Model-Based Design for Wireless Body Sensor Network Nodes

Ivan Beretta†, Francisco Rincon*, Nadia Khaled,
Paolo Roberto Grassi§, Vincenzo Rana§†, David Atienza†, Donatella Sciuto§
†Embedded Systems Laboratory (ESL), EPFL, Switzerland, {ivan.beretta, david.atienza}@epfl.ch
*DACYA, Universidad Complutense de Madrid, Spain, francisco.rincon@fdi.ucm.es
§Politecnico di Milano, Italy, {grassi, rana, sciuto}@elet.polimi.it

***Abstract*— Wireless body sensor networks (WBSNs) are a rising technology that allows constant and unobtrusive monitoring of the vital signals of a patient. The configuration of a WBSN node proves to be critical in order to maximize its lifetime, while meeting the predefined performance during signal sensing, preprocessing, and wireless transmission to the base station. In this work, we propose a model-based optimization framework for WBSN nodes, which is centered on a detailed analytical characterization of the most energy-demanding components of this application domain. We also propose a multi-objective exploration algorithm to evaluate the node configurations and the corresponding performance tradeoffs. A case study is discussed to validate the proposed framework, proving that our model captures the behavior of real WBSNs and efficiently leads to the determination of the Pareto-optimal configurations.**

I. Introduction

Wearable wireless body sensor networks (WBSNs) for health monitoring and diagnosis, as well as emergency detection, are gaining popularity and will deeply change healthcare delivery in the next years [1]. A WBSN node is a low-power device that collects vital signs, preprocesses and then sends them to a coordinator that performs most of the workload [2].

The design of WBSN nodes is mainly focused on maximizing the lifetime of the node by reducing the energy consumption, although other performance requirements such the delay and quality of the delivered data must be kept into account. As a design may depend on tens of parameters, an efficient multi-objective optimization framework is required to explore the design space and to identify the Pareto-optimal solutions. The most critical part of this process is to provide a fast yet reliable estimation of all the optimization metrics. Three possible techniques to evaluate a solution are: a) an exhaustive set of experiments, which however cannot be automated; b) a network simulation, which is slow and hence impractical when a large number of potential solutions needs to be explored; c) an analytical model of the node, which favors a quick optimization and a better analysis of the node behavior.

Model-based evaluation has a long history, as many models have been proposed to describe the basic components of a node (e.g., memory, radio, etc. [3][4][5]). However, combining those components to form a model of the entire node is no easy task, as the model should include meaningful information of the specific node, while being reusable and not requiring a massive amount of experimental data to be constructed. In order to cope with the difficulty of building reliable node characterizations, a promising trend is to generate statistical models from a properly-selected set of experimental data [6]. The experimental data is used to estimate the parameters of a set of simple equations, which however do not provide an application-aware evaluation of the node.

In this work, we tackle the model-based optimization problem from a different perspective, i.e., by narrowing the scope of the model to WBSNs and focusing on their most typical features (defined in [2]). In particular, this allows us to discard those aspects that are not generally required, such as complex networking or task assignment. In this way, we are able to analytically capture aspects like lifetime, transmission quality and application performance with a high precision, while still keeping the model general and reusable. We then propose an adapted version of the multi-objective *simulated annealing* algorithm to perform the design space exploration and find the Pareto-optimal configurations. The proposed approach has been tested on a real application for ECG monitoring that uses compressed sensing [7], and is implemented on the Shimmer™[8] state-of-the-art commercial platform.

The remaining of this paper is structured as follows. In Section II we describe the reference node architecture, while in Section III we review the literature regarding node modeling and optimization. We then propose our model-based framework in Section IV. A real-world case study that validates the proposed approach is provided in Section V. The main conclusions of the work are summarized in Section VI.

II. Architecture of the Target Node

As a reference, we consider the popular WBSN architecture where a software application for data processing is executed on a microcontroller, and the remaining services are delegated to an operating system. Figure 1 provides a structural outlook of this class of nodes. In this work, we assume that a node transmits its data to a central coordinator through the typical star topology used in WBSNs. Each node in the network generates a constant traffic, thus avoiding data bursts that may interfere with the periodic transmission of the other nodes in the network. The result of this assumption is that the data delivery delay can be analytically estimated, and that the characterization of the network is simplified and can be seamlessly included in the node model.

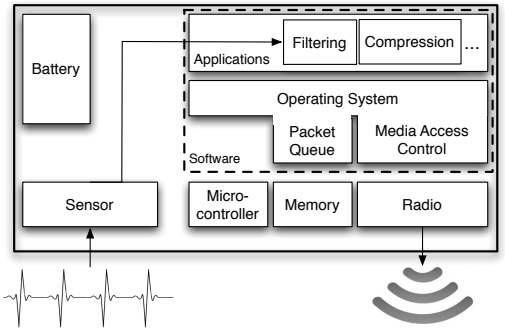

Fig. 1. Block diagram of the reference node architecture

The *sensor* component describes the hardware that conditions and samples the raw signal at a frequency that depends on the signal and on constraints such as the Nyquist–Shannon theorem. Then, the samples are quantized by an A/D converter using a number of bits that depends on its resolution.

The *applications* components comprise all the software programs used to process the sensed data, including filtering, feature extraction, compression and aggregation. For illustration and without loss of generality, we herein consider the *compressed sensing* application [7], which reduces the amount of data by exploiting the sparsity of many body signals. In particular, the sampled signal is compressed with a certain ratio, which is set equal to one if no compression is adopted.

The *operating system* (OS) provides services such as the interaction with the hardware, the software-level management of the sensing, the memory and the radio transmission. The OS also manages a set of queues, including the ones for interprocess communication, and the one containing the packets to be transmitted. Furthermore, the OS implements a MAC protocol to share the access to the wireless medium among the nodes in the star-topology network.

The *microcontroller* (μC) is the component of the platform in charge of executing the OS and the software applications. Depending on the hardware, techniques such as dynamic voltage scaling [4] might be available to allow the microcontroller to be active for a limited time (*duty cycle*), and to switch to a low-power state when there is no task to be executed.

The *memory* bank is used to store the data required by the applications and the OS. Although a larger memory exhibits a higher energy consumption [3], a limited size may affect the capacity of the internal queues, and hence the performance of the applications and the throughput of the transmission.

Finally, the *radio* component describes the hardware used to modulate and transmit the data through the wireless channel. Depending on the characteristics of the platform, the wireless transmission power and the modulation scheme can be adjusted, even dynamically [5], to determine the distance that is covered with a predetermined packet error rate (PER).

III. Related Work

Model-based optimization of wireless sensor nodes is a topic that has been already explored in the literature. However, most of the related works characterize the energy consumption of one of the node components shown in Figure 1, whereas just few of them aim at optimizing multiple components and performance metrics, which is instead the purpose of our work.

Power models of single components such as the microcontroller [4] and the memory [3] are available outside the scope of wireless sensor networks. In the context of sensor networks with low-power requirements, instead, a lot of effort has been put in the wireless communication part, with the design of new MAC protocols and the characterization of the radio. In [13], the authors propose an alternative MAC protocol for the *ZigBee* standard that introduces new power-saving policies. In [14], a model that relates the routing performed at the MAC level to the node lifetime is proposed. However, both works assume a multi-hop routing, and thus they cannot be applied to the star topologies used in WBSNs [2]. A characterization of the radio has been proposed in [5], where the energy consumption is related to parameters like the bit error rate and the modulation. Similarly, [15] provides a model for an IEEE 802.15.4 transmitter, which is supported by a set of physical measurements. However, the models of the single components cannot capture the interdependencies that exist between the different parts of the node, and in particular they often discard the effects of the application.

A few existing works have tried to propose an optimization that considers several components of the node. In [10] the authors propose a platform-based design methodology for industrial control. Although the work considers all the aspects of the node design, it is not based on an analytical model, and it mainly focuses on the generation of a complex network, which is not a critical aspect in WBSNs. In [12], different energy/delay tradeoffs are explored by exploiting voltage and modulation scaling. Similarly, [16] proposes a model-based optimization framework for star-topology networks, and a genetic algorithm to reduce the energy consumption by acting on the voltage and the modulation level. However, the number of parameters involved in the optimization is a small subset of the ones that can be tuned on real nodes. In [11], the authors propose an optimized transmission schedule to minimize the packet delay. The work shows good potential if transmission delay is the main objective, but it is not proved to scale in more sophisticated multi-objective optimization scenarios. A very relevant example of application-driven design is proposed in [9], where a multi-objective optimization involving all the system components is provided in the field of wildlife monitoring. The work shows how a deep knowledge of the final application can lead to an optimized node lifetime, while guaranteeing the quality of service (high data rate and low distortion). However, it heavily relies on the experimental data and hence is very specific for the target domain.

IV. The Proposed Model-Based Optimization Framework

The proposed model-based optimization framework is targeted to WBSN nodes whose architecture is similar to the one in Section II. The framework includes two parts: a node model that estimates the relevant quality metrics associated to the system, and an algorithm to find the optimal configurations.

A. Analytical Node Model

The node model includes fundamental parts that describe the common structures of every WBSN node, and advanced parts that can be further detailed according to the specific scenario (e.g., the application, the communication channel), and parameters that need to be determined through experimental data. The contribution to the energy consumption of each component described in Section II is characterized, although the level of detail differs from component to component depending on the number of relevant design parameters.

The proposed model is comprehensive as it captures the interdependencies between different components. The main metric, i.e., the overall energy consumption, is expressed as a function of the all hardware components of the system (sensor, microcontroller, memory and radio), whose behavior is influenced by the applications and the OS. In particular, the energy consumption per second is expressed as:

$$E_{node} = E_{sensor} + E_{\mu C} + E_{memory} + E_{radio} \ . \quad (1)$$

However, since the straightforward reduction of the energy consumption may lead to the loss of performance in one or more components of the node, we defined a set of performance metrics that only involve one or two components, in order to keep them monitored during the design.

1) Sensor: The sensor component consists of a transducer to detect the signal, and a hardware circuit to sample the data at a frequency $f_{sampling}$ and quantize it with a resolution of ρ_{ADC} bytes. The energy consumption of the former can be considered as a constant related to the specific sensor, while the latter is linearly related to the sampling frequency [17]:

$$E_{sensor} = E_{transducer} + [\alpha_1 \cdot f_{sampling} + \alpha_0] \ , \quad (2)$$

where α_1 is a constant depending on the capacitance and the square of the supply voltage [17] of the A/D converter, while α_0 describes leakage effects and is determined experimentally.

Although specific metrics can be defined to estimate the performance of the sensor, e.g. the signal-to-quantization-noise ratio, a more meaningful evaluation can be obtained by combining it with the subsequent application component.

2) Applications: The applications are software programs that do not directly dissipate energy, but influence the performance of the microcontroller, the memory and the radio. In particular, the applications define the duty cycle (ψ_{app}) of the microcontroller, the memory requirement (σ_{app}) and the average number of memory accesses per second (γ_{app}), which can be determined using software profiling. As mentioned in Section II, we assume that data can be compressed at the application level with ratio CR, which generates the following amount of packets to be transmitted per second (R_p):

$$R_p = f_{sampling} \cdot \frac{CR \cdot \rho_{ADC}}{H_{payload}} \ , \quad (3)$$

where $H_{payload}$ is a parameter that defines the number of data bytes included in each communication packet, and hence $H_{payload}/\rho_{ADC}$ denotes the number of samples per packet.

In order to evaluate the performance of the applications, domain-specific metrics can be defined. If compression is the only processing performed at the application level, such a metric is defined as the quality of the reconstructed signal.

3) Operating System: The OS is composed of software routines that implement services such as the packet queue and the MAC layer. The software routines of the OS can be modeled as any other software application, thus requiring a duty cycle ψ_{OS} from the microcontroller, and a maximum memory σ_{OS} that is accessed γ_{OS} times per second on average. For the sake of analysis, we separate the memory required by the transmission queue from the remaining memory occupied by the OS, because a detailed model of the transmission queue is crucial to characterize the throughput of the system.

The MAC layer implemented in the OS manages the access to the wireless channel shared among a known number of nodes (N_{nodes}) connected to the WBSN coordinator. The access policy defined by the MAC algorithm can be modeled using two quantities: a transmission window of length Δ_{tx} when the node can transmit without conflicts, and the number of times this window is repeated per second, N_{tx}. Those quantities can be directly computed for contention-free access mechanisms, but they can also be determined statistically for contention-based policies. In order to enforce the access policy, the algorithm may require a number of control messages to be exchanged from the node to the coordinator. We denote this number as $\Phi_{Node \to C}$, and the length of those messages as $H_{Node \to C}$), and the opposite as $\Phi_{C \to Node}$, of length $H_{C \to Node}$. Finally, the MAC protocol defines the control information that must be included in each packet (typically a header and a checksum), thus determining the final length H_{packet} of each packet. The transmission queue, which can contain up to λ packets, has then a size of λH_{packet} bytes.

A set of performance metrics can be defined for the OS component, and in this work we focus on the throughput and the packet delivery delay. In particular, the system should guarantee a throughput of R_p packets per second as required by the application, but certain packets might need to be retransmitted with a probability equal to PER. We define $R_p^{(r)}$ as the packet rate including the retransmissions, which is equal to $R_p \cdot [1 + \varpi(PER)]$, where $\varpi(PER)$ is the estimated number of retransmissions, including possible multiple retransmissions of the same packet. Then, a sufficient condition to guarantee that the desired throughput is met is the following:

$$N_{tx} \left\lfloor \frac{\Delta_{tx}}{T_{packet}} \right\rfloor \geq R_p^{(r)} \ \wedge \ \lambda \cdot N_{tx} \geq R_p \ , \quad (4)$$

where T_{packet} is the packet transmission time, and $N_{tx} \cdot \lfloor \Delta_{tx}/T_{packet} \rfloor$ is the maximum channel capacity guaranteed by the MAC layer. The second condition prevents the capacity of the queue from being a bottleneck and to avoid dropped packets. If the conditions are satisfied, we can provide the following worst-case estimation of the packet delivery delay:

$$Delay = \frac{1}{N_{tx}} \cdot \frac{\sum_{i=N_{tx}-\nu}^{N_{tx}-1} i}{\nu} \ , \ \nu = \left\lceil \frac{R_p}{R_p^{(r)}} \cdot N_{tx} \right\rceil \ , \quad (5)$$

where ν indicates the fraction of the N_{tx} transmission windows that are required to send R_p packets. $Delay$ is then computed by considering that all the retransmissions occur before the R_p packets are transmitted.

4) Microcontroller: Similarly to the sensor, the consumption of the microcontroller is expressed as a function of its frequency $f_{\mu C}$. The processor needs to be active for a duty cycle defined by the application and the OS, before switching to a low-power mode where only leakage effects occur:

$$E_{\mu C} = (\psi_{app} + \psi_{OS}) \cdot \beta_1 \cdot f_{\mu C} + \beta_0 \ , \tag{6}$$

where β_1 depends on the capacitance and on the square of the supply voltage, and β_0 describes the leakage effects and should be determined experimentally. Note that, if the specific scenario does not allow the microcontroller to switch to a low power state, $\psi_{app} + \psi_{OS}$ should be set equal to one.

5) Memory: The system memory is used for the execution of the applications and the OS, and to store the packets queue. The memory size M, which is also the main quality metric for the design of the memory component, can be written as:

$$M = \sigma_{app} + \sigma_{OS} + \lambda H_{packet} \ . \tag{7}$$

The energy consumption of a memory component is due to two factors [3]: a dynamic consumption due to the memory accesses, and a leakage that is known to be proportional to the memory size and appears when the memory is not being accessed. The software applications and the OS access the memory γ_{app} and γ_{OS} times per second, respectively. The transmission queue is filled with a number R_p of packets per second and, since we defined the throughput to guarantee that no packet is dropped, it is eventually read at the same rate. Assuming that the memory access in read and write modes has the same cost, we can express the energy consumption as:

$$\begin{aligned} E_{memory} = &(2 \cdot R_p + \gamma_{app} + \gamma_{OS}) \cdot T_{mem} \cdot \zeta_{access} + \\ &[1 - (2 \cdot R_p + \gamma_{app} + \gamma_{OS}) \cdot T_{mem}] \cdot M \cdot \zeta_{idle} \ , \end{aligned} \tag{8}$$

where T_{mem} is the access time in read or write mode, while ζ_{idle} and ζ_{access} are hardware parameters that define the consumption in idle and accessing modes.

6) Radio: The energy consumed by the radio depends on the number of packets that are sent and received. In particular, when a transmission of one bit takes place, the energy consumption can be expressed as:

$$E_{tx} = [P_{carrier} + P_r] \cdot T_{bit} \ , \tag{9}$$

where $P_{carrier}$ is the power required to generate the signal carrier, P_r is the remaining consumption related to the radio circuit. T_{bit} indicates the average time to transmit one bit, which also includes all the control information added by the physical layer [5]. The value of $P_{carrier}$ can be determined according to the desired PER. In particular, given the level of noise at the receiver, it is possible to compute the signal-to-noise ratio and consequently the bit error rate (BER), as a function of $P_{carrier}$ and the modulation scheme [5]. Once the BER is known, the packet error rate can be expressed as the probability of one bit being erroneous in a packet of length H bytes, i.e., $1 - (1 - BER)^{8H}$.

Algorithm 1 Multi-Objective Simulated Annealing

```
for i = 1 ... N_executions do
    S ← GenerateRandomConfiguration()
    T ← T_0
    repeat
        S' ← GenerateNeighbor(S)
        if Cost(S') dominates Cost(S) then
            S ← S'
        else if Cost(S) dominates Cost(S') OR
        no domination between Cost(S) and Cost(S') then
            if RandomNumber() < P(T) then
                S ← S'
            end if
        end if
        T ← Annealing(T)
    until T < T_min
end for
```

The energy required to receive a bit (E_{rx}) is computed as in (9), where $P_{carrier}$ is equal to zero, and P_r has a different value during the receiving phase. As a consequence, the energy consumption of the radio can be expressed as:

$$\begin{aligned} E_{radio} = &E_{tx} \cdot \left[R_p^{(r)} \cdot 8H_{packet} + \Phi_{Node \to C} \cdot 8H_{Node \to C} \right] \\ &+ E_{rx} \cdot \left[\Phi_{C \to Node} \cdot 8H_{C \to Node} \right] \ . \end{aligned} \tag{10}$$

B. Node Optimization

The proposed model provides an accurate evaluation of the node, hence it is suitable for scalar or multi-objective exploration to find a set of Pareto-optimal configurations. However, even in an existing platform where all the hardware constants are already fixed, the optimization may still involve tens of design parameters, ranging from low-level hardware aspects to the tuning of software algorithms. In order to show how the design space can be efficiently explored using the estimation provided by the proposed node model, we introduce a heuristic algorithm based on *simulated annealing* [18]. The choice of this technique is motivated by its ability to handle a large number of design parameters, its scalability from single-objective [18] to multi-objective optimizations [19], and its good convergence properties. The proposed Multi-Objective Simulated Annealing (MOSA) algorithm is shown in Algorithm 1, and it features a different non-dominance policy with respect to the standard MOSA to improve the stability of the algorithm and consequently the quality of the results.

In order to set up the optimization, one or more metrics (e.g., E_{node}, $Delay$, M, PER) should be picked as objective functions, thus generating a cost vector $Cost(S)$ for each node configuration S. Since the single-objective simulated annealing inherently finds a single solution, the MOSA requires multiple executions ($N_{executions}$) of the annealing process [19] to populate the Pareto set: the larger $N_{executions}$, the more accurate the Pareto set is, although the MOSA outputs Pareto solutions from the early executions.

Each execution starts from an initial temperature T_0 (typical values are 300 or higher) and a random initial configuration S.

A new candidate solution S' is then then obtained by randomly modifying one of the parameters of S and, if the S' dominates S, then it is accepted. If the candidate is dominated by the current best, it can still be accepted with a probability P. Different criteria to compute P can be found in the literature [19]: after analyzing them, we concluded that the best way to compute P only involves the temperature T as follows:

$$P(Cost(S), Cost(S'), T) = e^{-C/T} \ , \ C \in \mathbb{R} \ . \qquad (11)$$

According to the classical formulation of the MOSA [19], a non-dominance condition between $Cost(S)$ and $Cost(S')$ leads to the acceptance of the candidate, in order to explore more solutions. This approach, however, may discard a good solution even at low temperatures. In our framework, we propose a different technique, which employes the transition probability even in non-dominance scenarios and allows the algorithm to converge as the temperature decreases.

The annealing scheme should guarantee that the temperature decreases slowly enough to allow the algorithm to converge. A popular scheme is the geometric one, where the temperature T is multiplied by a constant value lower than 1 (typical values are 0.99 or higher) at each iteration. The execution finishes when the temperature reaches a lower bound T_{min} (typically close to 5), that corresponds to a low transition probability.

V. A Case Study

We applied the proposed optimization framework on a real sensor node for ECG monitoring. In this section, we show how the model can be easily adapted to cope with real applications and widely-adopted standards, and how the MOSA optimization can efficiently explore different performance tradeoffs.

A. Case Study Overview

We consider a real system that samples the ECG and uses the compressed sensing [7] to reduce the amount of data to be transmitted. The compressed signal is then sent to a smartphone that acts as a central coordinator, reconstructs the ECG and performs analysis and detection tasks. The choice of compressed sensing is motivated by the improved node lifetime, indeed, experimental results [7] showed that compressing and sending data can increase the node lifetime by 9.7% when compared to transmission of the raw ECG.

The system is implemented on the Shimmer™commercial platform [8], which features an ultra low-power microcontroller working at a maximum frequency of 8*MHz*, 10*kB* of RAM memory, and a radio implementing the IEEE 802.15.4 communication. A 3-lead ECG sensor is connected through a daughter board. On the software side, FreeRTOS [20] was ported on the node, and it controls the sensing, the queue services and the beacon-enabled mode of the IEEE 802.15.4 MAC layer [21]. In this MAC protocol, a beacon is periodically sent by the coordinator to define the time structure in terms of superframes. A superframe is a time interval divided into an inactive and an active part, the latter being further divided into a contention-free and a contention-active portions. In this case study, only the contention-free part is used, so the transmission only occurs during *guaranteed time slots* (GTSs).

B. Mapping the Case Study on the Analytical Model

The model we introduced in Section IV-A provides a good characterization of many parts of the target node, but additional information can be included to further describe the application, the memory, the MAC and the radio modulation.

At the application level, we estimated the duty cycle ψ_{app} required by the compressed sensing, and in the current implementation it only marginally depends on the value of CR. The performance metric considered for this component is the *percentage root-mean-square difference* (PRD), which quantifies the difference between the original ECG and the one reconstructed by the coordinator from the compressed data. By analyzing the experimental data provided in [7], the PRD can be expressed as a fifth-order polynomial function of CR:

$$PRD = \omega_5 CR^5 - \omega_4 CR^4 + \omega_3 CR^3 - \omega_2 CR^2 + \omega_1 CR - \omega_0 \ , \qquad (12)$$

where the coefficients ω_n are positive constant values.

The total available memory on the node is $10kB$: according to the experimental results, $6.5kB$ are required by the compressed sensing application (σ_{app}), while $3.5kB$ are reserved for the FreeRTOS routines (σ_{OS}) and for the transmission queue, whose size is then upperbounded. In particular, for a packet length H_{packet} of $127B$ (i.e., the maximum value for the selected MAC), λ must be lower than 10.

The beacon-enabled IEEE 802.15.4 MAC layer can also be easily included in the node. Two protocol-specific parameters need to be defined: the *Superframe Order* (SFO), and the *Beacon Order* (BCO). The former determines the active period or *superframe duration* (SD), while the latter defines the interval between two beacons (BI) as follows:

$$SD = 15.36ms \cdot 2^{SFO} \ , \ BI = 15.36ms \cdot 2^{BCO} \ . \qquad (13)$$

The superframe structure can be mapped on the transmission window Δ_{tx} of our model, as the average transmission time per second is equal to SD divided by the number of nodes in the network. Similarly, BI defines how many times a superframe is repeated, hence it can be related to N_{tx}:

$$\Delta_{tx} = \frac{SD}{N_{nodes}} \ , \ N_{tx} = \frac{1}{BI} \ . \qquad (14)$$

In terms of control messages, the standard does not require any control message from node (thus $\Phi_{Node \to C} = 0$), whereas the coordinator sends a number of beacons that depends on BI, and an acknowledgment for each transmitted packet, hence:

$$\Phi_{C \to Node} = R_p^{(r)} + \frac{1}{BI} \ . \qquad (15)$$

Finally, the estimation of the PER for this case study can be obtained by computing the BER for the 4-PSK modulation [5] with the selected value of $P_{carrier}$.

C. Node Optimization

We applied the proposed MOSA to determine a set of Pareto-optimal node configurations for the case study application. The design parameters available on the target platform are $f_{\mu C}$, CR, $H_{payload}$, λ, BCO, SFO, and P_{tx}, while the

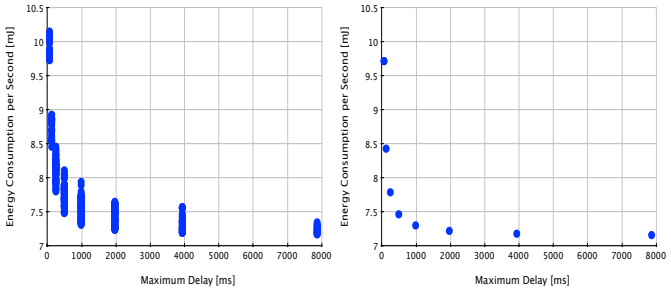

Fig. 2. Delay and energy consumption of the Pareto solutions found by the proposed approach (on the left), and the work in [12] (on the right)

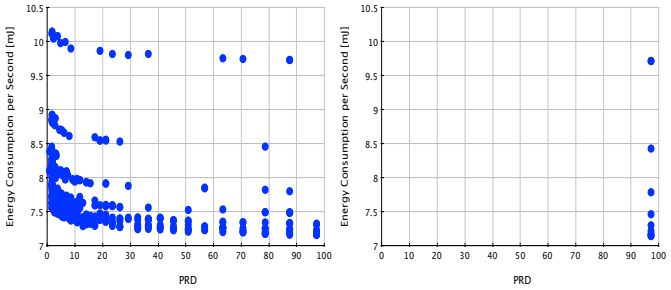

Fig. 3. PRD and energy consumption of the Pareto solutions found by the proposed approach (on the left), and the work in [12] (on the right)

cost function includes E_{node}, PRD, $Delay$, and PER. For the sake of illustration, we adopted a coarse discretization of the parameters and reduced the design space to 10^8 solutions, which can be explored by an exhaustive algorithm to provide a comparison between the MOSA and the real Pareto set.

The estimation provided by the proposed model proves to be effective as the error with respect to the experimental data is very low (i.e., it does not exceed the 1.9%). Moreover, results show that the proposed MOSA effectively explores the Pareto set, as the optimal solutions found by the MOSA perfectly match the ones found by the exhaustive algorithm, and cardinality of the Pareto set scales well with the number of executions. The solutions show a wide range of tradeoffs, e.g., the difference between the extreme values of E_{tx} exceeds 44%, the values of the PRD span from 0 to 93 (the maximum range is up to 100), and it is possible to achieve real-time transmission as well as packet latencies of tens of seconds.

Let us compare the proposed exploration technique with respect the other state-of-the-art approaches. The works designed for energy minimization, such as the one in [16], only produce a single solution that minimizes E_{node}, which is also found by the proposed MOSA. A more interesting comparison involves our approach and a multi-objective optimization of energy and delay, as proposed in [12]. Figure 2 shows that both approaches can identify the Pareto curve in terms of energy and delay. However, [12] shows some limitations when the application is considered, as it generates solutions with a high PRD (see Figure 3) and a high PER, and only finds 2.3% of the solutions found by the proposed approach. Finally, the executions of the MOSA and the approach in [12] take a few minutes, whereas the exhaustive search takes a few hours and does not scale well on larger solution spaces.

VI. Conclusion

In this work we have presented a model-based optimization framework for the design of WBSN nodes. We have analytically modeled the interdependencies between the most recurring components of a WBSN node, and their effects on the energy consumption. We have then proposed a multi-objective optimization algorithm to explore the optimal tradeoffs available in the design space. We have applied the proposed model to a case study, showing how it can effectively handle real standards, and how the optimization algorithm finds solutions that are consistent in terms of both energy and performance.

Acknowledgments

This research was partially supported by the Swiss National Science Foundation (SNF), under Grant 200021-127282.

References

[1] M. Patel and J. Wang, "Applications, challenges, and prospective in emerging body area networking technologies," *Wireless Communications, IEEE*, vol. 17, no. 1, pp. 80–88, Feb. 2010.
[2] S. Ullah et al., "A Comprehensive Survey of Wireless Body Area Networks," *Journal of Medical Systems*, pp. 1–30, 2010.
[3] H. Koc et al., "Minimizing Energy Consumption of Banked Memories Using Data Recomputation," in Proc. of *International Symposium on Low Power Electronics and Design*, 2006, pp. 358–361.
[4] V. Gutnik and A. P. Chandrakasan, "Embedded power supply for low-power DSP," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* , vol. 5, no. 4, pp. 425–435, 1997.
[5] C. Schurgers et al., "Power management for energy-aware communication systems," *ACM Trans. Embed. Comput. Syst.*, vol. 2, no. 3, pp. 431–447, Aug. 2003.
[6] L. S. Bai et al., "Automated construction of fast and accurate system-level models for wireless sensor networks," in Proc. of *Design, Automation Test in Europe Conference (DATE), 2011*, Mar. 2011, pp. 1–6.
[7] H. Mamaghanian et al., "Compressed Sensing for Real-Time Energy-Efficient ECG Compression on Wireless Body Sensor Nodes," *IEEE Transactions on Biomedical Engineering*, vol. PP, no. 99, p. 1, 2011.
[8] A. Burns et al., "SHIMMER - A Wireless Sensor Platform for Noninvasive Biomedical Research," *IEEE Sensors Journal*, vol. 10, no. 9, pp. 1527–1534, 2010.
[9] Z. He et al., "Energy-aware portable video communication system design for wildlife activity monitoring," *IEEE Circuits and Systems Magazine*, vol. 8, no. 2, pp. 25–37, 2008.
[10] A. Bonivento et al., "Platform based design for wireless sensor networks," *Mob. Netw. Appl.*, vol. 11, no. 4, pp. 469–485, Aug. 2006.
[11] S. Nabar et al., "Minimizing Energy Consumption in Body Sensor Networks via Convex Optimization," in Proc. of *International Conference on Body Sensor Networks*, 2010, pp. 62–67.
[12] G. S. A. Kumar et al., "End-to-End Energy Management in Networked Real-Time Embedded Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1498–1510, 2008.
[13] P. Suarez et al., "Increasing ZigBee network lifetime with X-MAC," in Proc. of *Real-world wireless sensor networks*, 2008, pp. 26–30.
[14] Y. Liu et al., "Static worst-case energy and lifetime estimation of wireless sensor networks," in Proc. of *Performance Computing and Communications Conference*, 2009, pp. 17–24.
[15] Casilari et al., "Modeling of Current Consumption in 802.15.4/ZigBee Sensor Motes," *Sensors*, vol. 10, no. 6, pp. 5443–5468, Jun. 2010.
[16] C. Yeh et al., "Energy-Aware Data Acquisition in Wireless Sensor Networks," in Proc. of *IMTC*, 2007, pp. 1–6.
[17] T. D. Burd and R. W. Brodersen, "Energy efficient CMOS microprocessor design," in Proc. of *Twenty-Eighth Hawaii International Conference on System Sciences*, vol. 1, Jan. 1995, pp. 288 –297 vol.1.
[18] S. Kirkpatrick et al., "Optimization by Simulated Annealing," *Science*, vol. 220, pp. 671–680, 1983.
[19] D. Nam and C. H. Park, "Multiobjective Simulated Annealing: A Comparative Study to Evolutionary Algorithms," *International Journal of Fuzzy Systems*, vol. 2, no. 2, pp. 87–97, 2000.
[20] "The FreeRTOS Project," http://www.freertos.org
[21] IEEE, *IEEE Std 802.15.4-2006*, 2006.