Approximately Uniform Random Sampling in Sensor Networks

Boulat A. Bash John W. Byers Jeffrey Considine

Computer Science Department
Boston University
{boulat, byers, jconsidi}@cs.bu.edu

Abstract

Recent work in sensor databases has focused extensively on distributed query problems, notably distributed computation of aggregates. Existing methods for computing aggregates broadcast queries to all sensors and use in-network aggregation of responses to minimize messaging costs. In this work, we focus on uniform random sampling across nodes, which can serve both as an alternative building block for aggregation and as an integral component of many other useful randomized algorithms. Prior to our work, the best existing proposals for uniform random sampling of sensors involve contacting all nodes in the network. We propose a practical method which is only approximately uniform, but contacts a number of sensors proportional to the diameter of the network instead of its size. The approximation achieved is tunably close to exact uniform sampling, and only relies on well-known existing primitives, namely geographic routing, distributed computation of Voronoi regions and von Neumann's rejection method. Ultimately, our sampling algorithm has the same worst-case asymptotic cost as routing a point-to-point message, and thus it is asymptotically optimal among request/reply-based sampling methods. We provide experimental results demonstrating the effectiveness of our algorithm on both synthetic and real sensor topologies.

1 Introduction

In the emerging research area of sensor databases, a central challenge is to develop cost-effective methods to extract answers to queries about conditions inside the sensor network. One typical sensor database scenario involves sensor elements that are prone to failure, are highly resource-constrained, and must communicate across a lossy network. Sensor networks comprised of small battery-powered motes are a representative instantiation of this scenario [7]. In such an environment, *aggregation queries* are particularly effective, as they are robust to node and link failures, can be resilient to incorrect or outlying responses, and are amenable to the use of in-network processing to minimize messaging cost. For these queries, approximate answers typically suffice, especially in light of the very high cost of ensuring 100% reliability in communications in sensor networks. Recent work has focused on computation of aggregates using a request/reply model in which a query is broadcast to a region of interest, individual sensors make best-effort replies, and responses are aggregated in-network en route to the origin of the query [3, 11, 20].

In this paper, we argue that there is a rich and relatively under-explored set of classic statistical methods that have not yet been extensively studied in the domain of sensor databases. In particular, we propose a more careful study of random sampling methods, which have long been used in other domains to approximately compute aggregates such as MEDIAN, AVG, and MODE [2, 12, 13]. Random sampling is a particularly good fit for approximate aggregation queries in the sensor network domain in light of the potentially modest messaging cost. While we view random sampling as especially useful in the context of data management and data aggregation problems, we also note that it is an integral component of other useful randomized algorithms that are potentially applicable to sensor networks, including randomized routing [18].

In the context of sensor networks, a natural abstraction is *spatial sampling*, i.e. sampling from geographical locations within the network uniformly at random. On a 2-D network with bounded spatial extent, such an objective can easily be realized by picking an (x, y) coordinate from within the space at random and using geographical routing to route to the node closest

The authors were supported in part by NSF grants ANI-9986397, ANI-0093296, ANI-0205294, and EIA-0202067.

Copyright 2004, held by the author(s)

Proceedings of the First Workshop on Data Management for Sensor Networks (DMSN 2004), Toronto, Canada, August 30th, 2004.
http://db.cs.pitt.edu/dmsn04/

to that point. While this is desirable for many applications, such as computing spatial averages [6], many other applications and database queries prefer to ignore geometry and instead wish to *sample uniformly from the set of nodes*. Examples include querying average sensor battery life, counting the number of nodes that are currently capable of executing a given sensing task, determining the 95th quantile of sensor CPU utilization, or estimating the number of sensors that will fail within the next day. Our focus is to develop practical algorithms for uniformly sampling from a set of sensor nodes with low messaging cost.

Since it is well-known that nodes in a sensor network often have highly irregular placements, spatial sampling will produce non-uniform samples of the nodes [5]. Our work relies on spatial sampling as a starting point, but uses practical methods for smoothing, or regularizing, the non-uniform samples to produce approximately uniform node samples. The key idea is to have each sensor node compute and maintain the area of its Voronoi cell. A uniform node sample is then realized by sending a sequence of spatial samples until one is "accepted". A targeted node in the network "accepts" by responding to a given spatial sample with an appropriate probability normalized by its Voronoi cell size, otherwise it "rejects". The specifics of this normalization depend on global statistics on the number of nodes in the network and on an appropriate k-quantile of Voronoi cell sizes across the network. We argue that these statistics can be updated infrequently and consistently. Ultimately, this application of von Neumann's rejection method [19] results in approximately uniform node samples.

As sketched above, our algorithm for generating a random sample has a messaging cost that is typically bounded by the messaging cost of a small constant number of spatial samples in the expectation. This cost is low since the messaging cost of computing a spatial sample is akin to routing a point-to-point message using a geographic routing method such as GPSR [8]. In the worst case, such a message traverses the diameter of the network. In contrast, the best existing methods for node sampling, which can compute an exactly uniform sample, necessitate contacting all nodes in the network [13]. We note that the additional infrequent global update costs incurred by our algorithm can be amortized by the potentially vast number of samples that can be taken between updates.

The remainder of the paper is organized as follows: Section 2 formalizes the uniform sampling problem and the limitations of existing methods. We summarize the building blocks of our proposed method in Section 3. Our rejection-based sensor sampling algorithm is presented in Section 4. Then in Section 5 we describe the practical implementation issues, and Section 6 concludes with the broader implications and applications of our work.

2 Sampling: Problems and Methods

We now formally define our sampling problems.

Definition 1 (Uniform random sampling) *An algorithm samples uniformly at random from a set of reachable sensors S if and only if it outputs a sensor ID $s \in S$ with probability $\frac{1}{|S|}$.*

Uniform random sampling is simple if the set of sensor IDs is known in advance and sensors neither fail nor move. However, it is much more challenging in the realistic case where the set of IDs may not be known and the set of reachable sensors dynamically changes over time. For these reasons, we will be content with the following close approximation to uniform sampling.

Definition 2 ((ϵ, δ)-sampling) *An algorithm performs (ϵ, δ)-sampling of a reachable set S if and only if it returns a sample $s \in S$ such that no element of S is returned with probability greater than $\frac{1+\epsilon}{|S|}$ and at least $(1-\delta)|S|$ elements are output with probability at least $\frac{1}{|S|}$.*

By this definition, our goal is to sample from almost all sensors nearly uniformly with tunable parameters ϵ and δ. Our definition allows us to under-sample a small fraction δ of the nodes.

In a sensor network scenario, we typically wish to sample from a set of pairs $\langle k, v \rangle$ where k identifies a particular sensor and is unique within the set, and v is some value associated with the sensor. This value might be a measurement by the sensor, such as the local temperature, or an internal statistic such as the remaining battery life. As motivated earlier, sampling in sensor networks is more challenging since neither a full list of sensors nor direct communication with them is available.

Prior to this work, the following two methods for near-uniform sampling were proposed in the context of sensor and other overlay networks.

Min-wise sampling: In [13], the use of min-wise samples [1] was proposed for sampling a sensor network uniformly at random. Given a hash function h on sensor IDs, they returned the value associated with the ID s such that $h(s)$ is minimal (i.e. $\forall_{s' \in S}(h(s) \leq h(s')))$. Each sensor would then propagate the value associated with the smallest observed $h(s')$. With careful control of the transmissions, this scheme can be implemented with each node in the sensor network sending a constant number of messages, for a total of $\Theta(|S|)$ transmissions. However, since the entire network is involved, this is an expensive operation.

Random walks: Another natural method for sampling is the use of random walks. In the sensor network domain, one could generate a random sample by propagating a request message along a randomly chosen k-hop path starting from the query sink, and sampling the kth sensor reached. Unfortunately, this procedure would both need to use a large value of k, and would need to compensate for the fact that the method is biased toward drawing samples from near the center of the spatial region where sensors are located, as we demonstrate in Section 5.1.

Our methods follow a rather different line. Like the random walk method, we ultimately seek out a single sensor, but our choice of the route to the sensor avoids many of the dependencies and complications of the random walk approach.

3 Prerequisites

Instead of choosing a path at random, we choose a location in the sensor coordinate space at random and route a probe to its closest sensor using geographic routing techniques. When we partition the coordinate space into regions of ownership by mapping the nearest neighbor regions (Voronoi cells) to sensors, we note that these regions are irregularly sized in most instances. Thus, this naive *spatial sampling* method is very likely to generate a biased sample. Therefore, our last key step is to use von Neumann's rejection method to normalize the samples. We now briefly summarize these three prerequisite ideas.

Geographic routing: If every node in a network is aware of its own coordinates (e.g. via GPS), then it is possible to route to a particular position using entirely local decisions. Most of these local routing decisions can be made in a greedy fashion, simply choosing the neighboring node which has the closest coordinates to the destination. This greedy routing fails when there is an obstruction, or "void", which must be circumnavigated to reach the destination. GPSR [8] provides an elegant solution to this problem with just two states. The default state of GPSR is greedy routing, while the other state follows the perimeters of voids until greedy routing can resume. When a packet reaches its target point, another round of perimeter routing is run to visit each of the immediately surrounding sensors so that it can find the sensor nearest to the target point. For typical topologies in 2-D, geographic routing takes $\Theta(\sqrt{|S|})$ steps.

Voronoi diagrams: Once routing to an arbitrary point is possible, we must also quantify the size of the region of points that are closest to a particular sensor s. Formally, the set of points closer to sensor s than any other sensor is called the *Voronoi cell* of s [4]. In the planar case which we consider, the Voronoi cell of s is a convex polygon containing s, where each edge of this polygon lies on a perpendicular bisector between s and another sensor. The exact boundaries of this Voronoi cell are easily determined exactly by locating all of the sensors in the immediate vicinity of s. The areas of these Voronoi cells have been used previously to weight sensor readings for spatial aggregates [6] and they are easily computable, but it is well known that these areas vary widely when the sensors are placed randomly [16]. This variation leads to a bias in spatial sampling – each sensor is chosen with probability in proportion to $A(s)$, the area of its Voronoi cell. For convenience, we assume the areas are normalized so that they sum to one, and thus $A(s)$ can also be interpreted as the probability a randomly chosen point is closest to s.

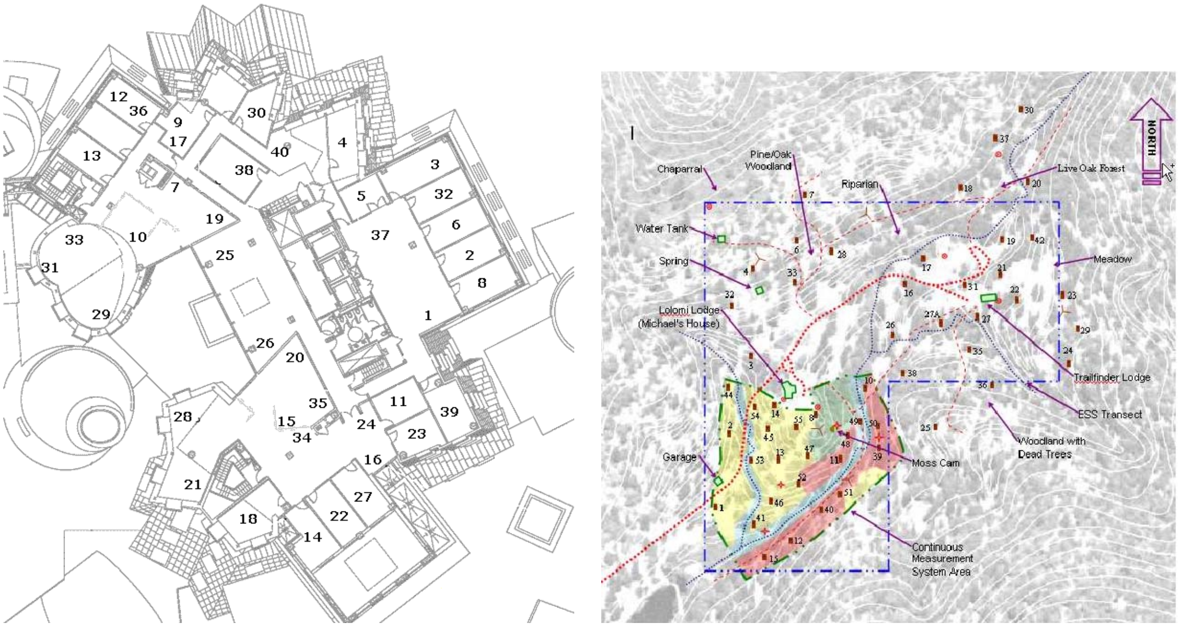
von Neumann's rejection method: Much of the early work on random sampling focused on sampling complex distributions, assuming the ability to sample simpler distributions. A well known example of this is von Neumann's rejection method [10, 19]. Suppose we wish to sample from a distribution with probability density function f (i.e. an event t has probability $f(t)$). If we can sample from a distribution with probability density function g, then we can sample from f as follows. First generate a sample t using g, but only accept and return sample t with probability $\frac{f(t)}{cg(t)}$, where c is a positive constant. If t is not accepted, it is rejected and the process repeats for a new sample t. Assuming that c is chosen so that $\frac{f(t)}{cg(t)} \leq 1$, then the probability of picking a particular event t on the first attempt is $g(t) \cdot \frac{f(t)}{cg(t)} = \frac{1}{c} f(t)$. It then follows that after c expected samples from g, we have one sample from f.

4 Rejection-based Sensor Sampling

We now describe our method to combine ideas of spatial sampling with von Neumann's rejection method to flatten out an irregular probability distribution into a nearly uniform one. For our application, the desired density function is uniform, i.e. $f(t) = \frac{1}{|S|}$, and the distribution which we can sample from, $g(t)$, is the distribution of Voronoi cell areas. One weakness in von Neumann's method for *exactly* reproducing a distribution f is that the constant c must be chosen so that for all events t, $\frac{f(t)}{g(t)} \leq c$. In our application, if there exists a very small Voronoi cell, then c, and hence the expected messaging cost, can be very large. Since we cannot rule out this possibility, we content ourselves for now with generating approximately uniform samples. Later, in Section 5.2, we consider strategies to boost sampling probabilities for the smallest cells to significantly reduce residual sampling bias. We employ the following basic algorithm.

Algorithm 1 (Rejection-based Sampling)

1 *The random sampler picks a random location in the sensor field and routes a message to the sensor s closest to this point, using geographic routing and pre-computation of Voronoi cells.*



(a) MIT sensor testbed. Reproduced with permission from [17]

(b) James Reserve sensor network. Reproduced with permission from [5]

Figure 1: Maps of real sensor deployments used in our experiments.

2 With probability $\frac{\min(A(s),\tau)}{A(s)}$, s accepts and reports its value, where τ is a threshold to be defined shortly.

3 Otherwise, s rejects and the random sampler repeats Steps 1-3. The random sampler also returns to Step 1 if it times out waiting for a response.

Intuitively, τ can be thought of as a threshold on Voronoi cell areas, in which we think of any Voronoi cell of area at least τ as *large* and any area less than τ as *small*. By our procedure, all large cells will be selected equiprobably, but small cells will be selected with smaller probability, in proportion to their area. To ensure that Algorithm 1 results in (ϵ, δ) sampling, we must guarantee that the fraction of small cells (sampled non-uniformly) is less than δ, and that the bias introduced by under-sampling small cells results in at most $(1+\epsilon)$-oversampling of large cells. In practice, we set τ to be the area of the cell that is the k-quantile, where $k = \min\left(\delta, \frac{\epsilon}{1+\epsilon}\right)$, and prove the following main result.

Theorem 1 *Running Algorithm 1 with $k = \min\left(\delta, \frac{\epsilon}{1+\epsilon}\right)$ and setting τ to be the cell area that is the k-quantile results in (ϵ, δ)-sensor sampling.*

Proof: By our problem definition, it suffices to show that the method ensures that no element of S is sampled with probability greater than $\frac{1+\epsilon}{|S|}$ and at least $(1-\delta)|S|$ elements are sampled with probability at least $\frac{1}{|S|}$. First, we show that all large cells, i.e. cells with area at least τ, are sampled in a given iteration of the sampling algorithm with probability at least $\frac{1}{|S|}$. The probability that a given sensor s is sampled in a particular probe is $p_s = \min(A(s), \tau)$, and thus the probability that a particular probe is successful is $\sum_s p_s \leq |S|\tau$. Now let E_ℓ denote the event that the algorithm ultimately samples from a large cell ℓ.

$$\Pr[E_\ell] = \frac{p_\ell}{\sum_s p_s} = \frac{\tau}{\sum_s p_s} \geq \frac{1}{|S|}.$$

Now since large cells are at least a $(1-\delta)$ fraction of all cells by the setting of $k \leq \delta$, we have that at least $(1-\delta)|S|$ elements are sampled with probability at least $\frac{1}{|S|}$.

Next we show that no element is sampled with probability greater than $\frac{1+\epsilon}{|S|}$. By construction, large cells are sampled with highest probability, so we restrict attention to those cells. Starting from the same probability bound as before:

$$\begin{aligned}
\Pr[E_\ell] = \frac{p_\ell}{\sum_s p_s} &= \frac{\tau}{\sum_{s|A(s)<\tau} p_s + \sum_{s|A(s)\geq\tau} p_s} \\
&= \frac{\tau}{\sum_{s|A(s)<\tau} p_s + \sum_{s|A(s)\geq\tau} \tau} \\
&\leq \frac{\tau}{\sum_{s|A(s)\geq\tau} \tau} \\
&\leq \frac{\tau}{(1-k)|S|\tau}
\end{aligned}$$

$$\begin{aligned} &\leq \frac{\tau}{(1-\frac{\epsilon}{1+\epsilon})|S|\tau} \\ &\leq \frac{1}{(\frac{1+\epsilon}{1+\epsilon}-\frac{\epsilon}{1+\epsilon})|S|} \\ &= \frac{1}{(\frac{1}{1+\epsilon})|S|} \\ &= \frac{1+\epsilon}{|S|}. \end{aligned}$$

Thus the theorem follows. ■

Relating this result back to von Neumann's method, this corresponds to a situation in which $c = \frac{1}{\tau|S|}$. As with the rejection method, the probability that a particular sensor s is picked and accepted on the first attempt is $A(s)\frac{\min(A(s),\tau)}{A(s)} = \min(A(s),\tau)$. It remains to select an appropriate threshold τ for our algorithm.

4.1 Threshold Management

Given user-specified values of ϵ and δ, the threshold τ should be set to the k-quantile of the Voronoi cell areas, where $k = \min\left(\delta, \frac{\epsilon}{1+\epsilon}\right)$ as discussed earlier. The k-quantile can be computed during an initial preprocessing step using recent techniques developed in the sensor database community. In particular, work such as [3, 11] shows how to efficiently count the number of sensors matching some criteria (e.g. with a cell area below a specified threshold) and deriving other simple statistics such as the average cell area. We note that while these values need to be updated to account for dynamic changes within the sensor network, they need not be exact, as bounds on the values suffice for our methods. Therefore, only infrequent updating of these global statistics is needed to maintain consistent and approximately correct values. Updating these statistics can easily be performed either by piggybacking them on the random probes or on various control and maintenance messages. Either way, once these statistics are available, the sampler recomputes τ, and sends it with each probe. Since the sampler's value of τ is included in the query, each sensor deciding to accept or reject a probe acts consistently.

5 Practical Implementation Issues

We now discuss the details of a practical implementation of Algorithm 1. We begin in Section 5.1 presenting experimental results using the basic implementation outlined in Section 4, and then discuss various refinements to improve the uniformity of sampling in Section 5.2.

5.1 Experiments

We experimentally validated our proposed sampling algorithm using three topologies: two from real sensor deployments and one synthetic topology with 2^{15} sensors placed uniformly at random on a unit square. The first real network, illustrated in Figure 1(a), is a testbed deployed at MIT [17]. These sensors were heuristically placed according to expected quality as a vantage point, and proximity to available power outlets. The second real deployment, illustrated in Figure 1(b), is a sensor network for micro-climate monitoring at the James Reserve [5]. These sensors are more concentrated in the lower left, where there is thick foliage.

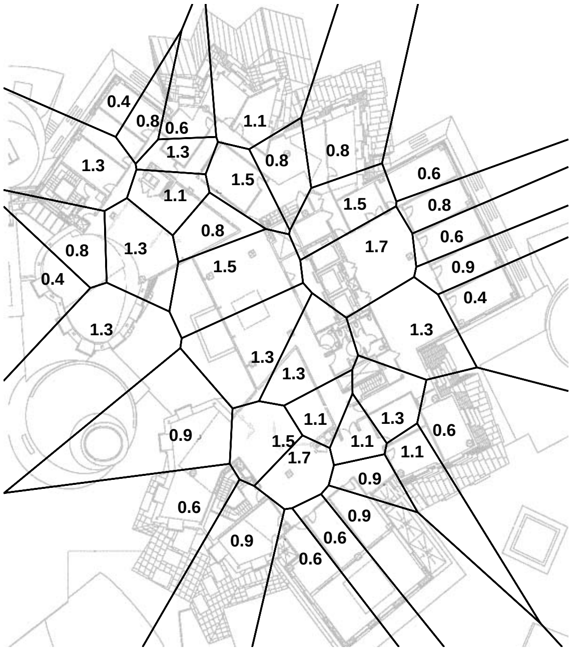
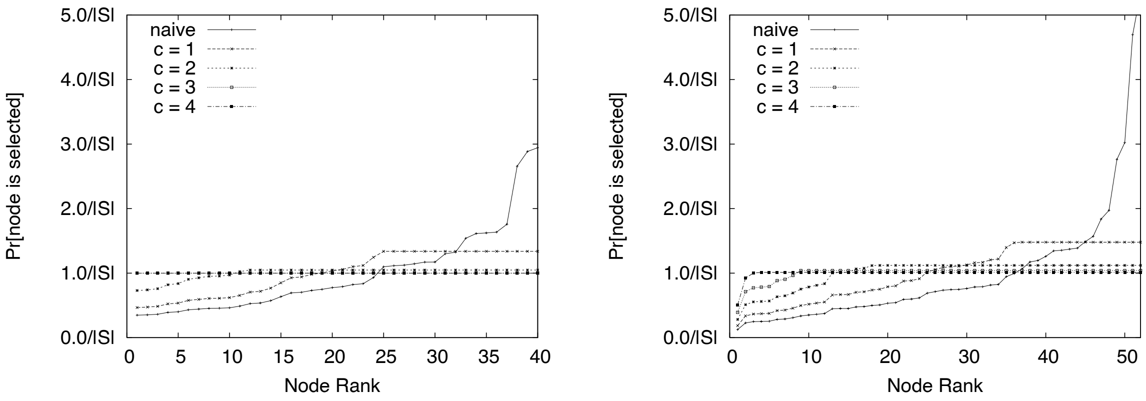


Figure 2: Sample distribution using long random walks along adjacent Voronoi cells. Each sensor's cell is labeled with its probability relative to the mean. For example, a sensor labeled 1.3 is picked with probability $1.3/|S|$.

The objective of these experiments was to demonstrate that we can cheaply obtain a close approximation to uniform sampling. Thus, besides examining ϵ and δ at for various choices of τ, we also examine the expected value of the random variable $\mathbf{Y}$, which is the number of probes sent before a sample is returned. The actual energy costs of our method depend heavily upon the geographic routing protocol in use. Since testing the performance of various geographical routing protocols is beyond the scope of this work, we do not implement geographic routing in our simulation.

First, we confirm our intuition that random walks are unsuitable for near-uniform random sampling. We consider the following random process. Starting at any sensor in the network, a query repeatedly considers the sensors with adjacent Voronoi cells and moves to one chosen uniformly at random. After a sufficient num-



(a) MIT sensor testbed

(b) James Reserve sensor network

Figure 3: Resulting distributions for real testbeds. Nodes are in increasing order of Voronoi cell area.

c	ϵ	δ	$\mathbf{E}[\mathbf{Y}]$
naive	1.9	0.6	1.00
1	0.34	0.45	1.34
2	0.047	0.25	2.09
3	0	0	3.00
4	0	0	4.00
5	0	0	5.00

(a) MIT sensor testbed

c	ϵ	δ	$\mathbf{E}[\mathbf{Y}]$
naive	4.3	0.69	1.00
1	0.48	0.46	1.48
2	0.12	0.23	2.24
3	0.041	0.15	3.12
4	0.012	0.038	4.05
5	0.0072	0.019	5.04

(b) James Reserve sensor network

c	ϵ	δ	$\mathbf{E}[\mathbf{Y}]$
naive	3.8	0.57	1.00
1	0.27	0.41	1.27
2	0.051	0.15	2.10
3	0.017	0.06	3.05
4	0.0079	0.029	4.03
5	0.0042	0.017	5.02

(c) 2^{15} randomly placed points

Table 1: Summary of experimental results

ber of steps to converge on the stationary distribution, the query outputs its current location. Figure 2 shows the Voronoi diagram of the MIT sensor testbed and the relative sampling probabilities of each sensor. As expected, the sensors most likely to be chosen are in the middle of the network, and the sensors least likely to be chosen are on the edges of the network. Sufficiently long random walks on this topology can achieve $(0.71, 0.52)$-sampling. This is better than naive spatial sampling, which would achieve $(1.90, 0.60)$-sampling on the same topology, but our rejection-based methods will give much better results.

Figure 3 shows the results of Algorithm 1 on the real topologies assuming that there are no faults and each sensor knows the area of its own Voronoi cell. The areas of both networks are the areas of their minimum bounding boxes. The threshold τ was set to $\frac{1}{c|S|}$ for $c = 1, 2, 3, 4, 5$, and the naive spatial sampling method is included as a baseline. As c increases and τ decreases, the distribution becomes more uniform and improvements in both ϵ and δ are clearly visible.

Tables 1(a) and 1(b) summarize the parameters of the resulting sampling distributions, along with the expected number of probes for each sample. With the MIT sensor testbed, setting $c = 3$ (equiv. $\tau = \frac{1}{3|S|}$) results in uniform sampling – this is because there are no sensors with less than a third of the average cell area in their Voronoi cell. With the James Reserve network, one sensor has a cell area of slightly more than one tenth of the average, so $c \geq 10$ is necessary for uniform sampling. However, this is the only sensor which is under-sampled for $c \geq 5$.

For comparison, Table 1(c) summarizes the corresponding results for a synthetically generated topology of 2^{15} randomly placed points on a unit square. The smallest Voronoi cell in this topology was slightly smaller than $\frac{1}{98|S|}$, so if exact sampling is desired, an average of $c \geq 99$ probes per sample are needed. However, just setting $c = 5$ achieves $(0.0042, 0.017)$-sampling.

Figure 4 shows the cell size distributions of our test topologies where the impact of human choices on sensor placement is present. First, humans are prone to favor interesting or easily accessible points, resulting in sensors being clustered together, each with below-average area. This is evident in Figure 4: the two real sensor networks have a larger fraction of sensors with below-average Voronoi cell areas than a randomly generated topology. At the same time, humans are unlikely to choose very poor placements where many sensors are extremely close together. Figure 4 also hints at this point, as the smallest Voronoi cells in syntheti-

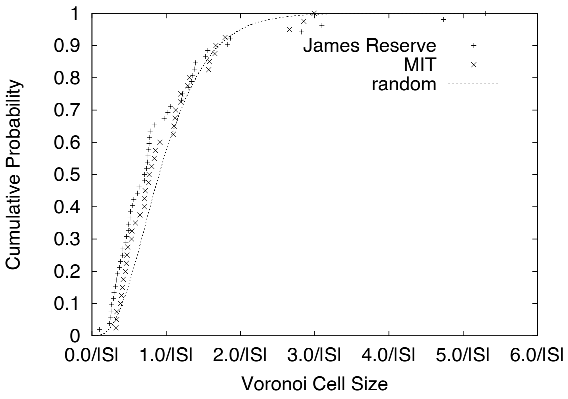


Figure 4: Cell size distributions for random and real testbeds

cally generated networks are significantly smaller than the ones in real topologies.

5.2 Algorithmic Modifications

We now consider a variety of heuristics for improving our baseline algorithm by reducing the impact of small Voronoi cells on the (ϵ, δ)-approximation.

Sleeping: Perhaps the simplest method for handling sensors with very small Voronoi cells is for some of these sensors to sleep. Sleeping sensors are deactivated, and sampling from them is thus rendered impossible. Putting one small cell to sleep will increase the size of adjacent cells (which are also likely to be small), so it is not necessary to put all small cells to sleep to remove their impact. We note that this approach is similar in spirit to some routing schemes which use sleep for power management, particularly in crowded areas [21]. Because the sensed values from the sleeping nodes are unavailable, this approach may not be appropriate for some applications.

Pointers: Another method for increasing the sampling probability of small cells is for larger cells to keep pointers to nearby small cells and forward some rejected probes to those small cells. That is, whenever a large cell would reject a probe, it may instead redirect the probe to a nearby small cell. The probability of forwarding a probe can be negotiated between the cells based on their respective sizes. Essentially, a large cell would donate part of its "unused" area to its small neighbor.

Virtual coordinates: Instead of using real-world geographic coordinates to map points to sensors, we can use virtual coordinates [14, 15], modified to include either a repulsive force between close sensors, or a hard lower bound on the inter-sensor distances. Virtual coordinate spaces also allow the boundaries of the sensor network to be pre-defined, instead of explored via periodic probing [5].

6 Future Work and Conclusions

Uniform random sampling is a standard and useful primitive underlying many algorithmic and statistical methods. Our work focused on the unique constraints imposed by sensor networks, and the problem of cheaply selecting one sensor node uniformly at random. In future work, there are numerous generalizations to consider. Our methods immediately generalize to queries that wish to sample nodes satisfying a geometric predicate, such as those within a region of interest, but we have not yet studied how to efficiently sample from nodes satisfying a non-geometric predicate. Another interesting question is how best to take advantage of parallelism when the number of samples needed or the expected number of attempts is high. Here, distinct probes may traverse common network links, so clever strategies may be able to reduce total transmission costs. We also plan to consider how to optimize sampling for queries which do not fall into a request/reply paradigm. For example, if query patterns are known in advance, such as periodic fixed queries, a more streamlined method for sampling that avoids explicit requests could be implemented in a decentralized fashion. However, our methods may still find use in answering such queries since their "on-demand" nature allows quick responses to unexpected events or failures.

Finally, we note that variants of our sampling methods can be applied much more broadly, outside the context of sensor networks. For example, uniform node sampling is also an important problem in structured P2P networks based on coordinate systems [9]. Variants of our methods apply to these P2P scenarios and provide a simpler and more topology-agnostic alternative to existing methods.

Acknowledgments

We are grateful to Deepak Ganesan and Stanislav Rost for the use of their sensor deployment maps and allowing them to be reproduced here. We thank Phil Gibbons, Kanishka Gupta, and Niky Riga for helpful conversations and feedback on earlier versions of this manuscript.

References

[1] A. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher. Min-wise independent permutations. *Journal of Computer and System Sciences*, 60:630–659, 2000.

[2] S. Chaudhuri, R. Motwani, and V. Narasayya. Random sampling for histogram construction: how much is enough? In *Proc. of ACM SIGMOD '98*, pages 436–447, 1998.

[3] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proc. of the IEEE Int'l Conf. on Data Engineering*, March 2004.

[4] M. de Berg, O. Schwarzkopf, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 2nd edition, 2000.

[5] D. Ganesan, S. Ratnasamy, H. Wang, and D. Estrin. Coping with irregular spatio-temporal sampling in sensor networks. In *Proc. of HotNets-II*, November 2003.

[6] C.-C. Han, S. Ganeriwal, A. Boulis, and M. Srivastava. Going beyond nodal aggregates: Spatial average of a physical process in sensor networks. Poster in *ACM SenSys*, Nov. 2003.

[7] J. Hill and D. Culler. Mica: A Wireless Platform for Deeply Embedded Networks. *IEEE Micro*, 22(6):12–24, Nov/Dec 2002.

[8] B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *ACM MobiCom*, Aug. 2000.

[9] V. King and J. Saia. Choosing a random peer. In *Proc. of ACM PODC'04*, July 2004.

[10] D. E. Knuth. *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Addison-Wesley, Reading, MA, 2nd. edition, 1981.

[11] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks. In *USENIX OSDI*, 2002.

[12] G. S. Manku, S. Rajagopalan, and B. G. Lindsay. Approximate medians and other quantiles in one pass and with limited memory. In *Proc. of ACM SIGMOD '98*, pages 426–435, 1998.

[13] S. Nath and P. Gibbons. Synopsis diffusion for robust aggregation in sensor networks. Technical Report ITR-03-08, Intel Research, Aug. 2003.

[14] J. Newsome and D. Song. GEM: Graph EMbedding for routing and data-centric storage in sensor networks without geographic information. In *ACM SenSys '03*, pages 76–88, 2003.

[15] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *ACM MobiCom*, Sept. 2003.

[16] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-centric storage in sensornets with GHT, a geographic hash table. *Mobile Networks and Applications*, 8(4):427–442, 2003.

[17] S. Rost and H. Balakrishnan. Lobcast: Reliable Data Dissemination in Wireless Sensor Networks. Under submission.

[18] L. G. Valiant and G. J. Brebner. Universal schemes for parallel communication. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*, pages 263–277. ACM Press, 1981.

[19] J. von Neumann. Various techniques used in connection with random digits. *U.S. National Bureau of Standards Applied Mathematics Series*, 12:36–38, 1951.

[20] Y. Yao and J. Gehrke. The Cougar approach to in-network query processing in sensor networks. *ACM SIGMOD Record*, 31(3):9–18, 2002.

[21] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proc. of IEEE Infocom*, pages 1567–1576, June 2002.