# Synopsis Diffusion: A Robust Paradigm for Computing Aggregates in Sensor Networks

Suman Nath[†,*] Phillip B. Gibbons[*]
[*] *Intel Research Pittsburgh* [†] *Carnegie Mellon University*

## 1. INTRODUCTION

In a large scale network of wireless sensors (motes), aggregate queries assume greater importance than individual sensor readings due to energy or communication constraints. When such a network is deployed in a harsh environment, the population and topology of the network become highly dynamic and unpredictable. New nodes may join the network and existing nodes may leave due to failures, resource depletion, duty cycling (sleeping) to conserve energy, etc. Moreover, communications between the nodes may be lost due to noise or collision, and this may happen at frequent intervals and for extended periods of time (hence, simply retransmitting the messages does not solve the problem). Existing query processing research on sensor networks has not adequately addressed this dynamicity. In this paper, we present a novel class of energy-efficient aggregate query processing algorithms that is robust against the dynamic nature of the network's population and topology. We believe that this robustness is very important, because in a large deployment of sensors, node and link failures will be the norm, rather than the exception.

**Existing Approaches and Their Limitations.** Most of the existing algorithms [4, 5] maintain a tree rooted at the querying node that spans all the nodes in the network and then perform in-network aggregation along the tree. These *single-tree based algorithms*, however, are very sensitive to node failure or packet loss, and hence are effective only in scenarios where the network is not very dynamic. For example, on failure of a node $u$, unless the spanning tree is constructed again (which is very expensive), the information of the subtree rooted at $u$ is absent from the final answer. The same problem occurs if the message containing the answer from $u$ to its parent is lost. These problems are very common in real test bed experiments, resulting in a significant error in the query answer [5].

To achieve robustness, Madden *et al.* [4] uses a DAG topology, where each node can have multiple parent nodes. A node with $k$ parents splits its accumulated value $v$, and sends $v/k$ to each of its parents. This approach minimizes the impact of link failures. However, due to the value splitting, correct answers for monotonic aggregates (such as `COUNT` or the `SUM` of positive values) are returned only when there are no link failures. Moreover, this approach does not alleviate the problem of node failures.

Gupta *et al.* [3] proposes a gossip-based fault-tolerant approach for computing aggregates over large process groups. However, the solution is not energy-efficient and some of their assumptions (*e.g.,* all the processes are synchronized with different phases of the algorithm) are not practical enough to be used in a real sensor deployment.

For *exemplary* queries (*e.g.,* `MAX`), Zhao *et al.* [5] uses a novel technique called *digest diffusion*, where each node broadcasts its current estimate of the aggregate to its neighbors. Each neighbor then recomputes the aggregate based on its own estimation and the estimations broadcasted by its neighbors. However, this approach fails for more general queries, where including the same reading more than once produces an incorrect result.

**Our Contributions.** In this research, we make the following contributions:

1. We propose a class of algorithms named *synopsis diffusion*, which decouples the aggregation process from the aggregation topology. By choosing a suitable aggregation topology, the aggregation process can be made much more robust and energy-efficient. Both one-time and continuous queries can be handled.

2. We formalize the semantics of the answers and the error bounds of the approximate answers provided by a synopsis diffusion algorithm. We also identify necessary and sufficient properties of different components of the algorithm in order to provide well-defined semantics and error bounds.

## 2. SYNOPSIS DIFFUSION ALGORITHMS

In a synopsis diffusion algorithm, the aggregation process is orthogonal to the underlying aggregation topology – hence the aggregation can be done over any arbitrary robust topology (*e.g.,* a 2-connected graph). With such a robust topology, a synopsis diffusion algorithm tries to propagate the intermediate answers in more than one path towards the querying node. Unlike [4], a node broadcasts the complete intermediate result for its neighbors and a sensed reading is correctly included in the final answer if it, possibly in an aggregated form, can reach the querying node through at least one path. Moreover, it performs in-network aggregation. The intermediate result is represented as a synopsis [1], a small representation (*e.g.,* histograms, bit-vectors, etc.) of the data. Synopses, being small by definition, keep the messages small and make is possible to piggyback them on the existing ad hoc networking protocol.

The algorithm is based on the following three functions. A *synopsis generation function* $SG(\cdot)$ takes the local data to be included into the query answer and generates a *local synopsis*. A *synopsis fusion function* $SF(\cdot,\cdot)$ takes two synopses and generates a new synopsis. Finally, a *synopsis evaluation function* $SE(\cdot)$ translates a synopsis to the final answer. The exact details of the functions $SG()$, $SF()$, and $SE()$ depend on the particular query to be answered.

We now present the synopsis diffusion algorithm over two example aggregation topologies.

**Rings: An Energy-Efficient Topology.** When the query is first propagated from the querying node $q$, nodes form a set of rings around $q$ as follows: $q$ is in ring 0 and a node is in ring $i$ iff it receives the query first from a node in the ring $(i-1)$ (thus a node is in ring $i$ iff it is $i$ hops away from $q$). The subsequent query aggregation period is divided into *epochs* and one aggregate answer is provided at each epoch. Like [4], nodes are loosely time synchronized and allocated with specific time intervals when they should be awake to receive synopses from other nodes.

At the beginning of each epoch, each node in the outermost ring generates its local synopsis $s = SG(v)$, ($v$ is the set of values to be included in the query answer) and broadcasts it. A node in the ring $i$ wakes up at its allocated time for receiving synopses from its neighbors and generates its local synopsis $s = SG(\cdot)$. Upon receiving a synopsis $s'$ from a neighbor in ring $(i + 1)$, it updates its local synopsis as $s = SF(s, s')$. At the end of its allocated time (by when it is done with fusing the synopses from its neighbors in ring $(i+1)$) within the epoch, it broadcasts the local synopsis $s$ if $s$ has changed from the previous broadcast. Thus the fused synopses gradually propagate towards the querying node in ring 0. At the end of the epoch, the querying node returns $SE(s)$ as the answer to the aggregate query. Since the underlying wireless communication is broadcast, *synopsis diffusion generates the same optimal number of messages as a single tree based algorithm while enjoying more robustness.*

**Flooding: A Robust Topology.** With this topology, each node receives synopses from all of its neighbors (not just from the neighbors in its adjacent ring), applies $SF()$ to fuse them with the local synopsis $s$ and broadcasts $s$ if it has not changed. The process converges in a time proportional to the diameter of the network, after which each node in the network can apply $SE()$ to translate its local synopsis to the final answer. However, this scheme generates more messages and is less energy efficient than the previous scheme.

**Challenge: Supporting Duplicate-Sensitive Aggregates.** The main challenge of a synopsis diffusion algorithm is to support *duplicate-sensitive* queries (*e.g.,* `COUNT`). Since there is no aggregation tree, a value may be aggregated along more than one path towards the querying node, and the algorithm should still be able to answer the query correctly. To achieve this, we map the target aggregate function to a set of *duplicate-insensitive*, *commutative* and *associative* synopsis generation and fusion functions. Such a set of functions ensure that repeating the same input or intermediate value does not hurt the correctness of the final answer.

We have found that a number of techniques for computing/estimating aggregates on a data stream [1] can be adapted to fit within our synopsis diffusion framework. For example, Flajolet and Martin's algorithm (see [1]) can be adapted to estimate how many distinct values occur among all the nodes in the network. Overall, we have found specific algorithms for computing/estimating a wide variety of aggregates including max/min, count, distinct-count, sum, set-resemblance, uniform sampling, and median. For lack of space, we describe only the sampling algorithm.

**Fixed Size Sampling of Values.** Suppose each node $u$ has a set $set_u$ of items. A fixed size sampling operation finds a uniform sample of a given size $K$ of the items occurring in all the nodes in the network. Traditional sampling procedures would not produce a uniform sample in the presence of multiple fusion paths. Instead, assume that each node knows a shared, uniformly random hash function $H : item \mapsto [0..l]$, for some constant $l > K$. The components of the algorithm are as follows: at each node $u$, the local synopsis $s$ is a sample of size $K$, $SG(set_u)$ is the $K$ items $\in set_u$ with maximum hash values, $SF(s, s')$ is the $K$ items $\in s \cup s'$ with maximum hash values, and $SE(s) = s$. The correctness of the algorithm can be shown using the uniform randomness property of $H$. Other aggregates can be estimated from this sample, e.g., the median value can be approximated using $SE(s) = \text{median}(s)$.

On the other hand, for other aggregates such as the second frequency moment/self-join size [1], the data stream algorithm breaks down in the presence of duplicated inputs, and we are exploring entirely new techniques.

## 3. ONGOING RESEARCH ISSUES

**Semantics of the Query Answer.** The dynamic nature of the network implies that the set of nodes that are reachable from the querying node $q$ can change even while the aggregate is being computed. To make the point concrete, assume that a query for the *total number of active nodes in the network* is made at time $t_0$ and the answer to the query is returned at time $t_1 > t_0$. If a number of nodes have joined and/or left within the time $(t_1 - t_0)$, then what does the answer really mean? Bawa *et al.* [2] addresses similar issues in the context of peer-to-peer networks. The results, however, do not hold for an arbitrarily dynamic wireless sensor network (*e.g.*, when links can fail and then return). We extend their results and formalize the semantics of the query answer in a dynamic sensor network.

**Error Bounds of the Approximate Answers.** Using synopses may provide only an approximate answer to the query. We provide a general framework to analyze the errors introduced by the function $SG()$, and accumulated through the successive applications of the function $SF()$. We show that for a large class of functions $SG()$ and $SF()$, the answer (and hence the error bound) provided by a (distributed) synopsis diffusion algorithm is the same as that provided by first collecting all the values at the querying node $q$ and then applying the function $SG()$ on those values locally (as is done by many data stream applications).

**Outliers.** We incorporate new in-network outlier detection techniques with the synopsis diffusion algorithms such that faulty readings of the sensors are automatically detected and reported, or excluded from the final aggregate answers. Previous work does not consider this issue.

**Supporting Mobile Sensors.** Since the functions $SG()$ and $SF()$ are duplicate-insensitive, associative, and commutative, a mobile sensor can correctly contribute its data to the final aggregate answer by broadcasting its local synopsis periodically or when it moves.

We are currently working towards quantifying the robustness of our approach for both static and mobile sensors, using simulation and real deployment in a mote network.

## 4. REFERENCES

[1] Babcock, B., Babu, S., Datar, M., Motwani, R., and Widom, J. Models and issues in data stream systems. In *PODS* (2002).

[2] Bawa, M., Garcia-Molina, H., Gionis, A., and Motwani, R. Estimating aggregates on a peer-to-peer network. http://www-db.stanford.edu/ bawa/Pub/aggregates.ps.

[3] Gupta, I., van Renesse, R., and Birman, K. Scalable fault-tolerant aggregation in large process groups. In *Proc. Conf. on Dependable Systems and Networks* (2001).

[4] Madden, S., Franklin, M. J., Hellerstein, J. M., and Hong, W. Tag: A tiny aggregation service for ad hoc sensor networks. In *OSDI* (2002).

[5] Zhao, J., Govindan, R., and Estrin, D. Computing aggregates for monitoring wireless sensor networks. In *SPNA* (2003).