14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

HYPOTHESIS TESTING OVER A RANDOM ACCESS CHANNEL IN WIRELESS SENSOR NETWORKS

Elvis Bottega[*,†], *Petar Popovski*[*], *Michele Zorzi*[†], *Hiroyuki Yomo*[*], *and Ramjee Prasad*[*]

∗Center for TeleInFrastructure (CTIF), Aalborg University
Niels Jernes Vej 12, DK-9220 Aalborg, Denmark
e–mail: {bottega, petarp, yomo, prasad}@kom.aau.dk
† University of Padova
e–mail: zorzi@ing.unife.it

ABSTRACT

In the design of the communication protocols for wireless sensor networks a specific requirement emerges from the fact that the data contained in an individual sensor is not important *per se*, but its significance is instantiated with respect to the contribution to the overall sensing task and the decision fusion. Therefore, the communication protocols should be application–aware and operate by reckoning the utility of the carried data. In this paper we consider the problem of hypothesis testing at the fusion center (sink) when all the sensors communicate with the sink via a random access channel. Each sensor contains a binary information 0 (event occurred) or 1 (event did not occur). In a traditional protocol design, an existing random–access protocol is used by which the sink collects the data from *all* sensors and subsequently makes the decision through majority voting over the received data. In this paper, we propose approaches for joint design of the communication and the decision fusion for the application of hypothesis testing. The fusion center terminates the data gathering through the random access channel as soon as it can make sufficiently reliable decision based on the data received so far. We describe two instances of the protocols, where the total number of sensors N is known and not known, respectively. Our results show that the proposed approaches provide optimized performance in terms of time, energy and reliability.

1. INTRODUCTION

The introduction of wireless communication capability in the realm of the sensing technologies has initiated novel perspectives for design and analysis of distributed sensing systems. A wireless sensor network can be defined as a system of sensing nodes that are using wireless communication to achieve a synergetic sensing task. This implies that the protocols for wireless sensor networks (WSNs) should be designed in an *application–centric* or *task–centric* manner instead of an *architecture–centric* manner, such as the IP–based networks or networking protocols based on the OSI–layering.

Let us consider for example a set of sensors deployed to detect occurrence of an event in a certain area. The end user of the sensory data is interested in the answer to the question "Has event X occurred?". The node that collects the sensory data and answers the query of interest is referred to as *data sink* or *fusion center (FC)*. The individual data collected at a sensor is not important *per se*, but to the extent to which it contributes to the answer to the aforementioned question. For example, if there are 100 deployed sensors, 98 of which detected the event, then the remaining 2 sensors do not need to send their data "no event" to the fusion center, since it will not affect the overall decision. In general, the communication protocol in a WSN should be designed in concert with the data fusion process, thus conforming to the application–centric networking design paradigm.

In this paper we consider a single–hop wireless sensor network. We consider a scenario that consists of a population of battery–operated failure prone sensors, randomly deployed in a field of interest, an unlimited energy supplied fusion center and a common shared channel where the up–link communications follow a random access scheme and the down–link transmissions are performed in a broadcast manner. Initially, FC or the interrogator[1] is probing the sensor field by broadcasting an inquiry in order to solicit replies from sensors which have detected/sensed data with a certain property A. Let us denote by N_A the number of sensors that have such a property. If $N_A > 1$, the sensors need to run some conflict resolution algorithm in order to access the random access channel. A sensor node is resolved if it successfully transmits its message to the FC. Such conflict resolution is governed or facilitated by the FC by providing feedback that collision or successful transmission has been detected. The initial replies from the N_A sensors arrive simultaneously and produce an initial batch conflict of multiplicity N_A. After completely resolving the conflict and receiving all the N_A packets successfully, FC knows that exactly N_A packets have the property A. Let the property of interest be whether or not an event occurred. Then, each sensor has a binary decision to support one of the two possible hypotheses: $H_1 = \{\textit{Event } \mathscr{E} \textit{ occurred}\}$ and $H_0 = \{\textit{Event } \mathscr{E} \textit{ did not occurr}\} = \{\textit{Event } \overline{\mathscr{E}} \textit{ occurred}\}$. Among the N devices, N_1 support hypothesis H_1 and $N_0 = N - N_1$ support hypothesis H_0. The FC tests the hypothesis by counting N_0 and N_1 and deciding based on majority voting i. e. event occurred if $N_1 > N_0$. The straightforward implementation of this hypothesis testing is as follows. The FC can first resolve the batch conflict among the N_1 sensors and thus retrieve the value of N_1. Subsequently, FC can run another batch conflict resolution and retrieve N_0, then compare N_1 and N_0 and make the decision. Nevertheless, this conventional method may spend an excessively high number of messages/time slots, without a significant improvement of the decision reliability.

The framework for resolution of batch conflicts proposed in [6] allows to make estimation of the conflict multiplicity N_i while running the random access algorithm. This estimate is progressively correct as more nodes are resolved and is trivially correct when all nodes are resolved. Now let us suppose that in the hypothesis testing $N_1 \gg N_0$. Let the FC run the conflict resolution algorithms in a time–division manner, such that at some instant it has the estimates $\hat{N}_1$ and $\hat{N}_0$ and with high probability $\hat{N}_1 > \hat{N}_0$. In that way, the FC does not need to run the complete conflict resolution procedure, but only until it can make a satisfactorily reliable decision, e. g. the probability of no detection $P(\hat{N}_0 > \hat{N}_1 | N_0 \leq N_1) < \varepsilon$. With such a partial conflict resolution, the approximate decision fusion consumes less time and messages from the sensors and thereby contributes to the efficient usage of the limited energy supply of a sensor.

Using the framework for approximate counting over a random access channel from [6], in this paper we introduce methods for approximate hypothesis testing through majority voting. The FC terminates the data gathering through the random access channel as soon as it can make a sufficiently reliable decision based on the data received so far. We describe two protocol instances, where the total number of sensors N is known and not known, respectively.

[1] The terms interrogator and fusion center will be used interchangeably.

For each case, we infer a stopping criterion that satisfies certain reliability constraint for the correctness of the hypothesis testing.

1.1 Related Work

The general problem of sequential detection with a sensor field is described in [3]. Hypothesis testing for sensor fields has been considered in [1], in which the authors consider a network where devices make binary decisions (H_0 or H_1) corrupted by probabilities of missed detection and false alarm. An optimal decision rule is derived, assuming that both the numbers of sensors which support the two hypotheses H_0 and H_1 are known. In [2], a fusion rule is presented to solve the problem of target detection. Binary hypotheses are used and a model of signal power attenuation is considered for the transmissions from the target to the N devices (N is considered to be a random variable that follows a Poisson distribution). A related problem is considered in [4], where the authors propose an algorithm to solve the problem of the decision fusion for binary hypotheses, exploiting the shared, time–slotted communication link: besides, the multiplicity N of the sensors is supposed known, and the computations required to the FC to run the algorithm grow exponentially with its value. In [5] a sequential polling is performed in order to estimate the number of sensors that are in an operating state. The sensors are numbered from 1 to N and the number is used as a sensor identity. The FC is polling each sensor by using its identity, which avoids the usage of random access channel, but this method is inapplicable when the sensors do not have unique identity.

2. SYSTEM MODEL

We use the model of *slotted channel*. The transmissions of the terminals start at predefined, equally spaced instants. The duration between two neighboring instants is denoted as *slot*. When k terminals transmit in the same slot t, then the interrogator perceives the channel in slot t as:

- *Idle slot* (I) if $k = 0$ i.e. no terminal transmits.
- *Successful reception* (S) or *resolution* if $k = 1$ i.e. only one terminal transmits.
- *Collision* (C) if $k \geq 2$. The messages of the terminals interfere destructively at the FC and error is detected.

Prior to the next slot $(t+1)$, all nodes receive the feedback I, S, or C, indicating the state perceived by the FC in slot t. We neglect other sources of error besides collisions. For the channel state S, the feedback must be explicit i.e. an acknowledgement from the FC. For idle or collision, the feedback is implicit - if $k = 0$ no terminal expects feedback and if $k \geq 2$ the feedback is the absence of acknowledgement. The slot duration is enough to accommodate a packet sent by a terminal and the acknowledgement from the FC. In the model of slotted channel, the slot duration is the same regardless of whether the channel state is I, S, or C.

The terminals should generate random bits to be used in the conflict resolution process. It is assumed that the sensors are indistinguishable i. e. a sensor does not have unique ID that is known to the FC.

3. APPROXIMATE MAJORITY VOTING OVER A RANDOM ACCESS (RA) CHANNEL

3.1 Conflict Resolution with Approximate Counting

In order to introduce the framework for approximate counting from [6], we start by describing the well–known splitting–tree algorithms [7]. Fig. 1 depicts one possible evolution of the tree algorithm for initial conflict of multiplicity $N = 8$. The *level* of a tree node is the path length from that node to the root of the tree. Each tree node is uniquely associated with a string called *address*, related to the tossing outcomes of a terminal. For example, d_2 and d_3 belong to the tree node with address 01, enabled in slot 4. The root has address ε (empty string) and is at level 0. In fact, the initial conflict among the terminals follows the probe with address ε from the interrogator. Using the tree representation, we can say that an address is *enabled* in slot t if the terminals that belong to that node are allowed to transmit in t. In the basic variant, the tree nodes are enabled in a pre–order fashion.

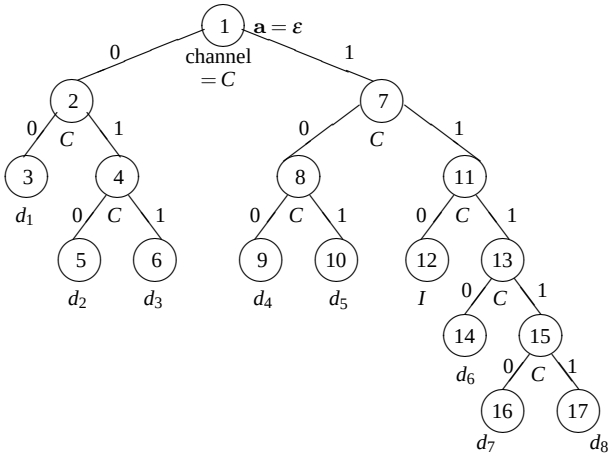


Figure 1: An instance of the binary tree algorithm for $N = 8$ terminals. The number denotes the slot in which a node is enabled. Below each node is the channel state in that slot. For channel state "single", d_i denotes the resolved sensor.

By representing the binary tree in an alternative framework, we can get an insight on how the conflict multiplicity can be estimated while the conflict is being resolved. Instead of tossing coins, let us assume that before the initial attempt to transmit, each terminal generates a random real number, referred to as token, uniformly distributed in the interval $[0,1)$. Now, instead of considering enabled addresses from the tree, we can equivalently consider enabled intervals. After the initial collision, the interval $[0, 0.5)$ is enabled in that sense that all nodes that have drawn tokens in that interval have the right to transmit in that slot. Fig. 2 shows how the instance of the algorithm on Fig. 1 can be represented through a sequence of enabled intervals. The interval $[0, p)$ is called resolved if all the sensor nodes that have their tokens in $[0, p)$ manage to transmit successfully to the interrogator. Let us assume that there are n tokens in the resolved interval $[0, p)$. Then the interrogator can estimate the unknown multiplicity N as $\hat{N} = \frac{n}{p}$. This fact has been used in [6] to speed up the conflict resolution by tuning the length of the enabled intervals to the estimated value $\hat{N}$. The fastest proposed version of such algorithms is the Interval Estimation Collision Resolution (IECR) algorithm. When the batch size $n \to \infty$, it is shown that the time efficiency of IECR reaches asymptotically the same time efficiency that the FCFS algorithm [7, pp. 289-304] achieves for Poisson instead of batch arrivals.

When applying the IECR (or any equivalent algorithm) for hypothesis testing by approximate counting of a set of N sensors, we need to find what is the reliability of such obtained estimate $\hat{N}$. The probability that $N = N_{est_p}$ given that exactly N_p^* sensors have tossed the tokens in $[0, p]$, can be modeled using a Bernoulli random variable n, with parameters p and N, where the latter is unknown:

$$P\{n = N_p^* \mid N = N_{est_p}, p\} = \mathscr{B}(N_p^*, N_{est_p}, p)\ . \tag{1}$$

For the conditional probabilities we can write:

$$P\{N{=}N_{est_p} | n{=}N_p^*, p\} = \frac{P\{n{=}N_p^* | N{=}N_{est_p}, p\} \cdot P\{N{=}N_{est_p} | p\}}{P\{n{=}N_p^* | p\}} \tag{2}$$

We can notice that if N_p^* devices tossed a uniformly distributed number smaller than p, there are $\frac{1}{p}$ values of N that make N_p^* the most probable outcome for n: for example, if $n = 8$ and $p = 0.25$, we

$$P[\text{error}] = P\left[\text{error} \mid N_0 \le \frac{t_{p_i}}{p_i}, N_1 < \frac{t_{p_i}}{p_i}\right] \cdot P\left[N_0 \le \frac{t_{p_i}}{p_i}, N_1 < \frac{t_{p_i}}{p_i}\right] + P\left[\text{error} \mid N_0 \le \frac{t_{p_i}}{p_i}, N_1 > \frac{t_{p_i}}{p_i}\right] \cdot P\left[N_0 \le \frac{t_{p_i}}{p_i}, N_1 > \frac{t_{p_i}}{p_i}\right] +$$
$$+P\left[\text{error} \mid \frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}, \frac{t_{p_i}}{p_i} \le N_1 < \frac{w_{q_i}}{q_i}\right] \cdot P\left[\frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}, \frac{t_{p_i}}{p_i} \le N_1 < \frac{w_{q_i}}{q_i}\right] + P\left[\text{error} \mid \frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}, N_1 < \frac{t_{p_i}}{p_i}\right] \cdot P\left[\frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}, N_1 < \frac{t_{p_i}}{p_i}\right] +$$
$$P\left[\text{error} \mid \frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}, N_1 > \frac{w_{q_i}}{q_i}\right] \cdot P\left[\frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}, N_1 > \frac{w_{q_i}}{q_i}\right] + P\left[\text{error} \mid N_0 > \frac{w_{q_i}}{q_i}\right] \cdot P\left[N_0 > \frac{w_{q_i}}{q_i}\right] \quad (5)$$

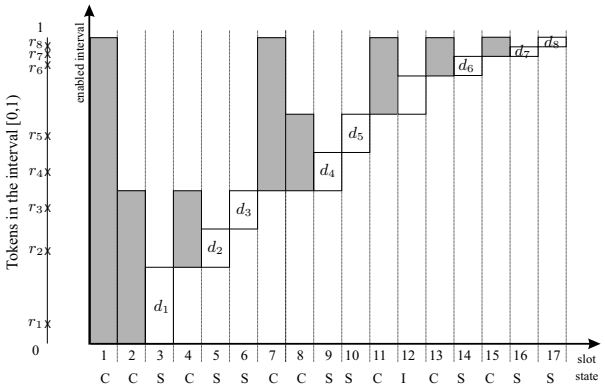


Figure 2: Representation of the example of binary tree from Fig. 1 by using tokens and sequence of enabled intervals.

have $\frac{1}{p} = 4$ values that make $n = 8$ the most probable outcome, i.e. $N = 30, 31, 32, 33$. If no knowledge of N is assumed, then the a priori probability for having each of those values is identical. Hence, using (2), the probability of N being equal to a particular value N_{est_p} can be approximated as:

$$P\{N{=}N_{est_p} \mid n{=}N_p^*, p\} = p \cdot \binom{N_{est_p}}{N_p^*} \cdot p^{N_p^*} \cdot (1-p)^{N_{est_p} - N_p^*} \quad (3)$$

3.1.1 *Majority Voting with Unknown N*

Recalling that the set of N sensors is divided into two subsets supporting respectively the binary hypotheses, the strategy we will pursue for the estimation of N_0 and N_1 relies on the IECR algorithm: a contention between two parallel runs of IECR is used to estimate both the unknown values, exploiting the TDMA technique. The slots are in fact alternatively used to resolve sensors of the N_1 set and of the N_0 set, so that we can make at each unit of time an error-prone decision, basing the choice on the value of the resolved nodes and the scanned intervals.

Considering at each slot of majority voting algorithm the values of the intervals (p_i and q_i) and devices (t_{p_i} and w_{q_i}) resolved for both N_0 and N_1 estimations, and using expression (3), we can easily calculate the probability of erroneous decision as follows: suppose for example we are in slot i, and the estimated value of N_1 is bigger than the one for N_0 ($\widehat{N_1} > \widehat{N_0}$). Referring to the previous results, the mass distributions of N_0 and N_1 are qualitatively shown on Fig. 3.

We recall here that the estimated values of N_1 and N_0 are given by the following:

$$\widehat{N_0}(ML) = \left\lfloor \frac{t_{p_i}}{p_i} \right\rfloor \quad \widehat{N_1}(ML) = \left\lfloor \frac{w_{q_i}}{q_i} \right\rfloor , \quad (4)$$

If we divide the set of possible values of N_0 into three subsets, namely $N_0 \le \frac{t_{p_i}}{p_i}$, $\frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}$ and $N_0 > \frac{w_{q_i}}{q_i}$, the probability of error can be written as in (5).

In slot i then we can calculate the probabilities:

$$\begin{aligned} P[N_0 = N_{0est_p}] \quad , \quad \forall N_{0est_p} \in \left[\frac{t_{p_i}}{p_i}, \frac{w_{q_i}}{q_i}\right] \\ P[N_1 = N_{1est_p}] \quad , \quad \forall N_{1est_p} \in \left[\frac{t_{p_i}}{p_i}, \frac{w_{q_i}}{q_i}\right] \end{aligned} \quad (6)$$

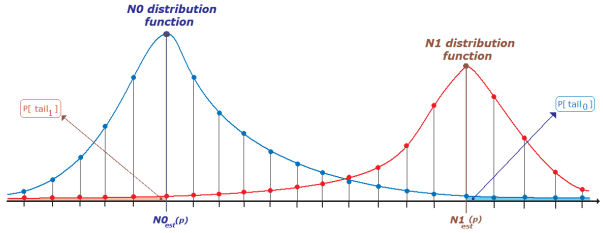


Figure 3: The a posteriori probability mass distribution for N_0 and N_1.

and use them to get the values of the four conditional probabilities of error of equation (3) as follows. First we define:

$$\mathcal{N}_0 = \sum_{N_0{}^* = \frac{t_{p_i}}{p_i}+1}^{\frac{w_{q_i}}{q_i}} P[N_0 = N_0{}^*] \quad , \quad \mathcal{N}_1 = \sum_{N_1{}^* = \frac{t_{p_i}}{p_i}}^{\frac{w_{q_i}}{q_i}-1} P[N_1 = N_1{}^*] \quad (7)$$

As shown in figure 3, we will call $P\left[N_1 < \frac{t_{p_i}}{p_i}\right]$ and $P\left[N_0 > \frac{w_{q_i}}{q_i}\right]$, respectively, $P[\text{tail}_1]$ and $P[\text{tail}_0]$, where:

$$P[\text{tail}_1] \le 1 - 2 * \mathcal{N}_1 - P[N_1 = \widehat{N_1}] \quad (8)$$
$$P[\text{tail}_0] \le 1 - 2 * \mathcal{N}_0 - P[N_0 = \widehat{N_0}] \quad (9)$$

As stated previously, the distributions of N_0 and N_1 are almost symmetrical around the values $\frac{t_{p_i}}{p_i}$ and $\frac{w_{q_i}}{q_i}$, at least when p is small.

Finally we observe that for each value $\frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}$, the error occurs if N_1 is smaller than N_0, and this happens with probability 1 if $N_1 < \frac{t_{p_i}}{p_i}$, whereas for $\frac{t_{p_i}}{p_i} \le N_1 < \frac{w_{q_i}}{q_i}$ the probability of error is given by:

$$P[\text{middle}] = P\left[\text{error} \mid \frac{t_{p_i}}{p_i} < N_0 \le \frac{w_{q_i}}{q_i}, \frac{t_{p_i}}{p_i} \le N_1 < \frac{w_{q_i}}{q_i}\right] = \sum_{N_1{}^* = \frac{t_{p_i}}{p_i}}^{\frac{w_{q_i}}{q_i}-1} \left(P[N_1 = N_1{}^*] \cdot \sum_{N_0{}^* = N_1{}^*+1}^{\frac{w_{q_i}}{q_i}} P[N_0 = N_0{}^*] \right) \quad (10)$$

Using derivations (7), (9), (8) and (10) in (5), and simplifying the terms equal to zero (all the cases in which we assume $N_0 < N_1$), we obtain the following:

$$\begin{aligned} P[\text{error}] \le\ & 1 \cdot P[\text{tail}_1] \cdot (1 - \mathcal{N}_1 - P[\text{tail}_0]) + \\ & + P[\text{middle}] \cdot \mathcal{N}_1 \cdot \mathcal{N}_0 + \\ & + 1 \cdot \mathcal{N}_0 \cdot P[\text{tail}_1] + 1 \cdot P[\text{tail}_0] \end{aligned} \quad (11)$$

We have in (11) the sign of inequality because we are overestimating $P[\text{tail}_1]$ and $P[\text{tail}_0]$ and the error probabilities related to them: in fact, when we consider $N_{0est_p} \in \text{tail}_1$ and $N_{1est_p} \in \text{tail}_0$, we upperbound the probability of error by 1.

3.2 Majority Voting with Known N

In this section we will consider the same scenario as the previous one, adding only one further hypothesis: we suppose to know the value N, that is the total number of sensors in the network. We expect to have better performance, since this information can only improve the probability of correct decision. We can in fact state now that the sum of estimated N_0 and estimated N_1 must be equal to N, otherwise we are facing an error, but the evaluation of the consequences of this observation on the estimation of the error probability requires a tactful approach.

We will again pursue the technique of sequential counting using the IECR algorithm to evaluate the value of N_0 and N_1: the primary target we have is then to find or to build a statistical model that describes the probability of correct decision under the following hypotheses:

- In interval $[0, p_i]$ t_{p_i} tokens are present;
- In interval $[0, q_i]$ w_{q_i} tokens are present;
- The total number of tokens is N, then we only need to have a precise knowledge of either N_1 or N_0 to estimate both of them;
- The value $\frac{N}{2}$ represents the critical value for the decision: if set $\mathscr{A}$ has multiplicity N^* larger than $\frac{N}{2}$ with high probability, consequently the other set $\mathscr{B}$ has multiplicity $N^{\triangle} = N - N^* < \frac{N}{2}$, and we can argue that $|\mathscr{A}| > |\mathscr{B}|$.

Given the above considerations, we can now develop the technique for the most probable correct decision: at any slot of time we compare $\widehat{N_0} = \frac{t_{p_i}}{p_i}$ and $\widehat{N_1} = \frac{w_{q_i}}{q_i}$ with the value $\frac{N}{2}$, and we proceed if and only if the two estimations reside in the two opposite halves of interval $[0, N]$. If the latter condition is encountered, then we compute for both sets the probabilities that the estimated value is in the right half using again expression (3). An example will clarify this procedure.

Suppose the following scenario: $N = 100$, $t_{p_i} = 3$, $\widehat{N_0} = 28$, $\widehat{N_1} = 75$ and the required maximum error is $p_{err}^{max} = 0.1$. In this case, the algorithm sets H_1 as the correct decision, and computes the probabilities that:

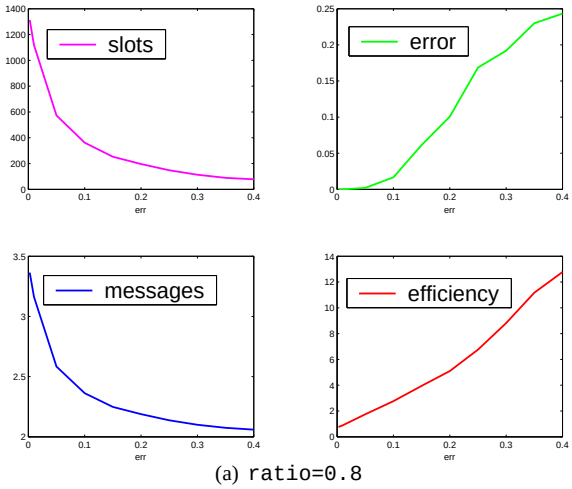
$$\sum_{N_{0est}=3}^{50} P[N_0 = N_{0est}] \quad , \quad \sum_{N_{1est}=50}^{97} P[N_1 = N_{1est}] \qquad (12)$$

The sums of expressions (12) represent respectively the probabilities that the two multiplicities are in the two halves, i.e. $P\{N_0 \in [3,50]\}$ and $P\{N_1 \in [50,97]\}$ (let's say that in our example they are 0.95 and 0.7). At this point, we have an upper-bound for the average probability of error we commit stating both that $N_0 \leq \frac{N}{2}$ and $N_1 \geq \frac{N}{2}$. If any of the two quantities is larger than $1 - p_{err}^{max} = 0.9$, the algorithm stops: concluding our example then, the decision is finally set for H_1, since $P\{N_0 \in [3,50]\} = 0.95 > 0.9$.

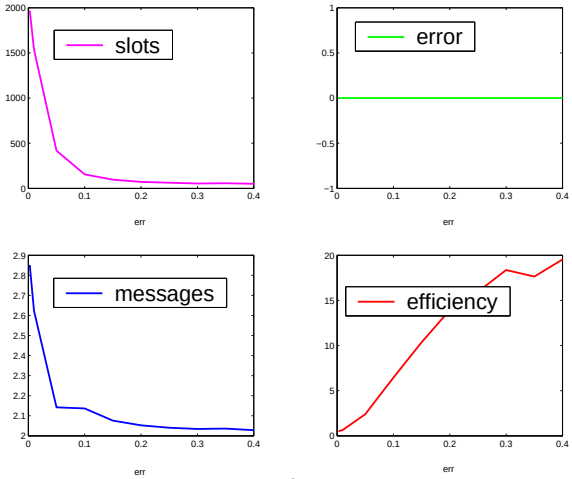
4. PERFORMANCE ANALYSIS

For the analysis of the introduced algorithms, we set the parameters as follows: $N = 1000$, runs $= 1500$, maximum error $= [1,5,10,15,20,25,30,35,40]$. Figures 4(a) and 4(b) show the performance of the algorithm with unknown N in two situations: $ratio = 0.8$ and $ratio = 0.1$, where $ratio = \frac{N_0}{N_1}$. We plot different parameters versus the maximal allowed error probability: the average number of slots used to run the algorithm, the mean error of the decision, the mean number of messages sent per each sensor and finally the time efficiency, defined as the number of slots used to run the algorithm divided by the number of sensors.

We can clearly see that for both values of *ratio* we can have an improvement for the time complexity and the message complexity if we accept to set an interval tolerance in the error probability. The two figures show that as the value of maximum allowed error grows, the curves representing the messages sent decrease, and the ones representing the time efficiencies grow.



(a) `ratio=0.8`



(b) `ratio=0.1`

Figure 4: Optimal Bound: results with unknown N.

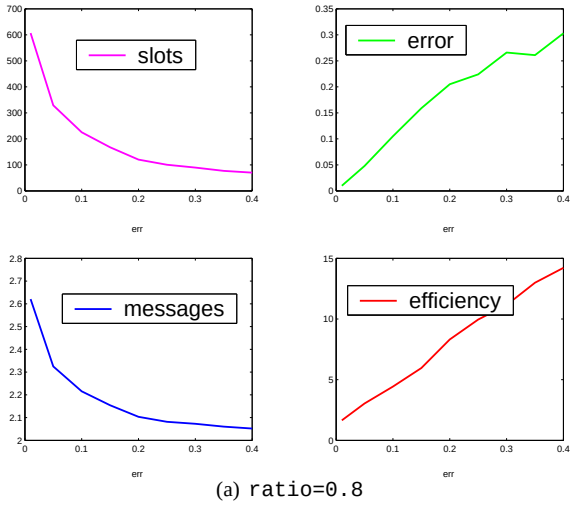
We can also notice in Fig. 4(b) that the error is always equal to 0: we know that one set of sensors is much smaller than the other, hence the algorithm counts all the sensors of that set before having the first feedback from the other set. Since the total N is unknown, the information related to the smaller set is then not useful until estimations of the second set are performed. The previous observations give reason also to the fact that in Fig. 4(b) the number of slots used by the algorithm approaches an asymptote different from 0.

For known N, the results are shown in figures 5(a) and 5(b), respectively for $ratio = 0.8$ and when $ratio = 0.1$.

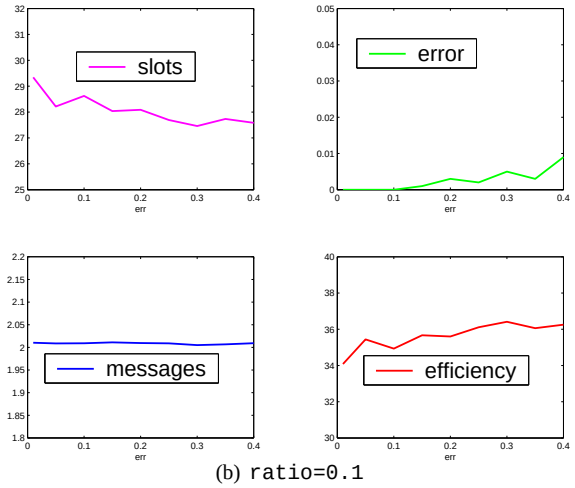
Here again we notice that the time complexity and the message complexity have a concrete improvement as soon we accept a controlled probability of error. What is more relevant is that we accomplished an improvement as compared with the previous results of figures 4(a) and 4(b): the number of messages sent when N is known is always smaller than in the case of unknown N, and the number of slots used to run the algorithm is larger when no information about N is given.

5. CONCLUSIONS

Wireless sensor networks (WSNs) require optimizations of the networking protocols according to the application i. e. the task that the WSN is supposed to fulfill. In this paper we have considered the problem of hypothesis testing for single–hop wireless sensor networks in which the sensors use a random–access channel to transmit information to the fusion center (FC). There is a set of N sensors,



(a) `ratio=0.8`



(b) `ratio=0.1`

Figure 5: Optimal Bound: results with known N.

N_1 of them detect that an event has occurred and $N_0 = N - N_1$ have detected the absence of that event. The fusion center needs to know N_0 and N_1 and uses majority voting to decide whether or not the event has occurred (i. e. it has occurred if $N_1 > N_0$). In a conventional approach, the FC runs a random access protocol to collect the data and find the exact values of N_0 and N_1 and then makes the decision. Here we have proposed methods for decision fusion based on approximate counting of N_0 and N_1. The FC runs two instances of the random access protocol, for the N_0 and N_1 nodes, respectively. The protocol utilizes the framework introduced in [6], which makes it possible to obtain estimates of N_0 and N_1 while resolving the conflicts. The precision of the obtained estimates is proportional to the resources (time and messages) used in resolving the conflict. Hence, by making a decision based on the estimates of N_0 and N_1, the FC can trade–off the reliability of the decision fusion with the energy that the sensor nodes invest in resolving the conflict.

As a future work, we need to consider more general scenarios for decision fusion over a random access channel. For example, an interesting question is how FC can approximately compute a function $f(x_1, x_2, \ldots x_N)$ over the random access channel, where x_n denotes the reading of $n-$th sensor. Furthermore, we need to investigate how the channel impairments affect the design of such fusion–aware networking protocols.

REFERENCES

[1] Z. Chair and P. Varshney ,“Optimal Data Fusion in Multiple Sensor Detection System,” *IEEE Trans Aerospace Electron Sys*, Vol.22, pp.98-101, January 1986.

[2] R. Niu and P. K. Varshney ,“Decision Fusion in a Wireless Sensor Network with a Random Number of Sensors,” *ICASSP IEEE*, 2005.

[3] R. Viswanathan and P. K. Varshney ,“Distributed Detection with Multiple Sensors: Part I—Fundamentals & Part II—Advanced Topics,” *Proceedings of the IEEE*, Vol.85, no.1, January 1997.

[4] Y. Yuan and M. Kam ,“Distributed Decision Fusion With a Random-Access Channel for Sensor Network Applications,” *IEEE Transactions on instrumentation and measurement*, Vol.53, no.4, August 2004.

[5] A. Leshem and L. Tong ,“Estimating Sensor Population via Probabilistic Sequential Polling,” *IEEE Signal Processing Letters*, Vol.12, no.5, May 2005.

[6] P. Popovski, F. Fitzek and R. Prasad ,“A Class of Algorithms for Batch Conflict Resolution with Multiplicity Estimation,” *Algorithmica, Special issue on Principles of Mobile Computing, Springer-Verlag*, accepted, 2005.

[7] D. P. Bertsekas, R. G. Gallagher ,“Data Networks,” *Prentice Hall, Second Edition*, 1992.