Efficient median estimation for large-scale sensor RFID systems

Huda El Hag Mustafa

Faculty of Mathematical Sciences,
University of Khartoum,
Khartoum 11115, Sudan
and
Department of Computer Science,
College of William and Mary,
Williamsburg, VA 23187, USA
Email: huda@cs.wm.edu

Xiaojun Zhu

State Key Laboratory for Novel Software Technology,
Nanjing University,
Nanjing, Jiangsu 210093, China
and
Department of Computer Science,
College of William and Mary,
Williamsburg, VA 23187, USA
Email: gxjzhu@gmail.com

Qun Li*

Department of Computer Science,
College of William and Mary,
Williamsburg, VA 23187, USA
Email: liqun@cs.wm.edu
*Corresponding author

Guihai Chen

State Key Laboratory for Novel Software Technology,
Nanjing University,
Nanjing, Jiangsu 210093, China
Email: gchen@nju.edu.cn

Abstract: We consider the median estimation problem in a large-scale sensor augmented RFID system. The large-scale deployment of RFID technology has opened the door to innovative ways to integrate RFID and sensor technology. Sensor-tags are tags that can report over 50 types of physical information to a reader. The traditional way to obtain information from sensor-tags is to query each tag. When the number of tags is large, however, it is prohibitive to query tags individually due to the high delay. In this paper, we present a probabilistic algorithm to estimate the median of a set of sensor-RFID tags without individually querying each tag. The median estimation problem is solved using binary search. Our evaluation demonstrates that the median search algorithm exhibits high accuracy and reasonable time latency. Moreover, we also design an exact algorithm for the continuous median update problem. Our algorithm can incrementally compute the exact median in less time.

Keywords: sensor; RFID tags; median; binary search; TCS; threshold checking scheme; continuous median update.

Reference to this paper should be made as follows: Mustafa, H., Zhu, X., Li, Q. and Chen, G. (2012) 'Efficient median estimation for large-scale sensor RFID systems', *Int. J. Sensor Networks*, Vol. 12, No. 3, pp.171–183.

Biographical notes: Huda El Hag Mustafa is currently a Visiting Research Scholar at the Computer Science Department in the College of William and Mary. She is an Assistant Professor at the Department of Computer Science in the University of Khartoum. She received her PhD degree in Electrical Engineering from Khartoum University. Her research interests include RFID, wireless networks, sensor networks, and security.

Xiaojun Zhu is currently a Visiting Student at the Computer Science Department in the College of William and Mary. He is currently pursing his PhD degree at Nanjing University, Nanjing, China. He received the BS in Computer Science from Nanjing University, Nanjing, China, in 2008. His current research interests include wireless sensor networks and vehicular networks. The first two authors make equal contributions to this paper.

Qun Li is currently an Associate Professor in the Department of Computer Science at the College of William and Mary. He received the PhD degree in Computer Science from Dartmouth College. He received the US National Science Foundation (NSF) Career award in 2008. His research interests include wireless networks, sensor networks, vehicular networks, RFID, pervasive computing systems, smart grid, social networks, and security and privacy.

Guihai Chen is a Professor in Nanjing University, China. He received the BS degree from Nanjing University in 1984, the ME degree from Southeast University in 1987, and the PhD degree from the University of Hong Kong in 1997. He has a wide range of research interests with focus on sensor networks, peer-to-peer computing, high-performance computer architecture, and combinatorics.

1 Introduction

There are a lot of arguments for integrating RFID tags and sensors (Liu et al., 2008). RFID tags are unobtrusive, inexpensive, have a long lifetime and do not consume a lot of power (Buettner et al., 2008). Sensors on the other hand have sensing ability, need a source of power on-board and can communicate using multi-hop communication protocols. Sensors are also exhibiting more and more accuracy and sensitivity. We consider an application scenario in which RFID technology is used for identification and communication while sensor technology is used for sensing. Both the tag and the sensor are attached to an object which needs to be monitored. These sensor-tags have the capabilities of sensing and more powerful computational capabilities (Smith et al., 2006; Zhang et al., 2009) than ordinary tags. An example of tags integrated with sensors is WISP tags (Smith et al., 2006; Roy et al., 2010). WISP tags can report quantities such as temperature, liquid level, and light. Inside the WISP tags, the energy taken from the reader operates a 16-bit programmable, low power micro-controller unit. The micro-controller can sample data from the sensors and report that data when probed by the readers (Smith et al., 2006; Roy et al., 2010). These sensor readings can be very large in quantity and very wide in the range of values represented by the readings. The statistical distributions of the data values can vary according to deployment scenario and the objects to which the tags are attached.

In reporting large amounts of data, it is sometimes important to estimate some order statistics, such as the median, of the data. Previous work has solved the median estimation problem in sensor networks (Greenwald and Khanna, 2001). In the sensor-RFID scenario, there are several challenges which must be overcome in order to compute the median in-network, at the reader. First, the communication pattern is very different because the tags can only communicate with the reader. Second, the tags are computationally weak compared with sensors. Another drawback is that if the median is computed at the reader then all tags must communicate with the reader simultaneously which will involve a lot of collisions, and be time consuming due to a lot of retransmissions. So, we propose a method to aggregate the data without reading all the tags.

In this paper, we extend the median selection problem to the sensor-tag context. We assume that there is a warehouse in which milk cartons are stored, and each milk carton has a sensor-tag attached to it. The attached sensor-tag is capable of measuring the milk carton temperature. At any instant, we are interested in the temperature which gives an indication of the overall storage conditions at the storage facility. We are interested in the median temperature of the milk cartons due to its insensitiveness to outliers. We study how to infer median values by querying the sensor-tags without the necessity of actually collecting data values from these devices. Collecting specific data values from each tag would be similar to tag identification protocols in terms of time and power cost. We estimate the median by leveraging tag cardinality estimation algorithms. We envision that our technique can be applied to obtain other quantiles. Our contributions can be summarised as follows:

- We design a median estimation algorithm based on binary search using a simple Threshold Checking Scheme (TCS; Sheng et al., 2008). TCS is used to test whether the number of active sensor-tags is more than a given threshold. By carefully selecting the relevant parameters, it is with a high probability that TCS returns true (Sheng et al., 2008) when the number of active sensor-tags exceeds a given threshold. To the best of our knowledge, this paper presents the first efficient median estimation algorithm for sensor-tags.

- We also design an algorithm to incrementally find the median when the median has to be updated continuously. Our algorithm merely queries a limited number tags that are significant to the median change. Our evaluation shows that the running time of the algorithm can be drastically lower than the naive scheme.

2 Related work

Numerous protocols to identify tags and estimate tag set cardinality have been proposed for RFID systems. All these protocols aim to minimise time and/or power consumption. Tag identification protocols can be divided into two main categories: the first category is tree-based protocols, in which the reader broadcasts an ID prefix and all the tags with IDs matching the prefix will reply. If there is a collision the reader will divide the tags into two groups and then query one group again. The reader continues splitting and probing until each individual tag is identified (Law et al., 2000; Micic et al., 2005; Myung and Lee, 2005). The second category is based on the ALOHA protocol and the Slotted ALOHA protocol (Madden et al., 2002a; Cha and Kim, 2005; Bonuccelli et al., 2006; Sheng et al., 2008; Tan et al., 2008; Sheng et al., 2010; Xie et al., 2010).

Probabilistic estimation algorithms which efficiently estimate the cardinality of RFID tags have been proposed in numerous works (Cha and Kim, 2006; Kodialam and Nandagopal, 2006; Qian et al., 2008; Han et al., 2010). These estimation algorithms differ in the level of accuracy, speed, and power consumption. For example, Kodialam and Nandagopal (2006) obtained an estimate of 50,000 tags within a confidence interval of ±500 tags in 4.5 s. The main concern with these protocols is that the upper bound on the number of tags must be known, in order to accurately estimate the cardinality of the set of tags within the specified error bound.

In sensor networks a lot of work has been done on data aggregation techniques. In the work of Madden et al. (2002a), a declarative query interface was proposed that allowed users to perform aggregate operations such as MIN, MAX, MEDIAN, and similar operations. This approach showed improved performance over centralised techniques, since the sensors collaborated to obtain an accurate result. In the work of Patt-Shamir (2004) two algorithms were proposed to compute the median, a randomised protocol which computed an approximate median and a deterministic protocol which computed an exact median. In the work of Singh and Prasanna (2003) an algorithm is proposed for the selection of the *i*-th largest (smallest) element in single hop dense sensor networks. Finding the median is an example of the selection problem.

Some research has been done on innovative systems which leverage both RFID and sensor technology. In an elderly group home floor mats were embedded with sensors, and RFID tags were attached to slippers in order monitor dementia patients in a discrete manner. This enabled caregivers to distinguish the patients and their movements (Miura et al., 2009). Some protocols for collecting sensor produced information from RFID tags are suggested in the literature (Chen et al., 2010; Qiao et al., 2011). To the best of our knowledge, no one has proposed a median search algorithm for sensor-tags. We believe the research in this paper on RFID sensors can be applied to other sensor network or security problems (Xuan et al., 2002; Ren et al., 2005; Xing et al., 2005; Wang et al., 2008).

3 System model and problem statement

3.1 Slotted ALOHA

We consider a system in which there are n RFID tags t_1, t_2, …, t_n, integrated with sensors, each tag has a unique ID. We assume that associated with each RFID tag is a data value y, which represents the quantity measured by the sensor.

Our communication model is built upon the slotted ALOHA protocol (Lee et al., 2005). In this protocol each frame is made up of a number of time slots. The reader broadcasts a frame size and a random seed S. A RFID tag uses a hash function $H(\cdot)$, f, S and its ID to pick a slot to communicate in. In addition to these parameters, the reader broadcasts a number y, and only the tags with value less than y will communicate in the following round.

In the slotted ALOHA protocol, an important parameter is the size of the frame. In our algorithm, the reader initiates the communication by sending a probe and zero or more tags will reply. Each tag chooses a slot to reply. If no tag replies in a slot it is called an empty slot. If a single tag replies in a slot it is called a singleton slot and if two or more tags reply in a slot it is called a collision slot.

Suppose s_0, s_1, and s_c are the number of empty, singleton, and collision slots, respectively, observed instantaneously by the reader. Let S_0, S_1, and S_c be random variable for no tags transmit in slot j, one tag transmits in slot j and more than two tags transmit in slot j. The expected values of the number of empty slots ($E[S_0]$) and the collision slots ($E[S_c]$) are then (Kodialam and Nandagopal, 2006):

$$E[S_0] = fe^{-\rho}$$
$$\sigma_0^2 = fe^{-\rho}\left(1-(1+\rho)e^{-\rho}\right)$$
$$E[S_1] = f\rho e^{-\rho}$$
$$\sigma_1^2 = fe^{-\rho}\left(1-(1+\rho)e^{-\rho}\right)$$
$$E[S_c] = f\left(1-(1+\rho)e^{-\rho}\right)$$
$$\sigma_c^2 = fe^{-\rho}\left((1+\rho)-\left(1+2\rho+\rho^2+\rho^3\right)e^{-\rho}\right)$$

For proof see Kodialam and Nandagopal (2006).

3.2 Problem definition

Given an RFID reader and a large set of tags, we wish to accurately estimate the median value of the tags without probing each tag individually. The algorithm takes several

parameters as input (see Algorithm 1). Those parameters can be determined according to user's accuracy requirement based on our later analysis. We omit how to determine those parameters in this paper.

Table 1 Abbreviations

Abbreviation	*Description*
E_{max}	Maximum value
E_{min}	Minimum value
n	Number of tags
n_c	Number of collision slots
n_0	Number of zero slots
ε	Error bound on median estimation
ρ	Load factor t/f
f	Number of slots in a frame

We use two performance metrics:

1 We define the accuracy of the median estimation algorithm as how close the estimated median is to the real median. The performance of the algorithm is determined by the parameters specified in the algorithm input.

2 The execution time of the median search algorithm is the time taken to estimate the median.

4 Median estimation algorithm

In this section, we describe our median estimation algorithm. We base our approach on binary search using TCS. The system is modelled as a reader and a set of tags. The reader probes the tags and gets a result when the queries terminate. We assume that each tag holds a data value y_j. The set of all data values is given by Y. The number of data values, which in this case also represents the tag cardinality, is given by n. The maximum possible data value is given by E_{max} and the minimum possible data value is given by E_{min}. The binary search algorithm divides the tag set into two non-overlapping subsets of RFID tags, tags with values above the median value (M^+) and tags with values below the median value (M^-). After any iteration of the binary search algorithm we test $|M^-|$ to see if it is equal to half the total number of tags $n/2$. At this point the algorithm can carry out one of three actions depending on whether $|M^-|$ is less than, greater than or equal to $n/2$. We cannot provide a deterministic guarantee that the estimated median has 100% accuracy. We can optimise for accuracy by increasing the number of rounds in the TCS algorithm and increasing the number of iterations in the median search algorithm. We can also optimise for time by decreasing the accuracy in both algorithms. In summary, this algorithm gives us the flexibility to capture the trade-off between accuracy and time.

The details of the algorithm are shown in Algorithm 1, where the input consists of frame size f, maximum number of tags n and E_{min} and E_{max}.

The reader computes the load factor $\rho = n/(2f)$ (Line 3) and calculates τ_c which is the expected number of collision slots (n_c), if the number of responding tags is $n/2$ and the frame size is f.

Algorithm 1: Median estimation binary search algorithm

Input: $\varepsilon, n, f, E_{max}, E_{min}$;
`(/* error bound, tag number, frame size, max. and min. values of the tags */)`
Output: x; the median
1 **begin**
2 $u = E_{max}$; $l = E_{min}$;
3 $\rho \leftarrow n/(2f)$;
4 $\tau_c \leftarrow f(1 - (1 + \rho)e^{-\rho})$;
5 **while** *true* **do**
6 $x \leftarrow (l + u)/2$;
7 Reader broadcasts f and x;
8 Each tag with value less than x randomly picks a time slot to reply;
9 Reader gets the number of collision slots n_c;
10 **if** $n_c > (1 + \varepsilon)\tau_c$ **then**
11 $u \leftarrow x$;
12 **else if** $n_c < (1 - \varepsilon)\tau_c$ **then**
13 $l \leftarrow x$;
14 **else**
15 return x;

The basis of the algorithm is a slotted ALOHA scheme (Lines 7–9). The estimated value for the median x is then computed. At the beginning, the RFID reader probes the tags by slotted ALOHA protocol with frame length f and the value x. Each tag picks a slot randomly and uniformly, and then transmits in that slot. For any slot in the frame, the reader can detect three events: no tag transmitting (empty slot), one tag transmitting (singleton slot), and multiple tags transmitting (collision slot). The reader will infer the number of tags based on the number of collision slots n_c. We intentionally choose the number of collision slots because the number of singleton slots is not monotonic with respect to the number of tags. On the other hand, the estimator based on empty slots performs poorly compared to that based on collision slots in our situation, since we prefer smaller frame size, which implies a larger load factor.

If the number of collision slots is greater or less than the $\tau \pm \varepsilon$ then the median is set to a new value and the reader broadcasts this new value, otherwise the value x is the median (Lines 10–15).

4.1 Threshold checking scheme

In this section, we elaborate on the algorithm underlying our binary search implementation, the TCS (Sheng et al., 2008). The basic idea of TCS is to test whether the number of concerned tags (say an unknown value n') is greater or less

than a threshold (n).We can first calculate the expected number of collision slots (τ_c) given the number of tags (n) and frame size (f).We will then count the number of collision slots (n_c') for the concerned tags and frame size f. If $n_c' > (1+\varepsilon)\tau$, then TCS asserts that the number of concerned tags is larger than n. If $n_c' < (1+\varepsilon)\tau$, TCS asserts that the number of concerned tags is less than n. The parameter ε determines the confidence level. Intuitively, the number of the collision slots is an indicator of the number of tags involved. TCS leverages the relationship between the number of collision slots and number of involved tags.

The algorithm uses a query-based approach in which all the tags that have value below x transmit while the others do not transmit. This can be implemented in the following way: the reader broadcasts three numbers x, w, and z and each tag t_i will transmit if the following condition holds $H(x, y_i) \bmod w = z$ (Sheng et al., 2008), where H is the hash function, and y_i is the value held by the tag. The RFID tag will be active only if $yi < x$. The TCS algorithm uses the ALOHA scheme in which frame slots can be zero, singleton slots, or collision slots. In a typical ALOHA scheme, the collision and the singleton slots occupy a much larger time interval than the zero slot because in the non-zero slots the tags transmit their ID with Cyclic Redundancy Check (CRC). In the TCS scheme each tag transmits a short bit string to advertise its presence (< 10) bits (Kodialam and Nandagopal, 2006), so all the frame slots have a length which is short and represents one short slot S. The identification slots occupy a long slot L.

The TCS algorithm can deliver accurate tag counts within the specified error bound when the parameters are carefully chosen, it is also very efficient because the frame is made up of short slots and, and the frame length is chosen by the user.

TCS is sensitive to signal loss because the threshold is determined by the observed number of collision slots. So if any tag signal is lost a collision slot may become a singleton slot, if more than one tag signal is lost, a collision slot may become an empty slot (Sheng et al., 2008). This loss can be compensated for by carefully choosing the error bound according to the deployment environment.

5 Theoretical analysis

In this section, we provide theoretical guarantee of our binary search algorithm. Binary search involves several iterations, each of which gives a range for the median. We say the algorithm makes a *wrong turn* in an iteration if that iteration gives a wrong range of the median. If all iterations do not make a wrong turn, then the search terminates with a correct median.

We first review some basic properties about RFID counting algorithm. For t tags and f time slots, the number of collision slots is a random variable, which we denote by n_c. As mentioned before, Kodialam and Nandagopal (2006) shows that n_c approximately follows normal distribution with

$$E[n_c] = f\left(1-(1+\rho)e^{-\rho}\right) \tag{1}$$

$$\sigma^2 = fe^{-\rho}\left((1+\rho)-(1+2\rho+\rho^2+\rho^3)e^{-\rho}\right) \tag{2}$$

where $\rho = t/f$ is the load factor.

We denote by $F(x)$ the number of tags with value less than x. Let $\Phi(x)$ be the cumulative distribution function for standard normal distribution such that

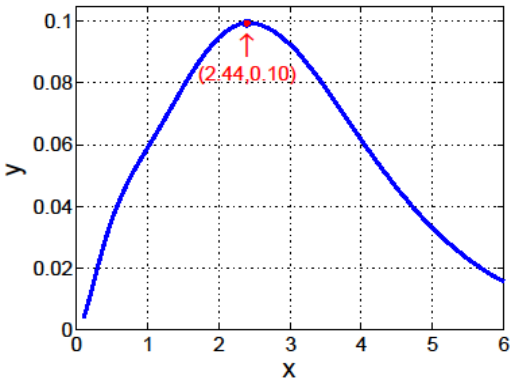
$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2}\, dt.$$

5.1 Probability of finding exact median

In this subsection, we assume that tag values are integers (non-integers can be scaled). If all iterations do not make a wrong turn, then the search terminates within $\log L$ iterations where L is the length of tag value range. Our analysis consists of three steps. First, we bound the probability that an iteration makes a wrong turn. Second, we derive the probability that the current iteration stops if it encounters the median. Third, we bound the overall probability that the algorithm gives the median.

We will give a probability guarantee that is irrelevant of tag value distribution. This is done by using a bound for the variance of collision slots. Consider equation (2). If we fix frame length f, then the variance σ^2 varies with respect to the number of tags. Note that varying t is equivalent to varying ρ, so we consider ρ as the variable. We plot the function $y = e^{-x}\left((1+x)-(1+2x+x^2+x^3)e^{-x}\right)$ in Figure 1. This function is not a monotonic function with respect to load factor. We find that $\sigma^2_{\max} = 0.1f$ at $\rho = 2.44$. In the following, we will use the term $\sigma_{\max}$ without mentioning its value. Note that our binary search algorithm also relies on equations (1) and (2). Specifically, τ_c is set according to equation (1). Its variance is $\sigma^2_{\tau_c} = fe^{-\rho}\left((1+\rho)-(1+2\rho+\rho^2+\rho^3)e^{-\rho}\right)$ where $\rho = n/(2f)$.

Figure 1 Plot for $y = e^{-x}\left((1+x)-(1+2x+x^2+x^3)e^{-x}\right)$ (see online version for colours)



Denote the median value as m. We assume that no more than 1 tag has the value m. Therefore, we have $F(m) = n/2$ (when n is odd, we can simply take a floor operation). Consider an iteration of the algorithm, and suppose the current candidate median is x. In an ideal case, if $F(x) < n/2$, then the iteration should turn right (i.e. $l \leftarrow x$); if $F(x) > n/2$, then the iteration should turn left (i.e. $u \leftarrow x$); otherwise, the iteration should break the loop. We will consider the wrong cases in the following.

Consider the iteration that takes a wrong turn. We have the following lemma.

Lemma 1: *Suppose all previous iterations of the while loop give correct range, then the current iteration gives a wrong range, or terminates prematurely, with probability at most* $\Phi\left(\frac{\in \tau_c}{\sigma_{\max}}\right)$.

Proof: Let $[l, u]$ be the range of the median value at the previous iteration of the *while* loop. By assumption, $m \in [l, u]$. Let $z = (l + u)/2$ and n_z be the estimated number of collision slots (n_c in the algorithm). Consider the two events that the algorithm takes a correct turn.

1 $F(z) > n/2$ and $n_z > (1 + \varepsilon)\tau_c$. Since function (1) is monotonically increasing with respect to the number of tags, we have $E[n_z] > \tau_c$. To see this, note that $F(z)$ is the number of tags in the expression $E[n_z]$, while $n/2$ is the number of tags corresponding to τ_c. Therefore,

$$\begin{aligned}
&\Pr\left[n_z > (1-\varepsilon)\tau_c\right] \\
&= \Pr\left(\frac{n_z - E[n_z]}{\sigma_{n_z}} > \frac{(1+\varepsilon)\tau_c - E[n_z]}{\sigma_{n_z}}\right) \\
&= 1 - \Phi\left(\frac{(1+\varepsilon)\tau_c - E[n_z]}{\sigma_{n_z}}\right) \\
&= \Phi\left(\frac{E[n_z] - (1+\varepsilon)\tau_c}{\sigma_{n_z}}\right) > \Phi\left(-\frac{\varepsilon\tau_c}{\sigma_{\max}}\right)
\end{aligned}$$

2 $F(z) < n/2$ and $n_z < (1 - \varepsilon)\tau_c$. Similarly, we have $E[n_z] < \tau_c$ and

$$\begin{aligned}
&\Pr\left[n_z < (1-\varepsilon)\tau_c\right] \\
&= \Pr\left[\frac{n_z - E[n_z]}{\sigma_{n_z}} < \frac{(1+\varepsilon)\tau_c - E[n_z]}{\sigma_{n_z}}\right] \\
&= \Phi\left(\frac{(1+\varepsilon)\tau_c - E[n_z]}{\sigma_{n_z}}\right) > \Phi\left(-\frac{\varepsilon\tau_c}{\sigma_{\max}}\right)
\end{aligned}$$

Since the two events are mutually exclusive, the proof is complete.

Lemma 2: *Suppose the current iteration of the while loop has candidate median x such that F(x) = n/2, then the algorithm terminates at current iteration with probability* $2\Phi\left(\frac{\in \tau_c}{\sigma_{\tau_c}}\right) - 1$.

Proof: Note that assumption $F(x) = n/2$ implies $E[n_c] = \tau_c$. Therefore,

$$\begin{aligned}
&\Pr\left[(1-\varepsilon)\tau_c \le n_c \le (1+\varepsilon)\tau_c\right] \\
&= \Pr\left(\frac{-\varepsilon\tau_c}{\sigma_{n_c}}\right) \le \frac{n_c - \tau_c}{\sigma_{n_c}} \le \frac{\varepsilon\tau_c}{\sigma_{n_c}} \\
&= 2\Phi\left(\frac{\varepsilon\tau_c}{\sigma\tau_c}\right) - 1
\end{aligned}$$

where the last equality comes from $\sigma_{n_c} = \sigma_{\tau_c}$.

Combining the above two lemmas, we have the following theorem.

Theorem 1: *Algorithm 1 finds the median with probability at least 1 – β log L where L is the length of the tag value range and*

$$\beta = \max\left\{2 - 2\Phi\left(\frac{\varepsilon\tau_c}{\sigma_{\tau_c}}\right), \Phi\left(\frac{\varepsilon\tau_c}{\sigma_{\max}}\right)\right\}.$$

Proof: Each iteration involves two desired events, a correct turn, and a correct termination. No matter what the ground-truth situation is, the undesired event of each iteration happens with probability at most β, according to Lemmas 1 and 2. Since there are at most log L iterations, the undesired event for the whole procedure happens with probability at most β log L by union bound. The proof is now complete.

It should be noted that the bound in Theorem 1 is independent of exact tag value distribution, and it is a loose bound in some situations. In the next subsection, we provide a tight performance guarantee that depends on tag value distribution. For the consumed time, note that there are at most log L iterations and each iteration takes a time interval proportional to the frame length f. Thus, the algorithm finishes within at most f log L time slots.

5.2 Probability of finding approximate median

Assuming that the underlying tag values are known, we can derive a tight bound. Recall that $F(x)$ is the number of RFID tags with values less than x, and n_x is the random variable indicating the estimated collision slots for $F(x)$. With a little abuse of notation, we refer to the expectation and standard deviation of n_x as μ_x and σ_x, respectively (instead of μ_{n_x} and σ_{n_x}). It should be noted that μ_x and σ_x are not random variables. They are determined by $F(x)$ and frame length. With the assumption that tag values are known, we can compute μ_x and σ_x for each x.

Define ξ as the event that the algorithm outputs a x with $(1-\varepsilon)\tau_c \le \mu_x \le (1+\varepsilon)\tau_c$. This event is desired, and we want to bound its probability. It contains the following special event η: the algorithm terminates correctly once the queried x is within the desired range.

We will compute the probability of η. The computation is still challenging due to the inherent dependency among consecutive iterations.

Consider the iteration that should not terminate.

Lemma 3: *Suppose x is the candidate median. If $\mu_x > (1+\varepsilon)\tau_c$, then the current iteration takes a correct turn with probability* $\Phi\left(\frac{\mu_x-(1+\epsilon)\tau_c}{\sigma_x}\right)$.

If $\mu_x < (1-\varepsilon)\tau_c$, then the current iteration takes a correct turn with probability $\Phi\left(\frac{(1-\epsilon)\tau_c-\mu_x}{\sigma_x}\right)$.

Proof: Note that n_x follows the normal distribution with expectation μ_x and variance σ_x^2. For the case $\mu_x > (1+\varepsilon)\tau_c$, the current iteration takes a correct turn if and only if $\mu_x > (1+\varepsilon)\tau_c$. We have $\Pr[n_x > (1+\varepsilon)\tau_c] = \Phi\left(\frac{\mu_x-(1+\epsilon)\tau_c}{\sigma_x}\right)$. For the case $\mu_x < (1-\varepsilon)\tau_c$, the current iteration takes a correct turn if and only if $\mu_x < (1-\varepsilon)\tau_c$, which happens with probability $\Phi\left(\frac{(1-\epsilon)\tau_c-\mu_x}{\sigma_x}\right)$.

Consider the iteration that should terminate.

Lemma 4: *Suppose x is the candidate median. If $(1-\varepsilon)\tau_c \le \mu_x \le (1+\varepsilon)\tau_c$, then the current iteration terminates with probability* $\Phi\left(\frac{(1+\epsilon)\tau_c-\mu_x}{\sigma_x}\right)-\Phi\left(\frac{(1-\epsilon)\tau_c-\mu_x}{\sigma_x}\right)$.

Proof: The theorem can be proved by noting that n_x follows the normal distribution with expectation μ_x and variance σ_x^2.

We now study the property of the event η.

Lemma 5: *For a given set of RFID values, the event η happens if and only if (a) every iteration takes a correct turn if the queried x is not in the desired range, and (b) the algorithm terminates if the queried x is in the desired range.*

Proof: It is straightforward to see the '$\Leftarrow$'. We show the '$\Rightarrow$' part. We only consider about (a), since (b) follows immediately from the definition of η. For (a), if at some iteration, the queried x is not in the desired range and either the algorithm terminates or the algorithm takes a wrong turn, then either the algorithm terminates prematurely so that η does not happen, or the algorithm cannot proceed to query a x in the desired range since all values in the desired range are excluded due to a wrong turn.

This lemma implies that, for a given set of RFID values, the number of iterations involved in event η is fixed and is not a random variable. More importantly, there is actually a unique sequence of queried x, as described by the following corollary.

Corollary 1: *Let R be the range $[(1-\varepsilon)\tau_c,(1+\varepsilon)\tau_c]$. For a given set of RFID values, there is a unique iteration number t and a unique sequence of queried x_i such that (a) $\mu_{x_i} \notin R$ for $i = 1, \ldots, t-1$ and $\mu_{x_i} \in R$; (b) at iteration $i \le t-1$, the algorithm queries x_i, and takes a correct turn; at iteration t, the algorithm terminates.*

According to Corollary 1, we may assume without loss of generality that the number of iterations is t. Let ζ_i be the event that iteration i takes a correct turn, and let ρ be the event that iteration t terminates. We can rewrite η in terms of ζ_i and ρ:

$$\eta = \zeta_1 \wedge \zeta_2 \wedge \cdots \wedge \zeta_{t-1} \wedge \rho \tag{3}$$

Note that ζ_i are not mutually independent. This is because, the previous ζ_i will inuence the current queried x, which can in turn influence the probability of the current ζ_i. In fact, this is the only causality link among ζ_i.

Theorem 2:

$$\Pr[\eta] = \prod_{i\in I^+}\Phi\left(\frac{\mu_{x_i}-(1+\varepsilon)\tau_c}{\sigma_{x_i}}\right)\cdot \prod_{i\in I^-}\Phi\left(\frac{(1-\varepsilon)\tau_c-\mu_{x_i}}{\sigma_{x_i}}\right)\cdot \left(\Phi\left(\frac{(1+\varepsilon)\tau_c-\mu_{x_t}}{\sigma_{x_t}}\right)-\Phi\left(\frac{(1-\varepsilon)\tau_c-\mu_{x_t}}{\sigma_{x_t}}\right)\right).$$

where x_i are defined in Corollary 1, $I^+ = \{i \le t-1 \mid \mu_{x_i} > (1+\varepsilon)\tau_c\}$ and $I^- > \{i \le t-1 \mid \mu_{x_i} < (1-\varepsilon)\tau_c\}$.

Proof: From equation (3), we have

$$\begin{aligned}\Pr[\eta] &= \Pr[\zeta_1 \wedge \zeta_2 \wedge \cdots \wedge \zeta_{t-1} \wedge \rho] \\ &= \Pr[\zeta_1]\cdot\Pr[\zeta_2][\zeta_1]\cdot\Pr[\zeta_3|\zeta_1,\zeta_2]\cdots \\ &\quad \Pr[\zeta_{t-1}|\zeta_1,\zeta_2,\ldots,\zeta_{t-2}]\cdot\Pr[\rho|\zeta_1,\zeta_2,\ldots,\zeta_{t-1}]\end{aligned}.$$

Let X_i be the random variable indicating the queried x at iteration i. According to Corollary 1, if all ζ_j happen for $j < i$, then $X_i = x_i$. We can check that the inverse is also true. Thus, for $i \ge 2$, we have

$$\Pr[\zeta_i|\zeta_1,\ldots,\zeta_{i-1}] = \Pr[\zeta_i|X_i = x_i].$$

Additionally, note that $X_1 = x_1$ is always true so that $\Pr[\zeta_1] = \Pr[\zeta_1|X_1 = x_1]$.

Plugging them into the expression of $\Pr[\eta]$ yields

$$\Pr[\eta] = \Pr\left[\zeta_1 \middle| X_1 = x_1\right] \cdot \Pr\left[\zeta_2 \middle| X_2 = x_2\right] \cdots \Pr\left[\zeta_{t-1} \middle| X_{t-1} = x_{t-1}\right] \cdot \Pr\left[\rho \middle| X_t = x_t\right].$$

Rearranging the terms and using the results of Lemmas 3 and 4 can prove the theorem.

By Theorem 2, given a set of RFID values and the parameter settings of the algorithm, we can compute the probability of η. We emphasise that this probability depends on the distribution of the RFID values. For example, we can easily construct a problem instance such that $\Gamma^+ = \varnothing$, and $\Gamma^- = \varnothing$, resulting in $\Pr[\eta] = \Phi\left(\frac{(1+\varepsilon)\tau_c - \mu_{x_1}}{\sigma_{x_1}}\right) - \Phi\left(\frac{(1-\varepsilon)\tau_c - \mu_{x_1}}{\sigma_{x_1}}\right)$

Recall that ξ is the event that the algorithm outputs a x with $(1-\varepsilon)\tau_c \le \mu_x \le (1+\varepsilon)\tau_c$.

Theorem 3: *For any parameter setting of the algorithm, it holds that*

$$\Pr[\xi] \ge \Pr[\eta].$$

and there exist problem instances such that $\Pr[\xi] = \Pr[\eta]$.

Proof: It is easy to see that $\Pr[\xi] \ge \Pr[\eta]$. We show that the equality holds in some cases, and we construct one such case. Construct the set of RFID tag values such that ξ is equivalent to η, i.e. if the algorithm does not terminate for the first time it encounters a desired x, then it will never encounter one again. Let the range R correspond to only one tag value, which can be done by setting enough number of tags with value equal to the median. In this case, any other tag value x will result in $n_x \in R$. Thus, ξ is equivalent to η.

Let μ_x^- be the inverse function of μ_x, i.e. $x = \mu_{\mu_x}^-$. Then ξ is equivalent to $\mu_{(1-\varepsilon)\tau_c}^- \le x \le \mu_{(1+\varepsilon)\tau_c}^-$ where x is the output of the algorithm.

Theorem 4: *Let x be the output of the algorithm. Then*

$$\Pr\left[\mu_{(1-\varepsilon)\tau_c}^- \le x \le \mu_{(1+\varepsilon)\tau_c}^-\right] \ge \Pr[\eta]$$

where Pr[η] can be computed by Theorem 2.

In this theorem, the performance guarantees, both $\mu_{(1-\varepsilon)\tau_c}^- \le x \le \mu_{(1+\varepsilon)\tau_c}^-$ and $\Pr[\eta]$, depend on tag value distribution. This follows with intuition in that different distributions require different number of iterations of binary search. This fact also motivates our simulation methodology where we try different tag value distributions.

6 Continuous median update

In this section, we consider the scenario that median value should be updated continuously. For example, in temperature monitoring applications, we may request the median every few minutes. In this case, the straightforward solution is to treat every median query request separately and for each request use either a tag identification algorithm or our previous median estimation algorithm. Here, we show a more efficient algorithm to solve the problem.

In practice, most of the sensor readings do not change drastically in a short period of time. Thus, the median does not change much. We can leverage this observation to design an algorithm that incrementally searches for the exact median.

Formally, there are n tags 1, 2, …, n and tag i has value x_{ij} at time slot j. Let X_j be the multi-set (A multi-set is a generalisation of the notion of set where elements are allowed to appear more than once.) of tag values at time slot j, i.e. $X_j = \{x_{ij} \mid i = 1,\ldots, n\}$. Denote by m_j the median value of X_j. Our task is to compute m_j for $j \ge 1$. In the rest of this section, unless otherwise specified, we assume the number n is odd so that the median value m_j is from X_j. We defer the case when n is even to the end of this section.

The basic idea of our algorithm is stated in the following theorem, which is straightforward to prove.

Theorem 5: *Given a real number x and an n-element multi-set X where n is an odd number, denote by $L_X(x)$ the number of elements in X that are no larger than x, and $S_X(x)$ the number of elements in X that are strictly less than x. Then x is the median of X if and only if (a) $L_X(x) > n/2$ and (b) $S_X(x) < n/2$.*

Theorem 5 can be used as a statement for median verification. Whenever we have a candidate median x, we can check whether both $L_X(x) > n/2$ and $S_X(x) < n/2$ hold. If not, then we adjust the candidate median until we find a value x that can satisfy both $L_X(x) > n/2$ and $S_X(x) < n/2$. The exact adjustment is based on the following theorem.

Theorem 6: *Let X be a multi-set with elements $x_1, x_2, \ldots, x_n$ sorted in ascending order where n is odd. Given a real number x, let*

$$x_1 \le x_2 \le \cdots x_t \le x < x_{t+1} \le x_{t+1} \le \cdots x_n.$$

If $L_X(x) > n/2$, then the first x_i in the list $x_t, x_{t-1}, \ldots, x_1$ that satisfies $S_X(x_i) < n/2$ is equal to the median; if $L_X(x) > n/2$, then the first x_i in the list $x_{t+1}, x_{t+2}, \ldots, x_n$ that satisfies $L_X(x_i) > n/2$ is equal to the median.

Based on Theorem 6, we maintain a variable $L_X(m)$ for each median computation request where m is the computed median. Whenever a new median computation request comes, we treat the previous median as a candidate median, test the condition $L_X(x) > n/2$, and then find the 'first' satisfying value by querying tags with values around the candidate median.

Specifically, suppose in the previous median computation, the tag value multi-set is X^- and the median is m^-. For a new median computation request, suppose the tag value multi-set is X. First, we treat m^- as a candidate median and compute $L_X(m^-)$.

This is done by using tag identification algorithm twice. At one time, we identify the number of tags whose previous values (in X^-) are $\le m^-$ but their current values (in X) are $> m^-$. Let this number be n_1. At the second time, we identify the number of tags whose previous values are $> m^-$ but their current values are $\le m^-$. Let this number be n_2. Trivially, $L_X\left(m^-\right)=L_{X^-}\left(m^-\right)-n_1+n_2$. Since $L_{X^-}\left(m^-\right)$ has been computed in last request, the consumed time by this step is the time to identify n_1+n_2 tags.

Second, find median by another round of tag identification. There are two cases depending on whether $L_X\left(m^-\right)>n/2$. (a) If $L_X\left(m^-\right)>n/2$, then the current median m is no larger than m^-. Suppose the median change is bounded by Δ (we consider the case when Δ is unknown later), then $m\in\left[m^--\Delta,m^-\right]$. Using tag identification algorithm, we identify the tags with values in this range and collect their values. Let the collected values be $t_1, t_2, \ldots$ in descending order, and let i be the first index such that $S_X(t_i)<n/2$, then the median is equal to t_i and $L_X\left(m\right)=S_X\left(t_i\right)+n_{t_i}$ where n_{t_i} is the number of collected values equal to t_i. During this process, note that $S_X\left(t_i\right)=L_X\left(m^-\right)-n'_{t_i}$ where n'_{t_i} is the number of collected values larger than or equal to t_i. (b) If $L_X\left(m^-\right)<n/2$, then the current median m is strictly larger than m^-. Again, suppose the median change is bounded by Δ, then $m\in\left[m^-,m^-+\Delta\right]$. We identify the set of tags with values in this range and collect their values. Let the collected values be $t_1, t_2,\ldots$ in ascending order, and let i be the first index such that $L_X\left(t_i\right)>n/2$, then the median is equal to t_i and $L_X\left(t_i\right)=L_X\left(m^-\right)+n''_{t_i}$ where n''_{t_i} is the number of collected values smaller than or equal to t_i.

It should be noted that the tag identification procedures mentioned above do not need to identify all tags. Instead, they only need to identify a fraction of tags that have corresponding property. This is in contrast to the naive scheme that identifies all tags.

Algorithm 2 describes how we compute a new median based on information of the last median. The time consuming steps are at Lines 2, 3, 6, and 16. In situations where tag values change smoothly, the number of tags appearing in either of the identification processes is small; thus only a small fraction of time is necessary.

As mentioned before, there are two issues left for further discussion. The first issue is that when Δ is unknown. In this case, we may set Δ as any value and iteratively increase Δ until we can find desired i. The resulting tag identification process does not necessarily take longer time than the case when Δ is known. The second issue is that when the number of tags is even. In this case, Algorithm 2 may not output the exact median. However, since the number of tags in our situation is large, one tag value bias can barely influence the value of the median. Additionally, it is possible to find the exact median by modifying the refining step of the algorithm (Lines 5–24). We omit the tedious modifications.

Algorithm 2: Continuous Median update

Input: m^-, L^-, n, Δ;
`(/* the previous median; the previous number of tags with values no larger than the median, i.e.` $L_X^-(m^-)$`; total number of tags; the maximum median change */)`
Output: m, L; current median and $L_X(m)$

1 **begin**
2 identify the set of tags whose previous values are no larger than m^- but their current values are larger than m^-, set n_1 as the number of such tags;
3 identify the set of tags whose previous values are larger than m^- but their current values are no larger than m^-, set n^2 as the number of such tags;
4 $L' \leftarrow L^- - n_1 + n_2$;
5 **if** $L' > n/2$ **then**
6 $T \leftarrow$ the multi-set of tag values who are in the range $[m^- - \Delta, m^-]$ (using tag identification algorithm);
7 sort T in descending order;
8 $i \leftarrow 1$;
9 $s \leftarrow n/2$;
10 **while** $s \ge n/2$ **do**
11 $n'_{t_i} \leftarrow$ the number of values in T larger than or equal to t_i;
12 $s \leftarrow L' - n'_{t_i}$;
13 $i \leftarrow i + 1$;
14 $m \leftarrow t_i$;
15 $n'_{t_i} \leftarrow$ the number of values in T equal to t_i;
16 $L \leftarrow s + n_{t_i}$;
17 **else**
18 $T \leftarrow$ the multi-set of tag values who are in the range $(m^-, m^- + \Delta]$ (using tag identification algorithm);
19 sort T in ascending order;
20 $i \leftarrow 1$;
21 $s \leftarrow n/2$;
22 **while** $s \le n/2$ **do**
23 $n''_{t_i} \leftarrow$ the number of values in T smaller than or equal to t_i;
24 $s \leftarrow L' + n''_{t_i}$;
25 $i \leftarrow i + 1$;
26 $m \leftarrow t_i$;
27 $L \leftarrow s$;

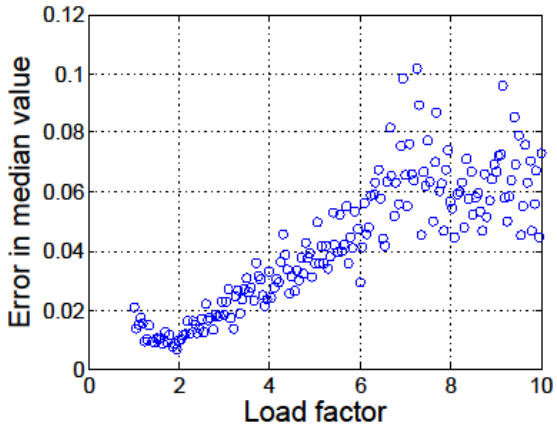
7 Performance analysis

We evaluate the performance of the median estimation algorithm by simulation, and by comparing the algorithm to

a tag identification algorithm formally introduced below. The metric used is the time taken to correctly evaluate the median and the accuracy. We evaluated our algorithms for different statistical distributions of the tag value data y.

The simulation parameters were set as follows. We set $\varepsilon = 0.15$, we also varied the number of tags n from 1000 to 50,000 tags. We carried out simulations on the uniform, normal and Weibull tag set distributions. The distribution parameters were set as follows: (a) uniform distribution: values drawn from the interval (0,1). (b) Normal distribution: values drawn from the standard normal distribution. (c) Weibull distribution: we set the scale parameter as 1 and the shape parameter as 1. We did not set the maximum and the minimum values. In Figure 2 we show the effect of load factor (ρ) on the error. We varied the load factor from 1 to 10 in steps of 0.05 and we fixed the number of RFID tags at 5000 tags. To reduce the variance of the experiment we conducted 100 simulations and then took the average. We used this simulation to set the load factor to 2.44. The frame length was then set to n/2.44.

Figure 2 Error versus load factor for 5000 RFID tags (see online version for colours)



The baseline tag identification method was used for comparison purposes.

7.1 Baseline median estimation algorithm

The baseline tag median estimation protocol collects all the tag IDs, and then queries each tag separately for its data value. This identification-like protocol gives the exact median value. In tag identification protocols each tag must be uniquely identified, so if there is a collision a tag will have to re-transmit its ID several times before the reader can correctly identify it. In DFSA (Cha and Kim, 2006) and EDFSA (Lee et al., 2005) each tag transmits its ID 2.72 times. So the protocol execution time is $2.72 \times M \times (L + S)$ (Li et al., 2010a; Li et al., 2010b), where L is a slot length which allows the transmission of a long response slot, S is a short slot which allows the transmission of only < 10 bits of information and M is the length of a slot which can transmit a tag ID.

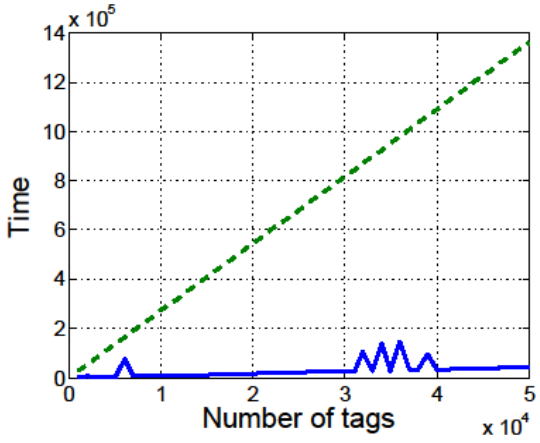
According to the specifications of the Phillips I-Code system $S = 0.4$ ms, $L = 2.4$ ms and $M = 0.8$ ms when the gap between transmissions are included.

7.2 Running time

We evaluate the time efficiency of the median binary search algorithm. We carried out simulations on the uniform, normal and Weibull distributions. The required threshold for the TCS scheme is set based on the total number of tags. The running time consists of two components: the time taken to check the threshold and the time (iterations) taken for binary search. In this paper, we use the fact that a long slot is equal to five short slots $L = 5S$ (Sheng et al., 2008).

Figures 3–5 show the results. In all the figures the solid line represents the time needed to converge for our algorithm while the dotted line represents the time needed by an identification algorithm for the median selection. We can see that the running time of our algorithm is significantly less than the tag identification algorithm. Additionally, for all three distributions, the running time of our algorithm increases slowly with the increase of number of tags. We find that the uniform distribution exhibited the best time performance. This is due to the fact that the tags are evenly distributed. The Weibull distribution has less improvement due to its fat tail.

Figure 3 Execution time for the uniform distribution in time units, the error was set to ε = 0:15 and the maximum and minimum tag values (E_{max}, E_{min}) were generated according to the parameters of the uniform distribution (see online version for colours)



7.3 Accuracy

In our simulation, the error is in the range of [0, 1]. We calculated the error as Absolute error = $|(M - M_{est})/(E_{max} - E_{min})|$, where M is the real median and M_{est} is the estimated median. The closer the error is to one, the larger the error. The binary median search algorithm was tested with different tag set distributions via simulation; we observed that all the tests showed a high level of accuracy with high probability.

In Figures 6–8, we analysed the absolute error for the different tag set cardinalities and found that the Weibull distribution exhibited a very large variance in the absolute error.

Figure 4 Execution time for the normal distribution in time units, the error was set to ε = 0:15 and the maximum and minimum tag values (E_{max}, E_{min}) were generated according to the parameters of the normal distribution (see online version for colours)

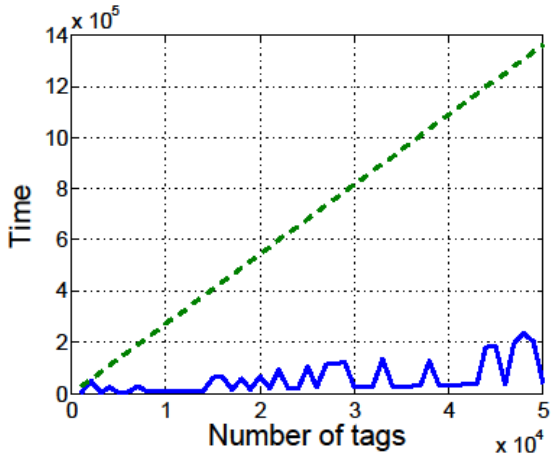


Figure 5 Execution time for the Weibull distribution in time units, the error was set to ε = 0:15 and the maximum and minimum tag values (E_{max}, E_{min}) were generated according to the parameters of the Weibull distribution (see online version for colours)

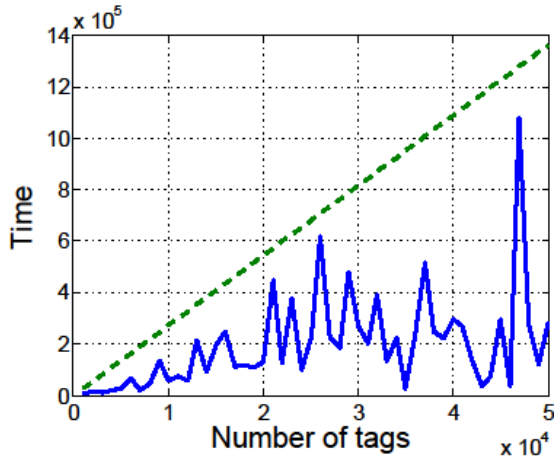


Figure 6 Absolute error in median value for the uniform distribution (see online version for colours)

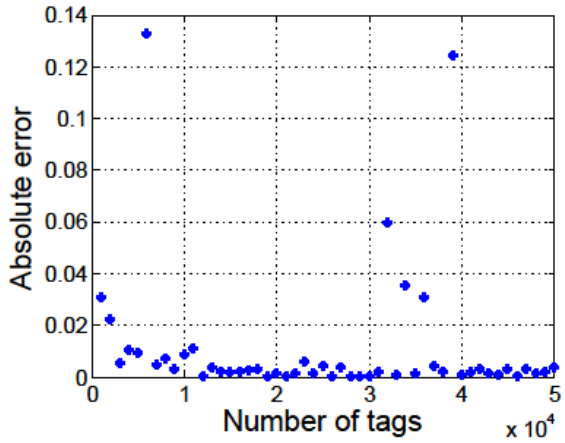


Figure 7 Absolute error in median value for the normal distribution (see online version for colours)

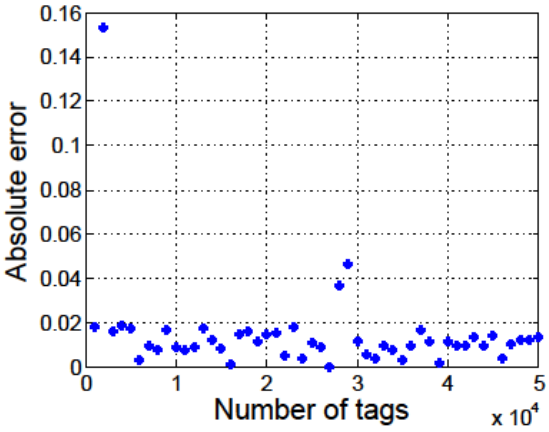
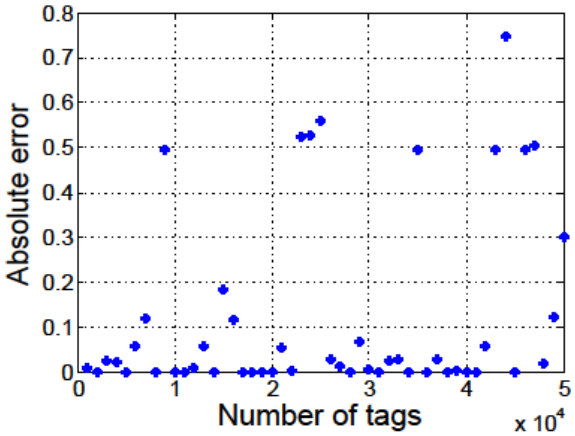


Figure 8 Absolute error in median value for the Weibull distribution (see online version for colours)



7.4 Continuous median update

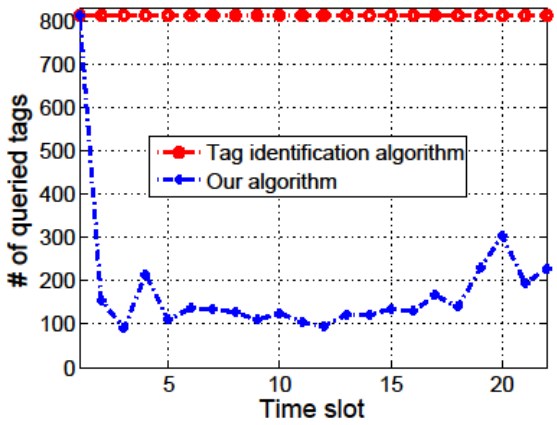
To study the effectiveness of Algorithm 2, we conduct trace-driven analysis. We use a temperature data set collected from 54 sensors, deployed in the Intel Berkeley Research lab between on 28th February and 5th April 2004 (Intel, 2004). Since the number of sensors in the data set is small, we emulated virtual sensors by generating between 12 and 15 sensors spatially around each real sensor.

These emulated sensors have values drawn uniformly and randomly from the real sensor's value range. The resulting data set consists of 813 sensors. Those sensors are considered as RFID tags. Our goal is to compute the median at each time slot. The naive tag identification algorithm requires identifying 813 tags in each time slot. Our algorithm, on the other hand, only queries all tags in the first time slot for initialisation, and then adjusts the median in other time slots by Algorithm 2. We mainly study the number of tags to be queried by our algorithm.

Figure 9 shows the improvement over the naïve scheme. We can see that, in most time slots, our algorithm only needs to query less than 25% of all tags, while the naive scheme needs to identify all of them. This improvement

does not sacrifice any accuracy, since both algorithms give accurate medians. In RFID systems, the number of queried tags is proportional to the time duration and energy consumption of identification process. Thus, our algorithm can save both time and energy.

Figure 9 Required number of queried tags (see online version for colours)



8 Conclusion

In this paper, we studied the median estimation problem in a large-scale sensor RFID system. This implementation can be used with real-time data collected from tags in a system. We showed that the median can be estimated efficiently by probing the RFID tags using binary search. We also proposed an algorithm for continuous median update. Based on the observation that the sensor readings do not change rapidly, our algorithm can perform much better than the naïve algorithm that queries all tags.

Acknowledgements

The authors would like to thank all the reviewers for their helpful comments. This project was supported in part by US National Science Foundation grants CAREER Award CNS-0747108 and CNS-1117412, the Faculty of Mathematical Sciences, University Of Khartoum, China NSF grants (60825205, 61073152, and 61133006) and China 973 project (2012CB316200).

References

Bonuccelli, M., Lonetti, F. and Martelli, F. (2006) 'Tree slotted aloha: a new protocol for tag identification in RFID networks', *Proceedings of the 2006 International Symposium on World of Wireless, Mobile and Multimedia Networks (WOWMOM'06)*, Buffalo-Niagara Falls, NY, USA, pp.603–608.

Buettner, M., Greenstein, B., Sample, A. and Smith, J. (2008) 'Revisiting smart dust with RFID sensor networks', *Proceedings 7th ACM Workshop on Hot Topics in Networks (HotNets-VII)*, Alberta, Canada.

Cha, J. and Kim, J. (2005) 'Novel anti-collision algorithms for fast object identification in RFID system', *Proceedings of the 11th International Conference on Parallel and Distributed Systems - Workshops (ICPADS 05)*, July, Fuduoka, Japan, pp.63–67.

Cha, J.R. and Kim, J.H. (2006) 'Dynamic framed slotted ALOHA algorithms using fast tag estimation method for RFID systems', *Proceedings of IEEE CCNC, 2006*, January, Las Vegas, Nevada, USA.

Chen, S., Zhang, M. and Xiao, B. (2010) 'Efficient information collection protocols for sensor-augmented RFID networks', *Proceedings of the 29th conference on Information communications (INFOCOM'10)*, March, San Diego, CA, USA.

Greenwald, M. and Khanna, S. (2001) 'Space-efficient online computation of quantile summaries', *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (SIGMOD'01)*, ACM, New York, NY, USA, pp.58–66.

Han, H., Sheng, B., Tan, C., Li, Q., Mao, W. and Lu, S. (2010) 'Counting RFID tags efficiently and anonymously', *Proceedings of the 29th conference on Information communications (INFOCOM'10)*, March, San Diego, CA, USA, pp.1028–1036.

Intel (2004, April) *Intel Lab Data*. Available online at: http://berkeley.intel-research.net/labdata/

Kodialam, M. and Nandagopal, T. (2006) 'Fast and reliable estimation schemes in RFID systems', *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking (MobiCom'06)*, September, Los Angeles, CA, USA, pp.23–29.

Law, C., Lee, K. and Siu, K-Y. (2000) 'Efficient memory less protocol for tag identification', *Proceedings of the 1st International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, ACM, New York, NY, USA, pp.75–84.

Lee, S-R., Joo, S-D. and Lee, C-W. (2005) 'An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification', *Proceedings of the 2nd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS'05)*, July, San Diego, CA, USA, pp.166–174.

Li, T., Wu, S., Chen, S. and Yang, M. (2010a) 'Energy efficient algorithms for the RFID estimation problem', *Proceedings of the 29th Conference on Information Communications (INFOCOM'10)*, March, San Diego, CA, USA, pp.19.

Li, T., Chen, S. and Ling, Y. (2010b) 'Identifying the missing tags in a large RFID system', *Proceedings of the 16th Annual International Conference on Mobile Computing and Networking (MobiCom'10)*, September, Chicago, IL, USA.

Liu, H., Bolic, M., Nayak, A. and Stojmenovic, I. (2008) 'Taxonomy and challenges of the integration of RFID and wireless sensor networks', *IEEE Network*, Vol. 22, No. 6, pp.26–32.

Madden, S., Franklin, M., Hellerstein, J. and Hong, W. (2002a) 'TAG: a tiny aggregation service for ad-hoc sensor networks', *SIGOPS Operating Systems Review*, Vol. 36, pp.131–146.

Madden, S., Szewczyk, R., Franklin, M. and Culler, D. (2002b) 'Supporting aggregate queries over ad-hoc efficient median estimation for large scale sensor RFID systems 13 wireless sensor networks', *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'02)*, IEEE Computer Society, Washington, DC, USA, pp.49–58.

Micic, A., Nayak, A., Simplot-Ryl, D. and Stojmenovic, I. (2005) 'A hybrid randomized protocol for RFID tag identification', *The 1st IEEE International Workshop on Next Generation Wireless Networks, WoNGeN'05*, December, Goa, India.

Miura, M., Ito, S., Takatsuka, R., Sugihara, T. and Kunifuji, S. (2009) 'An empirical study of an RFID mat sensor system in a group home', *Journal of Networks*, Vol. 4, No. 2.

Myung, J. and Lee, W. (2005) 'Adaptive binary splitting: a RFID tag collision arbitration protocol for tag identification', *Proceedings of the IEEE International Conference on Broadband Networks (BROADNETS 2005)*, October, Boston, USA, pp.375–383.

Patt-Shamir, B. (2004) 'A note on efficient aggregate queries in sensor networks', *Proceedings of the23rd Annual ACM Symposium on Principles of Distributed Computing (PODC'04)*, St. John's, Newfoundland, Canada, pp.283–289.

Qian, C., Ngan, H. and Liu, Y. (2008) 'Cardinality estimation for large-scale RFID systems', *Proceedings of the 2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PERCOM'08)*, March, Hong Kong, China, pp.30–39.

Qiao, Y., Chen, S., Li, T. and Chen, S. (2011) 'Energy-efficient polling protocols in RFID systems', *Proceedings of the 12th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'11)*, May, Paris, France, pp.1–9.

Ren, S., Li, Q., Wang, H., Chen, X. and Zhang, X. (2005) 'Analyzing object detection quality under probabilistic coverage in sensor networks', *Proceedings of the 13th International Conference on Quality of Service (IWQoS'05)*, Springer-Verlag, Berlin, Heidelberg, pp.107–122.

Roy, S., Jandhyala, V., Smith, J., Wetherall, D., Otis, B., Chakraborty, R., Buettner, M., Yeager, D., Ko, Y. and Sample, A. (2010) 'RFID: from supply chains to sensor nets', *Proceedings of the IEEE*, Vol. 98, No. 9, pp.1583–1592.

Sheng, B, Tan, C., Li, Q. and Mao, W. (2008) 'Finding popular categories for RFID tags', *Proceedings of the 9th ACM International Symposium on Mobile ad Hoc Networking and Computing (MobiHoc'08)*, May, Hong Kong, China, pp.159–168.

Sheng, B., Li, Q. and Mao, W. (2010) 'Efficient continuous scanning in RFID systems', *Proceedings of the 29th Conference on Information Communications (INFOCOM'10)*, March, San Diego, CA, USA, pp.1010–1018.

Singh, M. and Prasanna, V. (2003) 'Optimal energy balanced algorithm for selection in single hop sensor network', *Proceedings of the IEEE International Workshop on Sensor Network Protocols and Applications (SNPA) ICC*, May, IEEE Computer Society, Washington, DC, USA, pp.9–18.

Smith, J., Sample, A., Powledge, P., Mamishev, A. and Roy, S. (2006) 'A wirelessly powered platform for sensing and computation', *Proceedings of Ubicomp 2006: 8th International Conference on Ubiquitous Computing*, September, Orange Country, CA, USA, pp.495–506.

Tan, C., Sheng, B. and Li, Q. (2008) 'How to monitor for missing RFID tags', *Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems (ICDCS'08)*, June, Beijing, China, pp.295–302.

Wang, H., Tan, C-C. and Li, Q. (2008) 'Snoogle: a search engine for the physical world', *Proceedings of 27th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2008*,, April, Phoenix, AZ, USA, pp.1382–1390.

Xie, L., Sheng, B., Tan, C., Li, Q. and Chen, D. (2010) 'Efficient tag identification in mobile RFID systems', *Proceedings of the 29th Conference on Information Communications (INFOCOM'10)*, March, San Diego, CA, USA, pp.1001–1009.

Xing, K., Cheng, X. and Ding, M. (2005) 'Safety warning based on highway sensor networks', *Proceedings of IEEE Wireless Communication and Networking Conference (WCNC)*, March, New Orleans, LA, pp.2355–2361.

Xuan, D., Bettati, R. and Zhao, W. (2002) 'A gateway-based defense system for distributed dos attacks in high-speed networks', *IEEE Transactions on Systems, Man, and Cybernetics*.

Zhang, Y., Yang, L. and Chen, J. (2009) 'Integrated RFID and sensor networks: architectures and applications', *RFID and Sensor Networks: Architectures, Protocols, Security, and Integrations*, CRC Press, p.517.