Scale Free Aggregation in Sensor Networks

Mihaela Enachescu [a,1] Ashish Goel [a,2] Ramesh Govindan [b,3]
Rajeev Motwani [a,4]

[a] *Stanford University, Stanford, CA 94305-9025*

[b] *University of Southern California, Los Angeles, CA 90089-0781*

Abstract

Sensor networks are distributed data collection systems, frequently used for monitoring environments in which "nearby" data has a high degree of correlation. This induces opportunities for data aggregation, that are crucial given the severe energy constraints of the sensors. Thus it is very desirable to take advantage of data correlations in order to avoid transmitting redundancy. In our model, we formalize a notion of correlation, that can vary according to a parameter k. Then we relate the expected collision time of "nearby" walks on the grid to the optimum cost of scale-free aggregation.

We also propose a very simple randomized algorithm for routing information on a grid of sensors that satisfies the appropriate collision time condition. Thus, we prove that this simple scheme is a constant factor approximation (in expectation) to the optimum aggregation tree *simultaneously* for all correlation parameters k. The key contribution in our randomized analysis is to bound the average expected collision time of non-homogeneous random walks on the grid, i.e. the next hop probability depends on the current position.

Key words: Sensor networks, aggregation, random walk, collision time, routing, simultaneous optimization.

Email addresses: mihaela@cs.stanford.edu, ashishg@stanford.edu, ramesh@cs.usc.edu, rajeev@cs.stanford.edu (Rajeev Motwani).

[1] Research supported in part by NSF grant CCR-0126347.

[2] Research supported in part by NSF grant CCR-0126347 and NSF Career grant No. 0339262.

[3] Research supported in part by NSF grant CCR-0121778.

[4] Research supported in part by NSF Grants IIS-0118173, EIA-0137761, and ITR-0331640, and an SNRC grant.

1 Introduction

Consider a network where each node gathers information from its vicinity and sends this information to a centralized processing agent. If the information is geographically correlated, then a large saving in data transmission costs may be obtained by aggregating information from nearby nodes before sending it to the central agent. This is particularly relevant to sensor networks where battery limitations dictate that data transmission be kept to a minimum, and where sensed data is often geographically correlated. In-network aggregation for sensor networks has been extensively studied over the last few years [9,7,14]. In this paper we show that a very simple opportunistic aggregation scheme can result in near-optimum performance under widely varying (and unknown) scales of correlation.

More formally, we consider the idealized setting where sensors are arranged on an $N \times N$ grid, and the centralized processing agent is located at position $(0, 0)$ on the grid. We assume that each sensor can communicate only to its four neighbors on the grid. This idealized setting has been widely used to study broad information processing issues in sensor networks (see [12], for example). We call an aggregation scheme *opportunistic* if data from a sensor to the central agent is always sent over a shortest path, i.e., no extra routing penalty is incurred to achieve aggregation.

To model geographic correlations, we assume that each sensor can gather information in a $k \times k$ square (or, a circle of radius $k/2$) centered at the sensor. We will refer to k as the correlation parameter. Let $A(x)$ denote the area sensed by sensor i. If we aggregate information from a set of sensors S then the size of the resulting compressed information is $I(S) = |\bigcup_{x \in S} A(x)|$, i.e., the size of the total area covered by the sensors in S. Often, the parameter k can depend on the intensity of the information being sensed. For example, a volcanic eruption might be recorded by many more sensors and would correspond to a much higher k than a campfire. Accordingly, we will assume that the parameter k is not known in advance. In fact, we would like our opportunistic aggregation algorithms to work well simultaneously for all k.

There are scenarios where the above model applies directly. For example, the sensors could be cameras which take pictures within a certain radius, or they could be sensing RFID tags on retail items (or on birds which have been tagged for environmental monitoring) within a certain radius. Also, since we want algorithms that work well without any knowledge of k, our model applies to scenarios where the likelihood of sensing decreases with distance. For example, consider the case where a sensor can sense an event at distance r only if it has

"intensity" $f(r)$ or larger, where f is a non-decreasing function. Then, events of intensity y correspond to information with correlation parameter roughly $f^{-1}(y)$; if these events are spread uniformly across the sensor field then an algorithm which works well for all k will also work well for this case.

Thus, we believe that our model (optimizing simultaneously for all k) captures the joint entropy of correlated sets of sensors in a natural way for a large variety of applications, a problem raised by Pattem *et al.* [12].

For node (i, j), we will refer to nodes $(i-1, j)$ and $(i, j-1)$ as its *downstream* neighbors, and nodes $(i+1, j)$ and $(i, j+1)$ as its *upstream* neighbors. Since we are on a grid, we will also informally say that the neighbors are to the left or down/bottom (for downstream) and right or up/top (for upstream) of the original node (i, j). We would like to construct a tree over which information flows to the central agent, and gets aggregated along the way. Since we are restricted to routing over shortest paths, each node has just one choice: which downstream node to choose as its parent in the tree. In our algorithm, a node (i, j) waits till both its upstream neighbors have sent their information out [5]. Then it aggregates the information it sensed locally with any information it received from its upstream neighbors and sends it on to one of its downstream neighbors. The cost of the tree is the total amount of (compressed) information sent out over links in the tree.

Note that we do not need all sensors at a certain distance to transmit synchronously; we just need to make sure that a node sends its information only after both its upstream nodes have transmitted theirs. This can be enforced asynchronously by each sensor. In any case, Madden et al. [10] have developed protocols to facilitate synchronous sending of information by sensors (depending on the distance from the sink) which we can leverage if needed.

Our Results: We propose a very simple randomized algorithm for choosing the next neighbor – node (i, j) chooses its left neighbor with probability $i/(i+j)$ and its bottom neighbor with probability $j/(i+j)$. Observe that this scheme results in all shortest paths between (i, j) and $(0, 0)$ being chosen with equal probability [6]. We prove that this simple scheme is a constant factor approximation (in expectation) to the optimum aggregation tree *simultaneously* for all correlation parameters k. While we construct a single tree, the

[5] Of course maybe one, or both, of the upstream nodes may decide not to choose (i, j) as the parent node; however we assume that node (i, j) gets notified anyway when its upstream nodes send information out.

[6] Note that if you multiply the resulting probabilities, as the path approaches the origin the denominators are exactly the same for all the paths; the numerators are also the same (but permuted depending on the specific path).

optimum trees for different correlation parameters may be different.

The key idea is to relate the expected collision time of random walks on the grid [7] to scale free aggregation. Consider two neighboring nodes X and Y (i.e. nodes which can communicate with one another in our model), and randomly trace a shortest path from each of them to the sink. Define the collision time to be the number of hops (starting at say X) before the traces first meet. We first show (Sect. 3) that if the average expected collision is $O\left(\sqrt{N}\right)$, then we have a constant factor approximation algorithm to the optimal aggregation for *all* correlation parameters k. We then show that the average expected collision time for our randomized algorithm is indeed $O\left(\sqrt{N}\right)$ (Sect. 4). This analysis of the average expected collision time is our main technical theorem and may be of independent interest. To achieve this result, we first analyze the expected number of differing steps (where the two paths move in different directions) and then prove that the probability of a step being a differing step is a super-martingale.

We also present (Sect. 5) a slightly more involved hierarchical routing algorithm that is deterministic, and has an average collision time of only $O(\log N)$; hence the deterministic algorithm is also a constant factor approximation for all correlation parameters k. While this scheme has a slightly better performance, we believe that the simplicity of the randomized algorithm makes it more useful from a practical point of view.

Our results hold only for the total cost, and critically rely on the fact that information is distributed evenly through the sensor field. It is easy to construct pathological cases where our algorithm will not result in good aggregation if information is selectively placed in adversarialy chosen locations.

This result shows that, at least for the class of aggregation functions and the grid topology considered in this paper, schemes that attempt to construct specialized routing structures in order to improve the likelihood of data aggregation [6] are unnecessary. This is convenient, since such specialized routing structures are hard to build without some a priori knowledge about correlations in the data. With this result, simple geographic routing schemes like GPSR [8], or tree-based data gathering protocols are sufficient [7,10].

Related Work: Given the severe energy constraints and high transmission cost in the sensor network setting, data aggregation has been recognized as a crucial operation, which optimizes performance and longevity [4]. In the sensor network literature, aggregation can refer to either a database aggregate

[7] In the random walks considered here the probability of each move will depend on the current grid position.

operator (min,max,sum etc.) [1,10,11], or to general aggregation functions such as the one we consider in this paper.

Goel and Estrin [3] studied routing that leads to a simultaneously good solution (a $\log N$ approximation) for a large class of aggregation functions. In their model, the amount of aggregation only depends on the number of nodes involved, independent of location, and the network need not be a grid. In our problem, the amount of aggregation depends on the location of the sensors being aggregated: the closer two sensors are, the more correlated their data is. But it is also easier to aggregate data from nearby nodes. Hence, it seems intuitive that better simultaneous optimization may be possible for our case, an intuition that we have verified in this paper.

We build on the work of Pattem et al. [12] who study a closely related question, comparing three different classes of compression schemes for sensor networks: routing-driven compression, in which the routes from the nodes to the destination point just follow a shortest path, and in which compression is done opportunisticaly whenever possible, compression-driven routing which builds up a specialized routing structure, and distributed source coding which leverages a priori information about correlations. After a theoretical analysis of these schemes, they introduce a generalized cluster-based compression scheme in which correlated readings are aggregated at a cluster head, which is studied via simulations. They find that across a wide variety of correlations (roughly parameterized by the joint entropy of two sensors spaced d apart), the cluster-based compression scheme works reasonably well with a relatively fixed cluster size. Our model captures a wider range of joint entropy functions (since we also approximate any linear composition of k-correlated information for different values of k), one of the open problems they pose. Also, we present a formal proof of simultaneous optimization. It is easy to see that their cluster-based compression scheme does not perform well in our model, in that no single cluster size can be within constant factor of the optimal aggregation tree for all k.

Another study of routing schemes for correlated sensors has been performed by Cristescu et al. [2]. They showed that for a two-stage model where the amount of information depends only on whether a node is an internal node or a leaf, finding the optimum aggregation tree is NP-hard; they also present a constant factor approximation for this problem. Their result holds for an arbitrary sensor network (as opposed to just a grid).

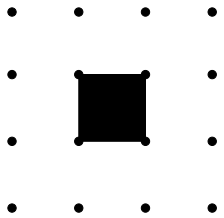

Fig. 1. The $k \times k$ sensors that detect a given value (the black square) for $k = 4$

2 Problem Definition

Recall our setting in which sensors are arranged in a $N \times N$ grid with a centralized processing agent at $(0, 0)$. Each sensor can only communicate with its immediate neighbors on the grid (at most four). We can assume that each sensor knows it's (x,y)-coordinates. This can be done for example via the fine-grain localization method described by Savvides et al. [13].

The sensor network can sense multiple kinds of data. For a specific type of data, we will refer to the information contained in a 1×1 grid square as a *value*. We then define this type of data k-correlated data if the following holds:

(i) Each value is sensed by all the sensors in a $k \times k$ grid centered at that location, as in Fig. 1. We will assume for simplicity that k is even, so that the notion of centering is well defined.

(ii) Let $A_k(x)$ denote the set of grid squares sensed by sensor x. If information from a set S of sensors is aggregated, the resulting information is of size $|\bigcup_{x \in S} A_k(x)|$

We will look at k-correlated data for which $k < N/2$, since otherwise we obtain a pathological case in which all information can be captured by a single sensor in the network. As stated in the introduction, we need not assume that the nodes equidistant from the central agents send data synchronously. However, we assume the transmission is partially synchronized, so that a node sends information only after receiving all data from its upstream neighbors and after finishing aggregating that information with its own data. We assume that a sensor cannot withhold information, and needs to send all information it can sense.

Given this setting, we want to find a tree on which to send information from all sensors to $(0, 0)$ so as to minimize the cost, simultaneously, for all values of k. In our cost model we focus on the transmission cost, assuming perfect aggregation, i.e. assuming a value v is transmitted across an edge e at most once. Each time such value is transmitted from some node to its downstream neighbor in the routing tree, the total cost increases by 1. More formally the cost we consider is given by the following equation:

$$\text{COST} = \Sigma_e \, |\{v|\text{value } v \text{ is transmitted across edge } e\}|$$

Theorem 2.1 *Lower bound on optimum cost (OPT) is* $\theta\left(N^3 + (Nk)^2\right)$.

Proof: Look at one individual value, at point (x, y) with $x, y \geq 0$ and construct the minimum cost routing for it.

The closest node to the origin that senses this value, at coordinates $(x - k/2, y - k/2)$, has to send the value all the way to the origin, so a cost of $D = x + y - k$ (distance from the point to the origin must be paid). All values incur this cost.

The value must be transmitted by all the nodes that can sense it, each node thus introducing a cost of 1. Thus, for all values for which the sensing $k \times k$ square is included in the $N \times N$ grid (it is easy to observe that there are $(N-k)^2$ such values), there is a cost of at least k^2 before the distinct values can be aggregated at a single node. We ignore this contribution to the cost for the other values.

Since we assume different values cannot be aggregated between them, we get a lower bound for the overall cost of at least:

$$\sum_{\text{values}} D + (N-k)^2 k^2 = 4 \left(\sum_{0<x<N, 0<y<N} (x + y - k) \right) + (k(N-k))^2$$

$$= N^2(N - 1 - k) + (k(N-k))^2$$

If we only consider parameters $k < N/2$ then $N - 1 - k \leq N/2$ and the above becomes $\theta\left(N^3 + (Nk)^2\right)$ as desired.

There may not exist a single tree which is optimum for each value v. This is because from the point of view of a value v_1 a certain sensor may need to communicate with one downstream neighbor for optimal aggregation, while from the point of view of another value v_2 that same sensor may need to communicate with the other downstream neighbor. This would lead to an impossible solution for the routing tree in which only one downstream neighbor can be selected. However, our analysis, while not solving for the exact value, does give a lower bound on OPT.

■

3 Relating Opportunistic Aggregation to Collision Time

Recall our definition of *opportunistic* algorithm, i.e. one in which the information from node X is sent to the processing agent on a shortest path. Note that the paths from any two neighbors X and Y, having the same destination, will eventually meet at some point Z. We call the distance from X to Z the *collision time* of X and Y.

Theorem 3.1 *An opportunistic algorithm with average expected collision time* $O\left(\sqrt{N}\right)$ *gives a constant factor approximation to the optimum aggregation cost for all* k.

Proof:

In a similar fashion as in our proof for the lower bound for the OPT, look from the point of view of a data value which is shared by $k \times k$ sensors. Inside the square we pay the same cost as in our lower bound for the OPT (i.e. at most k^2). Also the lower-left node transmits the value to $(0, 0)$ via a shortest path just as in OPT. So far, the cost is the same.

To analyze the extra cost incurred by the opportunistic algorithm from our hypothesis, consider the left and lower sides of the $k \times k$ sensing square of a given value. The paths from all sensors inside the square will go through one of the points on these sides. Consider these paths from the sides of the square to $(0, 0)$. There is some extra cost equal to the collision time between two adjacent nodes from the left and lower sides of the square, since two instances of the shared value are transmitted, instead of only one instance as would happen in the optimal case. It is easy to see that for each pair of adjacent nodes, there are k values that incur the extra cost due to the collision time, or put another way, k shared values for which these nodes are on the sides. Summing up the OPT cost and the extra cost we obtain the following equation for the total cost of our algorithm:

$$\sum_{\text{values}} D + k^2N^2 + k \sum_{\text{sensors}} (\text{collision time of the paths from two adjacent sensors})$$

$$= \theta\left(N^3 + (kN)^2\right) + O\left(kN^{2.5}\right)$$

The first two terms are the same as in the lower bound for OPT.

If $k < \sqrt{N}$ then the N^3 term dominates the $(Nk)^2$ term, as well as the $k \times N^{2.5}$ term, and we get an $O(1)$-approximation.

If $k > \sqrt{N}$ then the $(Nk)^2$ term dominates the other two terms, and we get again an $O(1)$-approximation.

Note that we compare to a lower bound for OPT, not OPT itself, which may be hard to compute, so the constant factor may be even less than what we can compute here. ■

4 The Probabilistic Distribution Shortest Path Algorithm

We will present a simple randomized opportunistic algorithm for constructing a tree. The path from each node will be a random walk towards the processing agent, but the walks are not independent. The main result is to prove that the average expected collision time of two adjacent paths in the resulting routing tree is $O\left(\sqrt{N}\right)$. The analysis of our random process may well be of independent interest. Applying theorem 3.1, we conclude that this algorithm produces a constant factor approximation of the optimal aggregation trees for any value of k.

The Probabilistic Distribution Algorithm: For every node, if the node is located at position (x, y), choose to include in the MST the left edge with probability $\frac{x}{x+y}$ and the down edge with probability $\frac{y}{x+y}$.

The Random Walk view: We can view the above process as a tree constructed from random walks originating from each sensor. At each time step the current node chooses one of the (at most) two downstream nodes as its parent. Because a node waits for its upstream nodes to transmit we can view the process as a flow in which the data gets closer by one to the origin at each time step. In our model, when two walks meet (passing some step through the same node) they "collapse" into a single walk and lose their independence. The analysis of the expected collision time for this random process is presented below. We believe our analysis is interesting since the random walk is non-homogeneous, thus standard random walk results do not apply.

4.1 Proving the Average Collision Time of the Random Walks

Theorem 4.1 (Random Walk Theorem) *The average expected collision time of two adjacent walks as generated by the randomized probabilistic distribution algorithm is* $O\left(\sqrt{N}\right)$.

Let us first introduce some notation, definitions, and lemmas which would help us prove the above result.

Two neighboring nodes can be either horizontal or vertical neighbors, and one, say the second, must be the upstream neighbor of the other. Thus there is a $\frac{x}{x+y}$ probability to meet initially. If they do not meet initially, then the

upstream node chooses as its parent the other downstream node, which on the grid is at distance 2 from the first node, and at the same distance from the origin.

Let's assume that the two walks do not meet initially. Thus, we will analyze the collision time of the random walks originating at $(x-1, y)$ and $(x, y-1)$. This new "diagonal" collision time provides a lower bound in the collision time of the initial "horizontal/vertical" neighbors. In fact we will prove the result stated in our Random Walk Theorem for this redefined notion of collision time, which then implies the original theorem.

Note that, because the nodes are at the same distance from the origin we can imagine them moving towards the processing agent "in sync" (this synchronicity assumption is not needed but it helps in thinking about the process). Look at the horizontal difference between the two paths, as a function of time, and let's denote this by $\Delta_t(x, y)$. Initially, $\Delta_0(x, y) = 1$. Because in general we focus our attention to a specific (x, y) we will drop these parameters from the notation. We want to analyze $E[t_c]$ where t_c is such that $\Delta_t = 0$ for the first time. Observe that t_c is precisely the collision time as defined earlier, since the two walks start from the same distance from the origin, and at every time step we assume the walks move one unit closer, so there is a one-to-one correspondence between time and distance from the initial point to the collision point. Once the horizontal distances become equal, the vertical distances must also be equal and the two paths would meet.

Let $M = x + y - 1$, the initial distance from the origin.

At each time step, Δ_t can stay the same or become different (increase or decrease by 1). We call a step at which Δ_t differs from Δ_{t-1} a *differing step*. By analyzing these differing steps we will transform the problem from a two dimensional process to a one dimensional process.

We will first analyze the number of differing steps before collision (Lemmas 4.2 and 4.3), and then bound the probability that a step is a differing step (Lemmas 4.4 and 4.5). These results together will lead to the proof of the main result.

Definition 4.1 *Let's denote by $D(x, y)$ the number of differing steps before Δ_t becomes 0 for the first time.*

Lemma 4.2 *$E[D(x, y)]$ is $O\left(\sqrt{min(x, y)}\right)$.*

Proof: At time t, when the first path is above at, say, point (x_1, y_1) and the second path is below at point (x_2, y_2) we know that $x_1 + y_1 = x_2 + y_2 = M - t$. Initially $x_1 < x_2$, so this above/below relation will continue to hold until $\Delta = x_2 - x_1$ first becomes 0. Also, initially, $x_2 = x$ and $y_1 = y$.

Based on our probabilistic model, and the above/below relation we derive the following for the next time step:

$$\Pr(\Delta_{t+1} - \Delta_t = 1) = \frac{x_1 y_2}{(x_1 + y_1)^2} \text{ and } \Pr(\Delta_{t+1} - \Delta_t = -1) = \frac{x_2 y_1}{(x_1 + y_1)^2}$$

Using $y_1 = M - t - x_1$ and $y_2 = M - t - x_2$ we obtain the following:

$$\Pr(\Delta_{t+1} - \Delta_t = 1) + \Pr(\Delta_{t+1} - \Delta_t = -1) = \frac{(M-t)(x_1 + x_2) - 2x_1 x_2}{(M-t)^2}$$

and $\Pr(\Delta_{t+1} - \Delta_t = 1) - \Pr(\Delta_{t+1} - \Delta_t = -1) = -\frac{\Delta_t}{M-t}$.

Now define $p_f(t) = \Pr(\Delta_{t+1} - \Delta_t = 1 | \Delta_{t+1} - \Delta_t \neq 0)$ and $p_r(t) = \Pr(\Delta_{t+1} - \Delta_t = -1 | \Delta_{t+1} - \Delta_t \neq 0)$ to be the conditional (normalized) probabilities of a forward (positive) change in Δ, and of a reverse (negative) change in Δ, respectively.

Also define λ as below: [8]

$\lambda = p_f(t) - p_r(t) = \frac{\Pr(\Delta_{t+1} - \Delta_t = 1) - \Pr(\Delta_{t+1} - \Delta_t = -1)}{\Pr(\Delta_{t+1} - \Delta_t = 1) + \Pr(\Delta_{t+1} - \Delta_t = 1)} = -\frac{\Delta_t (M-t)}{(M-t)(x_1 + x_2) - 2x_1 x_2}$

Since $p_f(t) + p_r(t) = 1$, we can rewrite $p_f(t)$ and $p_r(t)$ as:

$$p_f(t) = \frac{1}{2} + \frac{\lambda}{2} \text{ and } p_r(t) = \frac{1}{2} - \frac{\lambda}{2}$$

where λ still contains a dependence on t. The convergence to $\Delta = 0$ can only be slower if λ is smaller in absolute value. Note that by removing the $2x_1 x_2$ term from the denominator of λ we can only decrease the overall absolute value of λ. Also, we get the same effect if we replace $x_1 + x_2$ by $2\max(x_1, x_2) = 2x_2$.

Also, the $M - t$ factor will get simplified so we can replace λ by $-\frac{\Delta}{2x_2}$ to obtain new forward and reverse probabilities, independent of t and only dependent on Δ:

$$n_f(\Delta) = \frac{1}{2} - \frac{\Delta}{4x_2} \text{ and } n_r(\Delta) = \frac{1}{2} + \frac{\Delta}{4x_2}$$

Now consider an integer random walk in $[0, \max(x_1, x_2) = x_2]$, with an absorbing barrier at 0, and a reflecting one at $\max(x_1, x_2) = x_2$.

We analyze the behavior of this one dimensional random walk in lemma 4.3. By construction, the expected time for this new random walk to reach 0 starting

[8] Note that λ is negative.

from 1 is an upper bound to the expected time for Δ to reach 0 starting from 1.

We can then conclude that Δ reaches 0 in $O\left(\sqrt{x_2}\right)$ by directly applying the result in lemma 4.3. By symmetry we can also obtain time $O\left(\sqrt{y_1}\right)$. Since $x_2 = x$ and $y_1 = y$ initially, the theorem is proven.

■

Lemma 4.3 *Consider an integer random walk starting at point* 1 *on the interval* $[0, x]$. *Assume that, if we are at position* j *the random walk moves right with probability* $n_f(j)$, *and left with probability* $n_r(j)$ *in the interval* $[1, x-1]$ *where* $n_f(j)$ *and* $n_r(j)$ *are as defined in lemma 4.2. Assume that the point* 0 *is absorbing, and that the point* x *is reflecting (i.e. the walk moves to* $x-1$ *with probability* 1 *from* x*). If the walk starts at point* 1, *then the expected number of time steps necessary for this walk to first reach* 0 *is* $O(\sqrt{x})$.

Proof:

Note that at each step we move either in one direction or the other, since, by definition, $n_r + n_f = 1$.

Define $B(j)$ to be the expected number of steps before the point $j-1$ is first visited, assuming that the random walk starts at point j. We are then looking for the value of $B(1)$. We will use the properties of the walk, in particular the values of $n_f(j)$ and $n_r(j)$ to derive a recursive formula for $B(j)$ and then get a bound for $B(1)$.

If we pass exactly $i+1$ times through point j before reaching point $j-1$, the expected number of steps is $iB(j+1)+1$. The probability of this event is $n_r(j)n_f(j)^i$. Since i can range from 0 to ∞ we get the following relation for $B(j)$, where $j \in [1, x-1]$:

$$B(j) = \sum_{i=0}^{\infty} n_r(j)n_f(j)^i(iB(j+1)+1) = n_r(j)\sum_{i=0}^{\infty} n_f(j)^i + n_r(j)B(j+1)\sum_{i=0}^{\infty} n_f(j)^i i$$

$$= \frac{n_r(j)}{1-n_f(j)} + \frac{n_r(j)n_f(j)}{(1-n_f(j))^2}B(j+1) = 1 + \frac{n_f(j)}{n_r(j)}B(j+1) = \frac{2x-j}{2x+j}B(j+1)+1$$

Further note that $B(x) = 1$ because x is a reflecting barrier, so in the next step we move back with probability 1.

We want to solve for $B(1)$, the value of interest.

If we expand $B(1)$ in terms of $B(x)$ we obtain:

$$B(1) = \sum_{i=1}^{2x} \frac{2x-1}{2x+1} \times \ldots \times \frac{2x-i}{2x+i}$$

To simplify notation, denote $2x$ by X and $\frac{2x-1}{2x+1} \times \ldots \times \frac{2x-i}{2x+i}$ by T_i.

Note that the T_i's are decreasing as i increases, since all component factors are less than 1. Now, note that for $i \in \{1, \ldots, 2\sqrt{X}\}$ we have $T_i \leq 1$.

For $i \in \{2\sqrt{X}+1, \ldots, 3\sqrt{X}\}$ we have $T_i \leq \left(\frac{X-\sqrt{X}}{X+\sqrt{X}}\right)^{\sqrt{X}}$ since the last $\sqrt{X}$ factors in each of these T_i are all less than $\frac{X-\sqrt{X}}{X+\sqrt{X}}$.

In general, for any m, if $i \in \{m\sqrt{X}+1, \ldots, (m+1)\sqrt{X}\}$ we have $T_i \leq \left(\frac{X-\sqrt{X}}{X+\sqrt{X}}\right)^{\sqrt{X}(m-1)}$.

Thus $B(1)$ can be upper bounded by a geometric series with sum $\frac{X+\sqrt{X}}{2\sqrt{X}}$. Note that the $\left(\frac{X-\sqrt{X}}{X+\sqrt{X}}\right)^{\sqrt{X}}$ is approximately e^2, and thus constant, for large enough X, where $X = 2x$. Thus, the first term (the fraction) of this bound is a constant, and we conclude that $B(1)$ is $O(\sqrt{x})$.

■

Definition 4.2 *Define* $p_t(x,y) = \Pr[\Delta_t$ *is differing* $\mid$ *two walks have not collided yet*$]$.

As before, we will omit the arguments x, y since they are fixed.

Lemma 4.4 *For all* t, $p_{t+1} \geq p_t$

Proof:

Suppose the first walk is at coordinates (i,j) and the second one at coordinates $(i+\Delta_t, j-\Delta_t)$.

Case 1 ($\Delta_t \geq 2$): Then $\Delta_{t+1} \geq 1$, since the difference between Δ_t and Δ_{t+1} can be at most 1. Thus the random walks would not meet at time $t+1$, so we eliminate the conditioning for p_{t+1}, and we have the following:

$$\Pr[\Delta_t \text{ is differing}] = f(i,j,\Delta_t) = \frac{i(j-\Delta_t) + j(i+\Delta_t)}{(i+j)^2}$$

$$\Pr[\Delta_{t+1} \text{ is differing}] = g(i,j,\Delta_t) = \frac{i(j-\Delta_t)f(i-1,j,\Delta_t+1)}{(i+j)^2} +$$

$$+\frac{i(i+\Delta_t)f(i-1,j,\Delta_t)}{(i+j)^2} + \frac{j(i+\Delta_t)f(i,j-1,\Delta_t-1)}{(i+j)^2} + \frac{j(j-\Delta_t)f(i,j-1,\Delta_t)}{(i+j)^2}$$

It is easy to verify, using Mathematica for example, that $f(i,j,\Delta_t) - g(i,j,\Delta_t) = 0$, and hence, $p_t = p_{t+1}$.

Case 2 ($\Delta_t = 1$): In this case the conditioning in the definition of p_{t+1} implies that one of the cases in the above formula cannot take place. We still obtain that $p_{t+1} \geq p_t$, but the details are technical and are deferred to the appendix.

■

Lemma 4.5 *The expected time before a differing step between two adjacent walks originating at coordinates* (x, y) *and* $(x+1, y-1)$ *is* $O\left(\frac{x+y}{\min(x,y)}\right)$.

Proof:

From lemma 4.4 we see that at each time step the probability of a differing step is bounded below by p_0 which is the initial probability of having a differing step, given by:

$$\frac{x(y-1)+(x+1)y}{(x+y)^2} = \theta\left(\min(x,y)\frac{2\max(x,y)}{(x+y)^2}\right) = \theta\left(\frac{\min(x,y)}{x+y}\right)$$

This implies our lemma. ■

Proof: [Random Walk Theorem]

Consider two walks at (x, y) and $(x+1, y-1)$.

From Lemma 4.5 we bound the probability of a differing step to happen. Combining this with the result from Lemma 4.2 which bounds the expected number of differing steps before the two walks meet we obtain:

$$E[\text{collision time for adjacent walks at } (x,y)] = \theta\left(\sqrt{\min(x,y)}\frac{x+y}{\min(x,y)}\right) = \theta\left(\frac{x+y}{\sqrt{\min(x,y)}}\right)$$

Taking the sum over all x, y pairs we obtain: $\sum_{x,y}\left(\frac{x+y}{\sqrt{\min(x,y)}}\right) = \theta(N^{2.5})$.

Thus, since there are $O(N^2)$ pairs of adjacent nodes, the average is exactly $O(\sqrt{N})$, which concludes the proof of the expected average collision time theorem.

■

5 The Hierarchical Decomposition Approach

We now present a deterministic algorithm for constructing a tree that produces a constant factor approximation for any value of k. This algorithm has better properties (its average collision time is $O(\log N)$ instead of $O\left(\sqrt{N}\right)$ for example), but it is more involved. Also the approximation provided is still $O(1)$.

The solution is based on the idea of dividing the grid into sub-grids, and collecting all the values in a given sub-grid at the sensor closest to the origin

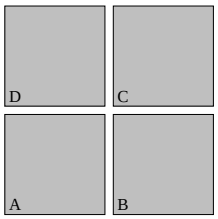

Fig. 2. Combining 4 smaller sub-grids to create the sub-grid at the next level

before forwarding it onto the next sub-grid.

5.1 The Hierarchical Decomposition Algorithm

We present the construction and the proof of correctness in parallel. We need two stages: a top-down stage in which we establish the sub-grids recursively, and a bottom-up stage in which we put the sub-grids together. We will assume for simplicity that N is a power of 2.

The Top-Down Stage: Divide the first quadrant in four sub-grids of size $N/2 \times N/2$, each of which is further divided in four size-$N/4 \times N/4$ sub-grids, and so on. For each sub-grid we will make sure that the MST converges to the sensor closest to the origin, i.e. if there is choice in what direction to move towards the origin, choose the choice that would not leave the sub-grid. If there is still choice choose arbitrarily.

The Bottom-Up Stage: We will prove by construction the following lemma.

Lemma 5.1 *If a $2^k \times 2^k$ sub-grid has the property that its average collision time is less than ck for all adjacent node pairs in the sub-grid, then we can construct a $2^{k+1} \times 2^{k+1}$ sub-grid with average collision time of $c(k+1)$ for all adjacent node pairs, where c is some constant greater than 2.*

Proof:

We assume the parent node is determined for all nodes inside the $2^k \times 2^k$ sub-grid, and thus we have constructed an MST, rooted at the sensor node closest to the origin, such that the property is true. If we combine four copies of this construction, as in Fig. 2 we need to establish the parent node of the three root sensors B, D, and C representing the upper-left, lower-right, and upper right sub-grids respectively. For the first two the choice is forced (the sensor at B needs to go left, and the one at D needs to go down). For the third (the sensor at C) let us route to the left.

Now calculate the new average for the $2^{k+1} \times 2^{k+1}$ sub-grid, assuming the hypothesis holds for the $2^k \times 2^k$ ones.

We have $2(2^k)^2$ pairs included in each of the 4 smaller sub-grids, and thus have

average less then ck, from the hypothesis. We also have 2^{k+2} new pairs (the ones spanning the white lines) that have collision time bounded by 2^{k+2}. Thus we obtain a new average collision time of: $\frac{8ck(2^k)^2+(2^{k+2})^2}{2(2^{k+1})^2} \leq ck+2 \leq c(k+1)$ as long as $c > 2$.

The base case is trivial.

■

6 Conclusions and Future Work

In this paper, we have argued that there exists a routing tree which is a constant factor approximation (in expectation) to the optimum aggregation tree *simultaneously* for all correlation parameters k. We present two constructions and prove that they obtain a constant approximation factor. Our result has important consequences – it obviates the need for specialized routing structures at least for the class of aggregation functions considered in this paper. This is convenient, since such specialized routing structures are hard to build without some a priori knowledge about correlations in the data.

There are several possible future research directions that this work leads to. It would be interesting to study the behavior of our randomized algorithm for non-grid topologies (for example on a random graph), or for the grid-topology model with generalized connectivity assumption, in which nodes have a larger number of neighbors. Another research direction would be to extend the aggregation model, either by defining a more general framework, or by analyzing the range of aggregation functions that can be obtained by combining the already defined functions.

References

[1] P. Bonnet, J. Gehrke, and P. Seshadri, Querying the Physical World, *IEEE Personal Communications Special Issue on Networking the Physical World*, October 2000.

[2] R. Cristescu, B. Beferull-Lozano, and M. Vetterli, On Network Correlated Data Gathering, *IEEE Proceedings of INFOCOM*, 2004, Hong Kong.

[3] A. Goel and D. Estrin, Simultaneous optimization for concave costs: single sink aggregation or single source buy-at-bulk, *Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2003, p. 499-505.

[4] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, Building Efficient Wireless Sensor Networks with Low-Level Naming, *Symposium on Operating Systems Principles*, 2001.

[5] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, Energy-Ecient Communication Protocol for Wireless Microsensor Networks, *33rd Hawaii International Conference on System Sciences (HICSS '00)*, 2000.

[6] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, The Impact of Network Density on Data Aggregation in Wireless Sensor Networks, *ICDCS*, 2002.

[7] C. Intanagonwiwat, R. Govindan, D. Estrin, J. S. Heidemann, and F. Silva, Directed diffusion for wireless sensor networking, *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, p. 2-16, 2003.

[8] B. Karp and H. T. Kung, GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, *Mobile Computing and Networking (MobiCom)*, 2000.

[9] B. Krishnamachari, D. Estrin, and S. B. Wicker, The Impact of Data Aggregation in Wireless Sensor Networks, *ICDCS Workshop on Distributed Event-based Systems (DEBS)*, 2002.

[10] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks, *Fifth Annual Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[11] S. R. Madden, R. Szewczyk, M. J. Franklin, and D. Culler, Supporting Aggregate Queries Over Ad-Hoc Wireless Sensor Networks, *Fourth IEEE Workshop on Mobile Computing and Systems Applications*, 2002.

[12] S. Pattem, B. Krishnmachari, and R. Govindan, The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks, *Symposium on Information Processing in Sensor Networks (IPSN)*, 2004.

[13] A. Savvides, C.C. Han, and M. B. Strivastava, Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors, *MobiCom*, 2001.

[14] A. Scaglione and S. D. Servetto, On the Interdependence of Routing and Data Compression in Multi-Hop Sensor Networks, *MobiCom*, 2002.

A Technical Details for Case 2 of Lemma 4.4

Since we want to maintain $\Delta_{t+1} \geq 1$ (no collision at time $t+1$), we eliminate the case in which the first walk moves from $(i, j-1)$ to $(i-1, j-1)$ and the second walk moves from $(i-1, j)$ to the same point as the first walk.

Thus our formula for p_{t+1} becomes:

$$\Pr[\Delta_{t+1} \text{ is differing} \mid \text{the two walks do not collide}] = \frac{i(j-\Delta_t)f(i-1,j,\Delta_t+1)}{(i+j)^2} +$$

$$+\frac{i(i+\Delta_t)f(i-1,j,\Delta_t)}{(i+j)^2}\frac{j(j-\Delta_t)f(i,j-1,\Delta_t)}{(i+j)^2} = g(i,j,\Delta_t) - \frac{j(i+\Delta_t)f(i,j-1,\Delta_t-1)}{(i+j)^2}$$

while the formula for p_t remains

$$\Pr[\Delta_t \text{ is differing}] = f(i,j,\Delta_t) = \frac{i(j-\Delta_t)+j(i+\Delta_t)}{(i+j)^2}$$

Taking the difference between the two, and simplifying, using Mathematica for example, we obtain:

$$(p_{t+1}-p_t)(i+j)^2 = i^3 - i^2(j-2) - i(j-1)^2 + (j-1)^2 j$$

If $j > i$ the right hand side reduces to $2i^2 + (j-i)[(j-1)^2] - i^2$ which is positive; if $i > j$ the right hand side reduces to $[i^2-j^2](i-j)+2i$ which is again positive; if $i = j$ the right hand side is just $2i^2$, again positive.

Combining this with the fact that $(i+j)^2 \geq 0$ for all i, j we deduce that $p_{t+1} - p_t$ is always positive, which is exactly what we wanted to prove.