Ensuring the sustainability of real-time embedded system under both QoS and Energy Constraints

Maissa Abdallah, Nadine Abdallah, Maryline Chetto

Ensuring the sustainability of real-time embedded system under both QoS and Energy Constraints

Maissa Abdallah[1], *Nadine Abdallah*[1]*and Maryline Chetto*[2]

[1]Faculty of technology, Lebanese University, Saida, Lebanon
[2]LS2N - Laboratoire des Sciences du Numérique de Nantes, 44322 Nantes, France

Abstract. Nowadays, wireless sensor networks (WSNs) are more and more used in applications such as environment monitoring, healthcare monitoring, etc...The challenge in sensor networks is to ensure the sustainability of the system by guaranteeing the required performance level. However, with the limited capacity of finite power sources and the need of guaranteeing a long lifetime of those systems, it is suitable to use energy harvesting which allows to supply low-power electronic systems by converting ambient energy into electric power. Hence, our study is concerned with the problem of soft periodic and aperiodic tasks scheduling in sensor nodes powered by energy harvesters. In this paper, we address this issue by proposing three energy-aware schedulers, namely BG-Green-RTO, BG-Green-BWP and Green-AWP which aim to improve the responsiveness of aperiodic tasks while still guaranteeing the execution of periodic tasks considering their timing and energy constraints. Such algorithms allow to gracefully cope with processing overload and energy starvation. Moreover, a simulation study permits to show their performance.

1Introduction

Wireless sensor networks have taken more and more place in diverse application domains. For example, in agriculture, it could be possible to deploy a wireless embedded network to detect various measures across a field such as temperature, light levels, soil moisture, ... This example implies that measures should be collected at a specific time and should be treated on time to react correctly to a given situation. Majority of wireless sensor networks represent real-time embedded systems. A real-time system is a system which must guarantee a response within a specified time (often referred to as "deadline"). In other sense, it is a system in which the correctness depends not only on the logical result but also on the time in which it was delivered. So that, if the time constraints are not respected it may lead to a system failure. According to their time constraints, real-time systems are classified into three categories: hard, soft and firm [6]. In a hard real-time system, all deadlines must be met imperatively. In a soft real-time system, missing deadlines will only cause a performance degradation without leading to a system failure. A firm real-time system is a specific case of a soft real-time system so that it must meet its deadlines but with a degree of flexibility. Batteries are the source of power for most embedded system applications. However a battery has a finite lifespan, limited energy density and capacity so when all its energy is consumed, the sensor must be retrieved to replace the battery. Nevertheless, It is not as simple to replace it, especially when the sensor is deployed in an inaccessible place where any human intervention may be either costly or impractical. Thus, such systems should be designed so as to ensure a continuous functioning without having a periodical maintenance due to replacing or recharging batteries. Renewable energy sources are available in an unlimited quantity and there is a variety of available techniques for energy harvesting (i.e. the process of extracting energy from the surrounding environment) such as solar, piezoelectricity, or

thermoelectricity, etc... Therefore, it is interesting to target research towards environmental energy harvesting which could be considered as a promising approach to provide power for long term applications. Now, the challenge is to utilize the information about future harvested energy to maximize the overall Quality of Service (QoS) and still guaranteeing the sustainability of the system.

In this paper, we consider a firm real-time embedded system which receives energy from the environment. The energy received is stored in an energy reservoir formed by either a battery or a super-capacitor. This energy reservoir is required to ensure the power requirements of the system even if no energy can be drawn from the environment. Our objective is to ensure the efficiency of the system and enhance the responsiveness of aperiodic task while still guaranteeing the Quality of Service (QoS) of the system according to the available energy level in the battery. The remainder of the paper is organized as follows. Section 2 gives the background material. Section 3 describes the model of the system. In Section 4 we describe our contribution with illustrative example.
Finally, Section 5 concludes the paper.

2 Background material

2.1 Scheduling Periodic Tasks under QoS constraints

The Skip-Over model [4] deals with the problem of scheduling firm periodic tasks which allow occasional deadline violations (i.e. skippable periodic tasks), on a uniprocessor system. A task τ_i is characterized by a worst-case computation time C_i, a period T_i, a relative deadline equal to its period $D_i = T_i$, and a skip parameter s_i. s_i gives the tolerance of this task to miss deadlines so that the distance between two consecutive skips must be at least s_i periods. We consider that a job is the occurrence of a task. In this model, a task is characterized by two kinds of jobs, red ones and blue ones. A red job must complete its execution before its deadline while a blue job can be aborted at any time. For each task τ_i, (s_i - *1*) consecutive jobs are red and the next one is blue. The more the parameter s_i is large, the less losses are allowed. Consequently, if s_i equals to infinity, no skips are allowed and the task τ_i is considered as a hard periodic task. In [5], the authors define the equivalent utilization processor U_p^* given by:

$$U_p^* = \max_{L>0} \left\{ \sum_{i=1}^{n} \frac{D(i,[0,L])}{L} \right\} \quad (1)$$

Where,

$$D(i,[0,L]) = \left(\left\lfloor \frac{L}{T_i} \right\rfloor - \left\lfloor \frac{L}{s_i T_i} \right\rfloor \right) C_i \quad (2)$$

2.2 Handling Aperiodic Tasks

In many applications, there are aperiodic and periodic tasks. An aperiodic task is typically activated by the arrival of external events (notified by interrupts). According to the application, it may have hard, soft or non real-time requirements. In our work, we assume that our system contains non critical aperiodic task (i.e. without a specific deadline). Therefore, a real-time system which executes periodic and non-critical aperiodic tasks have to guarantee that:

- all periodic jobs meet their deadlines, even in the worst-case.
- Aperiodic jobs have a good response time.

For a classical uniprocessor system, a lot of efficient server algorithms have been developed. In our work, we were interested in the Background server, proposed in [1], which are reasonable in memory requirement, runtime overhead. It consists of scheduling aperiodic tasks in background (i.e., during idle times left by periodic tasks) then, they do not interfere with any periodic task.

2.3 Scheduling soft Periodic Tasks under Energy Sustainability constraints

The problem of real-time task scheduling on one processor without considering energy constraint has been an active area of research for more than thirty years [6]. Since the last decade, researchers have become more and more interested to address power and scheduling issues. Most of the works in this area focus on either minimizing the energy consumption [3, 11] or ensuring a long-term sustainability of the energy system [7, 8, 9, 10]. In our work, we want to deal with Real-Time Scheduling for

Energy Sustainable Systems so that we consider both rechargeability of the batteries and real-time constraints.
In [10], The authors proposed a real-time scheduling algorithm called Earliest Deadline with energy guarantee (EDeg). It consists of scheduling a hard periodic task set (i.e., all jobs must meet their deadline) on a uniprocessor system which uses the energy stored in a rechargeable battery and the energy harvested from the environment. Every task is characterized by the energy consumption E_i in addition to the traditional timing parameters (i.e. C_i, D_i, T_i). All jobs are executed according to the *EDF* algorithm as soon as possible if the energy level in the battery is sufficient to provide energy for current and future occurring tasks without involving deadline violations. The system starts executing a job only if the so-called slack energy is positive and the battery is not empty. The slack energy represents the energy surplus that can be consumed by a job while still satisfying energy constraints of highest priority jobs. It is used to quantify the energy consumed by future jobs so that it prevents from violating deadlines in case of energy shortage. If energy is not sufficient to execute current or future occurring jobs, the system stops its activity as long as the slack time is positive and the battery is not fully replenished.

3. Model and terminology

We consider a uniprocessor system composed of n firm periodic tasks. This system is powered by a rechargeable battery. The battery is recharged by a renewable energy source. $P_r(t)$ is the instantaneous charging rate that incorporates all losses caused by power conversion and charging process. The nature and dynamics of the energy source are not specified, making our approach more easily implemented in real systems. Our system uses an ideal energy storage unit which could be a super-capacitor or a battery. It has a nominal capacity, namely C, expressed in Joules or Watts-hour.
The energy level has to remain between two boundaries E_{min} and E_{max} with $C = E_{max} - E_{min}$.
We assume that firm periodic tasks comply with the Skip-Over model so each task is characterized by computation, energy and QoS requirements. We assume the following notations in this paper. Let T be a firm periodic task set defined as follows: $T = \tau_i(C_i, D_i, T_i, s_i, E_i)$; i = 1 to n. E_i is the Worst Case Energy Consumption (WCEC). Tasks are independent, preemptable and deeply-red (i.e. synchronous activation at time t=0). Tasks have an implicit deadline ($D_i = T_i$). A task generates a sequence of jobs periodically. The k^{th} job of the task τ_i is denoted by $\tau_{i,k}$, and it has an absolute deadline denoted by $d_{i,K}$ equal to the released time of the next job, (i.e. $d_{i,k} = r_{i,k} + T_i$).
In [12], we define the equivalent energy factor U_e^* as follow:

$$U_e^* = \max_{L>0}\left\{\sum_{i=1}^{n} \frac{g^*(i,[0,L])}{E(0) + E_r(0,L)}\right\} \quad (3)$$

Where,

$$g^*(i,[0,L]) = \left(\left\lfloor\frac{L}{T_i}\right\rfloor - \left\lfloor\frac{L}{s_i T_i}\right\rfloor\right)E_i \quad (4)$$

$g^*(i,[0,L])$ represents the energy consumed by red jobs of task τ_i in the interval [0;L].
E(0) represents the initial energy level in the battery,
$E_r(0,L)$ represents the energy received by the battery during the interval [0,L].

In this paper, we assume that the power received by the environment is constant during a hyperperiod P (i.e. P = LCM(T_1s_1, .., T_is_i, ..,T_ns_n)). Then as $P_r(t) = P_r$, ∀t, $Er(0,L) = Pr.L$ where L represents the red job end points during the interval [0,P[.
In [], we define and prove the following feasible tests:

Theorem 1: *A skippable periodic task set T is schedulable under energy and timing constraints only* $\forall L|\ \sum_{i=1}^{n} g^*(i,[0,L]) \le E(0) + E_r(0;L)$

Theorem 2*: A set of skippable periodic tasks T is schedulable under energy and timing constraints only if :* $U_p^* \le 1$ *and* $U_e^* \le 1$

In [12], we proposed the Green-RTO and the Green-BWP schedulers which are based on the skipOver [4] model and EDeg algorithm [10].
Green-RTO runs as follows:

- When the battery is not empty, the processor is active if there is a pending red job and the system has enough slack energy. Ready red jobs are processed according to the EDF algorithm.

- The processor is inactive if the slack time is not null or there is no red jobs ready to be executed.

The slack time of the system at time *t* represents the maximum time available from *t* to postpone red jobs while still satisfying all timing constraints. Hence, it permits to put the processor to an idle state and recharge the battery. The Slack time is computed by using the Earliest Deadline as Late as possible (EDL) algorithm [2]. The slack energy of the system at time *t* represents the energy surplus that can be consumed by a red job which has the highest priority at time t while still satisfying energy constraints of future red jobs that will have a highest priority than the current job. It is used to avoid violating deadlines of red jobs in case of energy shortage.

4. Contribution

In this paper, we propose real-time scheduling techniques for reducing the response time of soft aperiodic tasks scheduled with real-time periodic tasks on an uniprocessor system under QoS and energy constraints. We consider that a soft aperiodic task is defined by $\varphi_i(r_i^a, C_i^a, E_i^a)$ where r_{ai} is its arrival time , C_{ai} is its worst case execution time, and E_{ai} is its worst case energy time. The response time of φ_i is expressed by $TR_i^a = f_i^a - r_i^a$ where f_i^a is its completion time.
Our work consists of a mixture of the BG server with Green-RTO or Green-BWP.

4.1 Schedulers

4.1.1 BG-Green-RTO

BG-Green-RTO consists of executing a ready aperiodic task if there is no ready red jobs. Hence at the time *t*, if there is no ready red jobs and there is at least one ready aperiodic task, an aperiodic task is chosen according to the FIFO algorithm (i.e. First In First Out). The aperiodic task chosen will be executed if it will not jeopardize the execution of future periodic tasks. Then the aperiodic task will be partially or entirely executed according to the surplus of energy available in the battery E_{av}. This surplus of energy available is computed to ensure that the battery will be fully charged at the next release of a future red job.
The surplus of energy available at the time *t* is computed as follows:
$E_{av}(t) = E(t) + (r_{min} - t)Pr - (E_{max})$ where, r_{min} represents the next release of a future red job and E_{max} corresponds to the maximal level of energy in the battery.

4.1.2BG-Green-BWP

BG-Green-BWP (i.e. Blue When Possible) relies on the same principle of BG-Green-RTO, but it executes blue jobs when there is no ready red jobs. Then an aperiodic task is executed if there is no ready periodic jobs (red or blue). Red and blue jobs are executed according to Green-BWP algorithm. If there is no ready periodic jobs, an aperiodic task is chosen from the ready aperiodic task list according to FIFO algorithm. Then E_{av} is computed so that the aperiodic task is executed only if the battery will be fully charged when the closest red job will be released.

4.1.3 BG-Green-AWP

Green-AWP (i.e. Aperiodic When Possible) has the same principle of BG-Green-BWP but it executes blue jobs when there is neither ready red job nor ready aperiodic task . Under Green-AWP, blue jobs have the smallest priority and an aperiodic task is executed if there is no ready red jobs. Red and blue jobs are executed according to Green-BWP algorithm. If there is no ready red jobs, an aperiodic task is chosen from the ready aperiodic task list according to FIFO algorithm. Then E_{av} is computed so that the aperiodic task is executed while E_{av} is positive. It means that the battery must be fully charged when the closest red job is released.

4.4 Illustrative examples

In this section, we consider the task set *T* which consists of 2 firm periodic tasks given by T $T = \tau_i(C_i, D_i, T_i, s_i, E_i) = \{\tau_1(3, 12, 12, 2, 7), \tau_2(3, 15, 15, 2, 8)$.

We assume that $P_r = 2$. The battery is initially fully charged, $E(0) = C = H.Pr = 20 = 6$.
$U_p = 0.45$ and $U_e = 0.54$ so that the system is not overloaded in energy and timing point of view. Moreover, $U_p^* = 0.4$ and $U_e^* = 0.45$
We assume that 3 aperiodic tasks arrive during the interval [0,60]:
$\varphi = \{\varphi_i(r_i^a, C_i^a, E_i^a)$; $\varphi_1(0, 9, 15)$; $\varphi_2(0, 3, 8)$; $\varphi_3(20, 7, 15)$.

Figure 1 shows the scheduling of the task set T under BG-Green-RTO during a hyperperiod P = PPCM(T_1, T_2) = 60 (the equivalent hyperperiod is equal to P^* = PPCM(T_1S_1, T_2S_2) =120).

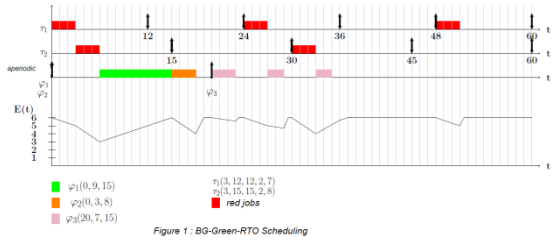

Figure 1 : BG-Green-RTO Scheduling

The aperiodic tasks $\varphi_1$and φ_2 arrive at time t=0. Butτ1 and τ2 are ready red jobs, so according to Green, RTO τ1 is executed first. Then τ2 is executed. At time *t=6*, there is no ready red job so that an aperiodic task could be executed if there is sufficient energy and if the battery will be full at the next red job release.

The amount of energy available until the next release of a red job (at time $t=r_{min} = 24$) is computed: $E_{av}(6) = E(6) + (r_{min} - 6)Pr - C = 3 + 36 - 6 = 33$.Eav(6) is positive then φ_1 begins its execution at time *t=6* until either a red job is released or the energy available in the battery becomes null. φ_1 complete its execution at time *t=15*. As φ_2 is waiting and there is no ready red job, the available energy is computed. $E_{av}(15)$ is positive then φ_2 begins its execution at time *t=15*. It completes its execution at time *t=18*. At time *t=20*, φ_3 arrives. There is no ready red jobs then the available energy is computed ($E_{av}(20) = 8$). As $E_{av}(20)$ is positive, φ_3 begins its execution at time $t = 20$. However the available energy becomes null at time t=23, then φ_3 stops its execution. The processor is idle for one unit of time to recharge the battery, etc..
We notice that in this example, φ_1, φ_2 and φ_3 have respectively the response time equal to 15, 18 and 15 time units. This shows that even if the system isn't overloaded in terms of time and energy, the response time of aperiodic tasks is still too long. Moreover, as blue job is always rejected, the QoS of the system depends on the loss parameter (s_i) of tasks. In this example, the QoS obtained is 50%.

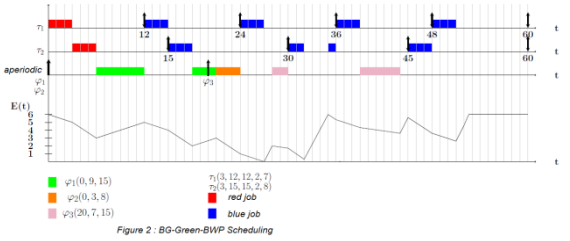

Figure 2 : BG-Green-BWP Scheduling

The Figure 2 shows the scheduling of the task set T under BG-Green-BWP during P = 60.
We notice that in this example, φ_1, φ_2 and φ_3 have respectively the response time equal to 21, 24 and 24 time units. In comparison with BG-Green-RTO, BG-Green-BWP gives a better QoS which is equal to 100% but the response time of aperiodic task is longer.

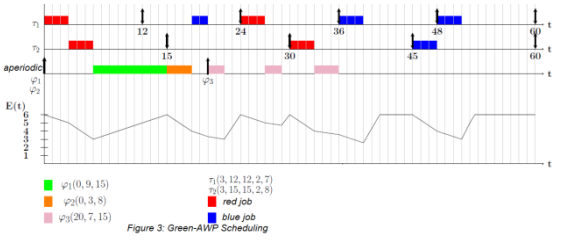

Figure 3: Green-AWP Scheduling

Figure 3 shows the scheduling of the task set T under Green-AWP during P = 60.We notice that in this example, φ_1, φ_2 and φ_3 have respectively the response time equal to 15, 18 and 16 time units. With Green-AWP, the response times obtained are better than the one obtained with BG-Green-BWP but the QoS obtained is worse : QoS=77.5%.In Comparison with BG-Green-RTO, the QoS obtained with Green-AWP is better (77.5% vs 50%) and the response times of aperiodic tasks are the same.

5. CONCLUSION AND FUTURE WORKS

In this paper, we have considered a firm real-time system that consists of firm periodic tasks and non critical aperiodic task. Our goal was to exploit losses due to the non execution of blue jobs to minimize the response time of aperiodic tasks. Based on the background server and the two energy aware schedulers Green-RTO and Green-BWP, we presented three techniques to execute aperiodic tasks in conjunction with periodic tasks namely BG-Green-RTO, BG-Green-BWP and Green-AWP. BG-Green-RTO gives the worst QoS among the three techniques but as the blue jobs are systematically rejected it will improve the response time of aperiodic tasks because there will be more energy available for them. BG-Green-BWP gives the best QoS among the three techniques by trying to execute blue jobs in priority but the response time of aperiodic tasks will be longer than the two other techniques. Finally, Green-AWP comes in the middle position in the sense that it improves the response time comparing to the one obtained with BG-Green-BWP, and it offers an acceptable QoS better than the one obtained with BG-Green-RTO. We conclude that according to the kind of the application, it will be better to choose one among the three techniques. If the event that occurs suddenly should be treated as soon as possible, it will be better to use BG-Green-RTO or Green-AWP. In future work, we are interested in minimizing the response time of aperiodic tasks by proposing new techniques to execute aperiodic tasks as soon as possible while still ensure acceptable QoS of the system.

References

1. J.-P. Lehozcky, L. Sha, and K.-K. Strosnider. Enhanced aperiodic responsivess in hard real-time environments. In proceedings of the 13th IEEE Real-Time Systems Symposium, 1987.
2. H. Chetto and M. Chetto. Some results of the earliest deadline scheduling algorithm. In IEEE Transactions on Software Engineering, 1989.
3. F. Yao, A. J. Demers and S. Shenker. A scheduling model for reduced CPU energy. In IEEE Symposium on Foundations of Computer Science,1995.
4. G. Koren and D. Shasha. Skip-over algorithms and complexity for overloaded systems that allow skips. In Proceedings of the 16th IEEE Real-Time Systems Symposium (RTSS'95), 1995.
5. M. Caccamo and G. Buttazo. Exploiting Skips In Periodic Tasks For Enhancing Aperiodic Responsiveness. In Proceedings of the 18th IEEE Real-Time Systems Symposium, 1997.
6. J.W.S. Liu. Real-Time Systems, Prentice-Hall, 2000.
7. A. Allavena and D. Moss,Scheduling of Frame-based Embedded Systems with Rechargeable Batteries, Workshop on Power Management for Real-Time and Embedded Systems (in conjunction with RTAS),2001.
8. H. Aydin, R. Melhem, D. Moss and P. Mejia-Alvarez. Determining optimal processor speeds for periodic real-time tasks with different power characteristics. In Euromicro Conference on Real-Time Systems, 2001.
9. C. Moser, D. Brunelli, L. Thiele and L. Benini. Real-time scheduling with regenerative energy. In 18th Euromicro Conference Real-Time Systems, 2006.
10. M. Chetto and H. Ghor. Real-time scheduling of periodic tasks in a monoprocessor system with rechargeable energy storage. In WIP RTSS09,2009.
11. N. Min-allah, Y. Wang, J. Xing, W. Nisar, and A. Kazmi. Towards dynamic voltage scaling in real-time systems - a survey. In IJCSES International journal of Computer Sciences and Engineering Systems, 2007.
12. M. Abdallah, M. Chetto and A. Queudet. Energy-aware schedulers for real-time energy harvesting systems with quality of service requirements. In Advances in Computational Tools for Engineering Applications (ACTEA), 2012 2nd International Conference, 2012.