithm for distributed heterogeneous hard real time systems based on a modification of the Earliest Deadline First (EDF) [9]. Each host executes a specific task set minimizing energy consumption ensuring the precedence order on the chain of tasks while meeting all time constraints.

The rest of this paper is organized as follows, in section 2 we present the distributed system framework. Section 3 is devoted to the scheduling algorithm to improve energy consumption. In section 4 we present the experimental results and finally in section 5 we draw the conclusions.

2. The distributed system model

The distributed system considered is formed by a group of heterogeneous hosts connected by a real time network. We consider a distributed hard real time application that consists of a set of independent periodic tasks, plus a set of end-to-end deadlines. The end-to-end deadline is formed by an ordered set of tasks $\Gamma_{Deek} = \{\tau_1, \ldots, \tau_N\}$ i.e. the set of successors of any τ_i is *succ*$_i = \{\tau_{i+1}, \ldots, \tau_N\}$. The tasks execution within the end-to-end deadline has to preserve the precedence order so that, as soon as a task finishes, the

[♣] This research is supported by MCYT project number TIC2001-0995-C02-01

immediate successor task is activated via message passing. The characteristics of an end-to-end deadline Γ_{Deek} are: a period T_{Deek}, a global deadline Dee_k that must be met in the precedence order and a delay associated with the message passing between different hosts C_{mess}.

We assume that all tasks are statically allocated to a specific host during the system design stage due to specific hardware characteristics. The independent periodic tasks ι_i— numbered $1 \leq i \leq n$ — are specified by their periods, worst case execution times and deadlines (T_i, C_i and D_i respectively).

3. On-line scheduling of the distributed system

Commonly, to study the feasibility of a distributed real time system, tasks are considered allocated to specific hosts and the global deadline statically distributed among the task that form the chain of precedence. The distribution of these global deadlines in the form of local deadlines for each task of the chain, allows the study of the distributed system as a set of independent uniprocessor [10]. In this case, the minimal local deadline determined for any task in the end-to-end deadline corresponds to the worst case response time (WCRT) of the corresponding task. Apart from the local deadline for each task, the distribution of the end-to-end deadline implies the generation of offsets (elapsed time from the activation of the first task of the end-to-end deadline). When the tasks within the end-to-end deadline are allocated to different hosts the consideration of these offsets are crucial for the scheduling analysis.

The main problem in this scheme is that the determination of the WCRT by any static priority scheduling can be very pessimistic. A first approach to solve this problem for static priority scheduling was proposed by Tindell et al [11] taking into account the real interference of the tasks within the end-to-end deadline (busy period). However, this static approximation still represents a worst case schedulability test that will discard many schedulable distributed systems heavily loaded. A similar approach for the dynamic priority scheduling scenario can be found in [12,13]. In [12] the main idea is to extend the holistic analysis proposed by Tindell adapted to EDF. In [13] a substantial improvement is achieved by considering dynamic offsets that result in new upper bounds for the worst case analysis.

We propose a general scenario where the end-to-end deadline is not distributed among precedence tasks. The main difference with the mentioned approaches is that we do not seek for schedulability bounds, but for an heuristic that will allow to save energy, without compromising deadlines. Moreover, we will consider a dynamic priority scheduling based on EDF [9]. In this case, the schedulability has been determined by simulating the execution of the system over the whole hyper-period at maximum speed, assuming that tasks consume their WCET. During this simulation we calculate the minimum WCRT of each task in the end-to-end deadline.

To ensure the global deadline, we provide the uniprocessor EDF algorithm with the following definition of the maximum local deadline:

$$D_i = Dee_k - \sum_{\forall\, j \in succ_i} C_j \qquad \tau_i \in \Gamma_{Deek}$$

where D_i stands for the local deadline of τi and Dee_k stands for the deadline of the chain k, and C_j stands for the worst case execution time of τ_j. The EDF scheduling with the above local deadline definition for the tasks of the end-to-end deadline, provides the necessary flexibility to accommodate the priorities of the tasks in every host according to the current load.

4. Energy aware algorithm based on EDF scheduler.

The EDF scheduler always decides to execute first the task with the earliest deadline. On the other hand, to save energy, we need to reduce the speed of the processor along with the voltage supply. Then, the scheduler before executing any task instance needs to fix the ratio of the processor speed according with the maximal tardiness the instance is allowed. The speed ratio is calculated following the heuristics proposed by Shin et al. [4] that is built on the assumption that the delay for this reduction is negligible. The safeness of the system under these conditions is proved on theorem 1 of the cited work. The speed is determined on the basis that all tasks consume their WCET, but in practice, the tasks will execute only part of this WCET. Time not used by the instance of the task will be called the “spare” time of the task. The “a priori” estimation of this spare time is not compatible with the hard real time constraints, where no missed deadlines are allowed. This implies that, the speed ratio should be fixed, in run-time, as the minimum value that still guarantees the execution of all WCET before the time constraints of the instance. The spare time generated by one task could be use for the following task.

We present, now, an energy aware algorithm based on EDF for distributed systems with global and local deadlines, LPDEDF (Low Power Distributed Earliest Deadline First). The energy reduction in the current scenario is obtained by reducing the speed of the processor when the only ready task in the system has not successor or when the previous executed task has not exhausted its WCET. The speed ratio is calculated as follows:

1. If there is not any task in the system, then set a timer to the next arrival time task ta_i minus the wake up delay (in this work we assume the wake up delay to be zero), and power down the processor. So, the speed ratio until the activation of any task will be:

$$\textit{Speed ratio} = 0$$

2. When the unique ready task in the system τ_i has not any successor task, then the speed ratio to execute τ_i until there is a new ready task is:

$$Speed\ ratio = \frac{\min(ta_k - tc, ta'_h - tc, remaining(C_i))}{\min(ta_k, ta'_h, td_i) - tc}$$

   where *tc* is the current time, ta_k stands for the next activation time of a task without any precedence task, ta'_h stands for the next predicted activation time of a task with a precedence task that has not finished yet, the *remaining(C$_i$)* corresponds to the remaining execution time of task τ_i in the WCET, and td_i is the deadline of task τ_i.

   We reduce energy by spreading the execution time of τ_i at low speed up to the arrival of the activation of any task. This time interval is calculated in the denominator as the difference between the current time *tc* and the minimum time among ta_k, ta'_h, and the current task deadline td_i. The work the task τ_i could execute in this interval is the minimum among the remaining execution time of its WCET and the activation of the next dependent or independent task.

3. When there is more than one ready task [4] or the unique ready task has a successor task then

$$Speed\ ratio = 1$$

The activation time of the task τ_i (ta_k) without precedent tasks (independent tasks or the first task in the chain of precedence) is known by its periodicity. The problem arises in determining the activation time for a task with a precedent constraint (ta'_h). This time depends on the finalization of the precedent task plus the time to deliver the activation message. That is, depends on the exact executing time of all the precedence task and on the interference of higher priority tasks of its hosts. The energy saving relies then in this predicted activation time for any successor tasks. If the prediction is very early the algorithm will have less energy saving, but if it is too late the deadline of the task could be compromised.

We propose to use a "conservative" prediction for the determination of this time (ta'_h). The prediction is conservative, in the sense that, we prefer to save less energy and preserve all the time constraints. Our prediction is estimated on-line as follows:

(i) Once the first task τ_1 in the chain of the Γ_h is activated, we compute the ta'_h for all successor tasks in its chain using the following iterative formula for each task in the $\Gamma_h = \Gamma_{Deek} = \{\tau_1, \tau_2 \ldots, \tau_i, \ldots, \tau_N\}$:

$$ta'_2 = C_1 + \sum_{\forall j \in hp(\tau_1)} remaining(C_j) + C_{mess}$$

$$ta'_i = C_{i-1} + ta'_{i-1} + C_{mess}$$

where $hp(\tau_1)$ are the tasks in the τ_1 host which have a higher priority level than τ_1.

(ii) When a task with a successor tasks τ_i finishes, we compute Δ as the difference between the predicted time and the exact activation time, due to the exact interference occurred. If Δ is positive, it means that the activation of the successor tasks will be later that the prior predicted time. In this case we refine the arrival prediction time of all its successor tasks, adding Δ to the actual activation time of all the successors. If Δ is negative we do not refine the predicted time, with the off-line simulation we assure the schedulability with the original predicted time. And, as we say before, a later predicted time implies more energy saving.

5. Results and discussion

To test the energy saving performance of the scheduling algorithm LPDEDF we perform several experiments using 100 different schedulable task sets in each one. Our task sets have been determined as follows: The number of hosts has been randomly chosen between 2 and 4. The number of end-to-end deadlines have been chosen as 2*number of hosts. The number of tasks of each end-to-end deadline has been randomly chosen between 2 and the number of host. The number of total tasks has been randomly chosen between number of hosts and 6 * number of hosts. Task periods have been randomly chosen, between 100 and 1000. The load of each task is determined by dividing randomly the fixed total load of the system. The end-to-end deadlines are equal to their periods. The time to send a message from one host to a different host has been fixed to 1 unit of time, and the time to send a message inside the same host has been fixed to 0. In particular, we consider the worst case execution time of every task as an upper bound on its execution time. It includes various architecture overheads such as the cost for the execution of the scheduler, for context switches, for packetizing and depacketizing messages, etc.

We use an heuristic algorithm to statically allocate tasks in the processors [10]. This heuristic algorithm tends to allocate tasks from the same end-to-end deadline to different host, and prioritises the allocation of tasks to the most loaded host. The result of this allocation is a complex distribution of tasks difficult to schedule. The hosts will have different utilizations, being at least one of them close to 100 %. Note that this situation gives a difficult scenario to save energy because in average we have two end-to-end deadline per processor, and the interference between them will be very high.

To prove the feasibility of the system, the schedulability has been determined by simulating the execution of the system over the whole hyper-period at maximum processor speed assuming that all tasks consume its WCET. At the same time, we calculate the minimum WCRT of ant tasks. We present results for 100 schedulable task sets executing over 5 hyper-periods, and comparing between the energy consumed by executing all tasks at maximum processor speed versus the energy consumed by executing all tasks using our energy saving scheduling algorithm. Note that when the processor is idle, the consumed energy in both solutions is the same.

In Figure 1, we show the results of the energy saving, in a heavily loaded scenario with values of the global utilization of the system ranging from 65% to 95%. The standard deviation values are around 15% of the average value presented for the total percentage of energy saving.

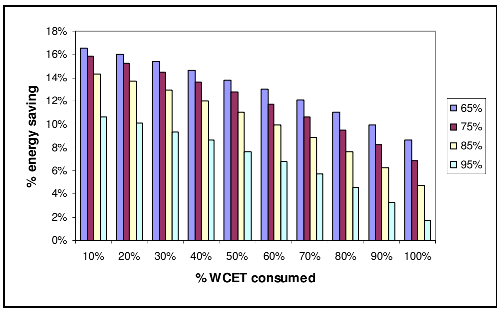

Figure 1. Percentage of energy saving depending on the % of WCET consumed. Different colours represent different values of the global load of the system

We observe that the scheduling algorithm proposed reduces energy consumption form 10% to 16 % where the global load of the system is 65%. In the most loaded case 95%, the algorithm still find place to save between the 2% to 10% of the energy consumption depending on the percentage of WCET consumed by tasks. The explanation is that below the minimal clock frequency there is not possible reduction, independently of the idle time allowed by the low load configurations. When there is no load both algorithms consume exactly the same energy.

Another case study is proposed for the sake of comparison, when the same algorithm is applied to a task set characterized as before, but without the precedence constraints. Note that now, we can reduce the processor speed when in the other study were a precedent constraint. See figure 2.

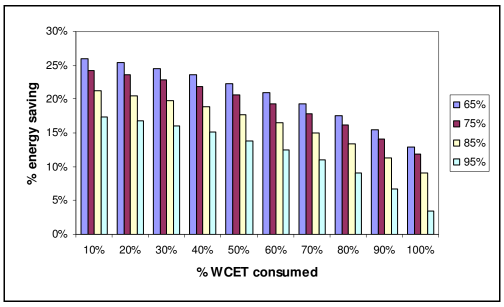

Figure 2. Percentage of energy saving depending on the % of WCET consumed for a case study without end-to-end deadlines. Different colours represent different values of the global load of the system.

Note that in this case, the energy saving increases as expected because we can execute at reduced speed whenever there is an only ready task in the system independently of the forthcoming tasks. The energy saving ranges from 15% to 27%, for different WCET consumptions, when the load is 65%. For the highest load simulated, 95%, the saving ranges from 3% to 16%. The standard deviation values are around 20% of the average value presented for the total percentage of energy saving.

6. Conclusions

We have presented an energy aware scheduling algorithm for distributed heterogeneous hard real time systems with precedence constraints and with local or global deadlines. The scenario proposed allows energy reduction when the global deadline are non-distributed among tasks in the precedence chain of tasks and no deadline could be compromised. In this scenario we show that the proposed algorithm adapts dynamically the scheduler to save energy even when the system is heavily loaded.

7. References

[1] J. Rabaey and M Pedram (Editors). "Low Power Design Methodologies". *Kluwer Academic Publishers*, Norwell, May 1996.

[2] S.T. Cheng, S.M. Chen and J.W. Hwang, "Low-Power Design for Real-Time Systems", *Real-Time Systems*,15, pp 131-148, 1998.

[3] A.P. Chandrakasan, S. Sheng and R. W. Brodersen, "Low-power CMOS digital design", *IEEE Journal of Solid-State circuits*, vol. 27, pp. 473-484, April 1992.

[4] Y. Shin and K. Choi, "Power conscious Fixed Priority scheduling in hard real-time systems" *Proceedings of the Design Automation Conference,* ACM 1-58113-7/99/06, 1999.

[5] P. Pillai and K.G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems" *18th ACM Symposium on Operating Systems Principles*, October 2001.

[6] S. Saewong and R. Rajkumar, "Practical Voltage-Scaling for Fixed-Priority RT-Systems" *Proceedings of the 9th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS),* May 2003

[7] R.Mishra, N.Rastogi, D.Zhu, D. Mossé and R. Melhem, "Energy Aware Scheduling for Distributed Real-Time Systems" *In Proc. of the International Parallel and Distributed Processing Symposium*, 2002

[8] J.Luo and N. K. Jha. "Static and dynamic variable voltage scheduling algorithms for real-time heterogeneous distributed embedded systems." *In Proc. of 15th International Conference on VLSI Design*, Jan. 2002.

[9] C.L. Liu and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment" *Journal of the ACM 20(1), pp 46,61, 1973.*

[10] M.A.Moncusi, J.M.Banus, J.Labarta, A. Arenas, "A New Heuristic Algorithm to Assign Priorities and Resources to Tasks With End-to-End Deadlines", *International Conference on Parallel and Distributed Processing Techniques and Applications*, Vol. IV, pag. 2102-2108, June 2001.

[11] K. Tindell and J. Clark, "Holistic Schedulability Analysis for Distributed Hard Real-Time Systems". *Microprocessing & Microprogramming*, Vol 50, pp 117-134, April 1994.

[12] M. Spuri. "Holistic Analysis of Deadline Scheduled Real Time Distributed Systems", RR-2772, INRIA, France, 1996.

[13] J.C. Palencia and M.González Harbour."Offset-Based Response Time Analysis of Distributed Systems Scheduled under EDF", *15th Euromicro Conference on Real-Time Systems*, July 2003.