Project 1. Optimization of energy consumption and end to end delay in a wireless sensor network using duty-cycle MAC protocols.

Jose M. Barcelo Ordinas

Universidad Politècnica de Catalunya (UPC-BarcelonaTECH),
Computer Architecture Dept.
joseb@ac.upc.edu

March 11, 2021

1 Project 1. duty-cycle MAC protocols - Energy conservation

We will consider deploying a sensor network with wireless technology. In this deployment each sensor samples a physical phenomenon such as NO_2, O_3, or any other environmental phenomenon and has to be sent to a gateway that is connected to the Internet. The energy consumption to reach the gateway, also called sink, in one hop is very high. Therefore, the sensor node will use a network of multi-hop sensors that retransmit the packets until they reach the gateway. We can consider several network topologies such as a random deployment of sensors or a more organized topology such as a grid topology.

With the topology fixed, we assume that a routing protocol has determined the path from each node to the sink. The first thing we have to solve is the traffic model we're going to get from the topology. The traffic model consists of determining on average how many packets each sensor node is going to transmit. We must take into account that each node samples the environment with a frequency F_s, and therefore generates a packet every F_s seconds. Besides the own traffic, each node has to retransmit packets from other nodes, since we have a multi-hop topology.

Each node

The Media Access Protocol (MAC) will set when we have to transmit packets. This protocol follows a duty-cycle type architecture, which consists of being in a sleep state, until it is your turn to transmit a packet, either because you have sampled and it is your turn to send a packet of your own, or because a node wants to retransmit in the chain of nodes until it reaches the sink. While a node is in a sleep state, it consumes very little power. When it wants to transmit, the MAC can use three states: a carrier sense state in which it is listening for packets or signals from other nodes, a receiving state in which it receives packets from other nodes, or a transmission state in which it sends packets to nodes which are closer to the sink. These three states consume at least ten times more energy than the sleep state, so it is important that the node is in this state as long as possible. On the other hand, while in this state, or you can monitor the medium to see if there are any nodes that want to use it as a forwarding node. So the longer you are in this sleep state, the more delay you will cause to a packet.

The objective of this project is to optimize a duty-cycle MAC protocol in order to minimize energy consumption and end to end (e2e) delay. The traffic model for a several set of MAC protocols can be found in reference [4], section 3.2. For the project, I have chosen one of the MAC protocols explained in [4] called X-MAC, [2], but you are free to choose another one.

Maximizing the time in sleep state ↓↓ energy consumption
↑↑ Delay in the packets

Objective of the project: minimize consumption and E2E delay

The first part of the project is to minimize energy consumption or minimize delay. For that it is necessary to learn the dynamic range at which the sleep time works, i.e., draw the Energy-Delay plot and choose a range. Secondly, you can minimize the energy consumption or the delay by finding the optimal interval used in the sleep state. Finally, you can try to find a trade-off point of working between energy and delay according to for example the Nash Bargaining model. You may use paper [3] as reference paper where we analyzed several MAC protocols and we optimize the energy consumption. In this paper, the process is evaluated for B-MAC, X-MAC, RI-MAC, SMAC, DMAC and LMAC, all of them forming MAC's for WSN (Wireless Sensor Networks). In the following, I summarize some of the concepts of these papers in order to do the project, but I recommend you to go directly to these papers to have a more detailed description.

~ To Do

1.1 Network Model

Let us consider an unsaturated network with low traffic, which is typical of WSN applications. For a sake of simplicity, a random topology is adopted following the same analysis as in [4], although our modeling can be easily adapted to other network topologies, e.g. grid topology. A spanning tree is constructed, where nodes are static and maintain a unique path to the sink and use the shortest path routing with a maximum length of D hops; the depth or number of rings of the tree. We assume a network with size of N nodes, a uniform node density on the plane and a unit disk graph communication model. There are C+1 nodes, in average, on the unit disk. Hence, all nodes are in communication range with an average number of neighbors, C, except the leaf nodes. The nodes are layered into levels according to their distance to the sink in terms of minimal hop count, d= 0, . . . , D, where d=0 is reserved for the sink.

Let us consider periodic traffic generation. That is to say, every source node generates packets with frequency F_s. That means that a each node generate a packet every T_s=1/F_s seconds. According to this, we will use in our modeling the same input F_I^d, output F_{out}^d, background F_B^d traffic and input links I^d equations for every node as derived in [4]. The different symbols introduced in the analysis related to network and traffic model are summarized in Table 1 with typical values. The neighboring nodes, then, can be classified as the set of children (input) nodes I and the set of overheard (background) nodes, B, such that, $C = |I| + |B|$.

Assuming a random (ring) topology, let us define the following parameters; N_d is the average number of nodes in ring d:

$$N_d = \begin{cases} 1 & \text{if d=0,} \\ Cd^2 - C(d-1)^2 = (2d-1)C & \text{otherwise.} \end{cases} \tag{1}$$

The average number of input links at a node at level d is:

$$I_d = \begin{cases} 0 & \text{if d=D,} \\ C & \text{if d=0,} \\ \frac{N_{d+1}}{N_d} = \frac{2d+1}{2d-1} & \text{otherwise.} \end{cases} \tag{2}$$

Assuming a sampling rate of F_s, the output frequency defined as the number of packets that leaves a node at level d can be calculated as a recursion as:

$$F_{out}^d = \begin{cases} F_s & \text{if d=D,} \\ F_I^d + F_s = I_d F_{out}^{d+1} + F_s = F_s \frac{(D^2 - d^2 + 2d - 1)}{(2d-1)} & \text{otherwise.} \end{cases} \tag{3}$$

As you can note, nodes at the same ring d produce in average the same number of packets, and this number of packets depends of the level d at which the node is placed and the sampling frequency. Nodes

at outer rings, e.g. near ring D, produce less packets than nodes near ring d=1. For example, a node at level D produces F^d_{out}=F_s packets in average, while a node at level d=1 produces D^2*F_s packets in average. That means that nodes near the sink will consume faster their batteries than nodes at other levels since they have to be in the transmitting state more time than nodes in other levels.

Since the input traffic (packets that a node has to retransmit from other nodes), F^d_I, is related to the output traffic, it is easy to obtain it as:

$$F^d_I = \begin{cases} F_s D^2 C & \text{if d=0,} \\ I_d F^{d+1}_{out} = F_s \frac{(D^2-d^2)}{(2d-1)} & \text{otherwise.} \end{cases} \quad (4)$$

Finally, the aggregated background traffic, F^d_B, is defined as traffic from other nodes that interfere with the transmission of a given node. This background traffic has an impact in wireless communications since the number of packets to be transmitted by a node depends on the neighboring nodes. Then $F^d_B = B^d F^d_{out} = (C - I_d) F^d_{out}$, where B^d is the average number of background nodes.

Table 1 summarizes the different parameters for a random topology, the values of typical radio chipsets (CC2420 radio chipset), and typical values in the X-MAC wireless sensor MAC protocol.

(a)

CC2420 Radio	**Parameter Description**	**Values**
R	Rate [kbyte/s]	31.25
T_{cs}	Time [ms] to turn the radio on and probe the channel (carrier sense)	2.60
T_{up}	Time [ms] to turn the radio on into RX or TX	2.40
L_{pbl}	Packet preamble length [byte]	4
Traffic & Network	**Parameter Description**	**Values**
P	data payload [byte]	32
F_s	Sampling rate [pkt/node/min]	F^*_s
F^d_I	Node's Input Traffic Frequency at level d	$F^d_{out} - F_s$=$F_s \frac{D^2-d^2}{2d-1}$
F^d_{out}	Node's Output Traffic Frequency at level d	$F_s \frac{D^2-d^2+2d-1}{2d-1}$
F^d_B	Background Node's Traffic Frequency at level d	$\|B^d\|F^d_{out} = C - \|I_d\|F^d_{out}$
N	Network Size (number of nodes) [#nodes]	200-512
D	Network Depth [#levels]	5-8
C	Network Density (Connectivity) [#neighbors]	4-8

(b)

MAC	**Parameter & Description**	**Values**
X-MAC	T_w X-MAC wake-up period [ms]	T^*_w
	T_{al} Acknowledgement listen period [ms]	0.95
	T_{ps} Strobe preamble duration [ms]	$\frac{5+L_{pbl}}{R}$
	T_{cw} Contention window size [ms]	15 * 0.62
	T_{hdr}, T_{ack} pkt header & Ack duration [ms]	$\frac{9+L_{pbl}}{R}$

Table 1: (a) CC2420 Radio Constants [1], Network and Traffic Model with Typical Parameter Values. (b) X-MAC, DMAC, and LMAC Symbols used in Energy & Delay Equations

1.2 XMAC: a duty-cycle MAC protocol

X-MAC [2] is an asynchronous preamble sampling based protocol where nodes wake up periodically every T_w seconds to perform carrier sensing for, $T_{cs} + T_{al}$, (1) as depicted in Fig. 1. To send a packet, a node first contends to access the channel within the contention window T_{cw}, and it transmits a sequence

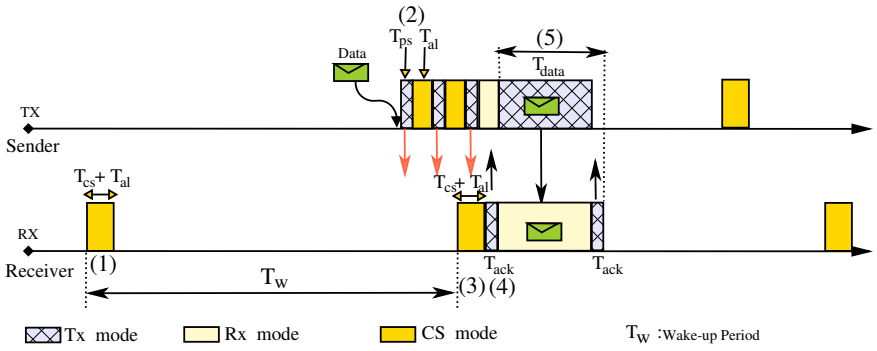

Figure 1: X-MAC's carrier sensing, transmission, and receiving modes.

of strobe preambles of duration T_{ps}, which are short packets containing the identifier of the receiver. It then listens to an acknowledgment for T_{al} (2). Strobes continue for a period sufficient to make at least one strobe overlap with a receiver wake-up (3). The receiver replies with an acknowledgment of duration T_{ack} (4) and keeps the radio on. After that, the sender transmits the data packet, T_{data}, which spans for the transmission of the header and the payload (5). The main adjustable parameter that affects the energy and delay performance is mainly the wake-up period, T_w, and hence the vector parameter for X-MAC protocol is given by X_{XMAC}=$[T_w]$. The per-node energy consumption based on the protocol operation modes, the e2e packet delay, and the bottleneck constraint are given in the following: a) The Energy of node n:

$$E^n = E^n_{cs} + E^n_{tx} + E^n_{rx} + E^n_{ovr} \tag{5}$$

where:

E^n_{cs}=$\frac{(T_{cs}+T_{al})}{T_w}$,
E^n_{tx}=$(T_{cs}$+T_{al}+$T_{tx})\, F^n_{\text{out}}$,
E^n_{rx} = $(\frac{3}{2}T_{ps}$+ T_{ack}+$T_{\text{data}})\, F^n_I$,
E^n_{ovr}=$(\frac{3}{2}\frac{T_{\text{tx}}}{T_w}T_{ps})\, F^n_B$

with

T_{tx}=$\lceil \frac{T_w}{T_{ps}+T_{al}} \rceil \frac{T_{ps}+T_{al}}{2}$+$T_{\text{ack}}$+$T_{\text{data}}$.

b) The delay of node n at level d^n:

$$L^n_{d^n} = \sum_{i=1}^{d^n} \left(\frac{T_w}{2} + \frac{T_{cw}}{2} + T_{\text{data}} \right) \tag{6}$$

where:

T_{data}=T_{hdr}+P/R+T_{ack}.

c) The bottleneck constraint:

$$|I^0|\, E^1_{tx} < 1/4 \tag{7}$$

Table 2: Eenrgy consumption and e2e delay formulas for XMAC.

Network Energy consumption	**End-to-End Delay**
$E^{\text{XMAC}} = \max_{n \in N}(\frac{\alpha_1}{T_w} + \alpha_2 T_w + \alpha_3) = \frac{\alpha_1}{T_w} + \alpha_2 T_w + \alpha_3$	$L^{\text{XMAC}} = \max_{n \in N}(\beta_1 T_w + \beta_2) = \beta_1 T_w + \beta_2$
$\alpha_1 = T_{cs} + T_{al} + \frac{3}{2} T_{ps}(\frac{T_{ps} + T_{al}}{2} + T_{ack} + T_{data}) F_B^{d^n}$	$\beta_1 = \sum_{i=1}^{d^n} 1/2$
$\alpha_2 = \frac{F_{out}^{d^n}}{2}$	
$\alpha_3 = (\frac{T_{ps}+T_{al}}{2} + T_{cs} + T_{al} + T_{ack} + T_{data}) F_{out}^{d^n}$ $+ (\frac{3}{2} T_{ps} + T_{ack} + T_{data}) F_I^{d^n} + \frac{3}{4} T_{ps} F_B^{d^n}$	$\beta_2 = \sum_{i=1}^{d^n} (\frac{T_{cw}}{2} + T_{data})$

The energy and e2e delay formulas for any node n belonging to a level d are summarized in Table 2, where the supscript d^n express level d of node n. We can observe that the formulas depend on constants α for the energy and β for the delay which in turn are dependent on the frequencies F_{out}, F_{in} and F_B. These frequencies are defined for each level. That means that the energy and delay formulas are the same for nodes at the same level but different for nodes of different levels.

We can also note that in the energy and delay formulas we consider the worst case for each of them. In the case of energy the worst case is to consider the node that produces the most energy consumption, i.e., maximize on n∈N. That means that as the nodes that consume more energy are in the level d=1, it is to say, with n such n is in d=1. Then, we can eliminate the maximize over n∈N, and restrict the equations with α_1, α_2 and α_3 parameters to the cases $F_{out}^{d=1}$, $F_{in}^{d=1}$ and $F_B^{d=1}$.

In the case of the delay the worst case is produced in those nodes that are in the most distant node, for what we will only consider the nodes n that are in the level d=D. Then, we can eliminate the maximize over n∈N, and restrict the equations with β_1 and β_2 parameters to the cases $F_{out}^{d=D}$, $F_{in}^{d=D}$ and $F_B^{d=D}$.

1.3 Project Realization

To carry out the project we will follow the following steps:

1. Use the python header provided with the typical values of a random topology with D levels. Set the parameters N, C, and D, and for that topology, draw the energy consumption E, and delay, L, curve for several sampling rates F_s. For example, set the sampling rate for some value such as 1 packet every 15 minutes and obtain the values of E and L for values of T_w in the interval [100, 500] ms. Then plot E against L and see how this plot is. Repeat the process for several values of F_s such as 1 packet every 1, 5, 10, 15, 20, 25 and 30 minutes. Observe that you have to use formulas E=$\frac{\alpha_1}{T_w} + \alpha_2 T_w + \alpha_3$ with $F_{out}^{d=1}$, $F_{in}^{d=1}$ and $F_B^{d=1}$ and L=$\beta_1 T_w + \beta_2$ with $F_{out}^{d=D}$, $F_{in}^{d=D}$ and $F_B^{d=D}$. You can plot:

   a) the energy as a function of Tw, where Tw∈[Tw^{min}, Tw^{max}] = [100, 500] ms,
   b) the delay as a function of Tw, and
   c) plot the curve E-L.

   For checking that you are in the correct way, I give you the values that I obtained for F_s = 1./(5*60*1000) = 3.3333e-06 (1 packet every 5 minutes, where T_s is given in milliseconds and therefore we have to divide by 1000), C=5, D=8 and N=C*D^2=320 nodes: α_1=3.5505, α_2=1.0667e-04 and α_3= 0.001911, β_1= 4 and β_2=52.048.

2. Using a solver to obtain the minimum energy consumption (Optimization Problem 1) and delay (Optimization Problem 2) for several values of L_{max} and E_{budget}. Draw plots showing the behaviour of parameters L_{max} and E_{budget} in energy consumption and delay optimal values. You

can vary $L_{max} \in [100, 5000]$ ms and $E_{budget} \in [0.5, 5]$ Joules. Use paper [3] as reference if necessary.

$$
\begin{array}{lll}
(P1) & Minimize & E^{XMAC}(T_w) \\
 & s.t. & L^{XMAC}(T_w) \leqslant L_{max} \\
 & & T_w \geqslant T_w^{min} \\
 & & |I^0|\, E_{tx}^1 \leqslant 1/4 \\
 & Var. & T_w
\end{array}
\tag{8}
$$

$$
\begin{array}{lll}
(P2) & Minimize & L^{XMAC}(T_w) \\
 & s.t. & E^{XMAC}(T_w) \leqslant E_{budget} \\
 & & T_w \geqslant T_w^{min} \\
 & & |I^0|\, E_{tx}^1 \leqslant 1/4 \\
 & Var. & T_w
\end{array}
\tag{9}
$$

3. In game Theory, a Bargaining scheme is an arbitration scheme in which a game is specified. In this game, players threat their opponents to play their best strategies, so, the opponents will get their worst pay-off. The arbitration scheme, then, says that players can collaborate to get a better pay-off than a worst one. The idea is that if a player threats another with playing his best (and then the opponent will get his worst), **all** players will get their worst pay-off. However, if they collaborate, even, since not all can get a best pay-off, at least, all will get something better than the worst. The Nash Bargaining Scheme (NBS) is specified as an optimization problem where if there are N players, each with $\mathrm{u}_{i,worst}$ pay-off, then:

$$
\begin{array}{lll}
\text{(NBS)} & \max & \prod_i (u_{\text{i,worst}} - u_i) \\
 & \text{s. t.} & (u_i) \geq u_{\text{i,worst}} \\
 & & u_i \in S \quad \forall i \\
 & \text{var.} & u_i \quad \forall i
\end{array}
$$

Using the Nash Bargaining scheme, find the trade-off between energy conservation and latency. The problem is expressed as:

$$
\begin{array}{lll}
\text{(NBS)} & \max & (E_{\text{worst}} - E(T_w))(L_{\text{worst}} - L(T_w)) \\
 & \text{s. t.} & (E_{\text{budget}}, L_{\text{max}}) \geq E(T_w), L(T_w) \\
 & & (E_{\text{worst}}, L_{\text{worst}}) \geq E(T_w), L(T_w) \\
 & & E(T_w), L(T_w) \in S \\
 & \text{var.} & T_w
\end{array}
$$

where the $E(T_w)$,$L(T_w)$∈S means that we have to include the MAC protocol intrinsic conditions and where the point $(E_{\text{worst}}, L_{\text{worst}})$ is the threat point. The problem (NBS) is non-linear non-convex, but this kind of problems can be transformed into a standard convex optimization problem without changing its solution. The idea is to define auxiliary variables E_1 and L_1 such that

$E_1 \geq E(T_w)$ and $L_1 \geq L(T_w)$, which should be satisfied by the optimal solution. Whenever the problem (NBS) is feasible, $E(T_w) \leq E_{\text{worst}}$, $L(T_w) \leq L_{\text{worst}}$, and application of (NBS) to the MAC protocols yields a concave problem.

$$
\begin{array}{lll}
\text{(NBS*)} & \max & log(E_{\text{worst}} - E_1) + log(L_{\text{worst}} - L_1) \\
& \text{s. t.} & (E_{\text{worst}}, L_{\text{worst}}) \geqslant (E(T_w), L(T_w)) \\
& & (E_1, L_1) \geqslant (E(T_w), L(T_w)) \\
& & (E_1, L_1) \leq (E_{\text{budget}}, L_{\text{max}}) \\
& & (E_1, L_1) \in S \\
& \text{var.} & E_1, L_1, T_w
\end{array}
$$

Consequently, the equivalent concave problem for XMAC is:

$$
\begin{array}{lll}
\text{(NBS-XMAC*)} & \max & log(E_{\text{worst}}^{\text{XMAC}} - E_1) + log(L_{\text{worst}}^{\text{XMAC}} - L_1) \\
& \text{s. t.} & E_{\text{worst}}^{\text{XMAC}} \geqslant E^{\text{XMAC}}(T_w) \\
& & E_1 \geqslant E^{\text{XMAC}}(T_w) \\
& & L_{\text{worst}}^{\text{XMAC}} \geqslant L^{\text{XMAC}}(T_w) \\
& & L_1 \geqslant L^{\text{XMAC}}(T_w) \\
& & T_w \geqslant T_w^{\min} \\
& & |I^0|\, E_{tx}^1 \leqslant 1/4 \\
& \text{var.} & E_1, L_1, T_w
\end{array}
$$

1.4 Solvers

In python, for solving optimization problems you can use the scipy python module. However, Optimization problems P1 and P2 are Non-convex, but are Geometric Programming (GP). Remember that GP can be transformed in a convex problem using logarithms. Instead of doing yourself the transformation and using scipy, you can use gpkit that is a library for GP, http://gpkit.readthedocs.io/en/latest/gp101.html.

In general, gpkit uses mosek (license) or cvxopt (free) as solvers. Then, follow the guide installation for using gpkit and the examples given in the gpkit URL. Better use cvxopt as solver since it is free. In any case, since this problem is simple, in general, scipy is able to provide good results.

The Game Theory part (optimization problem NBS) also is non convex. However, the log-transform problem NBS* is a concave problem (remember that maximum log(x) is the same than minimum -log(x)). Even if it is not convex, the range of working is convex and you can use the scipy module for solving this part.

References

[1] *CC2420 Single-Chip 2.4 GHz RF Transceiver, Texas Instruments*, March 2010.

[2] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *ACM SenSys*, pages 307–320, 2006.

[3] M. Doudou, J. M. Barcelo-Ordinas, D. Djenouri, J. Garcia-Vidal, A. Bouabdallah, and N. Badache. Game theory framework for mac parameter optimization in energy-delay constrained sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 12(2):10, 2016.

[4] K. Langendoen and A. Meier. Analyzing mac protocols for low data-rate applications. *ACM Transactions on Sensor Networks TOSN*, 7(2), 2010.