A hybrid control law for energy-oriented tasks scheduling in wireless sensor networks

Olesia Mokrenko, Carolina Albea-Sanchez, Suzanne Lesecq, Luca Zaccarian

A hybrid control law for energy-oriented tasks scheduling in wireless sensor networks

Olesia Mokrenko, Carolina Albea, Suzanne Lesecq, Luca Zaccarian

***Abstract*—Energy is a key resource in Wireless Sensor Networks (WSNs), especially when sensor nodes are powered by batteries. This work investigates how to save energy of the whole WSN, thanks to control strategies, in real time and in a dynamic way. The energy management strategy is based on a Hybrid Dynamical System (HDS) approach. This choice is motivated by the hybrid inherent nature of the WSN system when energy management is considered. The hybrid nature basically comes from the combination of continuous physical processes, namely, the charge/discharge of the node batteries; while the discrete part is related to the change in the functioning modes and an *Unreachable* condition of the nodes. This approach provides a decentralized controller with low computational load that reduces the number of switching as compared to existing approaches. The proposed strategy is evaluated and compared in simulation on a realistic test-case. Lastly, they have been implemented on a real test-bench and the obtained results have been discussed.**

***Index Terms*—Hybrid Dynamical Systems, Wireless Sensor Network, Energy management**

I. Introduction

Technological and technical developments performed in the areas of wireless communication, micro-electronics and system integration have led to the advent of a new generation of large-scale sensor networks suitable for various applications [1]. A set of small electronic devices (the so-called sensor nodes), autonomous, equipped with sensors and able to communicate with each other wirelessly forms a Wireless Sensor Network (WSN) capable of monitoring a phenomenon of interest, and possibly react to the environment. They can provide high level information about this phenomenon to users by the combination of measurements taken by the various sensors and, then, communicate via the wireless medium. This technology promises to revolutionize our way of life, working and interacting with the physical environment around us. Sensor nodes able to communicate wirelessly, together with distributed computing capabilities, allow developing new applications that were impractical or too expensive a few years ago. Today, tiny and inexpensive sensor nodes can be literally scattered on roads, bridges, buildings wings of planes or forests, creating a kind of “second digital skin” that can detect various physical phenomena such as vibrations created by earthquakes or the change in the shape of a mechanical structure, or fire appearance and evolution in forests. As a consequence, many applications deal with the detection and monitoring of disasters (earthquakes, floods), environmental monitoring and mapping of biodiversity, intelligent buildings, advanced farming techniques, surveillance and preventive maintenance of machinery, medicine and health, logistics and intelligent transportation systems.

WSNs are often characterized by a dense deployment of nodes in large-scale environments with various limitations. These limitations are related to processing and memory capabilities, radio communication ranges, but also to energy resources as the sensor nodes may be powered by batteries. Note that even if sensor nodes are connected to power lines, they must be power-efficient because it is not acceptable from an ecology viewpoint, to drastically increase the number of power plants just to feed all these new devices. For sensor networks powered by batteries, changing the batteries has an extra cost related to their recharge and/or replacement that must be taken into account. Moreover, as the batteries in the network will certainly not be drained at the same time, the network maintenance teams change all the batteries during a unique intervention, leading to a suboptimal discharge of some of the batteries. Also, the sensor nodes may be placed in locations that are hard to access, for safety or economic reasons. Indeed, it is widely recognized that the energy limitation is an unavoidable issue in the design and deployment of WSNs because it imposes strict constraints on the network operation. Basically, the power consumption of the sensor nodes plays an important role in the life of the network. This aspect has become the predominant performance criterion for sensor networks. If we want the sensor network to perform its functionality satisfactorily as long as possible, these energy constraints imply trade-offs among different activities both at the node level and at the network layers.

The WSN lifespan increase has already been addressed in the literature, from sensor-level to network-level [2]–[4]. [5] provides an overview of these techniques. We employed Model Predictive Control (MPC) [6] in a WSN composed of three states: 2 functioning modes (*Active* and *Standby*) and an *Unreachable* condition due to environmental disturbances as communication breakdown, or an insufficient energy level [7]. This MPC controller selects the devices to be in *Active* mode both to limit the WSN overall energy consumption and extend its lifespan while the WSN fulfils a given “mission”. Even if the results are appealing, the control law is centralized and requires a certain level of computational load. Moreover,

Olesia Mokrenko was with Univ. Grenoble Alpes, F-38000 Grenoble, France / CEA, LETI, Minatec Campus while this work was performed.

Carolina Albea is with LAAS-CNRS, Université de Toulouse, CNRS, UPS, Toulouse, France

Suzanne Lesecq is with Univ. Grenoble Alpes, F-38000 Grenoble, France / CEA, LETI, Minatec Campus.

Luca Zaccarian is with LAAS-CNRS, Université de Toulouse, CNRS, UPS, Toulouse, France and also with the Dipartimento di Ingegneria Industriale, University of Trento, Italy

Manuscript received July XX, 2016; revised DD MM, YYYY.

the solution obtained by this approach presents an important number the switches, by often turning on and off certain sensors.

In the present work, we propose another control approach in order to improve the scalability issues though a decentralized scheme that reduces both the computational load and the number of switches, while the network lifespan is not decreased when compared to the MPC approach, leading to an improvement in the network power consumption. This control makes use of a Hybrid Dynamical System (HDS) approach [8], where solutions may continuously flow according to some differential equations and may discontinuously jump according to some rules. Therefore, an innovative strategy that fulfils the control objectives is proposed for the control of the functioning modes of the nodes.

The work is organized as follows. Section II is first dedicated to the system description and, then, we formulate the problem. We present a hybrid dynamic scheduling law for a specific simplified context in Section III, extending it to a general case in Section IV. In Section V, we evaluate in simulation the HDS control strategy on a realistic benchmark and we compare it with an MPC approach also provided [9]. In Section VI, we finally validate the control strategy in experiments through a test-bench. Section VII summarizes the main results and proposes future work directions.

Notations: Throughout the work, $\mathbb{N}^*$ denotes the set of positive natural numbers. Vector e_h denotes column h of the identity matrix, or the hth vector of the Euclidean basis.

II. WSN SYSTEM MODELLING

A. System description

Consider a WSN that contains $n \in \mathbb{N}^*$ sensor nodes S_i, $i = 1, \ldots, n$, powered by batteries. The nodes may also be equipped with a harvesting system. All nodes are functionally equivalent; thus they are interchangeable but their hardware can differ, e.g. batteries, processors may be unalike. The communication can be multi-hop or single-hop clustered (see [10]): each node sends its data through a gateway to the supervisor. The nodes can exhibit different functioning modes M_h, $h = 1, \ldots, m$, $m \in \mathbb{N}^*$, characterized by a known power consumption over a given period of time. Typically, the functioning modes are “Active”, “Standby”, etc. This system model includes an *Unreachable* condition, i.e. during the lifespan of the WSN, some nodes S_i may become unavailable. This situation can occur because of node physical damages, a lack of power resources or strong perturbations of the radio channel. The sensor nodes can exit from this mode when, for instance, the battery is recharged by the harvesting system, the physical damages are repaired or the radio channel is back to normal condition. Due to this unpredictable appearance/disappearance of nodes, it is mandatory to supervise the number of reachable S_i at every instant time in order to collect enough measurements from the application viewpoint at the supervisor side. The supervisor chooses the functioning mode of each node thanks to an energy management control strategy presented hereafter.

In this wireless sensor network context, consider that each node S_i is characterized by two states x_i and u_i, and an output y_i:

- $x_i(t,j) \in \mathbb{R}_{>0}$ is the remaining energy in the node battery, where t is the continuous time and j is the total number of jumps of the solution, with the constraint on the state x_i:
$$0 < \underline{x}_i \leqslant x_i(t,j) \leqslant \overline{x}_i, \tag{1}$$
where $\underline{x}_i$ and $\overline{x}_i$ are the lower and upper bounds of the battery capacity. $x_i(0,0)$ (i.e. the state value at $t = 0$ and $j = 0$) denotes the initial remaining energy.
- $u_i = \begin{bmatrix} u_{i1} & \cdots & u_{ih} & \cdots & u_{im} \end{bmatrix}^T \in \{0,1\}^m$ denotes the control states, related to the functioning mode of the node. The components of u_i are equal to 0 or 1. Thus, $u_i = e_h$ means that S_i is in mode M_h. Moreover, $u_i = \mathbf{0}^m$ denotes the *Unreachable* condition of node S_i and $u_i(0,0)$ denotes the initial control state. We ensure that the values of $u_i(0,0)$ are chosen so as to take into account the “mission” (described hereafter) and possibly to penalize the sensor nodes with smaller lifespan.
- y_i is the measurement of the remaining energy in the node battery delivered by each node at each sampling time kT_c to the supervisor. The supervisor needs this measurement to calculate the control. Therefore, at time kT_c:
$$\underline{x}_i \leqslant y_i \leqslant \overline{x}_i. \tag{2}$$

Moreover the node S_i is characterized by:

- an exogenous input $\alpha_i \in \{0,1\}$ that denotes the *Unreachable* condition for the state x_i. α_i is 0 (resp. 1) if S_i is able (resp. *Reachable*). This *Unreachable* condition can occur because of node physical damages, strong perturbations of the radio channel or when $x_i \leqslant \underline{x}_i$ among others. Note that when $x_i = \underline{x}_i = y_i$, the lifespan of node S_i (see (3) below) is equal to zero. In this situation we consider that the node is *Unreachable*.
- a power consumption (line) vector $B_i \in \mathbb{R}^m_{>0}$. The component b_{ih} of B_i denotes the power consumed by S_i when operating in mode M_h, that is when $u_i = e_h$. From a practical viewpoint, the components of B_i are assumed unequal which is consistent with what is observed for current commercial nodes;
- a disturbance input $w_i \in \{0;1\}$ that corresponds to the ability for the node S_i to harvest energy. This disturbance input cannot be controlled but may be predicted in some situation. $w_i = 1$ (resp. $w_i = 0$) is associated to the capability for the harvesting system to harvest (resp. not to harvest) energy from the environment;
- the harvested power value E_i corresponds to the amount of power provided by the harvesting system of node S_i. Note that E_i is in essence a time-varying value in real-life conditions. E_i may be in some situations predicted or even measured;
- the switching power consumption $\delta_i^{h \to r} \in \mathbb{R}_{>0}$ between two functioning modes takes into account the fact that switching node S_i from mode M_h to mode M_l has an energy cost. Moreover, $\delta_i^{h \to h} = 0$;

The lifespan of the *Reachable* sensor node S_i at time t can be computed by the supervisor as:

$$L_i = \frac{y_i - \underline{x}_i}{B_i u_i}. \tag{3}$$

B. Control objectives

A dynamic energy saving control policy has to be implemented at the application level in order to increase the WSN lifespan while meeting a given "service" requested by the application built on top of the WSN and that makes use of the measurements provided by the nodes. For instance, this service is expressed as a given (minimum) number of measurements that must be provided by the WSN over a defined time lapse. Hereafter, this service is called "mission". Typically, the "mission" has to be guaranteed over a geographically limited area, where each node can exchange its role with another one without decreasing the performance of the whole network.

The control objectives are to extend the lifespan of the WSN by reducing the overall energy consumption of the nodes via an appropriate management of the functioning mode of each node while providing a given "mission". During the continuous-time evolution of the solution (x_i, u_i) for each node S_i, we must have the "mission" defined as:

$$\sum_{i=1}^{n} \alpha_i u_i^T e_1 = d_1, \tag{4}$$

where $d_1 \in \mathbb{N}^*$ defines the exact number of nodes in mode M_1 (*Active* mode) that can be seen as an external reference.

The constraints that define "mission" (4) can be dynamically changed, depending for instance on the time schedule or on external events. This *dynamic mission* allows to adjust the needs of the application during the system evolution. For instance, during the day period, when people are present in an office (controlled by the proposed strategy), the "mission" can by defined as $d_1 = a < n$ nodes in the *Active* mode while during the night period, when there is nobody in the office, the "mission" could be $d_1 = b < a$.

C. Hybrid representation and pairwise jump rules

We start our modeling framework in a context where no failures are considered (scalars α_i are disregarded) and no harvesting is in place (inputs w_i are all zeros). This simplifies our initial analysis.

Within the above setting, the flow dynamics of the n nodes are given by:

$$\left\{ \begin{array}{rcl} \dot{x}_i & = & -B_i u_i, \\ \dot{u}_i & = & \mathbf{0}^m, \end{array} \right. \quad i = 1, ..., n, \ (\mathbf{x}, \mathbf{u}) \in \mathcal{C}, \tag{5}$$

where the flow set $\mathcal{C}$ is to be designed. The state dynamics $\mathbf{x}$ can be compactly written as:

$$\left\{ \begin{array}{rcl} \dot{\mathbf{x}} & = & -B\mathbf{u}, \\ \dot{\mathbf{u}} & = & \mathbf{0}^{nm}, \end{array} \right. \quad (\mathbf{x}, \mathbf{u}) \in \mathcal{C} \tag{6}$$

where $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T \in \mathbb{R}^n_{>0}$, $\mathbf{u} = \begin{bmatrix} u_1 & u_2 & \cdots & u_n \end{bmatrix}^T \in \{0,1\}^{nm}$ are the states and, $B = \text{diag}(B_1, B_2, \cdots, B_n) \in \mathbb{R}^{n \times nm}$.

The jump dynamics of the n nodes comprise the possibility that the available (*Reachable*) nodes autonomously decide to swap their respective role within the network. Given (5), one readily understands that swapping role means simply swapping the values of u_i. Then, we may define the sets $\mathcal{D}_{il}$ to provide conditions under which two nodes S_i and S_l are required to swap their roles, under the straightforward assumption that $\mathcal{D}_{il} = \mathcal{D}_{li}$, so that swapping is simultaneously enabled from both sides.

The adopted paradigm intrinsically defines a distributed scheduling paradigm, as long as one restricts the sets $\mathcal{D}_{il}$ to be non-empty (therefore relevant for the *potential swap* evaluation) only for pairs (i, l), belonging to the edges of a suitable undirected interconnection graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ characterizing the nodes.

In the general case, the following sets will be designed:

$$\mathcal{D}_{il} = \mathcal{D}_{li}, \ \forall \text{ nodes } S_i, S_l : \ (i, l) \in \mathcal{E}, \tag{7}$$

where $\mathcal{E}$ is the set of all edges in the interconnection graph. Based on the sets in (7), which will be designed in Section III, we can represent the swapping as an instantaneous update of the state of the nodes S_i and S_l, corresponding to:

$$\left\{ \begin{array}{rcl} \begin{bmatrix} x_i^+ \\ x_l^+ \end{bmatrix} & = & \begin{bmatrix} x_i - (u_i^+)^T \Delta_i u_i \\ x_l - (u_l^+)^T \Delta_l u_l \end{bmatrix}, \\ \\ \begin{bmatrix} u_i^+ \\ u_l^+ \end{bmatrix} & = & \begin{bmatrix} u_l \\ u_i \end{bmatrix}, \end{array} \right. \quad (\mathbf{x}, \mathbf{u}) \in \mathcal{D}_{il}, \tag{8}$$

with the switching consumption matrix Δ_i defined as:

$$\Delta_i = \begin{bmatrix} 0 & \delta_i^{2\to1} & \cdots & \delta_i^{h\to1} & \cdots & \delta_i^{m\to1} \\ \vdots & & \ddots & & & \vdots \\ \delta_i^{1\to m} & \delta_i^{2\to m} & \cdots & \delta_i^{h\to m} & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{m \times m}. \tag{9}$$

Note that for swapping, the exogenous inputs α_i and α_l of nodes i and l are necessary equal to 1.

Equation (8) only indicates the instantaneous swap for a pair of nodes, but for a complete description of the dynamics, we should also specify that across these jumps all other nodes S_i, $i = 1, ..., n-2$ do not experience any change of their x_i and u_i states. In particular, the relations in (8) may be compactly represented as:

$$\begin{bmatrix} \mathbf{x}^+ \\ \mathbf{u}^+ \end{bmatrix} = \begin{bmatrix} g_x^{il}(\mathbf{x}, \mathbf{u}) \\ g_u^{il}(\mathbf{x}, \mathbf{u}) \end{bmatrix} =: g_{il}(\mathbf{x}, \mathbf{u}), \quad (\mathbf{x}, \mathbf{u}) \in \mathcal{D}_{il}, \tag{10}$$

where $g_{il} : (\mathbb{R} \times \{0,1\}^m)^n \to (\mathbb{R} \times \{0,1\}^m)^n$ can be straightforwardly expressed from (8). With the pairwise rules in (10) associated to a jump set $\mathcal{D}_{sw}$ corresponding to the set where at least one pair of nodes is ready for a swap operation (namely $\mathbf{x} \in \mathcal{D}_{sw}$ if $\mathbf{x} \in \mathcal{D}_{il}$ for at least one pair i, l):

$$\mathcal{D}_{sw} = \bigcup_{(i,l)\in\mathcal{E}} \mathcal{D}_{il}, \quad G_{sw}(\mathbf{x}, \mathbf{u}) = \bigcup_{(i,l)\in\mathcal{E}:\ (\mathbf{x},\mathbf{u})\in\mathcal{D}_{il}} g_{il}(\mathbf{x}, \mathbf{u}), \tag{11}$$

where, by construction, G_{sw} is a set-valued mapping (multiple pairs may be ready to swap at the same time) that possesses the useful property of having a closed graph because its graph is the union of the (closed) graphs of g_{il}.

Now, we focus on the solutions to (5) and (11), insisting that they evolve in a specific set $\mathcal{O}$ where the remaining energy in the batteries of each node is positive for all nodes, and the input vector $\mathbf{u}$ has components equal to 0 or 1. Within this set, the flow set $\mathcal{C}$ is the closed complement of the flow set $\mathcal{D} = \mathcal{D}_{sw}$ relative to $\mathcal{O}$. More specifically:

$$\mathcal{O} = \mathbb{R}^n_{>0} \times \{0,1\}^{nm}, \quad \mathcal{C} = \left(\overline{\mathcal{O} \setminus \mathcal{D}}\right) \bigcap \mathcal{O}. \tag{12}$$

Within the set $\mathcal{O}$, due to the positivity of the entries in the vectors B_i and the constraints on the state values x_i, it is evident that all solutions will be bounded and not complete (their domain is bounded). Thus, the objective is to design the jump sets $\mathcal{D}_{il}$ in an intuitive way, the goal being to maximize the length of the solution domain in the ordinary time direction, i.e. the WSN lifespan, until that the "mission" (Eq. (4)) is no more satisfied. The WSN lifespan is named hereafter the *lifespan of the solution*.

III. Scheduling law for the fault-free case with two nodes

In the previous section, the WSN power management has been expressed as a control problem that decides the functioning mode for each node, with the design of the pairwise sets $\mathcal{D}_{il}$. Indeed, dynamics (5), (10) already clarify what is happening during the flowing (i.e. when the batteries are discharging) and what should happen at each reconfiguration of the network. Basically, when the state $(\mathbf{x}, \mathbf{u})$ belongs to $\mathcal{D}_{il}$, the nodes S_i and S_l swap their roles. To suitably design $\mathcal{D}_{il}$ for all $i \neq l$, the adopted control paradigm focuses on a pairwise reconfiguration rule.

This rule is described for the simplified case of two nodes S_1, S_2 without harvesting systems ($\mathbf{w}(t) = \mathbf{0}^n$) and two modes M_1, M_2 (resp. *Active* and *Standby*), with $\mathcal{D}_{12} = \mathcal{D}_{21} = \mathcal{D}$. For this specific case, an optimal result is proved hereafter. Moreover, in this case, it does not make sense to have an *Unreachable* state because no replacement node is available. Therefore, we focus on the fault-free two-nodes case, with $\alpha_i = 1 \ \forall \ i = 1, 2$. Moreover, $d_1 = 1$.

The rationale behind the choice of $\mathcal{D}_{12}$ is that we would like to design an algebraic condition on $\mathbf{x}$ and $\mathbf{u}$ that expresses when it is convenient to swap roles between both nodes in such a way to maximize the *lifespan of the solution*. This quantity may be expressed as a cost function J to be maximized. When designing $\mathcal{D}_{12}$, the expected lifespan if the solution does not perform further jumps:

$$J(x, u) = \min_{i: u_i \neq \mathbf{0}^m} \frac{x_i - \underline{x}_i}{B_i u_i} \tag{13}$$

is introduced. A first condition to encode in the flow set is that it is not convenient to jump (or swap roles) whenever $J(x^+, u^+) < J(x, u)$. Thus, one must ensure that:

$$\begin{aligned} J^+(x, u) := J(g_x(x, u), g_u(x, u)) < J(x, u) \\ \Rightarrow \ (x, u) \notin \mathcal{D}_{12}. \end{aligned} \tag{14}$$

Intuitively, the condition (14) means that no switch is performed if it reduces the lifespan. Note that this condition is a function of both x and u.

Even though condition in (14) is reasonable, it may still induce undesirable behaviours in some cases. For instance, consider two identical nodes, namely $\Delta_1 = \Delta_2$, $B_1 = B_2$. The *Active* (resp. *Standby*) mode is supposed associated with a large (resp. small) power consumption. Assume also that Δ_i are relatively small when compared to the power consumption of the *Active* mode. In this case, the best strategy is clearly to keep one node in the *Active* mode until its battery is drained, and then swap once and only once along the solution. However, picking $\mathcal{D}_{12}$ as the closed complement of the left hand condition of (14) will enforce extra unnecessary jumps as soon as the energy level in the active node becomes sufficiently small compared to the energy in the other node, and *vice-versa*, and so on.

One way to avoid this situation is to encode in $\mathcal{D}_{12}$ another condition: nodes will not swap if there is no "emergency" to do so. Indeed, when the left hand condition in (14) holds, solutions will still keep flowing (thus, no switch is applied) if waiting any further does not cause any reduction in the expected lifespan after the potential switch. To characterize this reduction, denote by i^* the index (or the set of indexes) that minimizes J^+, namely:

$$i^* = \arg \min_{i=1,2} \frac{x_i^+}{B_i u_i^+} = \arg \min_{i: u_i \neq \mathbf{0}^m} \frac{x_i - u_{3-i}^T \Delta_i u_i}{B_i u_{3-i}}. \tag{15}$$

Note that i^* may contain both indexes if the two expected lifespans coincide. Then, characterize the reduction as the derivative of J^+ during flow (if no jump would be performed):

$$\delta J(x, u) = \min_{l \in i^*} \frac{\dot{x}_l^+}{B_l u_l^+} = \min_{l \in i^*} \frac{-B_l u_l}{B_l u_{3-l}}. \tag{16}$$

Note that the derivative of $u_{3-i}^T \Delta_i u_i$ is zero along the flow. The function in (16) captures the idea of how damageable it is to postpone the swap. In other words, (16) expresses how much smaller J^+ will become if the solution flows for some extra time, before jumping. Since the two components in B_i are supposed not equal, then $\delta J(x, u) \neq -1$.

Clearly, if we keep flowing, the decrease rate of J will simply be 1 as the lifespan decreases linearly as time flows. Therefore, one extra criterion for the selection of $\mathcal{C}$ may be:

$$\delta J(x, u) \geq -1 \quad \Rightarrow \quad (x, u) \notin \mathcal{D}_{12}, \tag{17}$$

which is well defined, as indicated above, because $\delta J(x, u) \neq -1$. In particular, what happens along solutions, as long as $J^+ \geq J$ is that $\delta J(x, u) > -1$ whenever there is no urge to jump, and then once the "argmin" in (15) changes, we start getting $\delta J(x, u) \leq -1$ and a jump occurs.

To summarize, one selects:

$$\mathcal{D}_{12} = \{(x, u) : \ J^+(x, u) \geq J(x, u) \text{ and } \delta J(x, u) \leq -1\}. \tag{18}$$

Note that the definition is commutative that is, there is not specific role of nodes S_1 and S_2 in the selection of $\mathcal{D}_{12}$. The selection (18) enables to prove the following optimality result. Notice that hereafter $\underline{x}_i = 0 \leq x_i \leq \overline{x}_i$. If this is not the case, a simple change in the variable is performed.

Theorem 1. *Assume that B_i, $i = 1, 2$ has positive and distinct components and that matrices Δ_i, $i = 1, 2$ have strictly positive off diagonal elements. Consider any initial condition with positive values of $x_i(0, 0)$ and with $u_i(0, 0)$, $i = 1, 2$ being two independent columns of the identity matrix. The solution of* (5)*,* (11)*,* (18) *is unique and it has a lifespan equal to the maximum lifespan that can be obtained by selecting arbitrary jump times for* (5)*,* (8) *starting from the same initial conditions.*

Proof. The proof is carried out by first (step 1) showing that the (optimal) solution providing the maximum lifespan performs at most one jump (one swap between both nodes), and then (step 2) showing that the solution generated by the proposed jump rule is optimal.

Step 1. Assume that the optimal solution $\varphi_{opt} = (x_{opt}, u_{opt})$, having lifespan T, performs more than one jump (or swap) and denote by t_1 and t_2 the first 2 jump times. These jump times clearly satisfy $0 \leq t_1 \leq t_2 \leq t_3$, where we denote by t_3 either the next jump time, or the lifespan time $t_3 = T$. Since T is the lifespan of the solution, we must have $x_{opt,1}(T, j) > 0$, $x_{opt,2}(T, j) > 0$, and that at least one of them is zero, where j is the total number of jumps of the solution. Consider now another solution (x^*_{opt}, u^*_{opt}) starting from the same initial condition $(x(0, 0), u(0, 0)) = (x_{10}, x_{20}, u_{10}, u_{20})$ and performing one jump at time $t^*_1 = t_1 + (t_3 - t_2)$, and then performing the same jumps as the original solution does, namely $t^*_2 = t_3$, …, $t^*_{j-1} = t_j$. Note that this new solution performs one more jump less than the original one. Then, it can be proved that at ordinary time $t^*_2 = t_3$ (and before the possible corresponding jump), both outputs x_1 and x_2 of the "*" solution are strictly larger than the corresponding states of the original *opt* solution. As a consequence:

$$0 < x_i(T, j) < x^*_i(T, j - 1), \quad i = 1, 2, \tag{19}$$

which implies that the lifespan of the "*" solution is larger, thus completing the proof of this step. We only prove (19) for the case $i = 1$ as the case $i = 2$ is identical. To this end, consider:

$$\begin{aligned} x^*_{opt,1}(t^*_2, 1) = x_{10} - (t^*_1 - 0)B_1 u_{10} - (t^*_2 - t^*_1)B_1 \Pi u_{10} - \\ (\Pi u_{10})^T \Delta_1 u_{10} = x_{10} - (t_1 + (t_3 - t_2))B_1 u_{10} - \\ (t_2 - t_1)B_1 \Pi u_{10} - (\Pi u_{10})^T \Delta_1 u_{10}, \end{aligned}$$

where the swaping matrix $\Pi := \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$ and $t^*_2 - t^*_1 = t_2 - t_1$ is chosen. Instead, for the original solution we have:

$$\begin{aligned} x_{opt,1}(t_3, 2) = x_{10} - (t_1 - 0)B_1 u_{10} - (t_2 - t_1)B_1 \Pi u_{10} - \\ (t_3 - t_2)B_1 \Pi^2 u_{10} - (\Pi u_{10})^T \Delta_1 u_{10} - \\ u^T_{10} \Delta_1 \Pi u_{10} = x_{10} - (t_1 + (t_3 - t_2))B_1 u_{10} - \\ (t_2 - t_1)B_1 \Pi u_{10} - (\Pi u_{10})^T \Delta_1 u_{10} - u^T_{10} \Delta_1 \Pi u_{10}, \end{aligned}$$

with $\Pi^2 = I$. This clearly shows that the second state is smaller because it performs one extra jump and the off diagonal terms of Δ_i, $i = 1, 2$ are all positive.

Step 2. Consider now the structure of the jump set in (18) and note that all along any solution $\varphi(t, j) = (x(t, j), u(t, j))$, the function $\psi(t, j) := t + J(\varphi(t, j))$ denotes the envisioned lifespan (in the future) of the solution in the case when no other jump happens, while $\psi^+(t, j) := t + J^+(\varphi(t, j))$ denotes the envisioned lifespan of the solution if a single jump happens at the current time. This is easily understandable from the fact that J and J^+ measure the remaining lifespan, so the total amount of the envisioned future flow. Then, at each hybrid time instant (t, j) in dom φ (hybrid time domain [8]), the envisioned lifespan corresponds to such an envisioned remaining lifespan plus the already elapsed time t. As an additional property, note that along any solution, the function ψ is non-decreasing. Indeed, its derivative along the flows is trivially zero, while the first condition in (18) implies that it is non-decreasing at jumps. Moreover, it is straightforward to conclude from (16) that:

$$\frac{d}{dt}\psi^+(t, j) = 1 + \delta J(\varphi(t, j)), \tag{20}$$

which can never be zero because of the assumption that the two components in the matrices B_i are not equal. We now split the proof in two cases comparing the optimal solution to our solution and establishing that they have the same lifespan.
Case 1: If the optimal solution φ_{opt} performs no jumps, then the envisioned lifespan from the initial condition is already T, namely $\psi(0, 0) = J(\varphi(0, 0)) = J(\varphi_{opt}(0, 0)) = T$. As a consequence, our solution will flow for at least (therefore exactly, from optimality) T ordinary time because the function ψ is non-decreasing.
Case 2: If the optimal solution φ_{opt} performs one jump, denote by $t^* \in [0, T)$ the time of that jump and note that it must satisfy:

$$t^* + J^+(\varphi_{opt}(t^*, 0)) = t^* + J(\varphi_{opt}(t^*, 1)) = T. \tag{21}$$

Denote also $\psi_{opt}(t, j) = t + J(\varphi_{opt}(t, j))$. Since the solution flows for all $(t, j) \in [0, t^*) \times \{0\}$, and since $\frac{d}{dt}\psi^+_{opt}$ can never be zero as emphasized after (20), it must necessarily be that $\frac{d}{dt}\psi^+_{opt}(t, 0) > 0$ for all $t \in [0, t^*)$. Otherwise there would be another solution jumping before t^* and lasting longer than φ_{opt} which is impossible due to optimality. As a consequence, since our solution φ starts from the same initial condition as φ_{opt}, it must hold, also based on (20), that $1 + \delta J(\varphi(0, 0)) = \frac{d}{dt}\psi^+(0, 0) = \frac{d}{dt}\psi^+_{opt}(0, 0) > 0$, meaning, according to the second condition in (18), that our solution does not jump at the initial time. Due to the uniqueness of solution of our flow dynamics, we also conclude that $1 + \delta J(\varphi(t, 0)) > 0$ for all $t \in [0, t^*]$, thus implying that our solution behaves optimally until t^*. The interval is now closed due to continuity of the solution along the flows. Then, we have from (21) that $\psi^+(t^*, 0) = t^* + J^+(\varphi(t^*, 0)) = T$, which completes the proof by the non-decreasing property of ψ established above. □

IV. Scheduling law for the general case

The approach of Section III is now extended to the case with $n > 2$ nodes, $m \geq 2$ modes, the *Unreachable* condition and harvesting systems.

In a real-life condition, the WSN system control is not based on the state x_i but on the measurement of outputs y_i characterized in (2). Moreover to take into consideration the harvesting action we now take into account the inputs w_i $i = 1, ..., n$ quantifying the harvested energy at node i. In particular, we generalize (5) to

$$\left\{\begin{array}{rcl} \dot{x}_i & = & -B_i u_i + E_i w_i, \\ y_i & = & \alpha_i x_i, \\ \dot{u}_i & = & \mathbf{0}^m, \end{array}\right. \qquad i = 1, ..., n,\ (\mathbf{y}, \mathbf{u}) \in \mathcal{C}, \quad (22)$$

which can be compactly written as

$$\left\{\begin{array}{rcl} \dot{\mathbf{x}} & = & -B\mathbf{u} + E\mathbf{w}, \\ \mathbf{y} & = & \alpha\mathbf{x}, \\ \dot{\mathbf{u}} & = & \mathbf{0}^{nm}, \end{array}\right. \qquad (\mathbf{y}, \mathbf{u}) \in \mathcal{C} \quad (23)$$

where $\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \cdots & w_n \end{bmatrix}^T \in \{0,1\}^n$, $\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}^T \in \mathbb{R}^n_{>0}$, $\alpha = \operatorname{diag}(\alpha_1, \alpha_2, ..., \alpha_n) \in \{0,1\}^{n\times n}$, and $E = \operatorname{diag}(E_1, E_2, \cdots, E_n) \in \mathbb{R}^{n\times n}$. Note that the Unreachable condition for node S_i corresponds to $\alpha_i = 0$, $u_i = \mathbf{0}^m$ and $w_i = 0$. Likewise, jump rules now only depend on the accessible measurements y. Within this context, J_{il}, J^+_{il}, k^*_{il}, δJ_{il} can be defined as those in (13), (14), (15), and (16), respectively, for each $(i, l) \in \mathcal{E}$ as follows:

$$\begin{aligned} J_{il}(\mathbf{y}, \mathbf{u}) &:= \min_{k=i,l;\ k: u_k \neq \mathbf{0}^m} \frac{y_k - X^{min}_k}{B_k u_k} & (24) \\ J^+_{il}(\mathbf{y}, \mathbf{u}) &:= J_{il}(g^{il}_y(\mathbf{y}, \mathbf{u}), g^{il}_u(\mathbf{y}, \mathbf{u})) & (25) \\ k^*_{il} &= \operatorname*{argmin}_{k=i,l;\ k: u_k \neq \mathbf{0}^m} \frac{y_k - u^T_{k^c} \Delta_k u_k}{B_k u_{k^c}} & (26) \\ \delta J_{il}(\mathbf{y}, \mathbf{u}) &= \min_{k \in k^*_{il}} \frac{-B_k u_k}{B_k u_{k^c}}, & (27) \end{aligned}$$

where k^c represents the "second" node in a pair, namely $k^c = i$ if $k = l$, and *vice-versa*.

With the above definitions, the pairwise jump sets $\mathcal{D}_{il}$ can now be defined by suitably generalizing the expression in (18):

$$\begin{aligned} \mathcal{D}_{il} = \{(\mathbf{y}, \mathbf{u}) :\ & J^+_{il}(\mathbf{y}, \mathbf{u}) \geq J_{il}(\mathbf{y}, \mathbf{u}) \\ & \text{and } \delta J(\mathbf{y}, \mathbf{u}) \leq -1 \text{ and } \alpha_i = \alpha_l = 1\}, \end{aligned} \quad (28)$$

where the swap is now inhibited if any of the two nodes in a pair is in the *Unreachable* condition. Selection (28) is then completed by the definition of G_{sw} and $\mathcal{D}_{sw}$ in (11).

Jump rules must now be defined to take into account the presence of the exogenous input α, that captures the possibility for nodes to fall into the *Unreachable* condition, and the dynamically changing "mission" during the system evolution. Thus, for each node, three supplementary jump rules are defined.

The first one corresponds to the situation when the node S_i is in a mode M_h and the corresponding α_i goes to zero, *i.e.* the node becomes *Unreachable*). Then, the input u_i is automatically switched to the *Unreachable* value, i.e. $u_i = \mathbf{0}^m$:

$$\left\{\begin{array}{l} x^+_i = x_i \\ u^+_i = \mathbf{0}^m, \end{array}\right. \qquad (\mathbf{y}, \mathbf{u}) \in \mathcal{D}^0_i, \quad (29a)$$

$$\mathcal{D}^0_i := \{(\mathbf{y}, \mathbf{u}) :\ u_i = e_h \text{ and } y_i \leq \underline{x}_i\}, \quad (29b)$$

$$\mathcal{D}_0 := \bigcup_{i=1,\ldots,n} \mathcal{D}^0_i, \quad G_0(\mathbf{y}, \mathbf{u}) = \bigcup_{i:\ (\mathbf{y},\mathbf{u}) \in \mathcal{D}^0_i} g^0_i(\mathbf{y}, \mathbf{u}). \quad (29c)$$

In this case, note that the supervisor did not receive information related to the remaining energy x_i for the *Unreachable* node with $\alpha_i = 0$. Moreover, $g^0_i(\mathbf{y}, \mathbf{u})$ has a similar structure to functions $g_{il}(\mathbf{x}, \mathbf{u})$ introduced in (10). In particular, $g^0_i(\mathbf{y}, \mathbf{u}) : (\mathbb{R} \times \{0,1\}^m)^n \to (\mathbb{R} \times \{0\}^m)^n$ is a compact way of representing the jump map in (29a) essentially comprising identities in all entries except for those related to node S_i.

The second jump rule corresponds to a more sophisticated action ensuring that the "mission" is fulfilled even when an *Active* node becomes *Unreachable*, and in the presence of nodes in the *Standby* mode. This means that there are nodes in mode M_h where $h \neq 1$. This second jump rule forces one *Reachable* node (without any pre-specified priority) to become *Active* if the "mission" is no more satisfied. This will typically happen if an *Active* node falls in the *Unreachable* condition because of a jump arising from (29). This jump rule is given by:

$$\left\{\begin{array}{l} x^+_i = x_i \\ u^+_i = e_1, \end{array}\right. \qquad (\mathbf{y}, \mathbf{u}) \in \mathcal{D}^1_i, \quad (30a)$$

$$\mathcal{D}^1_i := \{(\mathbf{y}, \mathbf{u}) :\ u_i = e_{h\neq 1} \text{ and } \sum_{i=1}^{n} \alpha_i u^T_i e_1 \leq d_1 - 1\}, \quad (30b)$$

$$\mathcal{D}_1 := \bigcup_{i=1,\ldots,n} \mathcal{D}^1_i, \quad G_1(\mathbf{y}, \mathbf{u}) = \bigcup_{i:\ (\mathbf{y},\mathbf{u}) \in \mathcal{D}^1_i} g^1_i(\mathbf{y}, \mathbf{u}), \quad (30c)$$

where $g^1_i(\mathbf{y}, \mathbf{u}) : (\mathbb{R} \times \{0,1\}^m)^n \to (\mathbb{R} \times \{0,1\}^m)^n$ is a compact way of representing the jump map in (30a) essentially comprising identities in all entries except for those related to node i. Basically, $g^1_i(\mathbf{y}, \mathbf{u})$ has a similar structure to functions g_{il} introduced in (10).

The third jump rule corresponds to the situation when the "mission" changes dynamically, *i.e.* d_1 is a time-varying integer function. In this case, node S_i jumps to the *Standby* mode from the *Active* one:

$$\left\{\begin{array}{l} x^+_i = x_i \\ u^+_i \in \{e_h : h \neq 1\}, \end{array}\right. \qquad (\mathbf{y}, \mathbf{u}) \in \mathcal{D}^2_i, \quad (31a)$$

$$\mathcal{D}^2_i := \{(\mathbf{y}, \mathbf{u}) :\ u_i = e_1 \text{ and } \sum_{i=1}^{n} u^T_i e_1 \geq d_1 + 1\}, \quad (31b)$$

$$\mathcal{D}_2 := \bigcup_{i=1,\ldots,n} \mathcal{D}^2_i, \quad G_2(\mathbf{y}, \mathbf{u}) = \bigcup_{i:\ (\mathbf{y},\mathbf{u}) \in \mathcal{D}^2_i} g^2_i(\mathbf{y}, \mathbf{u}), \quad (31c)$$

where $g^2_i(\mathbf{y}, \mathbf{u}) : (\mathbb{R} \times \{0\}^m)^n \to (\mathbb{R} \times \{0,1\}^m)^n$ is a compact way of representing the jump map in (31a) essentially comprising identities in all entries except for those related to node i. It has a very similar structure to function g_{il} introduced in (10). In the case, where $d_1(t_1) < d_1(t_2)$ with $t_1 < t_2$, the jump rule (30) is executed.

For these jump rules (11), (28), (29)-(31), the energy in the battery of node S_i does not experience any discontinuity, thus $x^+_i = x_i$, where $x_i \in [\underline{x}_i, \overline{x}_i]$. Remember that if $y_i = \underline{x}_i$, the node S_i has the lifespan $L_i = 0$ at the time t (see (3)), therefore it is considered as an *Unreachable* node.

With the definitions in (11), (28), (29c), (30c), (31c), the complete jump dynamics can be compactly represented as:

$$\begin{bmatrix} \mathbf{x}^+ \\ \mathbf{u}^+ \end{bmatrix} \in G(\mathbf{y}, \mathbf{u}), \quad (\mathbf{y}, \mathbf{u}) \in \mathcal{D}, \quad (32)$$

where $\mathcal{D} = \mathcal{D}_{sw} \cup \mathcal{D}_0 \cup \mathcal{D}_1 \cup \mathcal{D}_2$, and G is a set-valued map corresponding to the outer semicontinuous hull of G_{sw}, G_0, G_1, G_2 (the outer semicontinuous hull is a map whose graph is the union of the graphs of the composing maps) namely allowing for a jump whenever one of the corresponding rules is enabled. The discrete-time dynamics (32) together with (22) correspond to a hybrid description of the WSN system under the specific control action.

We may now express the following result, which is a straightforward consequence of the jump rules in (29) - (31) and of the fact that the jumps from (28) have no effect on the "mission" because they merely correspond to swapping the states of nodes S_i and S_l.

Proposition 1. *Given any solution to* (5), (32), *if at least* d_1 *nodes are not in the* Unreachable *condition at each time in its domain, then the "mission"* (4) *is always satisfied during the flows of the solution.*

The proof of the proposition is omitted because it is a straightforward consequence of the definition of jumps rules and of the fact that flowing is forbidden unless the "mission" is accomplished.

V. Simulation of the HDS strategy

To validate the theoretical results presented above, some simulations on a realistic benchmark are performed to illustrate the features of our control law based on HDS. The proposed scenario is built from device data-sheets and laboratory measurements. Firstly, we will describe the simulation conditions. Then, we will discuss the results provided by our HDS strategy and, they will be compared to another strategy based on MPC previously proposed by the authors [6].

A. Simulation conditions

We validate our proposed energy management strategy with a WSN composed of $n = 6$ nodes that may possibly embed a harvesting system. Each node can stay in one of three states, namely, $m = 2$ functioning modes and the *Unreachable* condition. The two functioning modes are defined by the real node abilities. This means that they can be placed in the *Active* or *Standby* mode, M_1 and M_2 respectively. Moreover, they may fall in the *Unreachable* condition due to environmental disturbances as communication breakdown, or an insufficient energy level. This context is similar to the one presented in [7].

The "mission" is defined by $d_1 = 3$ nodes in *Active* mode (M_1), which is considered sufficient to guarantee the service that the WSN must provide in its given application.

The components of matrix B_i in (5) are computed from the information provided in Table I. Likewise, Table II provides the initial energy capacity of the batteries associated with each sensor node and, the harvesting system availability.

B. Simulation of the HDS strategy

The simulations of the WSN with the proposed HDS control strategy are performed in the MATLAB/Simulink environment and the HyEQ Toolbox [11].

Table I: Average energy consumption B_i, $[mAV \cdot hour]$ of node S_i in the functioning modes M_1 and M_2.

Sensor node	**Mode M_1**	**Mode M_2**
S_1	34,854	5,846
S_2	36,482	6,031
S_3	36,593	6,105
S_4	36,482	6,105
S_5	36,556	6,105
S_6	33,041	5,735

Table II: Node battery and harvesting system characteristics

Node	Nom. battery capacity, $[mAV \cdot hour]$	Harvesting availability E_i, $[mAV \cdot hour]$	Energy coef. ξ_i, [1]	Harvesting period, *per* 24 hours
S_1	3885	–	1	–
S_2	3885	–	0,8	–
S_3	3885	77,7	0,9	7-12
S_4	3515	–	0,7	–
S_5	3515	99,9	1	13-18
S_6	3515	–	1	–

Figure 1 shows the evolution of the functioning mode for each node when the HDS control is applied. Note that nodes S_3 and S_5 present harvesting capability, which explains the oscillatory evolution of their remaining energy, depicted in Figure 2. We can observe that the remaining energy of all nodes is never equal to 0. Basically, this is related to the constraints (1) on the state x_i that avoid to fully drain the battery. From both Figures 1 and 2, we see that the "mission" ($d_1 = 3$ nodes in mode M_1) is fulfilled approximately until 196 hours, which represents the WSN lifespan.

During the simulation, the nodes jump between mode M_1 and mode M_2, following the rule (22)–(28). When a sensor node has no more enough energy, it falls in the *Unreachable* condition, following the rule (29). This same node may jump back to the *Reachable* condition at any time and be taken into account by the rule (28), which is decentralized.

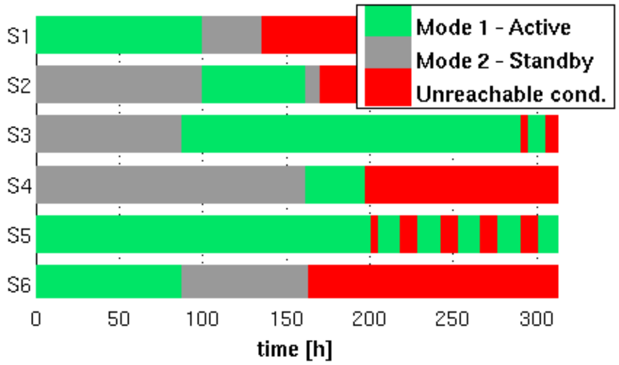

Figure 1: Evolution of the functioning modes of the nodes.

C. Comparison among HDS and other strategies

Now, we will compare the proposed HDS approach with other strategies, namely, a "basic controller" and a MPC-based one. Both of them are discussed in [6]. The "basic

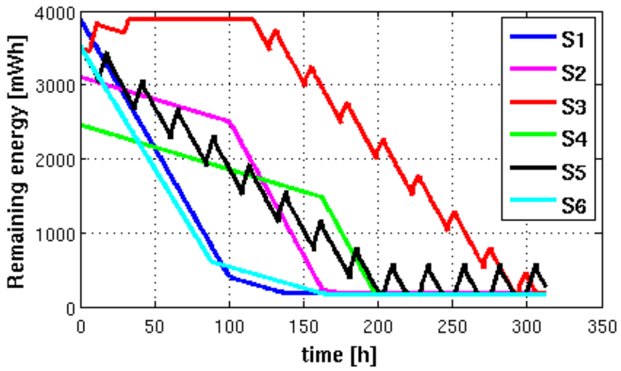

Figure 2: Evolution of the remaining energy in each node battery.

controller" manages the node activity in a centralised way under the constraint of fulfilling the "mission" and without any other constraint. On the other hand, the MPC considers all constraints mentioned in this work. However, small differences between the HDS approach and the MPC one must be highlighted (see [6] for details), namely, the MPC:

- is a centralized controller that penalizes the most consuming nodes;
- the switching energy loss ($\delta_i^{k\rightarrow l}$) is integrated in b_{il} [6].

For comparison purpose, we consider the WSN benchmark without and with the harvesting profile described above. Likewise, we will use two MPC formulations, namely a Mixed Integer Quadratic Programming (MIQP) and a Mixed Integer Linear Programming (MILP), this latter having some computational advantages due to the particular structure of the problem to be solved.

A summary of these strategies is shown in Table III (see [12] for details). It must be highlighted that the results obtained with the control strategies based on MPC and HDS approaches are promising. Compared to the "basic controller', the WSN lifespan obtained:

- using the MPC approach is 37$\%$ longer without harvesting and 49$\%$ longer with harvesting;
- likewise, using the HDS approach it is 28$\%$ longer without harvesting and 39$\%$ longer with harvesting.

Moreover, for the specific benchmark case, the WSN lifespan obtained using the MPC approach can be longer by 4.1% and 8.5% without and with (resp.) harvesting systems compared to the HDS approach. This phenomenon can be explained by the characteristics of the different controllers. On the one hand, the strategy based on MPC does not limit the number of switches compared to the strategy based on HDS, where a switch is enforced only if it significantly extends the WSN lifespan. As a consequence, we get a larger number of switches by using MPC, as shown in [6]. The reduction of the number of switches allows to further reduce the energy consumption (especially the one due to the switches) and to not increase the overload of the radio channel, because the number of communications also decreases. In fact, if we need to swap roles between nodes, extra message must be sent between the supervisor and the nodes.

Table III: Comparison of scenarios in terms of WSN lifespan and number of switches.

Strategies	WSN lifespan without harv. sys., [hours]	WSN lifespan with harv. sys., [hours]	Number of switches
Basic	128	192	3
MPC/MIQP	175	287	$\approx 10^2$
MPC/MILP	171	284	$\approx 10^2$
HDS	164	266	7 (5)

We can see that the benchmark presents different levels of complexity depending on the number of nodes. Therefore, we have an impact on the computational workload according to the implemented control strategy. Figure 3 shows the simulation time to solve the feedback algorithm for a control period $T_c = 1$ hour (one control step) for different numbers of sensor nodes. Note that the simulated system with the MPC strategy is discrete and, thus, a difference equation is computed. However, the simulated system with the control strategy based on HDS is hybrid (continuous/discrete). Thus, differential equations must be numerically integrated and, the control algorithm must be discretized (here we select a sampling time $T_c = 1$ hour). The computational times are obtained in the Matlab environment using the *tic-toc function*. Matlab runs on one core of the Intel Xeon Processor E5620 with 12 MB of cache, 2.40 GHz and 5.86 GT/s [13]. From these computational times, even if the evaluation is rough, it is evident that the control strategy solved with MIQP is more "demanding" compared to the problem solved with MILP. The comparison between HDS and MPC approaches shows that the HDS algorithm is in the range of two orders of magnitude faster than the MIQP approach for the benchmark with $n = 6$ nodes.

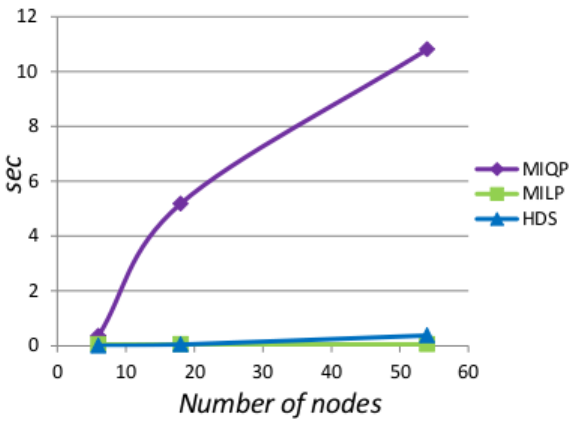

Figure 3: Complexity analysis of the algorithms.

To summarize, the advantages of using our developed HDS approach are:

- the control strategy is scalable and reliable;
- the number of switches is reduced and,
- the computational workload is smaller.

VI. Experimental results

The energy management strategy based on HDS described in this paper is now implemented in a real test-bench. This strategy consists of continuous physical processes of the energy charge/discharge of the node batteries (described by x_i) and, of finite-states for the functioning modes (described by u_i). Note that in real-life conditions, the remaining energy in the node battery is supposed to be measured and delivered by each node at each sampling time, i.e. we have $y_i = x_i$ at each time kT_c. In the present experiment, T_c is chosen equal to 1 hour.

A. Test-bench description

The hardware test-bench considered here consists of one supervisor, one router and $n = 6$ sensor nodes. The supervisor is a laptop where the control strategy is implemented. It is equipped for communication with a WiFi card. The WiFi router is used to increase the range of the network and to form the infrastructure topology. The sensor nodes are Flyport WiFi 802.11g modules developed by OpenPicus [14] connected to a sensing element. Each node has a battery, as shown in Figure 4. The sensing element contains a temperature & humidity sensor DHT-11 [15]. The Flyport WiFi 802.11g module is a programmable system-on-module with integrated WiFi 802.11g connectivity. For a programmed sensor node, two functioning modes are considered, namely the *Active* mode M_1 and the *Standby* mode M_2 that are described as follows:

- in the *Active* mode M_1, sensing, computing and communication units are "duty cycled" [16] with a sampling period $T_s = 1$min to sense, process and exchange data with the supervisor;
- the *Standby* mode M_2 is similar to mode M_1, *i.e.* it is duty cycled, but with a much larger sampling period $T_w \gg T_s$. In this mode, the node remains in the sleep state and, it wakes up each $T_w = T_c = 1$ hour to receive the commands from the supervisor and monitor the remaining battery capacity.

Based on the technical data-sheet of the Flyport module and on laboratory measurements, the numerical values of the energy consumption of the nodes were estimated. They are presented in Table I.

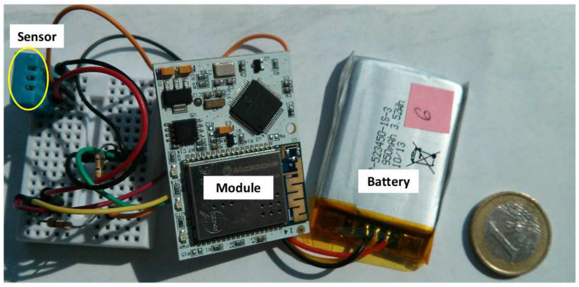

Figure 4: Used sensor nodes.

Each node presents one type over two Li-polymer rechargeable batteries [17], namely, *type 1* for sensor nodes $S_1 - S_3$, and *type 2* for $S_4 - S_6$. Note that harvesting systems are not available. The Li-ion batteries have nominal capacities $\overline{x}_i = 3885$mWh for *type 1*, and $\overline{x}_i = 3515$mWh for *type 2*. These batteries embed an electronic protection circuit ensuring a minimum State-of-Charge (SoC) value corresponding to 10% of the nominal capacity for *type 1* batteries, and corresponding to 16% of the nominal capacity for *type 2* batteries. As a consequence, the lower energy limit of the sensor nodes using a *type 1* battery is equal to $\underline{x}_i = 3885\text{mWh} \cdot 0.1 = 388,5\text{mWh}$ (for nodes $i = 1, 2, 3$). Instead for a *type 2* battery, it is $\underline{x}_i = 3515\text{mWh} \cdot 0.16 = 562.4\text{mWh}$ (for nodes $i = 4, 5, 6$).

After the battery calibration step described in [18], an accurate experimental model of the battery $Voltage - SOC$ curve is built. Figure 5 depicts an example of such SoC profiles for both types of new batteries, at 23°C, which corresponds to the typical environment temperature in the office where the sensor nodes are deployed. This calibration step together with the protection circuit allows to safely (i.e. without damaging the battery) and efficiently exploit the battery capabilities. The estimation of the remaining energy in the battery of all nodes is implemented together with the control law.

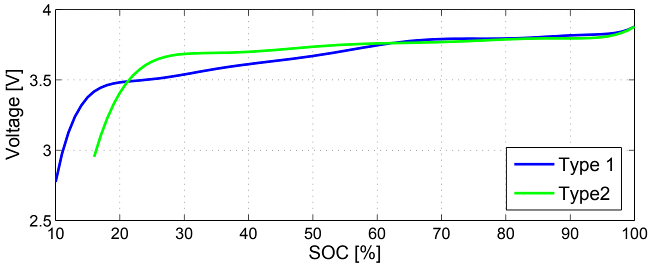

Figure 5: SOC profiles for both battery types.

The coordination between the sensor nodes and the supervisor is realized *via* the router, which uses the LINC coordination environment [19]. LINC is a resource-based middleware with a particular emphasis on the sensor/actuator network field. The LINC middleware addresses issues raised by applications considering a large number of sensor nodes (*i.e.* up to several hundreds), distributed over a wide field (*e.g.* a building or a set of buildings) and, connected via heterogeneous and unreliable communication protocols (*e.g.* various wireless networks) [20].

B. Control objectives

For this test-bench, the WSN with 6 sensor nodes has been deployed in a working office as shown in Figure 6. In order to control the air conditioning system, temperature and humidity are sensed through the sensor nodes. During the day, when the office is in use, the control of the air conditioning system requires measurements from 3 sensor nodes. During the night, measurements from only one sensor node are enough to ensure the appropriate control of the air conditioning unit. Therefore, the "mission" is split in two phases corresponding respectively to "day" and "night" periods of time. Then, the constraints that define the "mission" have to be dynamically adjusted, depending on the time schedule, leading to a dynamic "mission", as summarized in Table IV.

Table IV: Definition of the dynamic mission

Time period		d_1	Objectives
Day	8am−6pm	3	3 nodes in mode M_1
Night	6pm−8am	1	1 node in mode M_1

Figure 6: WSN deployed in a working office.

At time instant $t = 0$, all the sensor nodes of the system are in the *Active* (M_1) mode. Basically, they must transmit the initial remaining energy in their battery. Then, the supervisor checks whether the node batteries have enough energy so that any node S_i can fulfil the "mission" (i.e. being placed in mode M_1). If this is true, during the day period, 3 nodes are placed in mode M_1 while the other ones that are *Reachable* are placed in M_2. During the night period, only 1 node is placed in mode M_1 and the others are placed in mode M_2.

When a sensor node S_i is not seen by the supervisor, it is considered in the *Unreachable* condition. This situation occurs for instance when the remaining energy of a sensor node is lower than or equal to $\underline{x}_i$, or because of any other faulty conditions. Then, the control law assigns new modes to the remaining nodes in order to meet the dynamic "mission" while reducing the energy consumption of the sensor network. Note that this is the only centralized decision in our control law while all the other decisions are distributed. When the supervisor receives again information from a node that was beforehand in the *Unreachable* condition, it places this node in mode M_1 or in mode M_2 depending on the "mission" fulfilling.

C. Experimental results

The energy management strategy based on the HDS approach is now tested on the described test-bench. For this purpose, the control strategy is also written in Python and integrated in the LINC middleware.

The switching consumption matrices are supposed equal for all nodes. They are given by:

$$\Delta_i = \begin{bmatrix} 0 & 0.01 \\ 0.05 & 0 \end{bmatrix} mJ. \tag{33}$$

Note that the energy consumed for a switch is very hard to measure. It depends on the sensor node itself and on the environment conditions, especially the communication disturbances. However, the selection of Δ_i in (33) corresponds to an indirect way to penalize switches.

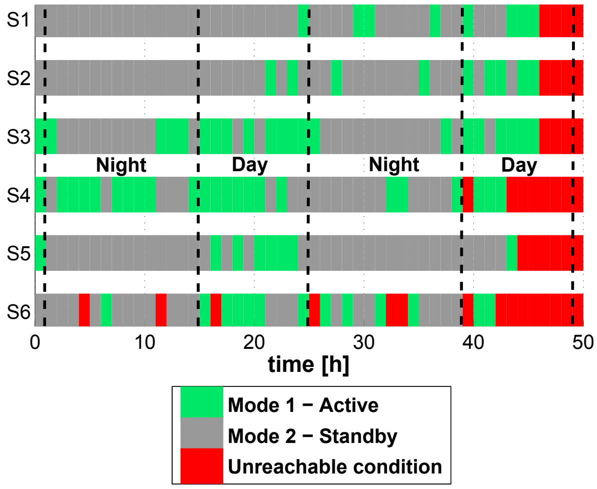

Figure 7: Evolution of the functioning mode of the nodes.

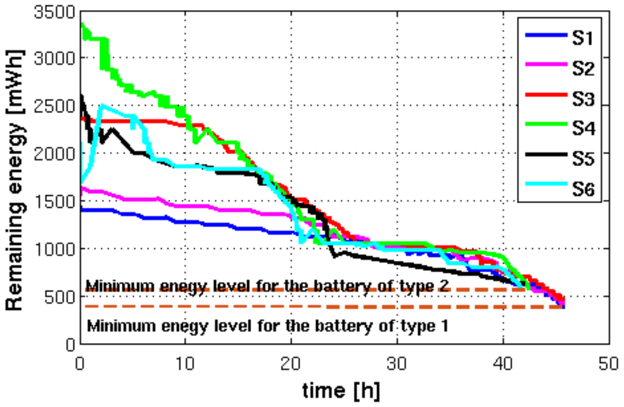

Figure 8: Estimated remaining energy of each sensor node.

The implementation in real-life conditions starts at 5 $p.m.$ and lasts 50hours. The evolution of the functioning modes and of the estimated remaining energy of all nodes are presented in Figure 7 and Figure 8 respectively. It can be observed that the WSN lifespan is equal to 46 hours, where the dynamic "mission" is fulfilled periodically for day and night periods of time. During the WSN lifespan, sensor nodes S_4 and S_6 fall in the *Unreachable* condition, which may be caused by perturbations of the radio channel. At the end of the experiment, the nodes still present remaining energy in their batteries, as shown in Table V.

Table V: Remaining energy at the end of the experiment.

Sensor node	$\underline{x}_i$ mWh	$x_i(t_{\text{final}})$ mWh
S_1	388,5	388,5
S_2	388,5	427,3
S_3	388,5	466,2
S_4	462,4	562,4
S_5	462,4	632,7
S_6	462,4	562,4

We can see in Figure 8 that the slope of the curves of the estimated remaining energy is also proportional to the energy consumption in the different functioning modes of the nodes. The number of switches is larger in real-life conditions compared to the simulation in Matlab. In fact, this can be the reflection of larger radio channel perturbations and of the dynamic "mission" that imposes extra switches. Therefore, we emphasize the importance of reducing the number of switching with the controller, because in real-life, additional switches may be expected.

VII. Conclusion

Wireless Sensor Networks (WSNs) have undergone in the last few years a tremendous growth, both in industry and in academia. This is mainly due to the different potentials of this technology, such as the essential absence of wiring costs and the possibility of application domains that were inaccessible for wired sensor nodes. However, WSNs also need to face significant design challenges because of their limited computational and storage capabilities, and the limited available energy as sensor nodes are usually supplied by a battery. The energy is a critical resource and it often constitutes a major obstacle to the deployment of sensor networks that will be used everywhere in the world of tomorrow.

In this work, a distributed control algorithm for energy management in a WSN has been proposed. The energy in the sensor nodes is modeled using a Hybrid Dynamical System representation. The WSN has to provide a given functionality (named "mission") while taking into account the possibility for nodes to fall in an *Unreachable* condition. The optimality of our approach has been proven in the case of two nodes and an extension of the theorem is developed to encompass a more general case considering n nodes with harvesting systems. Simulation results on a realistic benchmark and comparison with an Model Predictive Control approach show the potential of the proposed control strategy, since the present distributed controller reduces the number of switches between two modes and the computational workload, besides making the WSN scalable and reliable. These benefits have been validated in experiments on a real test-bench.

VIII. Acknowledgment

Work supported in part by ANR project LimICoS contract number 12 BS03 005 01, by grant OptHySYS funded by the University of Trento, by the H2020 TOPAs project, nb 676760, and by the ARTEMIS ArrowHead project nb 332987.

References

[1] B. Krishnamachari, *Networking wireless sensors*. Cambridge University Press, 2005.

[2] N. P. Mandru, "Optimal power mamagement in wireless senosr networks for enhanced life time," *Journal of Global Research in Computer Science*, vol. 3, no. 4, pp. 73–78, 2012.

[3] W. Hailong, S. Yan, and W. Tuming, "Dynamic power management of wireless sensor networks based on grey model," in *3rd International Conference on Advanced Computer Theory and Engineering*, vol. 1, 2010, pp. 133–137.

[4] V. Sharma, U. Mukherji, V. Joseph, and S. Gupta, "Optimal energy management policies for energy harvesting sensor nodes," *IEEE Transactions on Wireless Communications*, vol. 9, no. 4, pp. 1326–1336, 2010.

[5] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad hoc networks*, vol. 7, no. 3, pp. 537–568, 2009.

[6] O. Mokrenko, M. I. Vergara-Gallego, L. Lombardi, L. Lesecq, D. Puschini, and C. Albea, "Design and implementation of a predictive control strategy for power management of a wireless sensor network," *The 14th annual European Control Conference*, 2015.

[7] P. S. Sausen, J. R. de Brito Sousa, M. A. Spohn, A. Perkusich, and A. M. N. Lima, "Dynamic power management with scheduled switching modes," *Computer Communications*, vol. 31, no. 15, pp. 3625–3637, 2008.

[8] R. Goebel, R. G. Sanfelice, and A. R. Teel, *Hybrid Dynamical Systems: modeling, stability, and robustness*. Princeton University Press, 2012.

[9] O. Mokrenko, S. Lesecq, W. Lombardi, D. Puschini, C. Albea, and O. Debicki, "Dynamic power management in a wireless sensor network using predictive control," in *40th Annual Conference of the IEEE Industrial Electronics Society*, 2014, pp. 4756–4761.

[10] V. Mhatre and C. Rosenberg, "Homogeneous vs heterogeneous clustered sensor networks: a comparative study," in *IEEE International Conference on Communications*, vol. 6, 2004, pp. 3646–3651.

[11] R. Sanfelice, D. Copp, and P. Nanez, "A toolbox for simulation of hybrid systems in matlab/simulink: Hybrid equations (hyeq) toolbox," in *Proceedings of the 16th international conference on Hybrid systems: computation and control*, 2013, pp. 101–106.

[12] O. Mokrenko, "Energy management of a wireless sensor network at application level," Ph.D. dissertation, Universite Toulouse III Paul Sabatier, 2015.

[13] "Intel xeon processor e5620, ark.intel.com," 2015. [Online]. Available: ark.intel.com

[14] "www.openpicus.com," 2015. [Online]. Available: http://www.openpicus.com

[15] "www.aosong.com/en/products/details.asp?id=109," 2015. [Online]. Available: http://www.aosong.com/en/products/details.asp?id=109

[16] S. Du, A. K. Saha, and D. B. Johnson, "Rmac: A routing-enhanced duty-cycle mac protocol for wireless sensor networks," in *26th IEEE International Conference on Computer Communications*, 2007, pp. 1478–1486.

[17] "www.farnell.com/datasheets/1666650.pdf and 1666648.pdf," 2015. [Online]. Available: http://www.farnell.com/datasheets/1666650.pdf and 1666648.pdf

[18] O. Mokrenko, M.-I. Vergara-Gallego, W. Lombardi, S. Lesecq, and C. Albea, "WSN power management with battery capacity estimation," in *13th IEEE International NEW Circuits And Systems conference*, 2015.

[19] M. Louvel and F. Pacull, "Linc: A compact yet powerful coordination environment," in *Coordination Models and Languages*, 2014, pp. 83–98.

[20] L.-F. Ducreux, C. Guyon-Gardeux, S. Lesecq, F. Pacull, and S. R. Thior, "Resource-based middleware in the context of heterogeneous building automation systems," in *38th Annual Conference on IEEE Industrial Electronics Society*, 2012, pp. 4847–4852.

Olesia Mokreko was Researcher-Engineer at CEA-LETI in Grenoble, France when this work was performed. She received the M.S. degree in Aeronautics in 2013 from the National Technical University of Ukraine Kyiv Polytechnic Institute, and the Ph.D. degree in Automatic Control in 2015 from the University of Toulouse III. Her current research interests include energy management and wireless sensor networks.

Carolina Albea is an associate professor at the University of Toulouse III (Universit Paul Sabatier) from 2011. Her research is performed at the LAAS-CNRS. She received her PhD in automatic control in 2010 from the University of Sevilla, Spain, and the University of Grenoble, France. From 2010 to 2011, she held a post-doctoral position at the CEA-LETI Minatec campus in Grenoble, France, on the control of nanoelectronic circuits. Her research interests cover: nonlinear control, hybrid dynamical systems, consensus, control of electronic devices and control of networks.

Suzanne Lesecq passed the Agrgation in electrical engineering in 1992. She received the Ph.D. in process control from the Grenoble Institute of Technology, in 1997. She joined the University of Grenoble in 1998 where she has been appointed as Associate-Professor from 1998 to 2006 and full-time Professor from 2006 to 2009. She joined CEA-LETI in mid-2009. Her topics of interest are application of control theory to power consumption management of Many-Processor System-on-Chip and sensor networks, and control of power converters.

Luca Zaccarian has been Assistant and then Associate Professor at the University of Roma Tor Vergata (Italy) since 2000. Then in 2011 he became Directeur de Recherche at the LAAS-CNRS, Toulouse (France) and since 2013 he also holds a part-time associate professor position at the University of Trento, Italy. Luca Zaccarian's main research interests include analysis and design of nonlinear and hybrid control systems, modeling and control of mechatronic systems. He was a recipient of the 2001 O. Hugo Schuck Best Paper Award given by the American Automatic Control Council and he is a fellow of the IEEE.