A Note on EDF Scheduling for Real-Time Energy Harvesting Systems

Maryline Chetto, Audrey Queudet

A note on EDF Scheduling for Real-Time Energy Harvesting Systems

Maryline Chetto, *Member, IEEE,* and Audrey Queudet

Abstract—Energy harvesting is the capture of ambient energy, its conversion into a usable form, and its storage for immediate or future use. Interest in energy harvesting has increased over the last decade because of its environmental friendliness and its ability to power devices without electric wires. This term has been frequently applied in recent years in the context of small autonomous embedded devices such as wireless sensor nodes. In this paper, we address the scheduling problem for a single processor device that executes preemptable time critical tasks. Each one has a certain energy requirement and arrives at an unpredictable time. We ask the question whether the traditional task scheduling algorithm Earliest Deadline First (EDF) is convenient for energy harvesting environments. The paper shows that EDF has a zero competitive factor but nevertheless is optimal for online non-idling settings.

Index Terms—Earliest Deadline First, energy harvesting, non-idling, optimality, preemptive scheduling.

1 Introduction

Energy harvesting is a technology that allows to capture otherwise unused ambient energy and convert into electrical energy that can be used immediately or later thanks to a storage unit [20]. This approach extends the life of batteries (or eliminates them entirely) and decreases maintenance. A variety of techniques are available for energy harvesting, including solar and wind powers, ocean waves, piezoelectricity, thermoelectricity, and physical motions. Energy harvesting is a perfect match for wireless devices and wireless sensor networks that otherwise rely on battery power. Some of the main applications include operating as power source for human wearable electronics, supplement battery storage devices, etc. Another key application that is being investigated in great detail is miniature self-powered sensors in medical implants for health monitoring and embedded sensors in structures such as bridges, buildings for remote condition monitoring. Levels of harvested energy may vary significantly from application to application. Therefore spare usage of available energy is of utmost importance.

The system we target consists of three components (see Figure 1): a single processing unit with unique voltage and frequency, an energy harvester and a rechargeable energy storage.

We address the problem of scheduling that arises in an energy harvesting system with real-time constraints where tasks have to meet deadlines. Our task processing model exhibits three major assumptions.

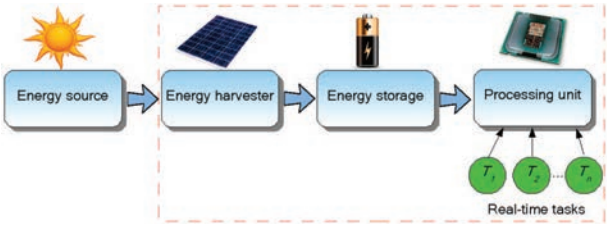

Fig. 1. A Real-Time Energy Harvesting System

First, as commonly assumed in the real-time energy harvesting literature [7], [18], the instantaneous power consumption of any task is no less that the incoming power from the harvesting unit. Indoor lighting control systems are one of the systems having such characteristics. Secondly, the energy consumed by a task is not necessarily proportional to its execution time. Thirdly, unlike previous work in this area [18], our approach to task scheduling is preemptive non-idling. So, the question we ask, can be formulated as follows: *Is Earliest Deadline First a convenient scheduler for energy harvesting systems under non-idling execution settings?*

The rest of the paper is organized as follows: We first present related work. The energy harvesting system model and some assumptions are explained in Section 3. We give background material in Section 4. Optimality analysis of EDF is described in Section 5. Section 6 concludes and describes future work.

2 Related Work

Most of the work in the area of energy-aware real-time scheduling focuses on either minimizing the energy consumption or maximizing the system lifetime under energy constraints. Research activities, such as

• *The authors are with the LUNAM University, University of Nantes, IRCCyN Institute, 1 Rue de la Noë, F-44321 Nantes FRANCE. E-mail: {maryline.chetto,audrey.queudet}@univ-nantes.fr*

Dynamic Power Management (DPM) [13], [17], [19] or Dynamic Voltage and Frequency Scaling (DVFS) [14], [23], focus on designing adaptive scheduling and voltage/frequency selection algorithms in order to effectively reduce the power consumption of a device, the limited energy in the battery being exhausted eventually. However, in many applications such as embedded sensors, it is impractical or costly to recharge or replace batteries.

Ideally, an embedded system should be designed to operate perpetually. This objective cannot be achieved when the battery is the only energy source, except by using energy harvesting. The first prototypes that demonstrate the feasibility of this technology are Heliomote [22] and Promotheus [10]. Many technical challenges must be solved to achieve energy autonomy and to make the embedded system work effectively. The energy source is unstable and changing from time to time. This requires to model the harvested power as a time-varying variable. Due to these intrinsic characteristics, energy harvesting systems are subject to new issues not encountered in conventional battery-powered systems.

Real-time scheduling in energy harvesting systems aims to find a schedule in which all the tasks meet their deadlines while consuming only as much energy as harvested. We then say that the system operates in an energy neutral mode [11] (i.e. the system never runs out of energy). The problem of scheduling real-time energy harvesting systems has gained little attention and has been addressed only in the last decade. The proposed scheduling techniques for such systems adapt the performance and power consumption of the application unit at runtime in response to the temporal variation in harvested energy.

Allavena et al. [1] propose an off-line scheduler restricted to a set of independent periodic tasks in a frame. Moreover the power scavenged by the energy source is constant and all tasks consume energy at a constant rate.

Later, the research presented in [18] is a variation of the famous Earliest Deadline First (EDF) scheduler [12], called Lazy Scheduling Algorithm (LSA). The authors use a similar model of the power source as we do hereafter. However, they assume for every task that its total energy consumption is directly connected to its execution time through the constant power of the processing device. But in a real application, instantaneous power consumed by tasks may vary along time depending on circuitry and devices required by the tasks. LSA was compared with EDF in terms of both schedulability and required capacity of the storage unit. The theory and the simulation study show that LSA outperforms EDF up to 45% in terms of achievable capacity savings.

More recently, we proposed another variation of EDF called EDeg [7] that relies on the EDS (Earliest Deadline as Soon as possible) rule initially described in [5].

More research works on real-time scheduling for energy harvesting embedded systems can be found in [15], [16].

3 Model and Terminology

3.1 System model

The Energy Harvesting model (referred to hereafter as EH) comprises a system composed of a set of tasks, an energy storage unit and an energy source as described above.

3.1.1 Task model

We consider a uniprocessor system that consumes negligible energy in idle state. Tasks enter that system and require to execute before deadlines. A four-tuple (r_i, C_i, E_i, d_i) is associated with a task τ_i and gives its release time, Worst Case Execution Time (WCET), Worst Case Energy Consumption (WCEC) and (absolute) deadline respectively. We assume that E_i is not necessarily proportional to C_i [21].

3.1.2 Energy model

The energy source is characterized by an instantaneous charging rate $P_r(t)$ that incorporates all losses. We define $E_p(t)$ as the energy produced by such a power source from time 0 to time t. We assume that the energy production times can overlap with the consumption times and the instantaneous power consumed by any task is no less than the instantaneous power drawn from the source. The energy produced on the time interval $[t_1, t_2)$ is denoted $E_p(t_1, t_2)$ while the energy consumed by tasks on the same interval is denoted $E_c(t_1, t_2)$.

The energy produced by the source is not considered as controllable and not necessarily predictable. Our system uses an ideal energy storage unit that has a nominal capacity, namely E. Let define $E(t)$ as the energy level of the system at time t. The stored energy may be used at any time later and does not leak any energy over time. Energy is wasted if the storage is fully charged at time t (i.e. $E(t) = E$) and we continue to charge it. In contrast, no task can be executed and the application definitively stops if the storage is fully discharged at time t (i.e. $E(t) = 0$). We assume that the energy level of the storage is never increasing every time a task executes.

A task τ_i can miss its deadline if one of the two following cases occurs:

- *time insufficiency*: when the task reaches its deadline at time t, its execution is incomplete since the time required to process the task within its deadline is not sufficient. There is available energy in the storage when the deadline violation occurs (i.e. $E(t) > 0$).
- *energy insufficiency*: when the task reaches its deadline at time t, its execution is incomplete

since the energy required to process the task within its deadline is not available. The energy in the storage is exhausted when the deadline violation occurs (i.e. $E(t) = 0$).

3.2 Terminology

We also include definitions of several well-known quantities and terms here for completeness:

Definition 3.1: A schedule Γ for τ is *valid* if the deadlines of all tasks of τ are met in Γ.

Definition 3.2: A scheduling algorithm is *optimal* if it finds a valid schedule whenever one exists.

Definition 3.3: A scheduling algorithm is *online* if it makes its decision at run-time with no information about the future.

Definition 3.4: A scheduling algorithm is *idling* if it is allowed to keep the processor idle even when there are pending tasks. Otherwise, it is *non-idling*.

Definition 3.5: A scheduling algorithm is *clairvoyant* if it has an a priori knowledge of all the tasks arriving in the system and profile of the energy produced by the source. Otherwise, it is *non-clairvoyant*.

Definition 3.6: The "value" of a task defines its contribution to the overall system performance. The value obtained is proportional to the computation time of the task. The system obtains the value of a given task if the task completes by its deadline. Otherwise, the system obtains no value for the task [2], [4].

Definition 3.7: The "goodness" of an algorithm is measured by its *competitive factor* : an on-line algorithm of competitive factor r $(0 < r < 1)$ is guaranteed to achieve a cumulative value at least r times the cumulative value achievable by any clairvoyant algorithm on any sequence of tasks [3].

Definition 3.8: An on-line scheduling algorithm is *competitive* if it has a constant competitive factor strictly greater than zero. Otherwise, it is *non-competitive*.

4 Background Material

The Earliest Deadline First (EDF) algorithm is the most popular approach for scheduling independent time critical tasks preemptively when there is no energy limitation and no processing overload [8], [12]. EDF is an on-line algorithm which schedules the ready task (i.e. the task that may be processed and is not yet completed) whose deadline is closest to the current time t. On-line scheduling is the only option in a system whose future workload is unpredictable. It can accomodate dynamic variations in user demands and resource availability.

EDF is optimal when used to schedule tasks on a processor as long as preemption is allowed and tasks do not contend for resources. If any algorithm can produce a valid schedule, then so can EDF. However, when a processing overload may occur (i.e. the processing required to handle all the tasks exceeds the system capacity), EDF is no longer optimal. In that situation, if we consider the value of a task as proportional to its execution time, Baruah et al. proved that no on-line scheduler – including EDF – can guarantee a competitive factor greater than 0.25 for tasks with uniform density settings [3].

5 Optimality analysis for EDF scheduling algorithm

A task set τ is *timely-feasible* if there exists a valid schedule for τ (i.e. all the deadlines of τ are met) without considering energy constraints. E_{tot} denotes the total energy needed to complete all the tasks in τ.

Theorem 5.1 simply states that EDF achieves the best possible performance if the energy storage unit size is sufficient to store the total energy required by all the occuring tasks.

Theorem 5.1: If $E_{tot} \leq E$, then EDF produces a valid schedule for any timely-feasible task set.

Proof: The storage is fully charged at time zero i.e. $E(0) = E$. Sufficient energy is available at time zero for executing all the tasks even if $P_r(t) = 0$ for any time t. So the proof directly follows from the optimality of EDF with no energy considerations [8]. □

It is natural to ask whether the EDF algorithm remains optimal if tasks exhibit energy requirements in energy harvesting settings. Let us restrict the study to the case of non-idling scheduling for proving that EDF is still optimal.

Theorem 5.2: EDF is optimal in the class of non-idling online algorithms for the EH model.

Proof: We prove that any valid schedule $\Gamma(X)$ produced by a non-idling online algorithm X can be transformed into a valid EDF schedule. Let us consider the first place in that schedule where it violates the EDF rule. We consequently find the first interval I_1 in which τ_i executes. Let τ_k be the task such that $d_i \geq d_k$ which executes in I_2 according to X and should execute in I_1 according to EDF. The question is: *Can we swap τ_i and τ_k?*

To prove the theorem, there are two cases to consider.

First case: We assume that tasks consume processing time only (i.e. $E_i = 0$, $E_k = 0$). In this case, direct the swap is always possible (as proven by Dertouzos in [8]), either I_1 being shorter than I_2 (Figure 2-(a)) or I_1 being longer than I_2 (Figure 2-(b)).

Second case: We assume that tasks consume energy from the energy storage (i.e. $E_i > 0$, $E_k > 0$). We consider that interval I_1 starts at time s_1. Let E_i (respectively E_k) be the energy consumption of τ_i within I_1 (respectively τ_k within I_2). The processor is continuously busy from s_1 to f_2 and there cannot exist an idle time since X is a non-idling algorithm. Without loss of generality, we may assume that I_1 is as long as I_2. By hypothesis, the instantaneous consumption power of every task is greater than or equal to the instantaneous power provided by the source. In other terms, the energy storage cannot recharge as the processor is busy executing any task. The difference between the total energy that will be consumed and the total energy that will be produced within $[s_1, f_2]$ is then available initially in the energy storage. In other terms, $E(s_1) \geq E_c(s_1, f_2) - E_p(s_1, f_2)$ where $E_c(s_1, f_2)$ (respectively $E_p(s_1, f_2)$) is the total energy consumed (respectively the total energy produced) between s_1 and f_2. The swap is consequently always possible either E_i being lower than E_k (Figure 3-(a)) or E_i being greater than E_k (Figure 3-(b)). Note that for simplicity reasons, $E_p(t_1, t_2) = 0$ and $E_c(t_1, t_2) = \frac{E_i}{C_i} * (t_2 - t_1)$ for all $0 \leq t_1 \leq t_2$ on the examples depicted in Figure 3. The result of that swap is that τ_i and τ_k are now scheduled with the EDF rule. We repeat this transformation for all pairs of tasks that are not scheduled according to EDF. As every valid schedule produced by a non-idling algorithm can be transformed into an EDF schedule which is valid too, this proves the optimality of EDF in the class of non-idling algorithms.□

From what precedes, the optimal schedule clearly depends on the knowledge of future arrivals and can be obtained by a clairvoyant scheduling algorithm only. The reason is that once a task has executed, it has consumed more energy than produced which impacts the energy availability for future occurring tasks. Now, let us undertake a competitive analysis of EDF.

Theorem 5.3: EDF has a zero competitive factor for the EH model

Proof: Directly follows from Lemma 1 in [9]. The following model is considered: the system has a fixed and limited energy budget. The energy requirement of every task is proportional to its execution time. The value of the task is added to the overall performance metric if and only if it meets its deadline. As EDF is non-competitive with this more restrictive model, EDF is necessarily non-competitive for the EH model where the energy budget is also initially limited.□

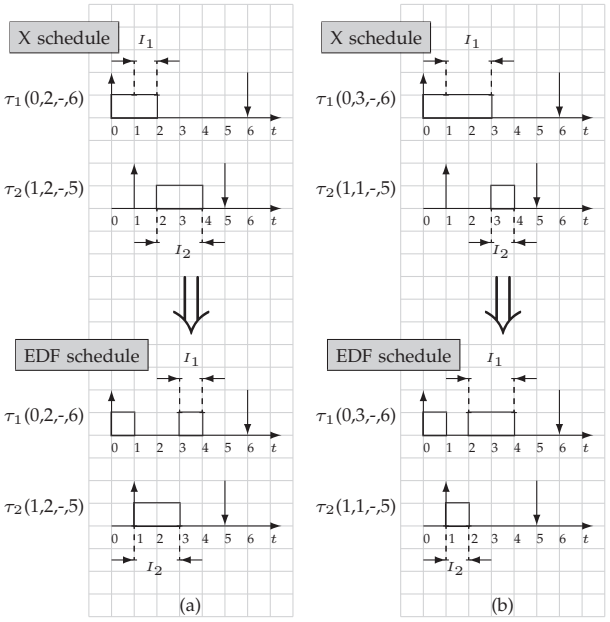

Fig. 2. Transformation into an EDF schedule with time consideration when (a) I_1 is shorter than I_2 and (b) I_1 is longer than I_2 [8]

By theorem 5.3, we show that if an online competitive algorithm exists, it is necessarily non-idling.

6 Discussion and Conclusions

We studied the case of a uniprocessor energy harvesting system that has to schedule a set of real-time tasks. Both time and energy constraints of the tasks are not known in advance. In a previous work, Moser et al. [18] introduced the LSA algorithm which is an online idling algorithm and proved it to be optimal under restrictive hypotheses: (i) WCEC and WCET of every task are proportional, (ii) the energy drained from the environment in the future is known and (iii) the computing system must be able to continously adapt its consumption power with respect to the power incoming from the source. The authors relaxed the (i) hypothesis in a more recent study by introducing the EDeg clairvoyant scheduling algorithm whose performance is demonstrated in a simulative study [7]. Nevertheless, this work leaves the proof of its optimality open.

Energy-aware scheduling algorithms exhibit the following drawbacks:

- they are energy-clairvoyant, i.e., the future harvested energy is assumed to be known precisely. Moreover, their performance in terms of optimality depends on the accuracy of the prediction.
- they rely on the capability to estimate accurately the remaining energy in the energy reservoir (e.g.

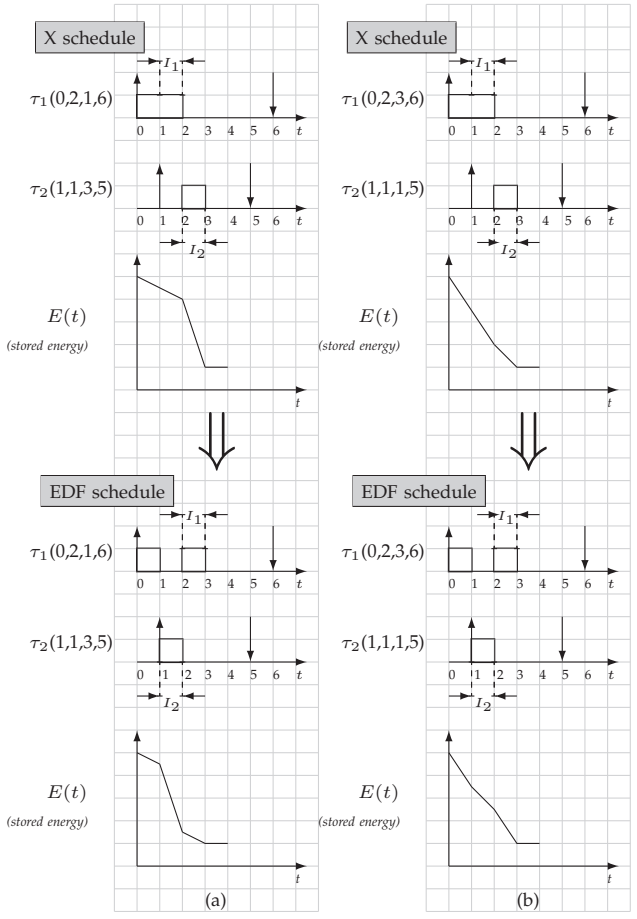

Fig. 3. Transformation into an EDF schedule with energy consideration when (a) E_1 is lower than E_2 and (b) E_1 is greater than E_2

battery or supercapacitor). Practical considerations about general implementation aspects and more accurate models for the energy reservoirs can be found in [6].

- they make the assumption that, at every time, the incoming power from the environment never exceeds the instantaneous consumption power of the running task.

In this paper, we investigated the optimality analysis of the most famous real-time scheduling policy – namely EDF – in the context of energy harvesting with no clairvoyance at all relative to both task arrival times and energy production. We proved that EDF is still optimal for online non-idling settings. Moreover, from a practical point of view, this greedy scheduler has not any specific technological requirement (e.g. additional circuitry for both the estimation and the prediction of the havested energy). Consequently, EDF clearly remains a class one priority-driven scheduler for energy harvesting applications looking at its simplicity.

REFERENCES

[1] A. Allavena and D. Mosse. *Scheduling of Frame-based Embedded Systems with Rechargeable Batteries*, Workshop on Power Management for Real-Time and Embedded Systems, 2001.

[2] S. Baruah, G. Koren, B. Mishra, A. Raghunathan, L. Rosier and D. Shasha. *Online scheduling in the presence of overload*, Symposium on Foundations of Computer Science, 1991.

[3] S. Baruah, G. Koren, D. Mao, B. Mishra, A. Raghunathan, L. Rosier, D. Shasha and F. Wang. *On the Competitiveness of On-Line Real-Time Task Scheduling*, Real-Time Systems, Volume 4, Issue 2, pp. 125-144, 1992.

[4] G. Buttazzo. *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications 2nd edn.*, Springer, Berlin, 2005.

[5] H. Chetto and M. Chetto. *Some Results of the Earliest Deadline Scheduling Algorithm*, IEEE TRansactions on Software Engineering, Volume 15, Issue 10, pp. 1261-1270, 1989.

[6] D. Dondi, A. Bertacchini, L. Larcher, P. Pavan, D. Brunelli and L. Benini. *A Solar Energy Harvesting Circuit for Low power Applications*, IEEE International Conference on Sustainable Energy Technologies, pp. 945-949, 2008.

[7] H. El Ghor, M. Chetto, R. Hage Chehade. *A real-time scheduling framework for embedded systems with environmental energy harvesting*, Computers & Electrical Engineering Journal, Volume 37, Issue 4, pp. 498-510, 2011.

[8] M.-L. Dertouzos. *Control Robotics: The Procedural Control of Physical Processes*, Proceedings of International Federation for Information Processing Congress, 1974.

[9] V. Devadas, F. Li, H. Aydin. *Competitive analysis of online real-time scheduling algorithms under hard energy constraint*, Real-Time Systems, Volume 46, Issue 1, pp. 88-120, 2010.

[10] X. Jiang, J. Polastre and D.-E. Culler. *Perpetuel Environmentally Powered Sensor Networks*, International Symposium on Information Processing in Sensor Networks, 2005.

[11] A. Kansal and J. Hsu. *Harvesting aware Power Management for Sensor Networks*, IEEE Design Automation Conference, 2006.

[12] C.-L. Liu, J.-W. Layland. *Scheduling algorithms for multiprogramming in a hard real-time environment*. Journal of the Association for Computing Machinery, Volume 20, Issue 1, pp. 46-61, 1973.

[13] Y.-H. Lu, L. Benini and G. De Micheli. *Low-power Task Scheduling for Multiple Device*, International Workshop HW/SW Co-design, pp. 39-43, 2000.

[14] J. Luo and N.-K. Jha. *Static and Dynamic Variable Voltage Scheduling Algorithms for Real-Time Heterogeneous Distributed Embedded Systems*, International Conference on VLSI Design, pp. 719-726, 2002.

[15] S. Liu, J. Lu, Q. Wu and Q. Qiu. *Harvesting-Aware Power Management for Real-Time Systems with Renewable Energy*, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, pp. 1-14, 2011.

[16] J. Lu and Q. Qiu. *Scheduling and Mapping of Periodic Tasks on Multi-core Embedded Systems with Energy Harvesting*, Journal of Computers and Electrical Engineering, pp. 498-510, 2011.

[17] S. Liu, Q. Qiu and Q. Wu. *Task Merging for Dynamic Power Management of Cyclic Applications in Real-Time Multiprocessor Systems*, ICCD, 2006.

[18] C. Moser, D. Brunelli, L. Thiele, L. Benini. *Real-time scheduling for energy harvesting sensor nodes*, Real-Time Systems, Volume 37, Issue 3, pp. 233-260, 2007.

[19] R. Mishra, N. Rastogi, D. Zhu, D. Mosse and R. Melhem. *Energy-aware Scheduling for Distributed Real-Time Systems*, International Symposium on Parallel & Distributed Processing, 2003.

[20] S. Priya and D.-J. Inman. *Energy Harvesting Technologies*. Springer-Verlag, New York (USA), 2009.

[21] R. Jayaseelan, T. Mitra, X. Li, *Estimating the Worst-Case Energy Consumption of Embedded Software*, 12th IEEE Real-Time and Embedded Technology and Applications Symposium, 2006.

[22] V. Raghunathan, A. Kansal, et al. *Design Considerations for Solar Energy Harvesting Wireless Embedded Systems*, International Symposium on Information Processing in Sensor Networks, 2005.

[23] F. Yao, A. Demers, et al. *A Scheduling Model for Reduced CPU Energy*, IEEE Symposium on Foundations of Computer Science, 1995.