On Cardinality Estimation Protocols for Wireless Sensor Networks

Jacek Cichoń, Jakub Lemiesz, and Marcin Zawada

Institute of Mathematics and Computer Science,
Wrocław University of Technology, Poland
{Jacek.Cichon, Jakub.Lemiesz, Marcin.Zawada}@pwr.wroc.pl

Abstract. We consider the problem of estimating a size of wireless sensor networks (WSNs). The problem arise in tasks that sensors have to quickly obtain approximate size of the network to use algorithms i.e. leader election or initialization problem that to work efficiently require this information. In this paper we present two phase algorithm based on order statistics and balls-bins model which effectively estimate the size of WSNs.

Keywords: cardinalities estimation, sensor networks

1 Introduction

Most wireless sensor network (WSN) algorithms require knowing, at least approximately, the size of the sensor network to work efficiently. In other applications the sensors should cope with mobility and changing connectivity and our goal is to count these sensors in a given area. For example, we want to effectively count the number of distinct people in mass events. Then, when the counter exceeds some threshold, we can send an alert message [1]. Thus increase security of those events.

At first, the solution to this problem seems quite simple. We can just store and count the number of different sensors identifiers. However, what if the number of sensors is large for example several hundreds or thousands. Let us assume for the moment that we have a thousand sensors and each has 32 bit identifier, then we need almost $4kB$ of memory to store those identifiers. Moreover, we need at least logarithmic time to insert each new encounter sensor or just check if the encounter sensor has already been inserted. Since, we restrict our attention to a quarter size sensors e.g. MICA2DOT which is based on the Atmel ATmega128 with available $4kB$ memory for data storage. Then, our simple algorithm will take all available memory. However, if the number of sensors exceeds thousand then this algorithm simply fails.

In this paper, we present probabilistic algorithm that are capable to count sensors with a lot smaller memory consumption and two of them run also in a constant time. Our algorithm can count even to $100\,000$ or more with less than $1kB$ of memory available and they have a very low variance of estimation.

1.1 Related Work

2 The Basic Algorithm

Our algorithm consists of two phases. To make the exposition clearer, we describe the intiutive way to look at the algorithm. In the first phase the ordered statistic is used to quickly obtain a rough estimate $\hat{n}_1$ of the network size. Next, in the second phase we use the balls and bins model with preselection to obtain a more accurate approximation, for preselection the estimate $\hat{n}_1$ is used. Namely, each sensor decides to participate in the second phase with probability

$$p = \min\{C/\hat{n}_1, 1\}, \tag{1}$$

where the optimal value of C is to be determined. Then, we obtain the estimation of the number of sensors $\hat{n}_2$ participating in the second phase. Dividing $\hat{n}_2$ by p gives the estimation of the total number of sensors $\hat{n}$. Notice that as a matter of fact each of this two phases could be used separately to approximate the number of sensors. However, in section 3 we show that together they obtain a lot better results.

2.1 First Phase: Order statistics

The first phase of our algorithm is based on ordered statistics. Initially each sensor allocates a table of real numbers T of size k and generates a random value x from the unit interval. Then, each sensor inserts x into its table T, broadcast x to all neighbours and goes to sleep. Upon receiving the message x, a sensor wakes up and tries to insert x into its table T. At first, the algorithm checks whether the value x has already occurred in the table. If so, the message is ignored. Otherwise, two cases are considered. If the table T is not yet completely filled, the value x is inserted into the table. If the table T is already filled up, a sensor checks whether the value x is smaller than the largest number in the table. If so, the largest element is replaced by x. Finally, in the case the table T has been changed, it is sorted and sensor forwards the new element to all neighbours. At any time a sensor can estimate the network size by counting the number of inserted elements or, if the whole table T is filled up, by calculating $(k-1)/T[k]$, where $T[k]$ is the largest number in the table.

Analysis Let $X_1, \ldots, X_n$ be independent random variables with the uniform density on the interval $[0, 1)$. The order statistics $X_{1:n}, \ldots X_{n:n}$ are the random variables obtained from $X_1, \ldots, X_n$ by sorting in the increasing order each of its realizations. The probabilistic density $f_{k:n}(x)$ of the variable $X_{k:n}$ equals

$$f_{k:n}(x) = \frac{1}{\mathrm{B}(k, n-k+1)} x^{k-1}(1-x)^{n-k} \quad . \tag{2}$$

(see e.g.[4]) where $\mathrm{B}(a, b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$. It is well known that $\mathbf{E}\left[X_{k:n}\right] = \frac{k}{n+1}$. Let

$$\mathcal{Z}_{k,n} = \frac{k-1}{X_{k:n}} \, .$$

Algorithm 1

Initialization

set $T[i] \leftarrow \square$ for $i = 1, \ldots, k$
$x \leftarrow$ generate uniformly at random a value from an interval $(0, 1)$
$T[1] \leftarrow x$
broadcast $\langle x \rangle$ to neighbours

Upon receiving a message

receive $\langle x \rangle$
if $\forall_{1 \leq i \leq k} T[i] \neq x$ **then**
    if $\exists_{1 \leq i \leq k} T[i] = \square$ **then**
        $T[i_0] \leftarrow x$ for i_0 such that $T[i_0] = \square$ and sort T
        broadcast $\langle x \rangle$ to neighbours
    else
        if $x < T[k]$ **then**
            $T[k] \leftarrow x$ and sort T
            broadcast $\langle x \rangle$ to neighbours
        end if
    end if
end if

Get estimate number of sensors

if $\exists_{1 \leq i \leq k} T[i] = \square$ **then**
    return $|\{i : T[i] \neq \square\}|$
else
    return $(k-1)/X[k]$
end if

Directly from definition of the density $f_{k:n}$ we deduce that for $k \geq 2$ we have

$$\mathbf{E}\left[\mathcal{Z}_{k,n}\right] = \int_0^1 \frac{k-1}{x} f_{k:n}(x)dx = n \,,$$

and for $k \geq 3$ we have

$$\mathbf{Var}\left[\mathcal{Z}_{k,n}\right] = \frac{n(n-k+1)}{k-2}.$$

Therefore, we proved the following theorem

Theorem 1. *Let $3 \leq k < n$. Then the random variable $\mathcal{Z}_{k,n}$ is an unbiased estimator of the number n and*

$$\frac{\sigma(\mathcal{Z}_{k,n})}{n} = \frac{1}{\sqrt{k-2}}\sqrt{1 - \frac{k-1}{n}} = \frac{1}{\sqrt{k-2}} + O\left(\frac{1}{n}\right). \tag{3}$$

For the estimator $\mathcal{Z}^*$ from [2] we have $E\mathcal{Z}^* = n(1 + o(1))$ and $\sigma(\mathcal{Z}^*)/n \sim \frac{1}{\sqrt{k-2}}$. Hence the estimator $\mathcal{Z}_{k,n}$ has better statistical properties (is unbiased) than the estimator $\mathcal{Z}^*$.

The equation 2 together with Chebyshev's inequality gives some information about the precision of the estimator $\mathcal{Z}_{k,n}$. However this approach is not precise. We will prove

a better estimates by reducing properties of order statistics to the Bernoulli distribution and using the classical Chernoff inequalities.

Let $B_{p,n}$ denotes a random variable with binomial distribution with parameters p and n, i.e. $\Pr[B_{p,n} = k] = \binom{n}{k} p^k (1-p)^{n-k}$.

Lemma 1. *Suppose that $k \leq n$ and $\alpha \in (0,1)$.*

1. *If $\alpha n < k$ then* $\Pr[X_{k:n} \leq \alpha] \leq \exp(-\frac{1}{3}\frac{(k-\alpha n)^2}{\alpha n})$
2. *If $\alpha n > k$ then* $\Pr[X_{k:n} \geq \alpha] \leq \exp(-\frac{1}{2}\frac{(k-\alpha n)^2}{\alpha n})$

Proof. We shall use in the proof the following well known form of Chernoff bounds for binomial distribution:

$$\Pr[B_{p,n} \geq (1+\delta)np] < \exp\left(-\frac{1}{3}np\delta^2\right) \tag{4}$$

$$\Pr[B_{p,n} \leq (1-\delta)np] < \exp\left(-\frac{1}{2}np\delta^2\right) \tag{5}$$

(see e.g. [5]).

Let $X_1, \ldots, X_n$ we a sequence of independent uniformly distributed random variable in the interval $(0,1)$. Let $Y_i = 1$ if $X_i \leq \alpha$ and $Y_i = 0$ otherwise. The random variable $B_{\alpha,n} = Y_1 + \ldots + Y_n$ have binomial distribution with parameters n and α.

Observe that the sentence $X_{k:n} \leq \alpha$ means that $|\{i : X_i \leq \alpha\}| \geq k$ i.e. that $B_{\alpha,n} \geq k$. Notice that $k = \alpha n(1 + \frac{k-\alpha n}{\alpha n})$. Hence if $\alpha n < k$ then from inequality 20 we get

$$\Pr[X_{k:n} \leq \alpha] < \exp(-\frac{1}{3}n\alpha\left(\frac{k-\alpha n}{\alpha n}\right)^2) = \exp(-\frac{1}{3}\frac{(k-\alpha n)^2}{\alpha n}) \,.$$

Suppose now that $\alpha n > k$. Observe that $X_{k:n} \geq \alpha$ is equivalent to $B_{\alpha,n} \leq k$. So we may use inequality 21 and in a similar way we get the result. □

Theorem 2. *Suppose that $\eta > 0$, $0 < \varepsilon < 1$ and $3 \leq k \leq n$. Then*

$$\Pr[\frac{n}{1+\eta}\frac{k-1}{k} < \frac{k-1}{X_{k:n}} < \frac{n}{1-\varepsilon}\frac{k-1}{k}] > 1 - \left(e^{-\frac{k\eta^2}{2(1+\eta)}} + e^{-\frac{k\varepsilon^2}{2(1-\varepsilon)}}\right) \tag{6}$$

Proof. Let $0 < \varepsilon < 1$. From the first part of the last Lemma we obtain

$$\Pr[X_{k:n} \leq (1-\varepsilon)\frac{k}{n}] \leq \exp(-\frac{1}{3}\frac{k\varepsilon^2}{1-\varepsilon})$$

Observe next that $X \leq (1-\varepsilon)\frac{k}{n}$ if and only $\frac{n}{1-\varepsilon}\frac{k-1}{k} \leq \frac{k-1}{X}$. Hence

$$\Pr[\frac{n}{1-\varepsilon}\frac{k-1}{k} \leq \frac{k-1}{X_{k:n}}] \leq \exp\left(-\frac{1}{3}\frac{k\varepsilon^2}{1-\varepsilon}\right)$$

In a similar way we show that

$$\Pr[\frac{n}{1-\eta}\frac{k-1}{k} \leq \frac{k-1}{X_{k:n}}] \geq \exp\left(-\frac{1}{2}\frac{k\eta^2}{1-\eta}\right)$$

□

After putting $\eta = \frac{1}{20}\left(1+\sqrt{41}\right)$, $\varepsilon = \frac{1}{40}\left(-3+\sqrt{249}\right)$ and $k = 400$ into the last formula from the last theorem we get the following bound:

Corollary 1. *Suppose that $n \geq 400$. Then*

$$\Pr[0.728n < \frac{399}{X_{400:n}} < 1.466n] \geq 1 - \frac{2}{e^{20}} \approx 1 - \frac{4}{10^9}$$

Remark 1. Numerical calculations with the incomplete regularized Beta functions shows that in this case for all $n <= 10^7$ we have $\Pr[0.728n < \frac{399}{X_{400:n}} < 1.466n] \geq 1 - 1.357057799 * 10^{-11}$.

Corollary 2. *Suppose that $n \geq 400$. Then*

$$\Pr[0.847n < \frac{399}{X_{400:n}} < 1.217n] \geq 0.99$$

Remark 3. Numerical calculations with the incomplete regularized Beta functions shows that in this case for all $n <= 10^7$ we have $\Pr[0.847n < \frac{399}{X_{400:n}} < 1.217n] \geq 0.9995$. Also, for all $n <= 10^7$ we have $\Pr[(1-\frac{1}{6})n < \frac{399}{X_{400:n}} < (1+\frac{1}{6})n] \geq 0.9987$

2.2 Second Phase: Balls-Bins Model with Preselection

The second phase of our algorithm is based on the balls and bins model. At first each sensor allocate a bit map of size m, which represent m bins, and initialize all entries to "0"s. The sensors at this point decides whether it will participate ($b = 1$) or not. If so, it generates random value x, sets xth bit to "1" and broadcast x to all neighbours. Next, sensor goes to sleep and wakes up upon receiving the message x. Then, it checks if bin $T[x]$ is empty, if so it sets xth bin to "1" and forwards message to neighbours. Otherwise, sensor does nothing and goes to sleep. Notice that if a sensor receives the same message twice then the proper bit is already set so it does not forward the message further. Since we have usually a lot more sensors than bins it also can happen that two stations generate the same value, thus further decreases the number of transmitted messages. At any time we can get the estimated number of sensors by calculating $\log(m/\hat{x})/\log(1-p/m)$, where $\hat{x}$ is the number of empty bins. As a hash function h we can even take identity function $h(x) = x$. However, notice that in our algorithm it is possible that sensors have different numbers of bins. For example, we can generate 32-bit random values. Then each sensor can use simple hash function $h(x) = x \bmod m$, where $m < 2^{32}$.

Analysis Let us assume that we have n balls and m bins. Let us fix a probability $p \in [0,1]$. Then, we consider the following two round process: in the first round each ball decides with probability p whether it will participate in the next round or not. In the second round each ball chooses uniformly at random one of the bins. Let random variable $X_{m,n}$ denotes the number of empty bins after n balls have been thrown. In the theorem below we shall generalize the result from [3], which holds only for $p = 1$.

Algorithm 2

Initialization

set bitmap $T[i] \leftarrow 0$ for all $i = 0, \ldots, m-1$

$b \leftarrow \begin{cases} 1 \text{ with probability } p, \\ 0 \text{ otherwise.} \end{cases}$

if $b = 1$ **then**

  $x \leftarrow \text{random}(0, m-1)$

  $T[h(x)] \leftarrow 1$

  broadcast $\langle x \rangle$ to neighbours

end if

Upon receiving a message

receive $\langle x \rangle$

if $T[h(x)] = 0$ **then**

  $T[h(x)] \leftarrow 1$

  broadcast $\langle x \rangle$ to neighbours

end if

Get estimate number of sensors

$\hat{x} = m - \sum_{i=0}^{m-1} T[i]$

return $\log(\hat{x}/m)/\log(1-p/m)$

Theorem 3. *Let $p \in [0,1]$. Then*

$$\mathbf{E}\left[X_{m,n}\right] = m(1-\frac{p}{m})^n \,. \tag{7}$$

Proof. Let A_i^n denotes the event that a i-th bin is empty for all $1 \leq i \leq m$ after n balls have been thrown. Then $\Pr[A_i^n] = (1-\frac{1}{m})^n$. Let

$$Y_i^n = \begin{cases} 1 \text{ if a } i\text{-th bin is empty}, \\ 0 \text{ otherwise.} \end{cases}$$

Thus, $X_{m,n} = \sum_{i=1}^{m} Y_i^n$. Moreover, let Z denotes the event that k balls have decided to participate in the second round. Then $\Pr[Z=k] = \binom{n}{k} p^k (1-p)^{n-k}$ and

$$\mathbf{E}\left[X_{m,n}\right] = \sum_{k=0}^{n} \mathbf{E}\left[X_{m,n} | Z=k\right] \Pr[Z=k].$$

Since $\mathbf{E}\left[X_{m,n}|Z=k\right] = \sum_{i=1}^{m} \mathbf{E}\left[Y_i^k\right] = \sum_{i=1}^{m} \Pr[Y_i^k] = m(1-\frac{1}{m})^k$, we obtain that

$$\mathbf{E}\left[X_{m,n}\right] = \sum_{k=0}^{n} m(1-\frac{1}{m})^k \binom{n}{k} p^k (1-p)^{n-k} = m(1-\frac{p}{m})^n.$$

□

Thus, we can derive the estimator $\hat{n}$ for the number of balls that have been thrown to m bins as

$$\hat{n} = \frac{\log(\frac{X}{m})}{\log(1-\frac{p}{m})} \,. \tag{8}$$

Theorem 4. *Let $p \in [0,1]$. Then*

$$\mathbf{Var}\left[X_{m,n}\right] = m((1-\frac{p}{m})^n - (1-\frac{2p}{m})^n) + m^2((1-\frac{2p}{m})^n - (1-\frac{p}{m})^{2n}). \quad (9)$$

Proof. We keep the notation from the proof of Theorem 3. By the law of the total variance

$$\mathbf{Var}\left[X_{m,n}\right] = \mathbf{E}\left[\mathbf{Var}\left[X_{m,n}|Z\right]\right] + \mathbf{Var}\left[\mathbf{E}\left[X_{m,n}|Z\right]\right] .$$

we have

$$\mathbf{Var}\left[X_{m,n}\right] = \mathbf{E}\left[\mathbf{E}\left[(X_{m,n})^2|Z\right]\right] - \mathbf{E}\left[X_{m,n}\right]^2 . \quad (10)$$

We will use the formula

$$\mathbf{E}\left[\mathbf{E}\left[(X_{m,n})^2|Z\right]\right] = \sum_{k=0}^{n} \mathbf{E}\left[(X_{m,n})^2|Z=k\right] \Pr[Z=k]. \quad (11)$$

Notice that

$$\mathbf{E}\left[X_{m,n}^2|Z=k\right] = \mathbf{E}\left[(Y_1^k + \ldots + Y_m^k)^2\right] = \mathbf{E}\left[\sum_{i=1}^{m}(Y_i^k)^2 + \sum_{i\neq j} Y_i^k Y_j^k\right].$$

Since $(Y_i^k)^2 = (Y_i^k)$ we obtain

$$\sum_{i=1}^{m} \mathbf{E}\left[(Y_i^k)^2\right] = \sum_{i=1}^{m} \mathbf{E}\left[(Y_i^k)\right] = \mathbf{E}\left[X_{m,n}|Z=k\right].$$

Let us now assume that $i \neq j$. Since the assignment of the balls is independent

$$\mathbf{E}\left[Y_i^k Y_j^k\right] = \mathbf{Pr}\left[A_i^k \cap A_j^k\right] = (1-\frac{2}{m})^k.$$

Then

$$\mathbf{E}\left[X_{m,n}^2|Z=k\right] = m(1-\frac{1}{m})^k + m(m-1)(1-\frac{2}{m})^k. \quad (12)$$

Finally, combining formulas (**??**) , (**??**) and (**??**) we obtain the assertion. □

As using the formula (6) is rather inconvenient, we will henceforth use the following good approximation

$$\mathbf{Var}\left[X_{m,n}\right] \approx m(e^{-np/m} - e^{-2np/m} - \frac{np^2}{m}e^{-2np/m}) \quad (13)$$

In the derivation of this approximation we use

$$m\left((1-2p/m)^n - (1-p/m)^{2n}\right) \cong \frac{np^2}{m}e^{-2np/m}$$

since...[3, p.225]

2.3 The Complete Algorithm

To optimize the complete algorithm we shall find the value of C in (1) minimizing the variance of $\hat{n}$. As $\hat{n}_1$ is unbiased estimator of n let us temporarily assume that $p = C/n$. Let random variable $V_{m,n} = X_{m,n}/m$ denotes the fraction of empty bins. We have

$$\mathbf{E}\left[V_{m,n}\right] = \left(1 - \frac{C/m}{n}\right)^n , \tag{14}$$

$$\mathbf{Var}\left[V_{m,n}\right] \approx \frac{e^{-C/m} - e^{-2C/m}(1 - \frac{C^2}{nm})}{m} . \tag{15}$$

To give an approximation of the variance of the estimator $\hat{n}$ we expand the function $\log(V_{m,n})$ by its Taylor series about $x_0 = e^{-C/m} \approx \mathbf{E}\left[V_{m,n}\right]$ and truncate after the second term

$$\hat{n} = \frac{\log(V_{m,n})}{\log(1 - \frac{C}{nm})} \approx \frac{1}{\log(1 - \frac{C}{nm})}\left(-\frac{C}{m} + \frac{V_{m,n} - x_0}{x_0}\right) . \tag{16}$$

The error caused by such a truncation was discussed in [3]. Then

$$\mathbf{Var}\left[\hat{n}\right] \approx \left(x_0 \log(1 - \frac{C}{nm})\right)^{-2} Var[V_{m,n}] . \tag{17}$$

For values of x close to 0 we have

$$\log(1 - x) = -x + O(x^2) .$$

Hence, $\log(1 - \frac{C}{nm}) \approx -\frac{C}{nm}$ (UZASADNIC). Finally, we obtain that

$$\mathbf{Var}\left[\hat{n}\right] \approx \frac{n^2 m}{C^2}\left(e^{C/m} - \frac{C^2}{nm} - 1\right) . \tag{18}$$

Let $V(C)$ denote the right hand side of the above equation. Now we are able to find an approximation of the optimal value of C

$$C^* = \underset{C \in (0,\infty)}{\operatorname{argmin}}\ V(C) = m\left(2 + \mathcal{W}(-\frac{2}{e^2})\right) \approx 1.594m . \tag{19}$$

Moreover, one can check that the standard error $\mathbf{SE}\left[\hat{n}\right]$ can be tightly upper bounded by the following function of the parameter C (SPRAWDZIC !)

$$\mathrm{SE}(C) = \frac{\sqrt{m}}{C}\left(e^{C/m} - \frac{C^2}{nm} - 1\right)^{1/2} . \tag{20}$$

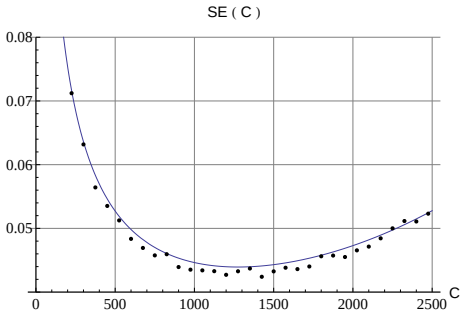

Fig. 1. Comparison of simulation results (dots) to SE(C) for $n = 10^4$, $m = 800$. Notice that $C^* \approx 1275$.(WYKRES DO POPRAWY)

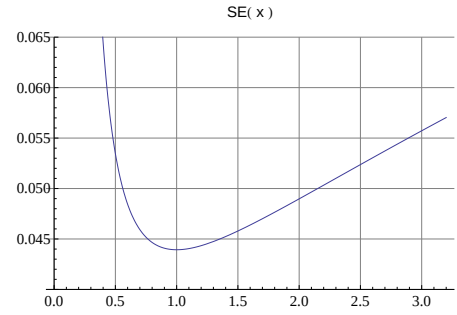

Fig. 2. Function SE(x) for $C = 1.594m$ and $m = 800$

DALEJ WSZYSTKO DO POPRAWY:

Setting $p = C^*/\hat{n}_1$ and $x = \hat{n}_1/n$, we may express the standard error as a function of x

$$\mathrm{SE}(x) = \frac{x}{1.594\sqrt{m}} \left(e^{1.594/x} - 1\right)^{1/2} .$$

For definiteness let us assume that each sensor has no more than 100 bytes of available memory. Then in the first phase we need to set $k = 16$ as we take 6 bytes for each statistic, to ensure the correctness of the procedure for networks up to size 1.5×10^7. In the second phase we set $m = 800$. Hence, as we put in the formula (22) $\eta = 1.1$ and $\varepsilon = 0.7$, we obtain that

$$\Pr[0.446 < x < 3.125] > 0.99 .$$

Thus, we can show that inequality $\mathrm{SE}(x) < 0.06$ holds with probability at least 0.99 (see Figure 2).

3 Experimental Results

4 Conclusions

References

1. Cichon, J., Kapelko, R., Lemiesz, J., Zawada, M.: On alarm protocol in wireless sensor networks. In: ADHOC-NOW. (2010) 43–52

2. Lumbroso, J.: An optimal cardinality estimation algorithm based on order statistics and its full analysis. In: AofA'10. Number 5333 in Discrete Mathematics and Theoretical Computer Science (2008) 491–506
3. Whang, K.Y., Zanden, B.T.V., Taylor, H.M.: A linear-time probabilistic counting algorithm for database applications. ACM Trans. Database Syst. **15**(2) (1990) 208–229
4. Arnold, B., Balakrishnan, N., Nagaraja, H.: A First Course in Order Statistics. John Wiley & Sons, New York (1992)
5. Mitzenmacher, M., Upfal, E.: Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press, New York, NY, USA (2005)