Sink-Free Audio-on-Demand over Wireless Sensor Networks

Hanhua Chen, *Member, IEEE*, Hai Jin, *Senior Member, IEEE*, and Lingchao Guo

Abstract—Audio represents one of the most appealing yet least exploited modalities in wireless sensor networks, due to the potentially extremely large data volumes and limited wireless capacity. Therefore, how to effectively collect audio sensing information remains a challenging problem. In this paper, we propose a new paradigm of audio information collection based on the concept of audio-on-demand. We consider a sink-free environment targeting for disaster management, where audio chunks are stored inside the network for retrieval. The difficulty is to guarantee a high search success rate without infrastructure support. To solve the problem, we design a novel replication algorithm that deploys an optimal number of $O(\sqrt{n})$ replicas across the sensor network. We prove the optimality of the energy consumption of the algorithm. We implement a sink-free audio-on-demand (SAoD) WSN system, and conduct extensive simulations to evaluate the performance and efficiency of our design. The experimental results show that our design can provide satisfactory quality of audio-on-demand service with short startup latency and slight playback jitter. Extensive simulation results show that this design achieves a search success rate of 98 percent while reducing the search energy consumption by an order of magnitude compared with existing schemes.

Index Terms—Audio-on-demand, wireless sensor networks

1 Introduction

The emerging wireless sensor networks (WSNs) [1], [2], [3], [4] have been revolutionizing the ways of collecting information from the physical world [5], [6], [7]. The community has envisioned a large variety of applications, such as environment monitoring, scientific observation, underwater surveillance, and structural health monitoring [8], [9], [10], [11]. So far, audio represents one of the most appealing yet least exploited modalities in sensor networks, mainly because high-frequency audio sampling can produce extremely large data volumes over bandwidth-limited links.

In this paper, we investigate a new paradigm of audio services, namely audio-on-demand, in wireless sensor networks. We consider a sink/infrastructure free environment targeting for earthquake disaster management scenario, where any base station/sink could be damaged during the disaster. We call our design SAoD, a sink-free audio-on-demand WSN system. The target application is post-earthquake search and rescue, which becomes extremely significant after the hitting of recent constant violent earthquakes [12]. When an earthquake occurs, recording and storing acoustic events and providing an on-demand retrieval service are essential to the later rescue in such systems. There are two main reasons: 1) The disaster area is often disconnected from the outside world, and 2) Most of the acoustic events are recorded before rescue could take place.

Since individual sensors are limited in their effective acoustic range, networks of acoustic sensors are needed to cover a disaster area. The requirements of a reliable audio-on-demand service are threefold. 1) Acoustic events should be recorded and stored inside the network because existing infrastructure if any may be destroyed in ruinous environmental conditions. 2) Without a base station or other infrastructures, it is difficult if not impossible to efficiently locate acoustic events. 3) The non-disruptive on-demand playback of the audio anywhere in the network requires parallel data transferring and efficient buffer pre-fetching mechanism due to the limited bandwidth capacity of WSNs.

With the recent advances in NAND flash memory, new mote prototypes are now available that interface Mica-class processing and radio hardware to up to 8GB of flash memory [13]. The increasing in-network storage capability indeed makes the above store-and-fetch paradigm possible for WSNs, where the sensory data is stored inside the network and can be retrieved on-demand. However, existing infrastructure-free systems investigate only the "store" [14] side of the store-and-fetch paradigm, leaving the other side an open issue. For example, EnviroMic [15] employs a distributed balanced storage mechanism to store the high-volume sensory acoustic event data inside the network. However, the sensory data stored inside EnviroMic cannot be readily accessed before all sensors are recovered from the experiment venue.

To support retrieval, a straightforward strategy is to locate the data using flooding. The problem is that flooding produces a large amount of traffic while the search success rate cannot be guaranteed. It is natural to utilize replication strategy to improve search efficiency. However, how to achieve an optimal replication strategy with minimum

• *The authors are with the Services Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China.*
E-mail: {chenhanhua, hjin}@hust.edu.cn, glc216722@163.com.

Manuscript received 6 Oct. 2013; revised 16 June 2015; accepted 24 June 2015. Date of publication 30 June 2015; date of current version 13 Apr. 2016.
Recommended for acceptance by H. Ammari.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2015.2451643

retrieval energy consumption is not trivial for audio applications, especially under an infrastructure-free and bandwidth-limited wireless sensor network. To address this problem, in this paper we propose a probabilistic exhaustive replication strategy. We show that, by replicating both data replicas and query replicas uniformly at random across the network, the proposed strategy guarantees a high search success rate with a determined lower bound while the replication cost is $O(\sqrt{n})$, where n is the network size.

Based on the efficient replication strategy, we implement the buffer pre-fetching module in SAoD system, a real world sink-free audio-on-demand system over WSNs. SAoD uses time-division cooperative recording technique for collecting audio sensory data, where multiple sensors detecting the same acoustic event form a group with an elected leader to assign the time slots. Thus, the nodes in the group record the acoustic event cooperatively in turn. Consequently, different chunks of an acoustic event are naturally recorded and stored in the form of time addressable audio chunks by different nodes around the source of the acoustic event in different time slots.

Instead of replicating raw audio chunks, SAoD encodes the metadata of chunks residing on a node into a Bloom filter (BF) and replicates the BF. A Bloom filter is a space-efficient probabilistic data structure for representing a set. A BF can support testing whether an element is a member of a set. The metadata of each chunk includes the time addressable identifier and the location where the chunk is stored. Thus, we can compress the set of chunks recorded by a node into a space-efficient bit vector. By replicating the bit vectors across the network, SAoD further reduces the communication cost for the replication. During retrieval, a query for a specific chunk can be evaluated against the BFs. If matched, the raw data chunk can be obtained from the origin location.

We implement the SAoD system with 30 IRIS motes equipped with MTS310 sensor boards. The experimental results show that SAoD provides high quality audio-on-demand service with very slight playback jitter and short startup latency. Results of extensive simulations in large-scale networks show that SAoD's buffer pre-fetching can achieve guaranteed success rate while reducing the energy cost by one order of magnitude compared to existing scheme.

The main contributions of this work are threefold:

- We design and implement a real audio-on-demand system over WSNs and evaluate the performance using 30 nodes.
- We propose a novel replication strategy which guarantees a high chunk search success rate with a determined lower bound at greatly reduced communication cost.
- We further reduce the communication cost of replication by encoding the chunk metadata using Bloom filter.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the probabilistic exhaustive replication model. Section 4 describes the detailed design of SAoD. Section 5 presents the system implementation and experimental results. Section 6 evaluates the design in large-scale environment using extensive simulations. Section 7 concludes the paper with possible extensions.

2 Related Work

Most of the existing work on audio sensor networks focuses on how to efficiently transfer the sensory data back to a base station (sink) [16] by either using online stream compression [17] or customizing high bandwidth sensor prototype [18].

In [16], Allen et al. deployed 16 sensor nodes on the upper flanks of the Reventador active volcano to collect the audio data. The nodes form a multi-hop routing topology and relay data via a long-distance radio modem to the observatory. They used the formed wireless sensor network to continuously sample acoustic data at the active volcano. A data collection protocol is designed to transfer continuously sampled acoustic data to the base station.

Soroush et al. [19] tackled the problem of online compression of data streams in a resource-constrained network environment, where traditional compression techniques are not applicable. Particularly, they aimed at fast piecewise linear approximation methods with quality guarantee. They studied two versions of the problem which explore quality guarantees in different forms. For the error bounded piecewise linear approximation problem, they designed a fast online algorithms running in linear time complexity and requiring a constant space cost.

Li et al. [18] designed and implemented a high bandwidth system for *quality-aware voice streaming* (QVS) in WSNs. QVS is built upon a new sensor hardware platform for high-rate audio communication. In their design they used the transceiver Chipcon CC1100 which has a 64 bytes FIFO buffer and maximum data rate of 500 kbps. They used dynamic voice compression and duplication adaptation, and distributed stream admission control techniques. Their experimental results show that QVS delivers satisfactory voice streaming quality.

The above existing work on audio services over WSNs assumes the existence of a base station [20]. The infrastructure-based schemes, however, may be problematic when applied to the audio-on-demand application addressed in this paper, because a user may hope to access only limited audio events of interest from any place in the WSN just as audio events are recorded everywhere. Transferring all the sensory audio data to a single base station is costly and infeasible. Moreover, a base station is a centralized point of failure. The failure of a base station in a disaster will paralyze the whole system.

To the best of our knowledge, we are the first to design and implement an audio-on-demand system over WSNs. The proposed retrieval scheme based on replication is different from existing flooding and a geographic hash table (GHT) [21]. Flooding does not guarantee the success rate without exhaustively searching all the sensor nodes. The GHT partitions the name space over the nodes and has good success rate for key-value search, while it suffers from the problem of exact match. Furthermore, although the problem of node failure for a key in GHT can be alleviated by using more than one node for a key, GHT cannot survive the catastrophic failure. However, the case that a large number of nodes may be destroyed is norm rather than the exception in the target application in this work.

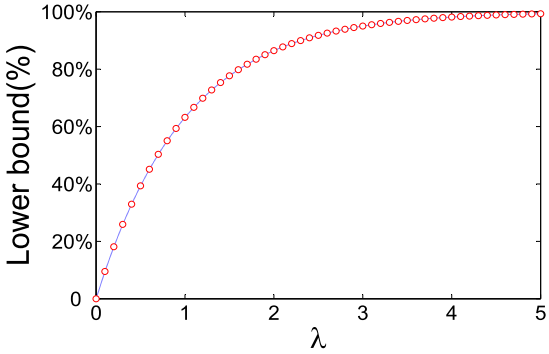


Fig. 1. Lower bounds of success rate.

3 Model

The conventional wisdom for searching in a wireless sensor network without infrastructure support, such as flooding algorithm, is to replicate the query onto many nodes, which then evaluate the query on the data they store. While this works, it does not scale: to guarantee the search success rate, exhaustive search would require replicating the query onto every node. SAoD takes a different approach. It performs exhaustive search probabilistically. Before going to the details of the design of SAoD, we specify the model of chunk search in SAoD.

3.1 Probabilistic Exhaustive Search Model

Let the replicas of a data be red balls and the replicas of a related query green ones. Query matching in a WSN is similar to a procedure of tossing the sets of green and red balls into a set of bins. Intuitively, with random casting, if the number of red balls or green balls is large enough, a collision of two kinds of balls in some bin can be guaranteed with high probability.

Theorem 1. *Given a sensor network with n nodes, if we replicate r copies of a data and g copies of related queries, both uniformly at random in the network, the probability that at least one sensor node has both a data replica and a query replica is not less than* $1 - \mathrm{e}^{-\frac{rg}{n}}$.

Proof. If all the replicas are distributed uniformly at random, for any given replica (data or query), the probability that it falls in a given sensor node is $\frac{1}{n}$, while the probability that it is not in the given sensor node is $1 - \frac{1}{n}$.

After all the number of r replicas are distributed uniformly at random, for a given sensor node, the probability that the node has no data replicas is $(1 - \frac{1}{n})^r$. Thus, for a given query replica, if it is deployed in a sensor node, the probability that the node has no data replicas is $(1 - \frac{1}{n})^r$.

Because all the query replicas are distributed independently, the probability that none of the nodes with at least one query replica having any data replicas is

$$\left(\left(1 - \frac{1}{n}\right)^r\right)^g = \left(1 - \frac{1}{n}\right)^{rg}. \tag{1}$$

Thus, the probability that at least one node has both a data replica and a query replica can be computed by

$$p = 1 - \left(1 - \frac{1}{n}\right)^{rg}. \tag{2}$$

For the given function $f(n) = (1 - \frac{1}{n})^{-n}$, we have

$$\begin{aligned}
&\lim_{n\to\infty} f(n) = \lim_{n\to\infty} \left(1 - \frac{1}{n}\right)^{-n} \\
&= \lim_{n\to\infty} \left[\left(1 + \frac{1}{n-1}\right)^{n-1} \cdot \frac{n}{n-1}\right] \\
&= \lim_{n\to\infty} \left(1 + \frac{1}{n-1}\right)^{n-1} \cdot \lim_{n\to\infty} \frac{n}{n-1} = \mathrm{e} \cdot \lim_{n\to\infty} \frac{n}{n-1} = \mathrm{e}.
\end{aligned} \tag{3}$$

We first prove that $f(n)$ is a decreasing function on n, i.e., $f(n) \geq f(n+1)$.

According to the inequality of arithmetic and geometric means we can find that for any list of n positive real numbers $x_1, x_2, \ldots, x_n$, we have

$$\frac{x_1 + x_2 + \ldots + x_n}{n} \geq (x_1 \cdot x_2 \cdot \cdots \cdot x_n)^{\frac{1}{n}}.$$

Based on the above inequality, let's consider the following relations first:

$$\begin{aligned}
f^{-\frac{1}{n+1}}(n) &= \left(\left(1 - \frac{1}{n}\right)^{-n}\right)^{-\frac{1}{n+1}} \\
&= \left(\left(\frac{n-1}{n}\right)^n \cdot 1\right)^{\frac{1}{n+1}} \leq \frac{1}{n+1} \cdot \left(\frac{n-1}{n} \cdot n + 1\right) \\
&= \frac{n}{n+1} = 1 - \frac{1}{n+1} \\
&= \left(\left(1 - \frac{1}{n+1}\right)^{-(n+1)}\right)^{-\frac{1}{n+1}} = f^{-\frac{1}{n+1}}(n+1).
\end{aligned}$$

Thus, we can achieve $f(n) \geq f(n+1)$.

According to Eq. (3) we have $(1 - \frac{1}{n})^{-n} \geq \mathrm{e}$. After performing some simple operations, we can have

$$\begin{aligned}
1 - \frac{1}{n} &\leq \mathrm{e}^{-\frac{1}{n}} \\
\left(1 - \frac{1}{n}\right)^{rg} &\leq \mathrm{e}^{-\frac{rg}{n}} \\
1 - \left(1 - \frac{1}{n}\right)^{rg} &\geq 1 - \mathrm{e}^{-\frac{rg}{n}}.
\end{aligned}$$

Thus $p \geq 1 - \mathrm{e}^{-\frac{rg}{n}}$ is proved. □

Theorem 1 shows that given a number of r data replicas and a number of g query replicas, if SAoD deploys all the replicas uniformly at random across the sensor network, the lower bound of the search success rate of a chunk is $1 - \mathrm{e}^{-\frac{rg}{n}}$. Given a network size, the lower bound of search success rate is determined by the product of numbers of data replicas and query replicas. For simplicity, we use λ to denote $\frac{rg}{n}$. Fig. 1 shows how the lower bound of success rate changes with λ. It shows that when $\lambda = 4$, the success rate is larger than 98.17 percent, while when $\lambda = 8$, the success rate is larger than 99.97 percent. Notice that only the product rg matters; if the product does not change, one can increase g while decreasing r and retain the same probabilistic bound.

It is not difficult to see that the traditional flooding scheme need to exhaustively replicate a number of n queries

to guarantee the search success rate. By leveraging our probabilistic exhaustive search model, we can avoid a significantly large number of replicas while sacrificing very slight search success rate. In practice, it is infeasible to conduct an exhaustive flooding scheme due to the prohibitively expensive communication cost. Thus our scheme, which grantees the success rate with high probability, is more practical in real systems.

It is clear that the model works when replicas are deployed uniformly at random across the sensor network. We will show how we sample random node and deploy replicas efficiently across the network in detail in Section 4.4.

3.2 Minimizing the Communication Cost

Given a fixed bound of success rate, it is important to save replicating cost by controlling the number of replicas to deploy. In Theorem 2 we show that the optimal number of replicas is determined by $\sqrt{n}$, where n is the network size.

Theorem 2. *Given the size of the data replica is S_d while the size of the query replica is S_q, the optimal numbers of data replicas and query replicas are in proportion to $\sqrt{n}$, where n is the size of the network.*

Proof. Let $\mathrm{e}^{-\frac{r \cdot g}{n}} = 1 - p = \mathrm{e}^{-c^2}$, we then have $r \cdot g = n \cdot c^2$, where $c = \sqrt{-\ln(1-p)}$ and p is the search success rate defined in Eq. (2).

Communication cost tends to dominate the energy consumption on sensor node in conventional wisdom. Intuitively, the communication cost is determined by the number of replicas to deploy. The optimal numbers of replicas should be achieved with the minimized cost

$$\begin{cases} \min\ Cost = r \cdot S_d + g \cdot S_q \\ s.t \quad r \cdot g = n \cdot c^2. \end{cases} \tag{4}$$

We can see that the minimized cost can be achieved only when $r \cdot S_d = g \cdot S_q$. By minimizing the above cost, we achieve the optimal values of r and g

$$\begin{cases} r = c\sqrt{n \cdot \frac{S_q}{S_d}}, \\ g = c\sqrt{n \cdot \frac{S_d}{S_q}}. \end{cases} \tag{5}$$

The result shows that the optimal numbers of data/query replicas are in proportion to $\sqrt{n}$. Thus proved. □

4 System Design

In this section, we present the design of SAoD. We first briefly describe how the audio events are recorded and stored. Then, we introduce how SAoD replicates the metadata of audio chunks in the compressed form. Finally, we describe the replicating and chunk discovery scheme.

4.1 Cooperative Recording

SAoD utilizes the cooperative recording scheme proposed in [15] to split the task of recording an acoustic event into units divided by time slots among multiple sensors around the acoustic event. When multiple nodes detect the same acoustic event simultaneously, they form a group. The members of the group coordinate to elect a leader, who assigns recording tasks to individual members in turn. Thus, an acoustic event file, a_i, is naturally segmented on time. Audio is partitioned into chunks of uniform unit to make the file addressable on time. Each chunk has a fixed playing time equal to the length of a time slot. Due to the limited memory capacity of sensor nodes [22] and the heavy loads caused by the analog-to-digital converter (ADC) sampling, choosing a proper size of the slot is not trivial. In Section 5, we will show how we obtain the optimal setting of time slots in SAoD system in greater detail.

By using the cooperative recording technique, the chunks of an acoustic event file can naturally be collected by different sensor nodes and stored in a distributed way. Without the cooperative recording technique, when an acoustic event occurs, a straightforward design will let all the nodes which detect the event perform the ADC sampling to record the event. Thus, the same event will be recorded and stored by multiple sensor nodes, making the scheme costly in both energy and storage consumptions. The time-division cooperative recording design can make audio chunks time addressable. Therefore in each time slot, one chuck will be recorded by one assigned node. Such a design greatly reduces the redundancy of sampling and storage. It also effectively achieves a better load balance. During retrieval, different chunks can be fetched from different nodes.

4.2 Metadata Encoding

Instead of replicating the raw audio chunks, we use Bloom filters [23] to encode the metadata of the chunks residing on a node. By replicating the metadata in a space-efficient way, SAoD greatly reduces the communication cost.

A Bloom filter is essentially a bit vector $bitvec\_m$ with m bits, initially all set to 0, which facilitates membership test to a finite set $S = \{x_1, x_2, \ldots, x_v\}$ of v elements from a universe U. It uses a set of k uniform and independent hash functions $\{h_1, h_2, \ldots, h_k\}$ to map the universe U to the bit address space $[1, m]$. For each element x belonging to S, the $h_i(x)$th bits are set to 1 for $1 \le i \le k$. To check whether or not an item y is in S, we check whether all the $h_i(y)$th bits are set to 1. If not, y clearly is not a member of S. If all the $h_i(y)$th bits are set to 1, y is in S with high probability which can be controlled by the parameters of BF.

After all v elements of S are hashed and inserted into the BF, the probability that a specific bit of $bitvec\_m$ is still 0 is

$$p = \left(1 - \frac{1}{m}\right)^{kv} \approx e^{-\frac{kv}{m}}. \tag{6}$$

The probability of a false positive after n elements inserted in the $bitvec\_m$ is the probability that a new element is not in S, but can be separately hashed by the k hash functions to a number k of "1" bits of the $bitvec\_m$

$$f = (1-p)^k = (1 - e^{-\frac{kv}{m}})^k. \tag{7}$$

Given an optimal choice of k hash functions, the false positive rate f can be minimized when $k = \frac{m}{v} lg2$ and the lower bound of the false positive rate is

$$f_{min} = 0.6185^{\frac{m}{v}}. \tag{8}$$

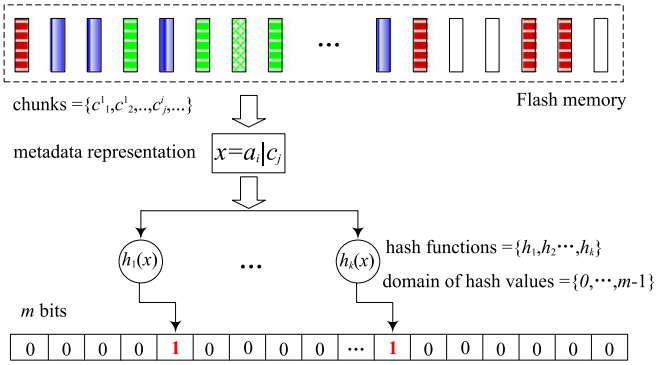


Fig. 2. Metadata encoding.

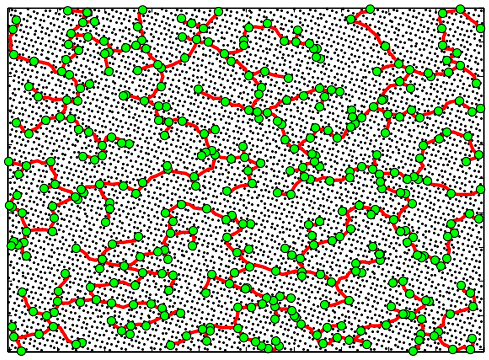


Fig. 3. Replica deployment.

As aforementioned, in SAoD each sensor stores chunks of different acoustic events in the equipped flash memory. Fig. 2 illustrates the process that a node in SAoD encodes the metadata of its chunks. On the top of the figure, the bars with the same color denote the chunks of the same acoustic event file. For simplicity, we use c_j^i to denote the jth chunk (denoted by c_j) of the ith acoustic event file (denoted by a_i). SAoD obtains a global unique time addressable identifier of chunk c_j^i by combining the identifier of the acoustic event file and the sequence number (time address) of the chunk, namely $a_i|c_j$.

By inserting all the identifiers of the chunks in the flash memory into the Bloom filter, a node in SAoD achieves a space-efficient bit vector for representing its chunks, which supports membership queries. The size of the Bloom filter can be determined by $m \geq \frac{kv}{\ln 2}$, where k is the number of hash functions used in the Bloom filter and v is the maximum number of chunks limited by the capacity of the flash memory.

4.3 Network Size Estimation

As aforementioned in Theorem 2, the optimal numbers of replicas are determined by the network size n. Without a base station, it is difficult to obtain such statistics [24]. To solve this problem, SAoD utilizes a variant of the gossip algorithm first proposed in [25] to estimate the network size. The robust algorithm enables every node to quickly collect the global statistics in the network. The main idea of the method is as follows. Initially, each node does the following experiment: it flips a coin up to l times and counts the number of times the head appears before the first tail. It saves this count r in a bit vector (all bits initially set to 0) by setting the rth (counting from the right end) bit of the vector to 1. Then the nodes in the system network perform a gossip algorithm. During each round of gossip, each node randomly selects a neighbor and sends its bit vector to the selected neighbor. The node receiving the bit vector performs a bitwise-or operation between the received bit vector and its local bit vector, and replaces the local bit vector with the resulting bit vector. The robust gossip scheme leads the computation of aggregated information to converge exponentially: after $O(\log n)$ rounds of gossip, all nodes will get the estimated network size with high probability [25]. Moreover, the statistical value of n is roughly $\frac{2^{t-1}}{0.77351}$ with high probability, where t is the position of the first "1" bit in the bit vector counting from the left end.

4.4 Replica Deployment

As shown in Theorem 1, the optimal replication model requires that the replicas are deployed at random. In SAoD deployment, we assume that the sensors are deployed uniformly at random in a specified area. After SAoD computes the optimal number of replicas to deploy according to Theorem 2, it samples the optimal number of random locations in the fixed area and deploys the replicas of the metadata bit vector to the nodes nearest to the locations. Before casting all the replicas to the set of selected nodes, it forms a minimal spanning tree among the randomly sampled/computed nodes. Here, the logical neighbors in the minimal spanning tree communicate with each other using the underlying geographic routing algorithms [26]. The replicas are deployed using multicast along the minimal spanning tree. Fig. 3 illustrates an example of the process of replica deployment across a rectangle with 4×10^4 sensors deployed uniformly at random. The bright green nodes are the ones with replicas deployed through the minimal spanning tree. Note, the location information of the source node is attached with the Bloom filter for chunk downloading during retrieval. Algorithm 1 describes the metadata replication strategy in detail.

Algorithm 1. Metadata Replication

Require: $EstimatedNetworkSize = n$ is achieved;
1: create an empty bit vector with m bits for node p, BF_p;
2: **for all** chunks in the local flash memory of node p **do**
3: insert the identifiers of the chunks into BF_p by setting the hashing functions $\{h_j(\cdot), 1 \leq j \leq k\}$;
4: **end for**
5: compute r, the optimal number of replicas according to Theorem 2 using the gathered statistics n;
6: multicast BF_p attached with $location_p$, the location of node p, through the minimal spanning tree formed with the nodes nearest to the number of n_0 sampled locations;
7: **return**

4.5 Query Evaluation

During playback, the SAoD buffer pre-fetching module issues queries asking for the set of missing chunks in the pre-fetching window. SAoD replicates the query replicas to an optimal number of randomly and uniformly sampled nodes in the similar way shown in Algorithm 1. Every receiving node checks all the Bloom filters replicated locally. The member verification mechanism of the Bloom filter can effectively enable chunk discovery. If any chunk is tested to

be contained in a Bloom filter, the replica is assumed to be the desired result with high probability. The $location_p$ attached with the chunk identifier is then returned for the buffer pre-fetching module to access the chunk. Note, due to the false positives of Bloom filters, the returned results may contain undesired ones with very low probability. This may lead to a slight decrease of the precision of the final results while keeping the recall rate guaranteed. If a false positive result is returned, SAoD filters it by simply sending the query to the node to evaluate the query locally. If no matching results, no actual chunks will be transferred. In Section 6, we will further discuss how to adjust the Bloom filter settings to achieve the tradeoff between the precision and the communication cost. It is not difficult to see that the chunk discovery algorithm can achieve better recall and latency than expected by Theorem 1, because the un-designated nodes on a path forwarding the queries can also provide data if they have. Algorithm 2 describes the query evaluation process in detail.

Algorithm 2. Query Evaluation

```
R ← ∅;
for all BFs replicated in a node do
  for all desired chunk t in the query Q do
    if ∀(j)(1 ≤ j ≤ k)s.t.BF_p[h_j(t)] = 1 then
      R ← R ∪ {(location_p, t)};
    end if
  end for
end for
return R.
```

5 System Experiments

We have implemented SAoD system over a real testbed. The wireless sensor testbed consists of 30 IRIS motes all equipped with a MTS310 sensor board and running TinyOS [27], [28]. The IRIS mote is a widely available, low-cost and low-power platform. It basically inherits the MICAz mote with the improvement of a larger RAM (from 4 KB to 8 KB) and a longer communication range. Each IRIS mote has a 512 KB flash memory for storing audio data. Although the flash memory is limited, it is sufficient to evaluate the performance of SAoD.

Fig. 4 illustrates the prototype system, deployed uniformly in a 5×6 grid with 30 nodes in the open field of a local sports stadium. The distance between two neighboring nodes is three meters. Thus the prototype system covers an area of 180 m^2. To set up the testbed, each node is equipped with a microphone capable of acquiring acoustic events. An audio source is broadcasted from a random location within the grid area. The sensors record the acoustic events using the cooperative recording technique in a time-division manner. All the chunks are stored into various nodes in the network. A node (i.e., IRIS mote) is placed randomly in the area to issue queries and gather acoustic chunks to feed the buffer of the player in the connected laptop. In the experiments, we let sensors start to work on the time of deployment [29]. In the real applications, to save energy the sensor nodes of SAoD are usually kept in the sleep mode [30]. In emergency of the target applications, nodes can be triggered by the sensor's accelerometer module [31] because the vibration of the sensor is likely to be the indication of the happening of earthquake.

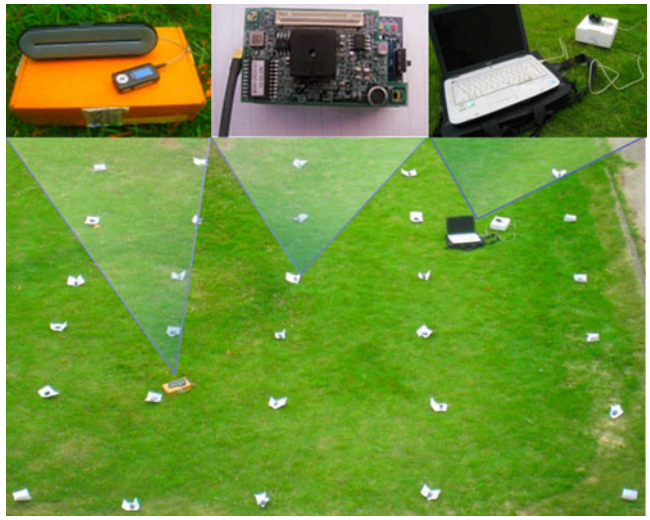


Fig. 4. System deployment.

As aforementioned in Section 4.1, the key parameter in the system design is the length of time slot during cooperative recording. In the system experiments, we found that choosing the length of the time slot is highly related to the memory size of IRIS mote and the sampling frequency of the sensor board. The IRIS MTS310 motes used in this experiment are equipped with 8 KB memory. We vary the sampling frequency and observed when sampling frequency is over 7 KHz the failures surge in flash write. This is because sampling and flash write cannot perform concurrently on IRIS MTS310. The flash write operation cannot start before the sampling operation finishes. Thus, a sensor node need to store the data in the memory during the time slot assigned to it. After the end of the time slot, the data will be written to the flash. Thus, we need to fix the bound of time slot with an acceptable sampling frequency to avoid memory overflow.

In the experiment, we set the sampling frequency to 7 KHz by setting the MTS310 sensor board's parameter SAMPLE_INTERVAL to $143\,\mu$s. At the same time, the length of time slot is set to 700 ms. With this setting, the program memory is almost full at 7,800 B, which pushes the performance examination close to the system limits. In fact, the sampling frequency of the MTS310 sensor board can be configured up to 16 KHz. Due to the limit of memory size in the experiment testbed, we limit the sampling frequency at 7 KHz in the experiments. In practice, a system designer can choose motes configured with larger memory capacity and adjust the sampling frequency to improve the quality.

In the experiment, the chunks are pre-fetched from the network by using the query evaluation algorithm in Section 4 and fed to the buffer for the player. In the prototype, the client node performs the following scheduling algorithm. Every time slot, the scheduler checks the next β chunks. If all the chunks are available, it starts/continues playing while keeping fetching the chunks in the pre-fetching window with the size of $\alpha + \beta$ chunks. If not, it stops and fetches the missing chunks in the next $\alpha + \beta$ chunks. Once the next β chunks are all available, it resumes the playback. For smooth playback, the player

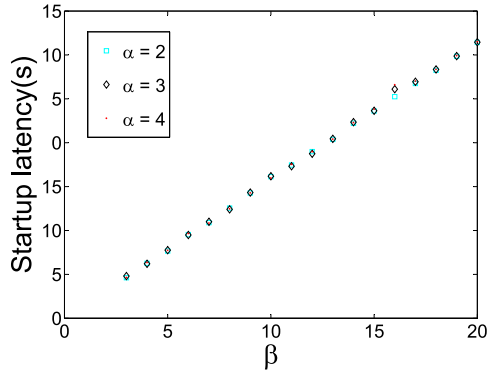


Fig. 5. Startup latency.

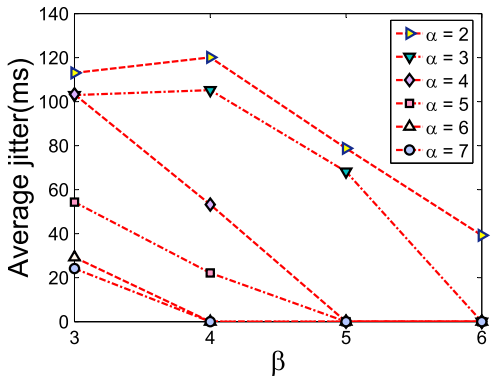


Fig. 6. Average jitter changes with β.

pulls the fetched chunks from the head of buffer list at the playback speed. It is clear that the buffer settings influence the system performance and quality of service. We vary the parameters α and β in the scheduler to optimize the buffer settings.

In the experiments we mainly examine the quality of service of SAoD. We consider three metrics, startup latency, jitter, and playback continuity.

Startup latency quantifies the waiting time from the audio event is requested until the playback starts. In SAoD design, startup latency is an important metric because in the target applications such as earthquake rescue, a quick response from the system is critical. In the experiments, we adjust the buffer settings and examine the startup latency.

Jitter quantifies the waiting time during playback. In the target earthquake rescue application, an acceptable playback quality with slight jitter is important for estimating the situation and determining the rescue plan for different cases. We can also use the waiting time per chunk to compute the average jitter. As aforementioned in the above scheduling algorithm, if any chunk is missing in the next β-chunk window in the buffer, the playhead stops for buffer pre-fetching until the next $\alpha + \beta$ chunks are all available.

Playback continuity is defined as the total delay time divided by the number of played chunks. Here, delays include startup latency and all playback jitters. We ignore the cases requiring the seek and pause operations in this prototype

$$Continuity = \frac{startup\ latency + playback\ jitters}{number\ of\ played\ chunks}. \quad (9)$$

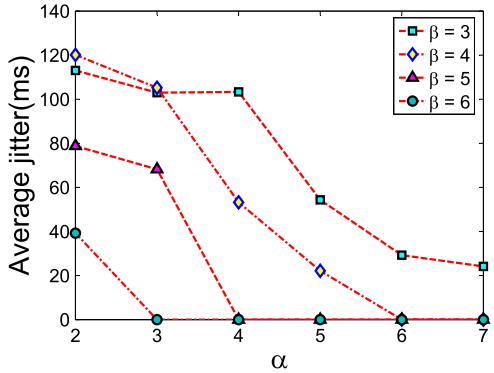


Fig. 7. Average jitter changes with α.

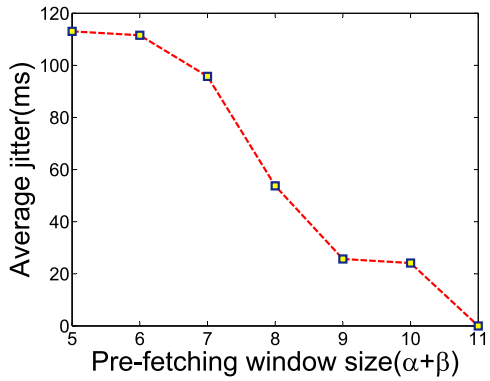


Fig. 8. Average jitter changes with α+β.

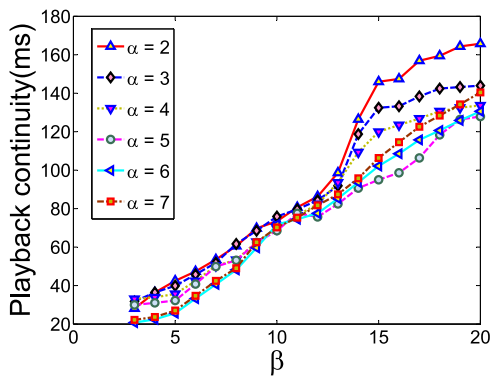


Fig. 9. Continuity changes with β.

The unit is seconds per chunk played. Lower continuity values are better, representing better playback continuity.

Fig. 5 shows that the startup latency increases nearly linearly when the parameter β increases, while the startup latency has very slight change with different settings of parameter α. Thus, we prefer to choose a smaller β for a short startup latency. However, Fig. 6 indicates when β is less than four, the average playback jitter is much worse and such worse jitter can be reduced by increasing α.

As we can see in Fig. 8 when the entire pre-fetching window size increases to 11, the average playback jitter decreases to zero. Fig. 7 shows that the optimal settings of the buffer parameter are $\beta = 5$ and $\alpha = 4$, which can achieve the minimized buffer size with no playback jitter and short startup latency.

Figs. 9 and 10 indicate that the playback continuity is dominated by the parameter β. This reveals that most of the user

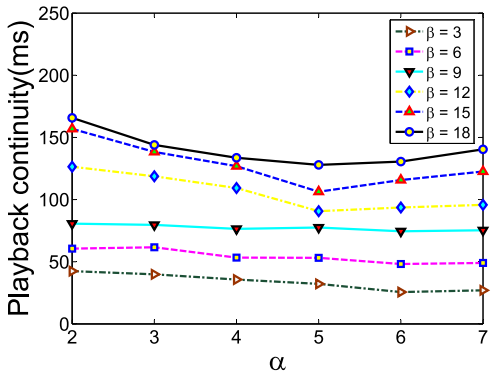


Fig. 10. Continuity changes with α.

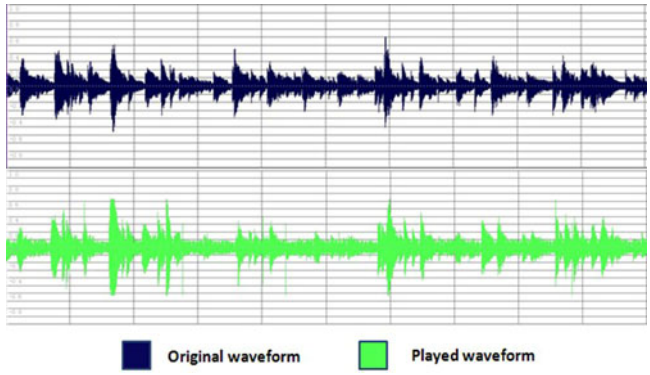


Fig. 11. Playback waveform.

waiting time in SAoD system is startup latency. Due to the slight jitter, startup latency is the most important factor and have much heavier weight in the continuity computation.

Fig. 9 shows that with a higher setting of β ($\beta \geq 10$), the best choice of α is a moderate value $\alpha = 5$. A smaller α, such as $\alpha = 2$, incurs frequent short stops. On the other hand, a larger α, such as $\alpha = 7$, leads to fewer but longer stops.

Based on the above analysis, in the SAoD system we set $\beta = 5$ and $\alpha = 4$. The experimental results reveal that playback is quite fluent with very slight jitter and short latency.

Fig. 11 shows the waveform of a traditional Chinese music named "Liangzhu" played on guitar, which is used in the experiment. The frequency of a guitar ranges from 70 Hz to 1 KHz, which is quite similar with the frequency range of the human voice. Moreover, the audio frequency in a single guitar music, such as "Liangzhu", varies more significantly than that of the voice from a single specific person. Using this method we can evaluate the performance of this design for different audio frequencies. Thus, we can examine the performance of this design for different audio frequency. The higher part of Fig. 11 is the waveform of the original music before transmission. The lower part is the waveform played by SAoD. Result shows how the waveform of SAoD reproduces that of the original one.

6 Simulations

In this section, we evaluate the design using simulations in large-scale networks. We first introduce our simulation methodology and the setups. Then we specify the metrics used for the evaluation. At last we present the results comparing with existing schemes. In the comparison we use the flooding scheme as the baseline, since it is extensively used in the literatures. To make a more fair comparison, in the baseline scheme we also replicate the data in the same way as SAoD.

6.1 Simulation Setups

In the simulation, we put $10,000$ nodes on the grid of a 1,000 m $\times$ 1,000 m rectangle and then perturb each point by a random shift following a normal distribution with $\sigma = 3$, which has been widely treated as an approximation for the manual deployment of sensor nodes in many literatures [32]. The energy dissipation for sensor communications follows the widely used path loss model [33], [34].

The communication radius of each node is set to 20 m. We randomly deploy a number of $10\sqrt{n}$ audio sources. Each source can be detected within a distance following a normal distribution with $\mu = 19$ and $\sigma = 2$. The length (ms) of an audio event follows a normal distribution with $\mu = 2,000$ and $\sigma = 500$. We simulate the cooperative recording by randomly assigning the audio chunk among the nodes within the radius the audio source can be sensed. According to the de facto standard used by TinyOS [27], every message in our design has a limited length of 46 bytes with 28 bytes payload, 11 bytes header information and 7 bytes metadata. During the search process, we randomly select a node to issue queries.

6.2 Metrics

SAoD design considers both quality of service and system efficiency. Quality focuses on user-perceived qualities, such as chunk search latency and success rate, while efficiency focuses on resource utilization, such as energy consumptions. We describe the metrics used in the evaluation as follows.

Energy consumption. Saving energy is critical in SAoD design. Since communication tends to dominate the energy consumption on sensor node in conventional wisdom [35], we carefully examine the communication cost of this design. Following the path loss model [34], the energy spent in transmitting one message of size κ bits over a distance d is computed by: $E_{tx}(d) = \kappa\varepsilon d^{\alpha} + \kappa E_{elec}$, where $E_{elec} = 50 \times 10^{-9}$, ε is the transmitter amplifier ($\varepsilon = 10^{-11}$ for $\alpha = 2$, and $\varepsilon = 13\times 10^{-16}$ for $\alpha \geq 3$) , and α is the pass-loss exponent ($2 \leq \alpha \leq 4$). The energy spent in receiving one κ-bits message is computed by $E_{rx}(d) = \kappa E_{elec}$.

Success rate. In this design, the chunk retrieval success rate is critical to the quality of audio services. Success rate is defined as the fraction of queries which have returned matched chunks if exist. Ensuring a high success rate is the main challenge in the design of a sink-free audio-on-demand WSN.

Energy efficiency. We define the energy efficiency as the ratio of energy consumption to the success rate. Compared with the energy consumption and the success rate, the energy efficiency is a more fair metric. Lower energy efficiencies represent greater success rate with less energy consumed.

Search latency. A short search latency is always desirable in the audio-on-demand service in WSNs [36], [37]. It is defined as the duration between the time when a query is issued and the result is returned. It is the sum of the latency during each

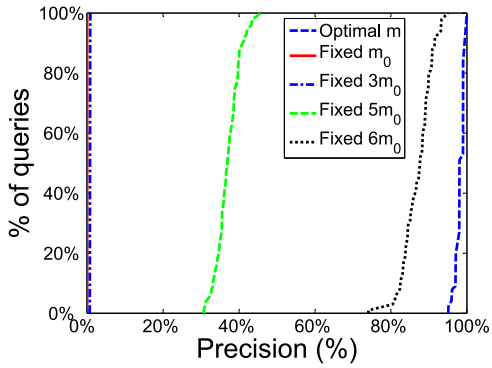


Fig. 12. Optimal settings of m in Bloom filters.

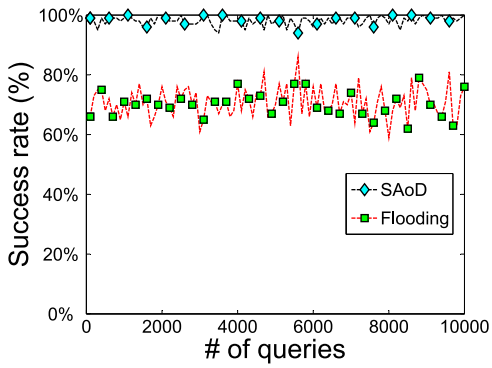


Fig. 14. Success rate.

hop in the underlying wireless links. The time required to transfer a packet includes the propagation time as determined by the distance between the underlying nodes and the transmission time. The transmission time of a κ-bits message is computed by: $ETX * \frac{\kappa}{B}$, where B is the bandwidth (bps) of a wireless link. ETX is the number of expected transmissions, which incorporate the possibility of re-transmission caused by packet loss following the packet loss model by Jerry et al. [38]. We ignore the backoff time in ETX for simplicity, although the backup time can be estimated easily. In the SAoD design, the playhead moves ahead only when the buffer is full. Fast locating and pre-fetching the chunks are quite important to the system [39], [40].

Precision. Precision captures the fraction of relevant chunks in the returned results. Due to the false positives of a Bloom filter, some returned results may not be the desired results. If a false positive result is returned, SAoD needs to evaluate the query by sending the query to the node holding the raw data.

6.3 Results

In the baseline flooding scheme we set the value of *Time to Live* (*TTL*) to 25. When simulating SAoD scheme, we set $\lambda = 4$ .To make the comparison fair, the flooding is performed with the same data replica deployment as that of SAoD.

It is clear that the settings of the Bloom filter parameters including m and k are critical to the performance of our scheme. We first vary the setting of k from 1 to 30 and find that when $k = 19$ the false positive of a Bloom filter is quite acceptable. Fig. 13 plots how the precision changes with the parameter k in detail.

Since different nodes in the network may have different numbers of chunks, the setting of m for the Bloom filter should vary according to this number. We set the value of m by: $m = \frac{vk}{lg2}$, which achieves the lowest false positive of Bloom filters, and compare the precision with the case with fixed m_0 setting throughout the network, where m_0 is the average value of the different m values chosen above. This means that using fixed m_0 throughout the network and using optimized m values in different nodes will have the same replication energy cost. Results in Fig. 12 show that with the optimal m value in each nodes, we can achieve an average precision of 98.35 percent while the precision with a uniformed m_0 is only 0.016 percent.

From the above results, we set k to 19 and compute an optimal m in the following experiments.

Fig. 14 shows the search success rate. It indicates that the SAoD scheme greatly outperforms the baseline scheme. The average search success rate of flooding is 70.8 percent, while the search success rate of SAoD is 98.1 percent. The results show that the SAoD achieves perfect chunk discovery success rate, which is critical for our target application. In practice, we can adjust the parameter λ to tradeoff between search success rate and resource consumption.

Fig. 15 shows how the search success rate changes with the network size. It shows that the search success rate of SAoD remains perfect when the network size increases, while the success rate of flooding decreases greatly. This reveals that the SAoD scheme is more scalable than the flooding scheme.

Fig. 16 examines the energy consumption of SAoD. The lowest energy consumption of the flooding scheme is about

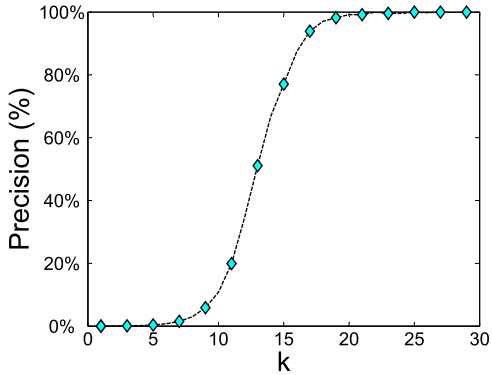


Fig. 13. Optimal setting of k in Bloom filters.

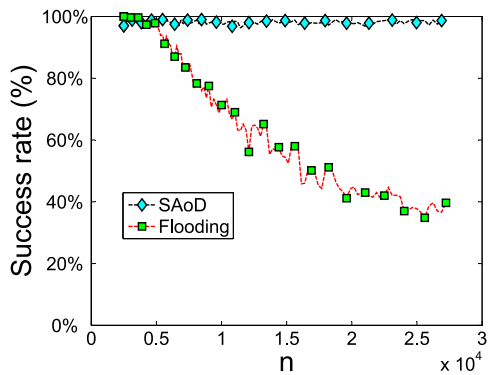


Fig. 15. Success rate changes with network size.

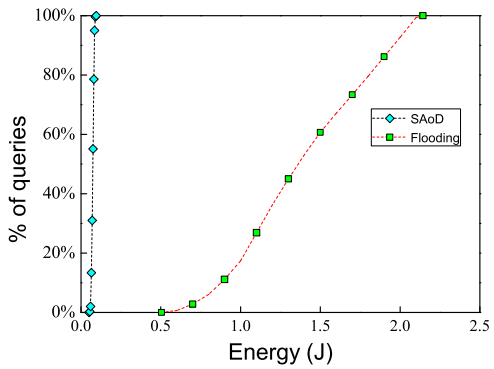


Fig. 16. Energy consumption.

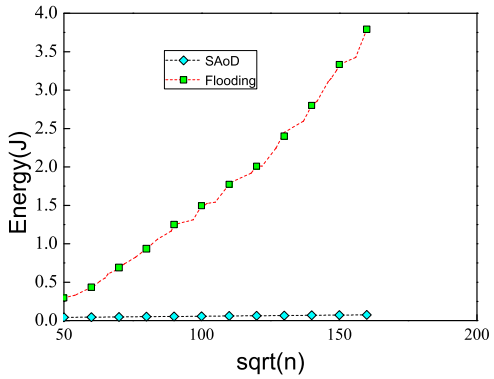


Fig. 17. Energy consumption change with network size.

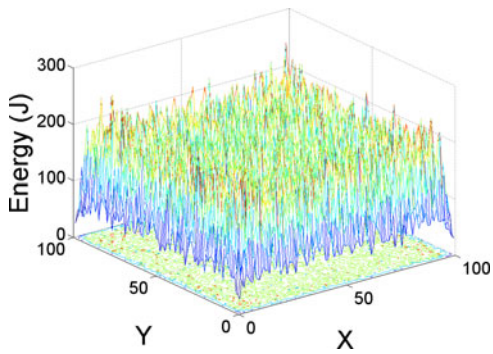


Fig. 18. Load balance.

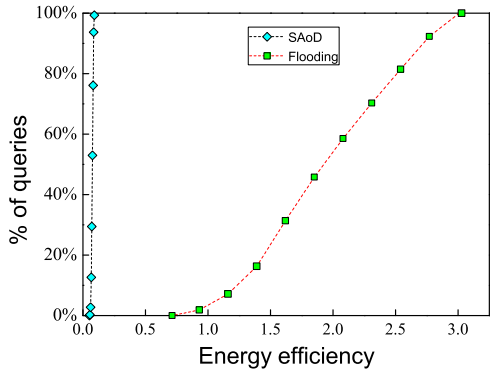


Fig. 19. Energy efficiency.

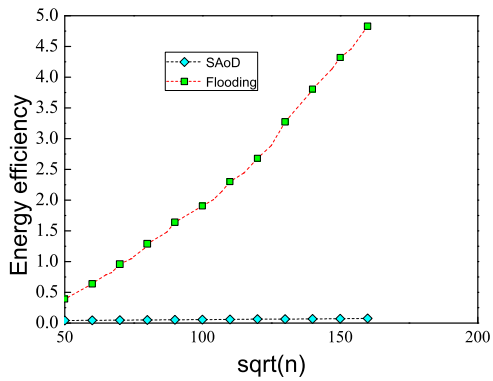


Fig. 20. Energy efficiency change with network size.

0.5 J while the highest energy consumption of SAoD is 0.09 J. The average energy consumption of a query using flooding is 1.36 J, while the average energy consumption of a query using SAoD is 0.073 J. The results show that the SAoD replication strategy is indeed cost-efficient. It reduces the energy consumption to more than one order of magnitude lower than that of flooding scheme.

Fig. 17 plots how the energy consumption changes with the network size. It shows that the energy consumptions of flooding and SAoD both increase with the network size. The energy consumption of flooding increases sharply when the network size increases, while the energy consumption of SAoD changes slightly with the network size. It is not difficult to see that the energy consumption of SAoD is linear with the square root of the network size. This demonstrates that the replica deployment strategy of SAoD is quite efficient.

Load balance is important for a wireless sensor network [41], because the overload of some sensor nodes will make them die quickly and this can paralyze the entire network. Fig. 18 shows the loads of all the nodes in the network. It reveals that the loads are balanced among the nodes in the SAoD network.

Figs. 19 and 20 plot the energy efficiency of SAoD. Fig. 19 shows that the average energy efficiency of SAoD query is 0.103 while that of flooding is 1.921. This means a 19× performance improvement. Fig. 20 illustrates the energy efficiency of SAoD is much better than that of flooding in different network size. When the network size increases the improvement also increases.

Fig. 21 illustrates the precision of SAoD. It shows that the average precision of SAoD query is 98.4 percent which is quite acceptable. Although due to the false positives of Bloom filters, the system cannot achieve a precision of 100 percent, the cost of removing the false positive is cheap (Section 4.5).

Fig. 22 shows that the precision of SAoD scheme decreases very slightly when the network size increases. This is because that when the network size increases, the number of query replicas increases. Given a fixed false positive of Bloom filter, the more the queries are evaluated, the more false results will be returned. Thus, the precision may decrease slightly.

Fig. 23 plots the latency of SAoD. It shows that 19.7 percent SAoD queries have shorter search latency than those of flooding, while overall the latency of SAoD is slightly larger. This is because SAoD utilizes the minimal spanning tree to multicast the query replicas while the flooding scheme follows the shortest path to propagate the queries. The average latency of SAoD is 28.72 ms while the average latency of

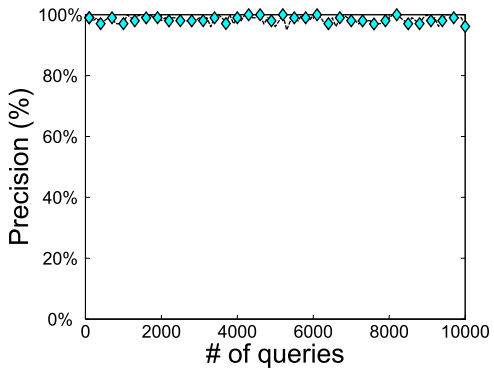


Fig. 21. Precision.

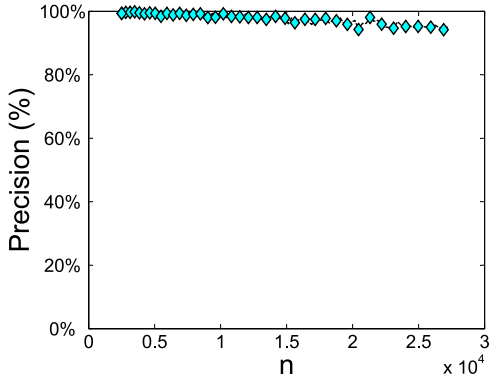


Fig. 22. Precision change with network size.

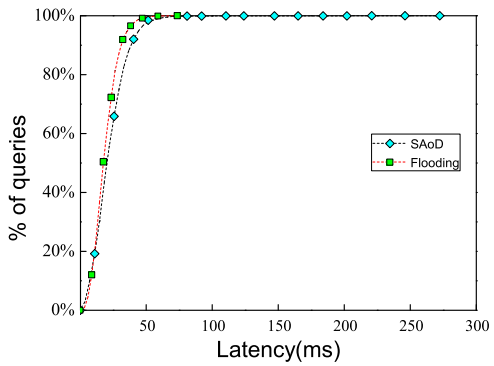


Fig. 23. Latency.

flooding is 26.54 ms. The results show that in the 1,000 m × 1,000 m area with 10,000 nodes the longest latency of SAoD is about 281 ms. Such a low latency is quite acceptable in real-world systems.

Fig. 24 shows that the search latencies of both scheme increase with the network size. The results show that the search latency is linear with the network size in square root scale.

Fig. 25 shows the data replication costs. We assume that each chunk identifier takes 20 bytes. It shows that by replicating Bloom filters, the SAoD scheme greatly reduces the energy cost for replicating.

7 Conclusions and Future Work

In this paper, we design and evaluate SAoD, an audio-on-demand system over WSNs. SAoD stores the audio data

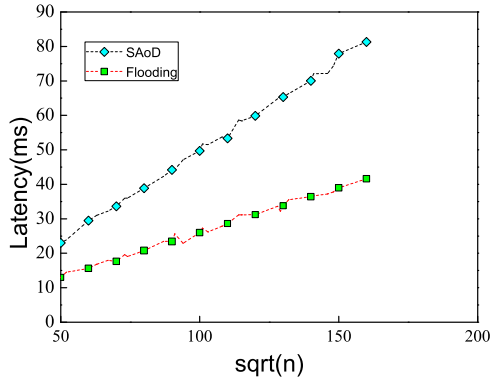


Fig. 24. Latency change with network size.

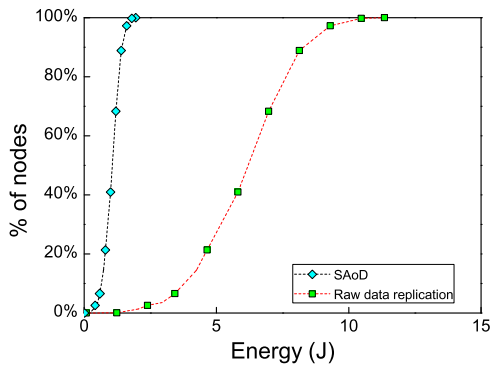


Fig. 25. Energy efficiency.

inside the network for retrieval. We show mathematically that SAoD achieves high success rate with determined lower bound for chunk retrieval at the cost of $O(\sqrt{n})$. Instead of replicating the raw audio data, we use Bloom filters to compress the metadata of chunks. We implement a real system based on IRIS mote as well as conduct comprehensive simulations to evaluate this design.

Equipped with the results obtained in this study, we see great potentials of applying SAoD to a wide range of audio-on-demand applications. Meanwhile, there are also some interesting open questions for our future work. For example, the potential breakdown of motes may lead to loss of chunks. We will investigate distributed chunk-grain data redundancy and recovery mechanism, in our design in the future.

Acknowledgments

This work was supported in part by NSFC under grants 61370233, 61422202, Ministry of Education and China Mobile Communications Corporation Research Founding under grant MCM20130382 and Foundation for the Author of National Excellent Doctoral Dissertation of PR China under grant 201345.

References

[1] T. Zia and A. Y. Zomaya, "A secure triple-key management scheme for wireless sensor networks," in *Proc. IEEE INFOCOM*, 2006, pp. 1–2.

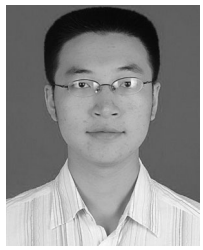
[2] H. M. Ammari and S. K. Das, "Centralized and clustered k-coverage protocols for wireless sensor networks," *IEEE Trans. Comput.*, vol. 61, no. 1, pp. 118–133, Jan. 2012.

[3] Y. Zou and K. Chakrabarty, "Distributed mobility management for target tracking in mobile sensor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 8, pp. 872–887, Aug. 2007.

[4] W. Huangfu, Z. Zhang, X. Chai, and K. Long, "Survivability-oriented optimal node density for randomly deployed wireless sensor networks," *Sci. China Inf. Sci.*, vol. 57, no. 2, pp. 1–6, 2014.
[5] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless ad-hoc sensor networks," in *Proc. 7th Annu. Int. Conf. Mobile Comput. Netw.*, 2001, pp. 139–150.
[6] X. Wen, L. Shao, Y. Xue, and W. Fang, "A rapid learning algorithm for vehicle classification," *Inf. Sci.*, vol. 295, pp. 395–406, 2015.
[7] Y. Wu, K. Kapitanova, J. Li, J. A. Stankovic, S. H. Son, and K. Whitehouse, "Run time assurance of application-level requirements in wireless sensor networks," in *Proc. 9th ACM/IEEE Int. Conf. Inf. Process. Sens. Netw.*, 2010, pp. 197–208.
[8] T. Zhang, D. Wang, J. Cao, Y. Q. Ni, L. Chen, and D. Chen, "Elevator-assisted sensor data collection for structural health monitoring," *IEEE Trans. Mobile Comput.*, vol. 11, no. 10, pp. 1555–1568, Oct. 2012.
[9] Y. Li, C. Ai, C. T. Vu, Y. Pan, and R. A. Beyah, "Delay-bounded and energy-efficient composite event monitoring in heterogeneous wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 9, pp. 1373–1385, Sep. 2010.
[10] J. Xu, K. Li, and G. Min, "Reliable and energy-efficient multipath communications in underwater sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 7, pp. 1326–1335, Jul. 2012.
[11] J. Shen, H. Tan, J. Wang, J. Wang, and S. Lee, "A novel routing protocol providing good transmission reliability in underwater sensor networks," *J. Internet Technol.*, vol. 16, no. 1, pp. 171–178, 2015.
[12] (2011). [Online]. Available: http://earthquake.usgs.gov/earthquakes/eqinthenews/
[13] K. Mihic, A. Mani, M. Rajashekhar, and P. Levis, "MStore: Enabling storage-centric sensornet research," in *Proc. ACM/IEEE Int. Conf. Inf. Process. Sens. Netw.*, Cambridge, MA, USA, Apr. 2007.
[14] J. Xu, X. Tang, and W.-C. Lee, "A new storage scheme for approximate location queries in object-tracking sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 2, pp. 262–275, Feb. 2008.
[15] L. Luo, Q. Cao, C. Huang, T. Abdelzaher, J. A. Stankovic, and M. Ward, "EnviroMic: Towards cooperative storage and retrieval in audio sensor networks," in *Proc. 27th Int. Conf. Distrib. Comput. Syst.*, 2007, p. 34.
[16] G. WernerAllen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proc. 7th Symp. Oper. Syst. Des. Implementation*, Seattle, WA, USA, Nov. 2006, pp. 381–396.
[17] X. Deng and Y. Yang, "Online adaptive compression in delay sensitive wireless sensor networks," *IEEE Trans. Comput.*, vol. 61, no. 10, pp. 1429–1442, Oct. 2012.
[18] L. Li, G. Xing, L. Sun, and Y. Liu, "QVS: Quality-aware voice streaming for wireless sensor networks," in *Proc. Int. Conf. Distrib. Comput. Syst.*, Montreal, QC, Canada, June 2009, pp. 450–457.
[19] E. Soroush, K. Wu, and J. Pei, "Fast and quality-guaranteed data streaming in resource-constrained sensor networks," in *Proc. MobiHoc*, Hong Kong, China, May 2008, pp. 391–400.
[20] S. Misra, M. Reisslein, and G. Xue, "A survey of multimedia streaming in wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 10, no. 1-4, pp. 18–39, Fourth Quarter 2008.
[21] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "GHT: A geographic hash table for data centric storage," in *Proc. 1st ACM Int. Workshop Wireless Sens. Netw. Appl.*, Atlanta, GA, USA, Sept. 2002, pp. 78–87.
[22] A. Bernauer and K. Römer, "A comprehensive compiler-assisted thread abstraction for resource-constrained systems," in *Proc. ACM/IEEE Int. Conf. Inf. Process. Sens. Netw.*, 2013, pp. 167–178.
[23] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, 1970.
[24] L. Chitnis, A. Dobra, and S. Ranka, "Aggregation methods for large-scale sensor networks," *ACM Trans. Sens. Netw.*, vol. 4, no. 2, p. 9, 2008.
[25] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," in *Proc. 2nd Int. Conf. Embedded Netw. Sens. Syst.*, 2004, pp. 250–262.
[26] D. Koutsonikolas, S. M. Das, Y. C. Hu, and I. Stojmenovic, "Hierarchical geographic multicast routing for wireless sensor networks," *Wireless Netw.*, vol. 16, no. 2, pp. 449–466, 2010.
[27] (2008). Tinyos. [Online]. Available: http://www.tinyos.net/
[28] A. Rezgui and M. Eltoweissy, "μRACER: A reliable adaptive service-driven efficient routing protocol suite for sensor-actuator networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 5, pp. 607–622, May 2009.
[29] A. Boukerche and D. Turgut, "Secure time synchronization protocols for wireless sensor networks," *IEEE Wireless Commun.*, vol. 14, no. 5, pp. 64–69, Oct. 2007.
[30] B. Jiang, B. Ravindran, and H. Cho, "Probability-based prediction and sleep scheduling for energy-efficient target tracking in sensor networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 4, pp. 735–747, Apr. 2013.
[31] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in *Proc. 2nd Int. Conf. Embedded Netw. Sens. Syst.*, Baltimore, MD, USA, Nov. 2004, pp. 13–24.
[32] J. Bruck, J. Gao, and A. A. Jiang, "MAP: Medial axis based geometric routing in sensor network," in *Proc. 11th Annu. Int. Conf. Mobile Comput. Netw.*, Cologne, Germany, Aug. 2005, pp. 835–853.
[33] S. Funke, A. Kesselman, U. Meyer, and M. Segal, "A simple improved distributed algorithm for minimum CDS in unit disk graphs," *ACM Trans. Sens. Netw.*, vol. 2, no. 3, pp. 444–453, 2006.
[34] H. M. Ammari, "Investigating the energy sink-hole problem in connected-covered wireless sensor networks," *IEEE Trans. Comput.*, vol. 63, no. 11, pp. 2729–2742, Nov. 2014.
[35] J. Zhang, S. Iyer, P. Schaumont, and Y. Yang, "Simulating power/energy consumption of sensor nodes with flexible hardware in wireless networks," in *Proc. 9th Annu. IEEE Commun. Soc. Conf. Sensor, Mesh Ad Hoc Commun. Netw.*, 2012, pp. 112–120.
[36] W. Tu, C. J. Sreenan, and W. Jia, "Worst-case delay control in multigroup overlay networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 10, pp. 1407–1419, Oct. 2007.
[37] S. Xie and Y. Wang, "Construction of tree network with limited delivery latency in homogeneous wireless sensor networks," *Wireless Pers. Commun.*, vol. 78, no. 1, pp. 231–246, 2014.
[38] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *Proc. 1st Int. Conf. Embedded Netw. Sens. Syst.*, Los Angeles, CA, USA, 2003, pp. 1–13.
[39] H. Nishiyama, A. E. A. A. Abdulla, N. Ansari, Y. Nemoto, and N. Kato, "Hymn to improve the longevity of wireless sensor networks," in *Proc. IEEE Global Telecommun. Conf.*, Miami, FL, USA, Dec. 2010, pp. 1–5.
[40] P. Guo, J. Wang, X. H. Geng, C. S. Kim, and J.-U. Kim, "A variable threshold-value authentication architecture for wireless mesh networks," *J. Internet Technol.*, vol. 15, no. 6, pp. 929–935, 2014.
[41] H. Zhang and H. Shen, "Balancing energy consumption to maximize network lifetime in data-gathering sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 10, pp. 1526–1539, Oct. 2009.

Hanhua Chen received the PhD degree in computer science and engineering from the Huazhong University of Science and Technology, China, in 2010, where he is currently a professor at the School of Computer Science and Technology. His research interests include distributed systems, wireless sensor networks, social network systems, and peer-to-peer systems. He received the National Excellent Doctoral Dissertation Award of China in 2012. He is a member of the IEEE.

Hai Jin received the PhD degree in computer engineering from the Huazhong University of Science and Technology (HUST), China, in 1994. He is a Cheung Kung Scholars chair professor of computer science and engineering at HUST. In 1996, he was awarded a German Academic Exchange Service fellowship to visit the Technical University of Chemnitz in Germany. He worked at the University of Hong Kong between 1998 and 2000, and as a visiting scholar at the University of Southern California between 1999 and 2000. He received Excellent Youth Award from the National Science Foundation of China in 2001. He is the chief scientist of ChinaGrid, the largest grid computing project in China, and the chief scientist of National 973 Basic Research Program Project of Virtualization Technology of Computing System. He has coauthored 15 books and published more than 600 research papers. His research interests include computer architecture, virtualization technology, cluster computing and grid computing, peer-to-peer computing, network storage, and network security. He is named the steering committee chair of several International Conferences including GPC, APSCC, FCST, and ChinaGrid and is a member of the steering committee of CCGrid, NPC, GCC, ATC, and UIC. He is a senior member of the IEEE and a member of the ACM.



Lingchao Guo received the master's degree from the School of Computer Science and Technology, Huazhong University of Science and Technology, China, in 2013. His research interests include online social networks and wireless sensor networks.