Estimation of the Number of Operating Sensors in a Sensor Network

Cristian Budianu and Lang Tong[†]
School of Electrical and Computer Engineering
Cornell University
Ithaca, NY 14853
Email: {cris, ltong}@ece.cornell.edu

Abstract—**This paper investigates the estimation of the number of operating sensors in a wireless sensor network. The basic model considered is equivalent to an urn model with replacement. For this model, we propose an estimator based on the Good-Turing non-parametric estimator of the missing mass. We show how this estimator can be applied to other related problems like estimation of class histograms. It is also shown that the estimator proposed is robust to model changes; more exactly its performance degradation is small when it is applied to a batch sampling model that models a receiver with multiple packet reception (MPR) capabilities. A modified estimation method that takes into account the batch sampling model is given and its performance is discussed.**

I. Introduction

This paper considers the problem of estimating the number of operating sensors in a wireless sensor network. Usually, wireless sensor networks are made up of a large number of sensors; after the sensors are deployed the number of operating sensors can vary in time due to battery consumption and/or external factors. The network is designed to operate properly only with a fraction of the deployed nodes as long as this fraction is above a certain level; knowing the number of operating nodes is important for taking decisions like deployment of additional sensors.

A. Network Architecture

The sensor network considered is a sensor network with mobile access points (SENMA), whose architecture was proposed in [1]. The main feature of SENMA architecture is the presence of mobile access points which are nodes with high processing power that act like mobile base stations for the sensor nodes, see Fig.1. In SENMA, the sensors transmit the collected data to the mobile access points, which are connected to a control center by a high rate data link.

It is considered that each sensor transmits the information in packets and each packet contains the identity (ID) of the transmitting node. The packet transmission is done according to a slotted ALOHA protocol and the packets are subject to channel fading.

The essential features of the sensor networks considered are the energy constraint, the low rate traffic and the large number of sensors deployed. Thus, there is no attempt to provide a reliable link between each sensor and the mobile access point; there are no acknowledgements (their processing by the sensors will consume extra power) and there is no packet retransmission at the MAC layer. In this approach, a fraction of the transmitted packets will be lost because of the fading channel and the multiple access. This loss of information is compensated by the large number of sensors deployed, so that the random field of interest is oversampled.

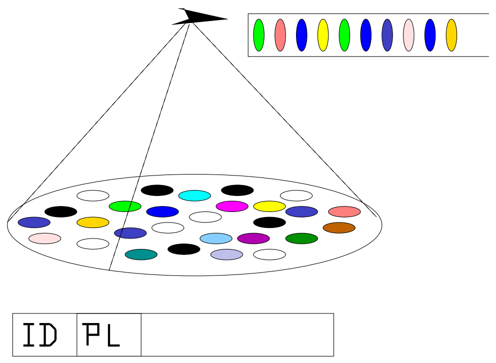

Fig. 1. The Sensor Network with Mobile Access Point; the packet structure contains an ID field and a PL "Power level" field.

The basic model considered in this paper assumes that only one packet is collected at a time, and each received packet can come from any of the operating sensors. This model is extended by considering that the mobile access point has multiple packet reception (MPR) capabilities.

Besides the number of active sensors, the operator might be interested in other quantities related to the network, like the distribution of the energy available to each sensor. In this case, the data packets will include the quantity of interest, besides the sensor ID. In Fig.1 each packet has the field "PL" (power level).

B. Estimation of the Number of Operating Sensors

The number of operating sensors will be estimated based only on the sensor IDs embedded in the received packets. As mentioned before, in the network considered, not all transmitted packets are received. In this case the receiver would need a really large amount of time to receive at least one packet from each operating sensor. Moreover, this approach implies a waste of energy, because the sensing information can be retrieved only from the information provided by a subset of sensors.

Thus, in our setup it would be useful if the operator was able to *estimate* the number of sensors in a network based on as few samples as possible. Alternatively, the problem can be thought of as estimating the number of operating nodes that were *not* seen by the receiver.

The setup of the problem considers a symmetric system with

[†]Corresponding author

This work was supported in part by the Multidisciplinary University Research Initiative (MURI) under the Office of Naval Research Contract N00014-00-1-0564, and Army Research Laboratory CTA on Communication and Networks under Grant DAAD19-01-2-0011.

i.i.d. sampling, model that is equivalent to an urn model with replacement. Urn models were investigated for a long time, see [3] for details.

For the i.i.d. uniform sampling model mentioned we propose an estimator of the number of operating sensors based on the Good-Turing nonparametric estimator [2] of the missing mass (Given a vector of observed sensor IDs (sample), the missing mass is the probability of receiving a message from a sensor that was not seen yet.) The approach is easily applicable to class histogram estimation.

A modified model considered is the batch sampling model, in which the receiver can receive multiple packets at a time, all packets received simultaneously being transmitted by *different sensors*. It is shown that the estimator given previously can be applied successfully to this modified model that does not satisfy the basic i.i.d. assumption.

This estimator choice over the traditional ML estimator is explained in the next section.

II. Model and Motivation

In this section we present the models considered and the motivation behind our approach.

Consider a sensor network having N operating sensors, each of them being identified by an ID that is an element of set $\mathcal{N}$, with $|\mathcal{N}| = N$. In this paper N is assumed constant during the collection time, but unknown. The operator collects n packets, each packet i containing the ID of the transmitting sensor, $X_i \in \mathcal{N}$. The vector of samples $\mathbf{X} \triangleq (X_1, \ldots, X_n)$ is called the sample. The collection is done i.i.d. and in each time slot the received packet can belong to any of the sensors with equal probability:

$$\forall\, x \in \mathcal{N} \;:\; p_x \triangleq \mathbb{P}[X_i = x] = \frac{1}{N}.$$

This model identical to an urn model with replacement.

Since the sample can contain multiple packets from the same sensor, we introduce the set $\mathcal{S}$ (script font) of received labels

$$\mathcal{S} \triangleq \{x \in \mathcal{N} : \exists k \in \{1, \ldots, n\}, X_k = x\};$$

its size $S = |\mathcal{S}|$ represents the total number of (different) sensors that appear in the sample.

With the notations given before, $S_0 \triangleq N - S$ represents the number of operating sensors that do not appear in the current sample. The problem is to estimate N using the received sample $\mathbf{X}$. Since S is observed, this is equivalent to estimating S_0, the number of sensors that are hidden to the operator.

A slightly more complicated model is the *class partition model*. Consider that the set of sensors is partitioned into classes and each sensor transmits in each packet its class index besides its ID. An example of class partition is the battery energy level distribution, where each class corresponds to an interval for the energy available to a sensor.

Let C denote the total number of classes and $c(x)$ the class of sensor x. Denote by $N(c)$ the number of sensors that belong to class c. For each class c, we define $\mathbf{X}(c)$ as a vector made of those elements of $\mathbf{X}$ that have class c, and the corresponding $\mathcal{S}(c)$, $S(c)$

$$\begin{aligned} \mathcal{S}(c) &\triangleq \{x \in \mathcal{N} : \exists k \in \{1, \ldots, n\}\; X_k = x,\; c(k) = c\} \\ S(c) &\triangleq |\mathcal{S}(c)|. \end{aligned}$$

We make the extra assumption that $C \ll \min(n, N)$ so that the $\min(S(c), N(c)) \gg 1$. This says that all the classes appear in the current sample so that we don't need to consider the estimation of the total number of existent classes. In this setup, the problem is to estimate how many sensors are in each class, *i.e.*, the histogram vector $\mathbf{N} = (N(1), \ldots, N(C))$. A possible power profile histogram is shown in Fig. 2. In each bar, the lower part represents the observed part and the upper one is the hidden part.

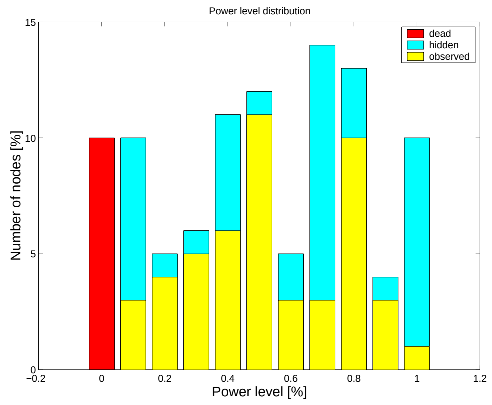

Fig. 2. The histogram of energy available to each sensor. The sensors that are not operating are considered to have 0 energy available (the first bar).

The ML parametric estimator based on the unordered sample $\mathbf{X}$ is given by

$$\hat{N} = \arg\max_{N \geq S} \frac{N!}{N^n\,(N-S)!}. \tag{1}$$

In the case of histogram estimation, the ML solution is given by

$$\hat{\mathbf{N}}_{ML} = \arg\max_{\mathbf{N}} \frac{1}{N^n} \prod_{c=1}^{C} \frac{N(c)!}{(N(c) - S(c))!}.$$

This problem requires a C-dimensional search, which is considerable harder than the optimization problem of the first case. However, in both cases, besides the difficulty associated with numerical evaluation, the ML solutions give little insight into the problem itself, since the analysis is hard, if not impossible. Also, the robustness and/or portability to model changes is unknown.

The estimator proposed in this paper has a simple expression, thus it is easy to implement and it can be analyzed. Also, being based on the non-parametric Good-Turing estimator, it is quite robust to model changes.

III. Estimation of the Number of Operating Sensors and Class Histograms

A. Background - the Good-Turing estimator

Consider a finite or countable set $\mathcal{N}$, a probability distribution P on this set, and a sample $\mathbf{X} = (X_1, \ldots, X_n)$, where $X_i \in \mathcal{N}$ are i.i.d random variables with distribution P. As before, for $x \in \mathcal{N}$, denote $p_x \triangleq \mathbb{P}[X_i = x]$ the probability of class x. Note that a uniform distribution is not required, besides the fact that $\mathcal{N}$ can be an infinite set.

For the observed sample $\mathbf{X}$, define the function $t : \mathcal{N} \to \mathbb{N}$, where $t(x)$ gives the number of samples in $\mathbf{X}$ equal to x. Using the multiplicity function t, we group the classes that appear the same number of times into sets :

$$\mathcal{S}_k \triangleq \{x \in \mathcal{N} : t(x) = k\}.$$

Note that the function t and the sets $\mathcal{S}_k$ are function of the observed sample $\mathbf{X}$, thus they are random variables. We use the notation $S_k \triangleq |\mathcal{S}_k|$. Now we define P_k to be the probability that a new sample, drawn (i.i.d.) with distribution P, belongs to set $\mathcal{S}_k$

$$P_k \triangleq \sum_{x \in \mathcal{S}_k} p_x.$$

For $k = 0$, P_0 is the probability that if a new item is observed, it belongs to a new class. The probability P_0 is called the missing mass and $1 - P_0$ is called the coverage of sample $\mathbf{X}$. The probabilities P_k depend on the sample $\mathbf{X}$, and thus are random variables.

The following estimator for the missing mass, known as Good-Turing estimator was proposed in [2]

$$\hat{P}_0 = \frac{S_1}{n}. \tag{2}$$

This estimator estimated the missing mass using the number of classes that appear in the sample exactly once.

In [4] it is shown that if all classes are equally likely, the Good-Turing non-parametric estimator has an asymptotic efficiency very close to the one of the best estimator derived under the assumption mentioned. This suggests that if the classes are equally likely, the Good-Turing estimator can be used to derive an estimator of the total number of sensors with “good” asymptotic properties.

In this section we used the term “class” in a general way; the reader should not confuse this with the class partition model described previously. When the Good-Turing estimator will be applied to the basic problem, each sensor ID will correspond to one class.

B. Estimation of the Number of Operating Sensors

We propose the following estimation method for the number of operating sensors in a sensor network. First, use the Good-Turing formula (2) to estimate the missing mass P_0. Then, using the assumption of equally likely classes, the missing mass is given by $P_0 = 1 - \frac{S}{N}$. Using the estimated value of P_0, we have the following estimator for N:

$$\hat{N} = \frac{S}{1 - \hat{P}_0} = \frac{S}{1 - \frac{S_1}{n}}. \tag{3}$$

C. Estimation of Static Power Profile

To estimate the distribution of classes among the sensors, for each vector $\mathbf{X}(c)$ we define the sets $\mathcal{S}_k(c)$ and the corresponding $S_k(c) \triangleq |\mathcal{S}_k(c)|$ in the same way as $\mathcal{S}$ and $|\mathcal{S}_k|$ respectively.

For each class c, define the corresponding missing mass as the probability that a new sample will be new *and* belonging to class c

$$P_0(c) \triangleq \mathbb{P}\{X_{n+1} \in \{x \in \mathcal{N} : c(x) = c\} \setminus \mathcal{S}(c) | \mathbf{X}\}.$$

The Good-Turing estimator can be used to estimate the missing mass for each class separately

$$\hat{P}_0(c) = \frac{S_1(c)}{n}.$$

To estimate the number of sensors in each class $N(c) = S(c) + S_0(c)$, we use the relation $P_0(c) = \frac{S_0(c)}{N}$, and the estimates $\hat{N}$ and $\hat{P}_0(c)$ obtained previously to get $\hat{S}_0(c) = \hat{P}_0(c)\hat{N}$, and further, $\hat{N}(c) = S(c) + \hat{S}_0(c)$.

Thus,

$$\hat{N}(c) = S(c) + S_1(c)\frac{S}{n - S_1}.$$

One can verify easily that $\sum_c \hat{N}(c) = \hat{N}$.

D. Estimation of Time-Varying Power Profile

If the classes of sensors change in time, denote by $c^{(t)}(x)$ the class of sensor x at time slot t. We want to estimate $N^{(n+1)}(c)$, the number of sensors in each class at time slot $n + 1$. It is assumed, however, that the class changing process does not modify the distribution of the received samples (for example, the case in which some sensors stop operating during the sample collection process is not included).

If the class $c^{(n+1)}(x)$ can be determined for each element $x \in \mathcal{S}$ based on the received sample, then we can use the previous result for fixed classes considering the classes of all elements of $\mathcal{S}$ given by $c^{(n+1)}(x)$.

One difficulty that appears is related to the prediction of the class of a sensor. This depends strongly on the system model used, *i.e.,* transmission scheme, channel and reception. For example, if the system uses an ideal CSMA MAC protocol, in which only one packet is transmitted at a time and no packets are lost, then the current class of each sensor in $\mathcal{S}$ can be determined exactly. The same happens if each sensor attempts to transmit a packet in every slot, but at most one packet per slot can be received.

However, both situations described above are ideal. A more realistic protocol choice is one in which each sensor transmits a packet in each slot with a fixed probability, and at most one packet per slot is received. We assume that the transmission (and reception) in each slot is independent of the previous slots. In this case, assuming that after n slots, the last received packet from sensor x is in slot t, then given $c^{(t)}(x)$, only the distribution of the class $c^{(n+1)}(x)$ can be determined. One possible way to cope with this situation is to obtain an estimate of $c^{(n+1)}(x)$ using for example the maximum likelihood rule. However, a “soft” approach can lead to better results. In this approach, the sets

$\mathcal{S}^{(n+1)}(c)$ can't be defined, so we define directly the quantities $S^{(n+1)}(c)$, that previously were the corresponding sizes of sets mentioned (define $\mathbf{C} \triangleq (c^{(1)}(X_1), c^{(2)}(X_2), \ldots, c^{(n)}(X_n))$):

$$S^{(n+1)}(c) \triangleq \sum_{x\in\mathcal{S}} \mathbb{P}\{c^{(n+1)}(x) = c|\mathbf{X}, \mathbf{C}\}$$
$$S_1^{(n+1)}(c) \triangleq \sum_{x\in\mathcal{S}} \mathbf{1}_{\{t(x)=1\}}\mathbb{P}\{c^{(n+1)}(x) = c|\mathbf{X}, \mathbf{C}\}.$$

The probabilities in the equation above are specified by the system model. For example, if the class of each sensor varies according to a Markov process with known transition matrix, then one can calculate the probabilities $\mathbb{P}\{c^{(n+1)}(x) = c|\mathbf{X}, \mathbf{C}\}$ easily, using the position of the most recent apparition of x in the data sample $\mathbf{X}$ and the power of the transition matrix.

With these definitions, we apply directly the algorithm given in the previous section.

E. The Batch Sampling Model

A simple extension of the basic model is the so-called batch sampling model, which models a sensor network with a mobile access point that has MPR capability. In this scenario the samples are collected in batches of random size. Denoting by b_i the size of batch i, the sample $\mathbf{X}$ collected in m batches is

$$\mathbf{X} = (X_{1,1}, \ldots, X_{b_1,1}, \ldots, X_{1,m}, \ldots, X_{b_m,m}).$$

The property that the samples in one batch have to be different can be written $\forall\, i, j, k, \quad 1 \le i < j \le b_k, \quad X_{i,k} \ne X_{j,k}$. It is clear that the samples collected in the same batch can't be considered realizations of i.i.d random variables, and thus the main condition under which the Good-Turing estimator was derived is violated.

However, the formula of the Good-Turing based estimator (3) can be applied directly to this model, using $\sum_{i=1}^{m} b_i$ for the sample size and ignoring the restriction imposed by the batch sampling:

$$\hat{N}_{GT} = \frac{S}{1-\hat{P}_0} = \frac{S}{1-\frac{S_1}{\sum_{i=1}^{m} b_i}}. \tag{4}$$

The performance of the Good-Turing estimator for the missing mass can be improved if S_1 is scaled by an appropriate factor. Denote by $\mathbf{X}^m$ a sample made of m batches and add the superscript (m) to all the quantities related to $\mathbf{X}^m$. The estimation procedure proposed is iterative, and it is given briefly by

$$\beta^{(m)} \triangleq \beta^{(m-1)} + \left|\mathcal{S}_1^{(m)} \setminus \mathcal{S}_1^{(m-1)}\right| \left(1 - \frac{b_m}{\hat{N}^{(m-1)}}\right) - \left|\mathcal{S}_1^{(m-1)} \setminus \mathcal{S}_1^{(m)}\right| \left(1 - \frac{b_{m-1}}{\hat{N}^{(m-2)}}\right)$$
$$\hat{P}_0^{(m)} = \frac{\beta^{(m)}}{\sum_{i=1}^{m} b_i}$$
$$\hat{N}^{(m)} = \frac{|S^{(m)}|}{1-\hat{P}_0^{(m)}}. \tag{5}$$

The main idea of the procedure above is that if the batch size b is fixed, then scaling S_1 by a factor $\left(1 - \frac{b}{N}\right)$ produces an unbiased estimator; however, N is not available, and the batch size is random, so we proposed a scaling factor that is function of the batch size and of an estimated value of N.

The performance of the Good-Turing based estimator $\hat{N}_{GT}$ (4) and the adjusted one (5) was investigated by simulations; the results and discussion are given in the simulations subsection.

IV. Simulation Results

In this section we present some simulation results for the algorithms presented. The performance measure used is the confidence interval for the relative estimation error. For a fixed ε, the x-axis represents the total number of samples available while the y-axis gives the levels c such that $\tilde{\mathbb{P}}\left\{\frac{\hat{N}}{N} > c > 1\right\} = \varepsilon$ (upper bound) and $\tilde{\mathbb{P}}\left\{\frac{\hat{N}}{N} < c < 1\right\} = \varepsilon$ (lower bound), where we denoted by $\tilde{\mathbb{P}}$ the observed empirical probability of an event.

In Fig. 3 the performance of the Good-Turing estimator is compared to the performance of the ML estimator given by (1). For the situation analyzed, *i.e.*, $N = 1000$, $Monte = 10000$ Monte-Carlo runs, and $\varepsilon = 0.01$, the confidence intervals for the two methods are virtually identical. Other combination of parameters also shown that the performance loss by using the Good-Turing estimator instead of the ML one is negligible.

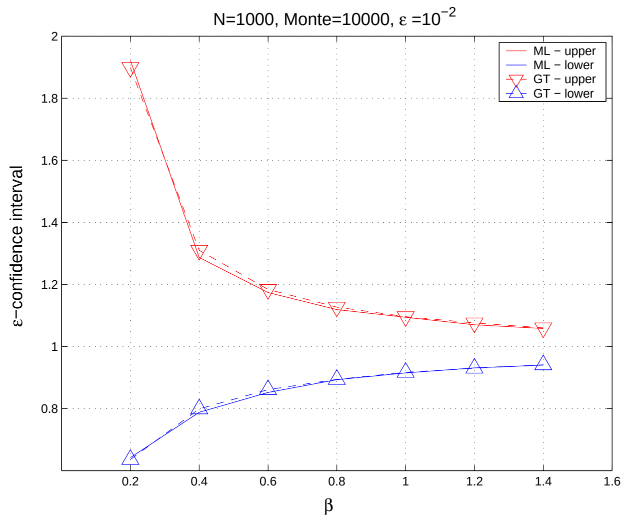

Fig. 3. ML vs Good-Turing; the performance difference is negligible

For the histogram estimation case, we consider again $N = 1000$; the sensors belong to 4 classes, each class representing a certain percentage of the total number of operating sensors; the class distribution is $\mathbf{C}_d = [0.1, 0.2, 0.3, 0.4]$. The confidence intervals plots reveal the fact that the performance plots for the number of operating sensors in each class is better when the number of sensors in each class is larger. Also, one can see that the performance of the estimator for the "larger" classes is very close to the performance of the Good-Turing estimator of the total number of samples.

In Fig. 5 the performance of the Good-Turing based algorithm applied to the batch sampling model (model mismatch) is compared to the performance of the same estimation algorithm applied to the basic model (i.i.d. sampling) and the same number of samples available. The parameters used were $N = 1000$

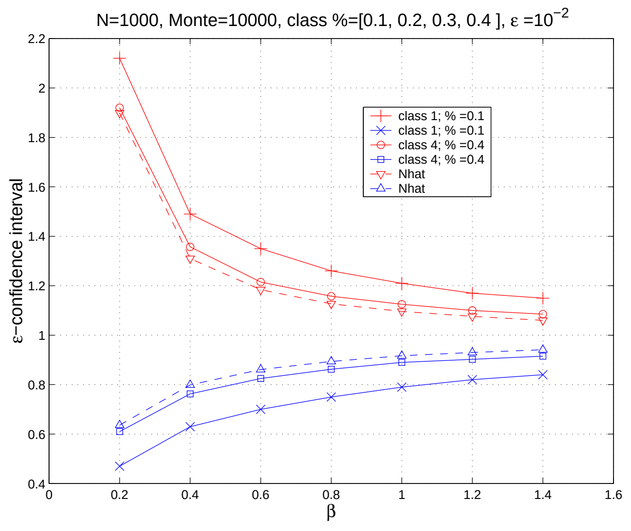

Fig. 4. Histogram estimation vs the basic model

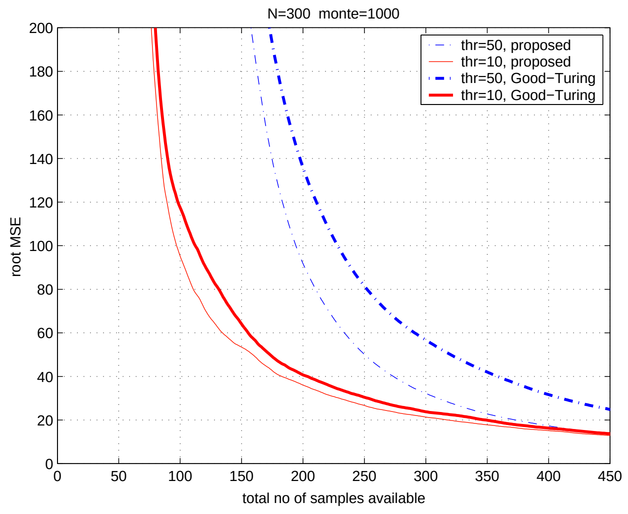

Fig. 6. RMSE vs the total number of of samples available for the Good-Turing algorithm and the proposed variant; the throughput is a parameter

and 10000 Monte-Carlo runs. The batch sizes were generated randomly with a uniform distribution $\mathcal{U}[1,\ldots,40]$. The figure reveals that the model mismatch introduces a small bias, in the sense that the confidence interval curves are shifted upwards. However, the loss is small and in practice it can be important to obtain the necessary number of samples by doing far less sampling operations.

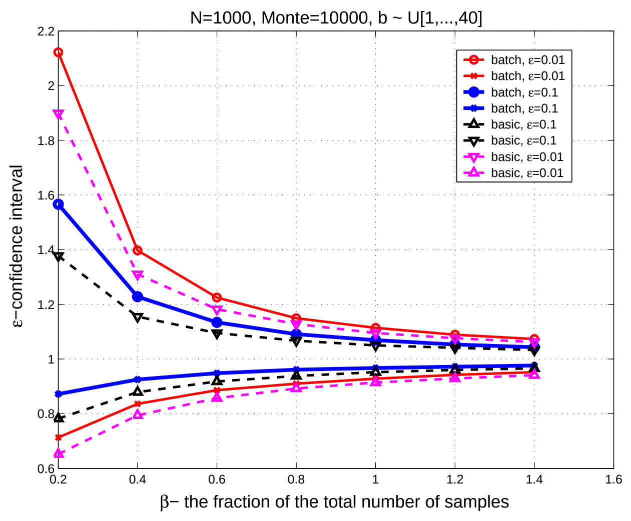

Fig. 5. Batch sampling model (model mismatch) vs the basic model

The Good-Turing based estimator is quite robust to the model change imposed by the batch sampling; however, we want to see its behavior in an extreme case. We consider $N = 300$ and (large) batch sizes generated with uniform distributions $\mathcal{U}[1,\ldots,20]$ and $\mathcal{U}[1,\ldots,100]$. Note that in the second case the batch size is comparable to the number of operating sensors. The performance metric considered is the root mean square error (RMSE). In Fig.6 the performance of the adjusted estimator (5) is compared with the one of $\hat{N}_{GT}$. In this extreme situation, the RMSE can be decreased in half, while for small throughputs the improvement provided by the modified algorithm is small.

V. Conclusions

We proposed the use of Good-Turing type estimators for the number of operating nodes in a sensor network. This estimator has a simple expression and its performance is almost identical to that of the optimal ML estimator. This estimator can be used for more complicated models like histogram estimation. It is also robust to model changes; when applied to the batch sampling model the performance loss due to model mismatch was shown to be small. This showed that the MPR capability of the mobile access point can can decrease significantly the number of time slots necessary to obtain a certain performance.

References

[1] L. Tong, Q. Zhao, and S. Adireddy, “Sensor Networks with Mobile Agents,” in *Proc. 2003 Military Communications Intl Symp.*, (Boston, MA), Oct. 2003.

[2] I. Good, “The population frequencies of species and the estimation of population parameters,” *Biometrika*, vol. 40, pp. 237–264, 1953.

[3] N. Johnson and S. Kotz, *Urn Models and Their Application*. John Wiley & Sons, 1977.

[4] W. Esty, “The Efficiency of Good’s Nonparametric Coverage Estimator,” *The Annals of Statistics*, vol. 14, pp. 1257–1260, Sept. 1986.