DIP: Density Inference Protocol for wireless sensor networks and its application to density-unbiased statistics*

Niky Riga Ibrahim Matta Azer Bestavros
Computer Science Department
Boston University
Boston, MA, USA

{inki, matta, best}@cs.bu.edu
Technical Report BUCS-TR-2004-023
May 31, 2004

ABSTRACT

Wireless sensor networks have recently emerged as enablers of important applications such as environmental, chemical and nuclear sensing systems. Such applications have sophisticated spatial-temporal semantics that set them aside from traditional wireless networks. For example, the computation of temperature averaged over the sensor field must take into account local densities. This is crucial since otherwise the estimated average temperature can be biased by over-sampling areas where a lot more sensors exist. Thus, we envision that a fundamental service that a wireless sensor network should provide is that of estimating local densities.

In this paper, we propose a lightweight probabilistic density inference protocol, we call DIP, which allows each sensor node to *implicitly* estimate its neighborhood size without the explicit exchange of node identifiers as in existing density discovery schemes. The theoretical basis of DIP is a probabilistic analysis which gives the relationship between the number of sensor nodes contending in the neighborhood of a node and the level of contention measured by that node. Extensive simulations confirm the premise of DIP: it can provide statistically reliable and accurate estimates of local density at a very low energy cost and constant running time. We demonstrate how applications could be built on top of our DIP-based service by computing density-unbiased statistics from estimated local densities.

1. INTRODUCTION

Motivation: Over the past few years sensor networks have received much attention as they are envisioned to support a wide range of important applications, *e.g.* surveillance systems, biological monitoring systems, environment control systems, equipment supervision systems, etc. A large number of such sensor applications are based on small, inexpensive, battery operated, electronic microsensor devices (*e.g.* Berkeley/Crossbow Motes [12], MIT μAMPS nodes [4]) with radio, sensing and processing components. Due to the size and cost restrictions, these *wireless* sensor devices have limited storage and computation capabilities. Furthermore access to the sensors maybe difficult or even impossible after the original deployment, which implies that the energy spent must be minimized to increase the lifetime of the system. The most energy intensive operation in these wireless devices is the radio operation which suggests that also the communication should be limited.

Many solutions have been proposed to cope with the above restrictions and improve the performance of this kind of wireless sensor networks. However, due to the wireless communication aspect, a significant body of previous work assumes that wireless sensor networks are just a variant of ad-hoc wireless networks with additional constraints. As pointed out by Ganeriwal $et\ al.$ [7] there are fundamental differences between ad-hoc wireless networks and sensor networks. Any kind of sensor network, resource constrained or not, is deployed to monitor the physical environment and therefore is highly coupled with the physical world. The sensors are periodically queried by an external source for summaries and statistical information about the underlying physical process.

In most of the previous work the sensors are thought to be uniformly distributed in the field and sometimes even to form a grid. In [9] Ganesan $et\ al.$ argue that in the majority of the cases there are going to be spatial irregularities. Thus the performance of many previous proposed solutions can be seriously affected when applied in non-uniform configurations. Besides performance deterioration, the correctness of the statistics computed from measurements collected by sensor nodes can also be affected. Ganeriwal $et\ al.$ [7] present a case where even a simple query like the average value of an area (e.g. temperature) can be miscalculated if the local density is not taken into account. This problem can be solved by having the sensors aware of density.

Our Contribution: Although there have been many proposals for neighborhood (local) density and topology discovery [15, 2, 26], to the best of our knowledge, all of these proposals require the *explicit* exchange of messages containing the node addresses/identifiers (and sometimes even their coordinates in the physical space). This typically requires some form of reliable broadcast which makes these schemes very expensive in terms of energy consumption and convergence time. In this paper, we present a *lightweight* distributed protocol for inferring (*implicitly* estimating) local density (neighborhood size) at each node. We henceforth refer to our Density

*This work was supported in part by NSF grants ANI-0095988, ANI-9986397, EIA-0202067, and ITR ANI-0205294.

Inference Protocol as DIP. DIP has the following salient features:

- DIP is based on a simple probabilistic analysis, thus it is easy to analyze the relationship between the level of contention observed at a node and the size of the contenting population (local density);
- Inferring local density without explicitly constructing it makes DIP communication-/energy-efficient since it avoids the (reliable) sending of specific messages carrying node identifiers, thus explicit retransmissions are not needed;
- Not relying on explicit messages makes DIP more resilient as well as more accurate in its estimation of local density since even sensed collisions contribute to the estimate—they are not simply lost messages that need to be retransmitted;
- The probabilistic basis of DIP allows for computing confidence levels for the computed estimate of local density, thus the running time of the protocol (henceforth referred to as *Density Inference Phase*) can be more easily controlled to achieve a certain energy-accuracy tradeoff;
- DIP provides a general basic service that could be used by a variety of sensor applications. We present in this paper such application which calculates density-unbiased approximate statistics via uniform spatial sampling over non-uniform sensor fields.

Paper Organization: Section 2 presents the sensor network model we assume in this paper, along with our DIP protocol and its analytical basis. Section 3 compares by simulation DIP against an explicit density discovery protocol that is typical of existing proposals. Section 4 describes an algorithm to calculate approximate statistics based on our proposed DIP protocol. Section 5 reviews related work (in addition to that mentioned throughout the paper), and Section 6 concludes the paper with future work.

2. DENSITY INFERENCE PROTOCOL

In this section we describe our network model and the probabilistic basis of our density inference protocol.

2.1 Model

Our model is summarized as follows:

- Without loss of generality and for ease of presentation, we assume that the sensor field is a square of side a
- There are N sensors in the field distributed according to a general density function $\lambda(x, y)$ defined over the x and y coordinates of the square field
- When a sensor transmits, every node within a range of radius r can hear its transmission
- All the sensors are synchronized using some synchronization protocol
- The energy spent for listening and receiving is proportional to the number of bits received[1]

[1]In the case of idle listening a node still has to intercept every bit to check the status of the carrier and to check the destination of each message.

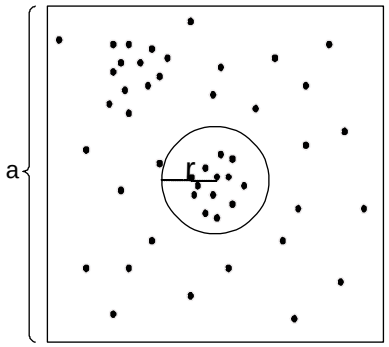


Figure 1: Non-homogeneous sensor field

- The energy spent for transmitting is larger than that for listening/receiving

Figure 1 shows an example of a non-homogeneous field.

2.2 Our Proposed DIP Protocol

We propose a probabilistic Density Inference Protocol (DIP) to estimate the number of neighbors of a node. Unlike existing *explicit* neighbor discovery protocols (e.g. [26, 2]), DIP runs in constant time and attempts to minimize energy. Using statistics the error of our approach can be bounded.

DIP is a contention-based MAC layer protocol. It uses the level of contention experienced by each node to estimate the density of the node's neighborhood. The contention that a node experiences while trying to transmit depends on the number of nodes trying to send at the same time. Our goal is to find a simple relationship between the contention a node observes on the carrier and the number of nodes competing.

Many contention-based MAC protocols have been proposed for wireless networks, and a lot of work have been done in analyzing their performance. The IEEE 802.11 protocol has drawn most of the attention and is widely deployed. The goal of IEEE 802.11 is to provide reliability while at the same time be efficient. Due to the complicated nature of the protocol, the analysis is quite complex. In the performance analysis provided in [25] and [1], the authors use numerical methods to solve the formulas that relate the number of nodes competing with the probability of a successful transmission.

Since we only care about measuring the level of contention and not provide a reliable MAC layer protocol, we can use a much simpler version of a CSMA protocol. Instead of exponential backoffs in times of collision, under DIP, nodes compete over a predefined number of slots, denoted by m. Each node chooses a slot out of these m slots. At the chosen slot a node transmits a message at a predefined range r, independent of whether it collides or not.

Let's find what is the expected number of nodes that will transmit in each slot. Let's assume that there are n nodes competing. Node i will choose to transmit its message during slot j with probability $p_{ij} = \frac{1}{m}$. We can think of this problem as a typical "bins and balls" problem. The sensors are equivalent to balls—each sensor will transmit one message in one slot—and the slots are equivalent to bins. Thus, by analogy, we have n balls and m bins. We throw

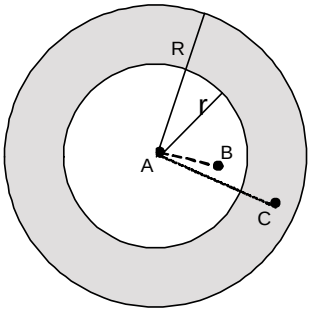


Figure 2: Interference problem in wireless communication. Although node A can only receive messages from node B, node C can interfere with the transmission of node B

each ball into one bin at random. Each throw is independent of the previous one. The quesion is: What is the expected number of balls in each bin? This problem is known as the "occupancy" problem [16].

Let's define a random variable X_{ij} as follows:

$$X_{ij} = \begin{cases} 1 & \text{if ball } j \text{ goes to bin } i \\ 0 & \text{otherwise} \end{cases}$$

The probability that ball j will go to bin i is $\frac{1}{m}$. So X_{ij} represents a Bernoulli trial. Let X_i be a random variable that counts the number of balls that go to bin i, i.e. $X_i = \sum_{j=1}^{n} X_{ij}$. Hence X_i follows the binomial distribution and we have:

$$Pr[X_i = l] = \binom{n}{l} \left(\frac{1}{m}\right)^l \left(1 - \frac{1}{m}\right)^{(n-l)}$$

From the above equation we can calculate what is the expected number of bins to **containing exactly l balls:**

$$E(l, n) = m\binom{n}{l} \left(\frac{1}{m}\right)^l \left(1 - \frac{1}{m}\right)^{(n-l)} \quad (1)$$

Now let's go back to our original problem. We have calculated what is the expected number of slots in which exactly l nodes collided. Although a node can tell whether or not there was a collision, it has no way of knowing now many nodes (within its reception range) had collided. Furthermore a node can not count the number of successful transmissions ($l = 1$) accurately due to the interference problem in wireless communication. A node correctly receives only messages that are within its reception range, which means that the signal is strong enough to guarantee an acceptable signal-to-noise ratio. Although the node does not correctly receive messages from nodes outside its reception range, these nodes can still interfere with the transmission of nodes within the reception range. Let's consider the example in Figure 2. Node A wants to count the number of successful transmissions of nodes within its reception range r. The interference range for node A is R. At slot i nodes B and C deside to transmit. Since only node B is within the reception range of A, A should count one successful transmission. However the transmission of C will make A observe a collision.

A sensor, though, can easily and accurately count the number of idle slots, i.e. $l = 0$.[2]

Substituting by $l = 0$ in Equation (1) we get:

$$E(0, n) = m\left(1 - \frac{1}{m}\right)^n \quad (2)$$

By inverting Equation (2), we can calculate n knowing m and the average number of idle slots. To obtain a *statistically* reliable estimate of that mean number of idle slots, we can repeat the experiment more times. We henceforth refer to these repeated experiments as *iterations* or *runs* of our DIP protocol.

The number of measured idle slots follows a distribution with mean given by Equation (2). Each time we run the protocol we draw a sample, say x_i, from this distribution. So if we repeat the protocol enough number of times, say ν, we can estimate the mean of the distribution as $\bar{X} = \frac{\sum_{i=1}^{\nu} x_i}{\nu}$. From statistics we know that the actual (true) mean of the distribution is bounded with probability α in an interval:

$$E(0, n) \in \{\bar{X}(0, n) \pm t_{\nu-1,1-\alpha/2} \times S(\bar{X}(0, n))\} \quad (3)$$

where $t_{\nu-1,1-\alpha/2}$ is the "critical point" of the t-Student distribution, and $S(\bar{X}(0, n))$ is the sample standard deviation given by $\sqrt{\frac{\sum_{i=1}^{\nu}(x_i - \bar{X})^2}{\nu(\nu-1)}}$.

From Equation (3) we can thus estimate a lower bound, L and an upper bound, U, on $E(0, n)$. By inverting (2) we get:

$$n = log\frac{E(0, n)}{m} / log\left(1 - \frac{1}{m}\right) \quad (4)$$

By substituting L and U for $E(0, n)$, we obtain a lower and an upper bound on our estimate of the neighborhood size n—this estimated range of the number of competing nodes will be within the true value with probability $\alpha\%$. Figure 3 shows the average number of nodes in the neighborhood of each sensor node, estimated using our DIP protocol, along with confidence intervals. The nodes in this example were distributed non-uniformly over the sensor field and the estimated neighborhood sizes are ordered on the X-axis from lower to higher density.

Our DIP protocol can be invoked periodically, so the sensors can update their estimate. How often the protocol should run depends on the specific application and on the dynamics of the field (e.g. how often sensor nodes die, new nodes join the network, nodes move). We refer to the time during which the nodes execute the DIP protocol as *density inference phase*.

In summary, DIP works as follows. We assume that every sensor has already received a message containing the input to the DIP algorithm, namely the number of slots m, the neighborhood range r, and the number of times to repeat the experiment ν. When the sensors enter the density inference phase then each sensor:

- chooses a random number s_i from $[1, m]$;

[2]Note that in the example of Figure 2, node C is outside the neighborhood (reception) range of node A and should not be counted in the estimate n. Indeed, if only node C transmits, node A would count this slot as an *idle* slot.

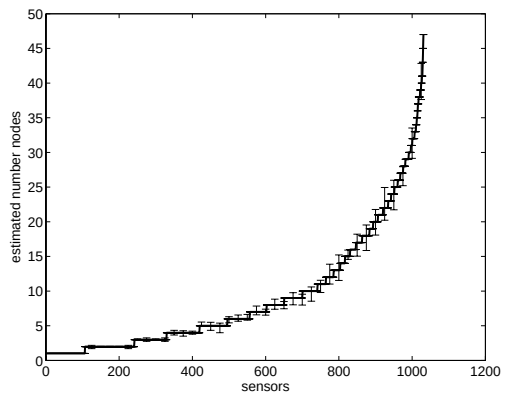


Figure 3: Estimated number of nodes in the neighborhood of each node with confidence intervals

- it transmits a message on the selected slot s_i; and
- for the remaining slots the sensor is in listening mode and counts how many slots were idle.

These steps are repeated ν times. After all the sensors have gathered ν samples, each sensor j estimates the number of nodes in its neighborhood n_j. Then it estimates its local density as $\lambda_j = \frac{n_j}{\pi r^2}$ and exits the density inference phase. Figure 4 shows in pseudocode an implementation of our DIP protocol.

Note that the actual message being transmitted during the DIP protocol is of no importance. It can be an empty message (the smallest message in Berkeley/Crossbow Motes is 27 bytes [12] [3].) Since the algorithm is based on the statistics of the transmissions, no retransmissions are necessary. The running time of the algorithm is a constant—it runs for $m \times \nu$ slots. If the bandwidth of the wireless link is 20 Kbps, each message is 27×8 =216 bits then each slot is 10.8 $msec$. Furthermore our algorithm runs in constant energy. Each node has to transmit only ν messages in each density inference phase and it has to be in idle/listen state for the rest of the $(m-1) \times \nu$ slots.

```
// m = number of slots per iteration
// ν = number of samples (iterations)
// r = neighborhood range
for s = 1 to ν do
   idle[s] = 0;
   choose at random j ∈ {1, m}
   for i = 1 to m do
      if j == i then
         transmit();
      end if
      if i is idle then
         idle[s]++;
      end if
   end for
end for
λ = calculate_density(idle[ ], m, ν, r)
```

Figure 4: Algorithm DIP(m, ν, r)

3. SIMULATIONS

In this section, we present simulation results that validate our proposed DIP protocol and compare its performance against an explicit neighbor discovery protocol typical of traditional approaches. We ran our simulations in Matlab [24]. We assume that all the links have equal bandwidth, so our base time unit is the time it takes for a node to receive/transmit one bit.

3.1 Parameters and Performance Measures

In our evaluation we use the following two performance metrics:

Normalized Error. Let $\hat{n}_j$ be the number of nodes in the neighborhood of node j estimated by DIP, and n_j be the actual number of nodes. The normalized error err_j is given by:

$$err_j = \frac{|n_j - \hat{n}_j|}{n_j}$$

Energy Consumed. This measure is the energy, E, expended during the execution of the protocol due to communication overhead. E is the sum of E_{tx}, the energy expended to transmit, E_{rx}, the energy expended to receive, and E_s, the energy expended while sensing the carrier. We assume that $E_s = E_{rx}$.

It is known that in radio communications the energy expended to transmit a message over a distance r is proportional to r^e where e is the path loss exponent, while the receive/sense energy is proportional only to the time the radio is on. Following the energy model used in [11], we take $e = 2$ and the following expressions for E_{tx} and E_{rx}:

$$E_{tx}(k, r) = E_{elec} \times k + \epsilon_{amp} \times k \times r^2$$

$$E_{rx}(k) = E_{elec} \times k$$

where k is the size of the transmitted message in bits, E_{elec} is the cost for just operating the radio, and ϵ_{amp} is the amplifier that adjusts the transmission power (range).

Table 1 lists the configuration parameters used in all our simulations. The packet size used is the smallest packet size for Berkeley/Crossbow Motes [12], as illustrated in Figure 5. Since in DIP each sensor calculates its local density, the distribution of sensors in the field does not affect DIP's performance. However for presentation and comparison reasons we assume that nodes are distributed uniformly over the sensor field with $\lambda = 0.1$, unless otherwise specified.

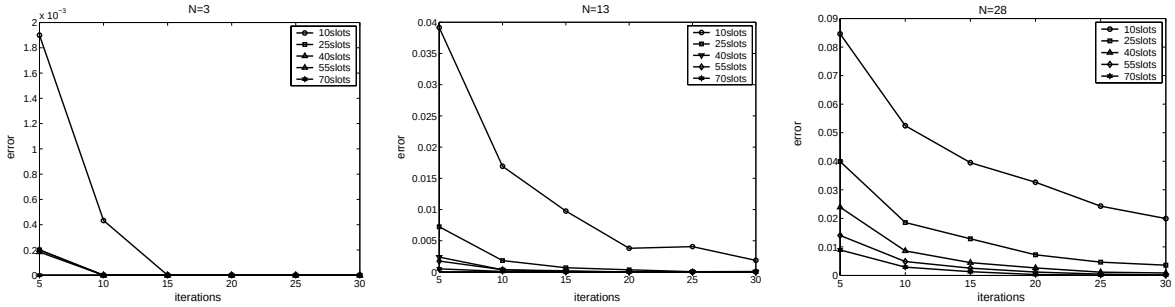
E_{elec}	$50nJ/bit$
ϵ_{amp}	$100pJ/bit/m^2$
field size	$100m \times 100m$
packet size	27 bytes

Table 1: Parameters used in the simulations

3.2 Results for DIP

Figure 6 shows the error err (averaged over all nodes) for increasing number of iterations and for varying number of slots, when the

[3]Since the contents of the packet are of no importance to DIP, the packet can be even smaller—it can be just a special radio signal so that the rest of the nodes know that someone in their neighborhood is transmitting.



(a) Avg no of competing nodes=3 (b) Avg no of competing nodes=13 (c) Avg no of competing nodes=28

Figure 6: The error err for various configuration parameters of DIP

PREAMBLE	SYNC	HEADER	PAYLOAD	CRC
18	2	5	(0-29)	2

Figure 5: Packet format

level of contention (represented by the average number of competing nodes) is set to 3, 13 and 28, which correspond to setting the range r to 1, 3 and 5, respectively. We notice that as long as the number of slots is sufficiently larger than the competing nodes then the error is very low even for a small number of iterations. Therefore if we choose a large enough number of slots, the iterations needed can be as low as 5, and that will provide a very small estimation error for any possible density in our field.

Figure 7 shows the energy consumption of DIP for various set of parameters. Note that the energy expended is independent of the density—DIP runs for a given number of slots and iterations. Based on the level of energy that an application is willing to expend, the DIP parameters can be adjusted accordingly, independent of the network topology, provided the resulting error in the estimation of local densities is acceptable.

3.3 Comparison against Explicit Approach

Next we compare our DIP protocol with a simple message exchange protocol based on unreliable broadcast. Each node broadcasts a hello message containing its identifier for l times. It sends the message more than once to increase the number of neighbors that will receive it successfully. We chose not to compare with a reliable broadcast protocol, since in most of these protocols the message is transmitted more than once with the additional cost of RTS/CTS messages for each transmission to ensure reliability. This would only increase the number of messages transmitted by each sensor and so deteriorate performance.

As a MAC layer transmission protocol we use the protocol implemented in Berkeley/Crossbow Motes [12]. This protocol is a CSMA/CA protocol using random backoff. When a node has a message to transmit it chooses a random delay between 1 and 128 bits. During this backoff period if the node hears another transmission it resets the backoff counter to a new random value and it starts counting at the end of the current transmission. After a node sends the l messages, it waits until it senses a silent period larger than the backoff timeout—with high probability that will indicate that there is no sensor in this node's neighborhood that has a message to transmit. The node then counts the number of distinct node identifiers that it had received and takes that as the number of neighbors it has. We henceforth refer to this protocol as the *explicit* (density discovery) protocol.

Figure 8 shows estimation error e versus the energy consumed during the execution of the explicit protocol. Each point on the graph represents the energy and accuracy achieved by sending l messages where l varies from 1 to 4. We show the results for different contention levels. Notice that the energy expended increases exponentially with the number of neighbors (the X-axis is in logarithmic scale). Since the number of messages sent by each node is constant the overhead of energy comes from the sensing of the channel. That means that it is also the case that the time it takes for the explicit protocol to terminate increases exponentially as the contention in the sensor field increases.

Table 2 reports the minimum energy consumption needed for the two protocols (DIP and explicit) to achieve an estimation accuracy of 95% for different levels of contention. We also report the parameters used to achieve this level of accuracy. We can see that the required energy for our DIP protocol is 1-2 orders of magnitude smaller than that required by the explicit density discovery approach.

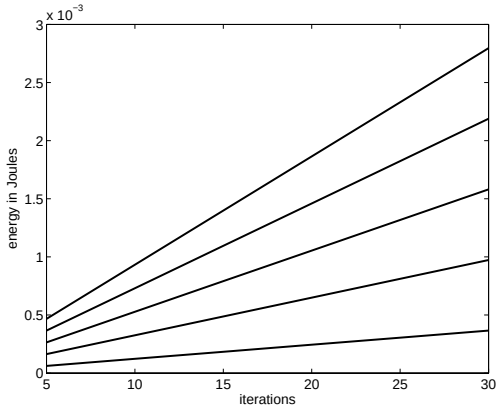


Figure 7: Energy consumption of DIP for various parameters

For the next experiment we used a non-homogeneous field. To create such a field, we divide the whole area into smaller regions and in each region we create a homogeneous subfield of sensors. The

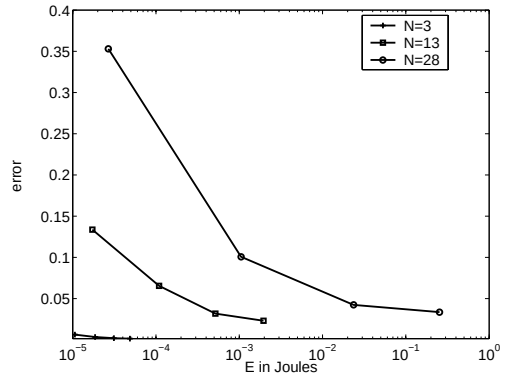


Figure 8: Energy consumption of the explicit protocol for calculating node density

		N=3	N=13	N=28
Explicit	energy	0.12mJ	0.5mJ	20mJ
	msgs	1	3	3
DIP	energy	0.061mJ	0.061mJ	0.16mJ
	slots	10	10	25
	iterations	5	5	5

Table 2: Minimum energy consumption needed to achieve accuracy of 95% for DIP and for the explicit protocol

value of the density of each region is chosen uniformly in $[\lambda_L, \lambda_H]$, so the standard deviation is $\sigma(\lambda) = \sqrt{\frac{1}{12}}(\lambda_H - \lambda_L)$. In the explicit density discovery protocol the time it will take each sensor to terminate the protocol, i.e. the sensor successfully sends the l messages carrying its identifier, depends on its local density. Therefore the time and energy expended by each node in a non-homogeneous environment is not a constant. Figure 9 illustrates how the standard deviation of the energy expended by each node within the field increases as the field becomes more irregular, i.e. as the standard deviation of the density increases. This is in sharp contrast to our DIP protocol, where the energy expended by all the sensors is constant and only depends on the DIP parameters, namely number of slots and number of iterations, and on the chosen neighborhood range r.

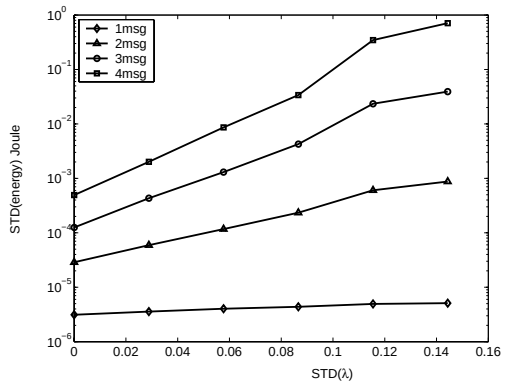


Figure 9: Standard deviation of the energy consumption in a non-homogeneous field as a function of the standard deviation of the density of the field

4. APPLICATION: COMPUTING APPROXIMATE STATISTICS

In this section we present an application that could be built on top of DIP. We use DIP to calculate unbiased approximate statistics for the underlying monitored process. For this application we consider the following setup: There are N sensors distributed in a field monitoring a physical process (e.g. temperature). There is a special node, called the "sink", that is interested in calculating statistics for this underlying process.

4.1 Limitations of Existing Approaches

In some of the previously proposed protocols (e.g. [8, 17]), the sink gets all the information, or a summary thereof, from the sensor nodes and then it is able to evaluate any aggregate query (e.g. average, maximum). Another set of approaches (e.g. [11, 14, 27, 28, 6]), propose to do in-network aggregation in order to reduce the number of messages traveling through the network and thus to consume less energy. In all of these approaches the objective is for the sink to gain knowledge about the values, or summaries on them, sensed by the sensor nodes.

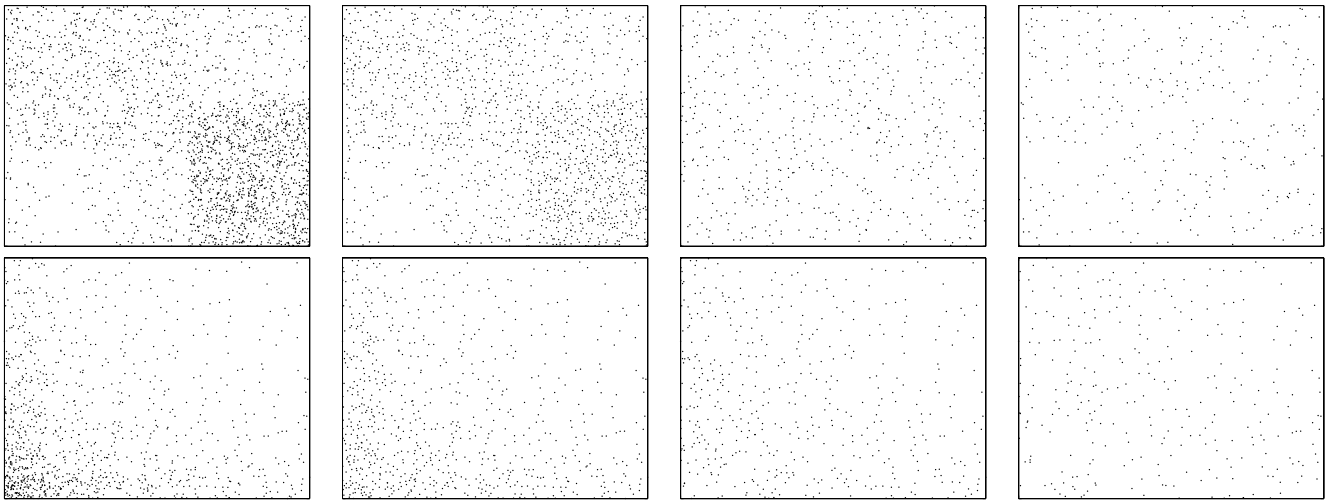
However the sink is interested in statistics concerning the physical process monitored by the sensors. To that end, Ganeriwal $et\ al.$ [7] have introduced this distinction between so-called *nodal aggregates* and *spatial aggregates*. They define as nodal aggregate the value calculated by simply applying an aggregate function over the set of values received at the sink, and as spatial aggregate the value calculated when the function is applied on the physical process. The latter accounts for the amount of physical space represented by a particular sensor node. Obviously if the sensors are uniformly distributed over the field then the nodal aggregate is a good approximation of the spatial aggregate. However, as discussed in [9] we shouldn't expect the distribution of the sensors over the field to be uniform. In most of the cases there are going to be areas of high density and areas of low density.

Without loss of generality let's consider a square field of size a and a set of sensors S distributed over this field. Let (x_i, y_i) be the physical location of sensor i. Let $V(x, y), x, y \in [0, a]$ be the function of the monitored process (e.g. temperature). The sink is interested in calculating functions like maximum$(V(x, y))$ or average$(V(x, y))$. Since the function $V(x, y)$ is unknown, the sink can only sample the function at specified points (where the sensors are physically located).

The best approximation is to try to reconstruct the function $V(x, y)$ from the known values. This solution requires, either the sensors to be location aware, or for the sink to know the physical locations of the sensors. An approximate approach is proposed in [7], where the Voronoi tesselation of the field is computed. The value of each sensor is assigned a weight proportional to the area covered by the Voronoi cell of the sensor. Two algorithms are proposed, a localized and a centralized one—the former requires that the sensors be location aware, and the latter does not allow for in-network aggregation.

4.2 Our Approach using DIP

A more natural way to provide a spatial unbiased estimate of an aggregate function is by removing the condition that introduced the bias in the first place. In other words the solution should attempt to make the distribution of the reported values uniform by drawing a spatial uniform sample of the sensors. Let A_1 ... A_k be subareas of



(a) Original field (b) Uniform sampling r=1 (c) Uniform sampling r=2 (d)Uniform sampling r=3

Figure 11: Uniform sampling: The output of our DIP-based uniform sampling algorithm: (a) the original non-homogeneous field; (b)-(d) uniform sampling for different values of range r

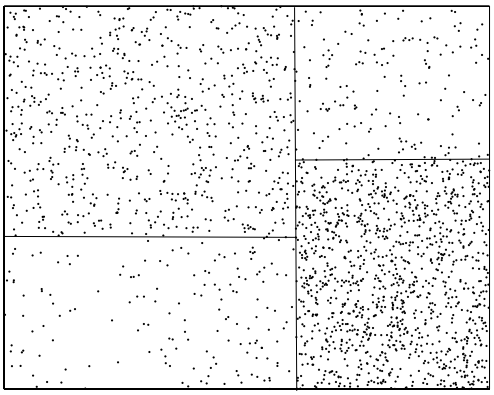


Figure 10: Partitioning of a non-uniform field into homogeneous subregions

the field such that $A_1 \bigcup A_2 \bigcup ... \bigcup A_k = A$, $A_1 \bigcap A_2 \bigcap ... \bigcap A_k = \emptyset$ and $\forall i \in [1 \cdots k] \lambda_{A_i}$ is constant where λ_{A_i} is the density of area A_i—see illustration in Figure 10. Assume each area contains N_i sensors where $\sum_{i=1}^{k} N_i = N$. Let n be the number of samples that the sink wants to receive. Then each area A_i should contribute n_i uniformly distributed samples where $n_i = \frac{A_i}{A} N$. Obviously if $N_i < n_i$ all of the N_i sensors would be part of the reported samples. The probability that a sensor j of area i will be part of the sample is given by:

$$p_j = \begin{cases} \frac{n_i}{N_i} & \text{if } N_i > n_i \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

Each sensor, after running DIP, has an estimate about N_i within an area $A_i = \pi r^2$, where r is the neighborhood range used in DIP. If we assume that A_i has a uniform density then each sensor using Equation (5) can estimate its probability of being part of the reported samples from area i. In summary, the protocol that we propose, on top of DIP, for performing uniform sampling over non-uniform fields involves the following tasks:

- The sink specifies a neighborhood range r, the number of samples n_i per neighborhood, and the round duration (which determines the frequency of reporting)
- Each sensor, using DIP, estimates the density of its local area within the range r, i.e. N_i, and then sets the probability that it sends its sensed value according to Equation (5)
- In each round each sensor decides if it is going to transmit its value to the sink with probability p_j

If the field is not very dynamic then the sensors don't have to run DIP in each round but every l rounds where again l is a parameter defined by the sink.

Figure 11 shows some execution examples of our DIP-based approach applied in different non-homogeneous fields, for different r. The sink requests 1 sample per neighborhood. We can see that for small values of r the sample is not uniform. This is due to the underlying physical limitations, *i.e.* if there are areas of size larger than πr^2 with not even one sensor inside they are not going to be represented. In essence, our DIP-based approach assumes that the requested density of the sample is smaller than the density of the most sparse area in the non-uniform field. Note that if the sink requests a sample density that is infeasible to provide, the sink could realize this after only a few rounds. Specifically, the sink knows that in each round it expects a total of n samples. If on average it gets less than n samples, this means that some areas are not represented, then the sink can decrease the requested density through a larger r (or smaller n_i).

4.3 Comparison against Density-oblivious Approach

In order to verify our claims about biased results produced by traditional density-oblivious approaches, we compare our density-aware

(DIP-based) sampling method with a simple density-unaware method. In the latter method the sink asks for n samples but since the sensors don't know their local density all the sensors have the same probability, $\frac{n}{N}$, to be part of the samples reported to the sink. This density-unaware sampling would inherit the spatial bias of the original distribution and will provide a biased answer. For this example we assume the *average* aggregate.

In order to be able to evaluate the quality of each method, we compare against the *true* mean and standard deviation of the monitored process. Let $\mu(V)$ be the mean and $\sigma(V)$ be the standard deviation of $V(x, y)$ where

$$\mu(V) = \frac{\int_0^a \int_0^a V(x,y)dxdy}{a^2}$$

$$\sigma(V) = \sqrt{\frac{\int_0^a \int_0^a (V^2(x,y) - \mu(V)^2)dxdy}{a^2}}$$

Figure 12 shows the average calculated with the two methods, for increasing sampling size. We also report the nodal aggregate and the spatial aggregate. The density-oblivious method gives the same results as the nodal aggregate, while our approach is much closer to the true aggregate value. As the sample size increases our approach also starts to diverge from the correct value. This is because of the underlying physical limitations of the field, meaning that given the original distribution of the sensors over the field there is no sample of the requested size that is uniformly distributed over the field, as a result of the desired density of the uniform sample being larger than the actual density of the most sparse area in the field.

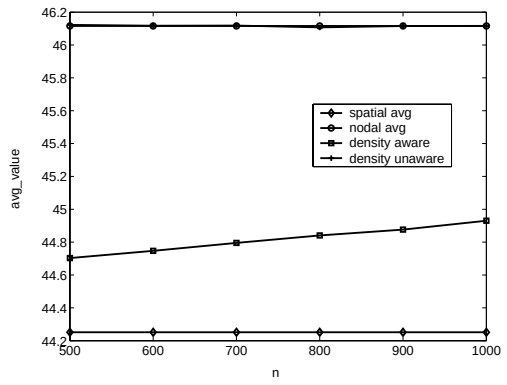


Figure 12: Average calculated using density-aware sampling and density-unaware sampling

4.4 Energy Savings using DIP

Besides providing unbiased statistics, our DIP-based approach also provides reduction in energy cosnsumption. After an initial cost of figuring out local densities, each sensor j sends on average, one value every $\frac{1}{p_j}$ rounds. Of course the energy savings depends on how dynamic the field is, i.e. how often the DIP protocol should run. If the topology doesn't change very often then one can achieve great savings using uniform DIP-based sampling. To validate this claim we ran experiments comparing the uniform sampling method versus sending all the values to the sink.

To disseminate all values to the sink we use LEACH [11]. LEACH is a routing protocol for data dissemination over wireless sensor networks based on clustering. Each sensor elects itself as a clusterhead with a probability p that is common for all the nodes, and then advertises its decision. The rest of the sensors send a join message to the clusterhead that is closest to the node. Then each clusterhead broadcasts a message to all its children with a schedule of when each child should send its value to the clusterhead. The clusterhead, after gathering all messages from its children, aggregates the values and sends only one message to the sink. In our simulation we don't take into account collisions during the clusterhead advertisement and join phase, and assume that all the messages are reliably transmitted and are received only by the receiver, i.e. the rest of the nodes don't overhear the transmission. Of course taking all these assumptions into account will only deteriorate the performance of the LEACH-based method.

For the transmission to the sink one-hop high-powered transmission is used for both our DIP-based scheme and for the LEACH-based scheme. Again collisions are not taken into account. Since LEACH is designed to run over uniform fields we ran the experiments over a uniform sensor field. Table 3 lists the configuration parameters used.

FIELD	
a	100
λ	0.2nodes/m^2
LEACH	
control packets	50bits
data packets	100bits
DIP	
slots	30
iterations	10

Table 3: Configuration Parameters used in the experiments comparing LEACH and DIP

Figure 13(a) shows how many nodes are alive in each round. The network lifetime with the LEACH-based scheme is 1170 rounds, while with our DIP-based scheme it is 4300, more than 300% increase. Figure 13(b) shows the average value obtained by both schemes. During the last 200 rounds in our approach the calculated value oscillates a lot. The reason is that in these last 200 rounds, less than 13 nodes are still alive out of 2000 nodes. In order to evaluate the quality of the result obtained using our approach, Figure 13(c) shows average value calculated over the first 4100 rounds, where there were still enough sensors to monitor the field, along with the true mean and standard deviation of the monitored process.

5. RELATED WORK

Work most closely related to ours is that done for neighborhood or topology discovery. If a node knows either its neighborhood information or the topology of the network then it clearly can estimate its local density. The main drawback of all these approaches is that their goal is to *explicitly* discover all neighbors of a node which significantly increases the cost.

Some of the protocols, such as AODV [18] and ZRP [10], propose that the neighborhood information be extracted by intercepting existing traffic. This approach implicitly makes two assumptions. First, packets contain the sender's ID, which is not always the case as in the MAC layer protocol of Berkeley/Crossbow Motes [12] where packets contain only the destination ID. Second, nodes are

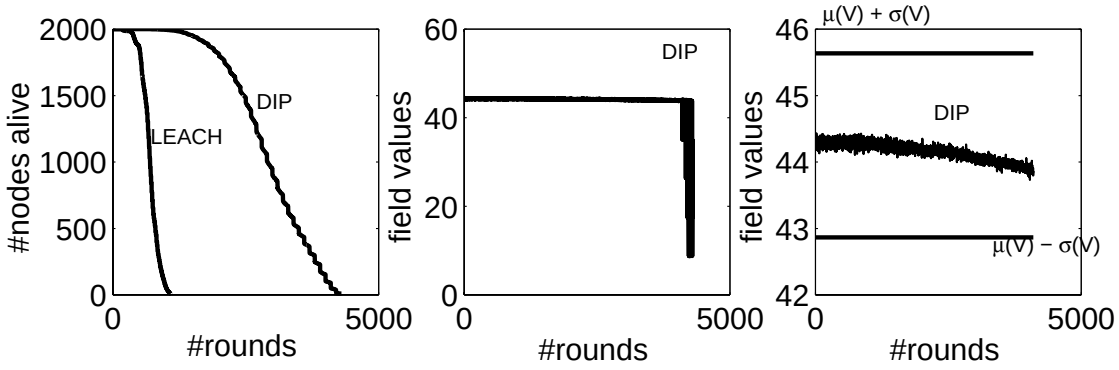


Figure 13: (a) Network lifetime; (b) Calculated average value using DIP; (c) The average calculated by our DIP-based approach compared with the true mean and standard deviation of $V(.)$

constantly awake to intercept the messages of all the nodes. This is not desirable in a sensor network environment, where the radio communication is a scarce resource that needs to be wisely used.

Other protocols, such as GAF [26], BMW [23], [5] and many others, assume the periodic broadcast of HELLO messages. In wireless networks the implementation of the broadcast functionality is not so easy to implement. For example the 802.11 standard doesn't provide reliable broadcast/multicast. Some solutions have been proposed to provide reliability on top of 802.11. Some of them [20, 21, 22, 23] attempt to extend the unicast scheme of RTS/CTS to provide reliable transmission for broadcast/multicast. Given that the size of the RTS and CTS messages is comparable to that of HELLO messages, communication overhead to add reliability is quite large. Another drawback of protocols relying on such explicit message exchange is the high contention during the discovery phase due to synchronized broadcasts of HELLO messages. On the other hand, if the nodes don't enter the discovery phase at the same time, as proposed in GAF [26], this implies that the nodes should stay awake longer to make sure that they will hear the HELLO transmission.

A different protocol proposed for neighbor discovery is the Birthday Protocol [15]. Although this protocol attempts to minimize contention by having nodes alternate between listening and sending states, its main objective remains to be that of maximizing the number of successfully transmitted messages resulting in increased communication cost. Furthermore in their analysis, McGlynn and Borbash only account for the energy consumed in transmitting, ignoring the energy due to receiving/listening. Work in the area of energy consumption have shown that the idle:receive:transmit ratios are 1:1.05:1.4 [19] while more recent studies show ratios of 1:2:5 [13].

Neighborhood/density estimation has also been a subject of study in the area of self-configuration in sensor networks [2, 3]. The goal of these studies has been to take advantage of the network density for routing purposes and to extend the lifetime of the system. However, the proposed architectures also need explicit neighborhood information so they use an explicit message exchange approach.

6. CONCLUSION

We introduced a density estimation service for wireless sensor networks. Our service is implemented by a lightweight probabilistic density inference protocol (DIP), which uses a simple relationship between the number of contending nodes in the neighborhood of a node and the contention level measured by that node to implicitly estimate its local density. We envision many applications built upon our service. We demonstrated one such application by computing density-unbiased approximate statistics. Our simulations confirm the superiority of our approach over existing explicit message exchange approaches in terms of consumed energy and convergence time while providing statistically reliable and accurate estimates of local densities.

We are currently developing other density-aware applications on top of DIP. We are also implementing DIP in Berkeley/Crossbow Motes and examining deployment issues including the dynamic triggering of DIP based of current network conditions. DIP code and results will soon be publicly available at *http://csr.bu.edu/dip*.

7. REFERENCES

[1] G. Bianchi. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *Journal on Selected Areas in Communications (J-SAC)*, 18(3), March 2000.

[2] N. Bulusu, J. Heidemann, D. Estrin, and T. Tran. Self-configuring Localization Systems: Design and Experimental Evaluation. *Trans. on Embedded Computing Sys.*, 3(1):24–60, 2004.

[3] A. Cerpa and D. Estrin. ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies. In *Proceedings of IEEE INFOCOM 2002*, New York, NY, June 2002.

[4] A. Chandrakasan, R. Min, M. Bharwaj, S.-H. Cho, and A. Wang. Power Aware Wireless Microsensor Systems. In *Proceedings of ESSCIRC*, September 2002.

[5] S. Y. Cho, J. H. Sin, and B. I. Mun. Reliable Broadcast Scheme Initiated by Receiver in Ad Hoc Networks. In *Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks*, pages 281–282. IEEE, 2003.

[6] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate Aggregation Technique for Sensor Databases. In *Proceedings of 20th IEEE International Conference on Data Engineering (ICDE)*, 2004.

[7] S. Ganeriwal, C. C. Han, and M. B. Srivastava. Spatial Average of a Continuous Physical Process in Sensor Networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, pages 298–299. ACM Press, 2003. Poster Abstract.

[8] D. Ganesan, D. Estrin, and J. Heidemann. DIMENSIONS: Why do we need a new Data Handling Architecture for Sensor Networks? In *Proceedings of the ACM Workshop on Hot Topics in Networks*, pages 143–148. ACM Press, 2002.

[9] D. Ganesan, S. Ratnasamy, H. Wang, and D. Estrin. Coping with Irregular Spatio-temporal Sampling in Sensor Networks. *SIGCOMM Comput. Commun. Rev.*, 34(1):125–130, 2004.

[10] Z. Haas, M. Pearlman, and P. Samar. The Zone Routing Protocol (ZRP) for Ad Hoc Networks, July 2002. Internet draft.

[11] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proceedings of the 33rd International Conference on System Sciences (HICSS '00)*, 2000.

[12] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next Century Challenges: Mobile Networking for Smart Dust. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 271–278. ACM Press, 1999.

[13] O. Kasten. Energy Consumption, 2001. www.inf.ethz.ch/~ kasten/research/bathtub/energy_consumption.html.

[14] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a Tiny AGgregation Service for Ad-hoc Sensor Networks. *SIGOPS Oper. Syst. Rev.*, 36:131–146, 2002.

[15] M. J. McGlynn and S. A. Borbash. Birthday Protocols for Low Energy Deployment and Flexible Neighbor Discovery in Ad Hoc Wireless Networks. In *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 137–145. ACM Press, 2001.

[16] R. Mortwani and P. Raghavan. *Randomized Algorithms*. Cambridge Univerity Press, 2000.

[17] R. Nowak, U. Mitra, and R. Willett. Estimating Inhomogeneous Fields Using Wireless Sensor Networks. *IEEE Journal on Selected Areas in Communications*, 2004.

[18] C. E. Perkins and E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computer Systems and Applications*, pages 90–100. IEEE Computer Society, 1999.

[19] M. Stemm and R. H. Katz. Measuring and Reducing Energy Consumption of Network Interfaces in Hand-held Devices. In *IEICE Trans. Fund. Electron Commun. Comp. Sci.*, volume 80, pages 1125–1131, 1997. Special Issue on Mobile Computing.

[20] M. Sun, L. Huang, A. Arora, and T. Lai. Reliable MAC Layer Multicast in IEEE 802.11 Wireless Networks. In *Proceedings of IEEE ICPP 2002*, pages 527–537, 2002.

[21] K. Tang and M. Gerla. MAC Layer Broadcast Support in 802.11 Wireless Networks. In *Proceedings of IEEE MILCOM 2000*, pages 544–548, 2000.

[22] K. Tang and M. Gerla. Random Access MAC for Efficient Broadcast Support in Ad Hoc Networks. In *Proceedings of IEEE WCNC 2000*, pages 454–459, 2000.

[23] K. Tang and M. Gerla. MAC Reliable Broadcast in Ad-Hoc Networks. In *Proceedings of IEEE MILCOM 2001*, pages 1008–1013, 2001.

[24] I. The MathWorks. MATLAB. http://www.mathworks.com/.

[25] H. Wu, Y. Peng, K. L. S. Cheng, and J. Ma. Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis and Enhancement. In *Proceedings of INFOCOM*, 2002.

[26] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed Energy Conservation for Ad Hoc Routing. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pages 70–84. ACM Press, 2001.

[27] Y. Yao and J. Gehrke. The Cougar Approach to In-network Query Processing in Sensor Networks. *SIGMOD Rec.*, 31(3):9–18, 2002.

[28] J. Zhao, R. Govindan, and D. Estrin. Computing Aggregates for Monitoring Wireless Sensor Networks. In *Proceedings of SANPA*, 2003.