Scalable Fault-Tolerant Aggregation in Large Process Groups

Indranil Gupta, Robbert van Renesse, Kenneth P. Birman
Dept. of Computer Science
Cornell University
Ithaca NY USA 14853
{gupta,rvr,ken}@cs.cornell.edu

Abstract

This paper discusses fault-tolerant, scalable solutions to the problem of accurately and scalably calculating global aggregate functions in large process groups communicating over unreliable networks. These groups could represent sensors or processes communicating over a network that is either fixed (eg., the Internet) or dynamic (eg., multihop ad-hoc). Group members are prone to failures. The ability to evaluate global aggregate properties (eg., the average of sensor temperature readings) is important for higher-level coordination activities in such large groups. We first define the setting and problem, laying down metrics to evaluate different algorithms for the same. We discuss why the usual approaches to solve this problem are unviable and unscalable over an unreliable network prone to message delivery failures and crash failures. We then propose a technique to impose an abstract hierarchy on such large groups, describing how this hierarchy can be made to mirror the network topology. We discuss several alternatives to use this technique to solve the global aggregate function evaluation problem. Finally, we present a protocol based on gossiping that uses this hierarchical technique. We present mathematical analysis and performance results to validate the robustness, efficiency and accuracy of the Hierarchical Gossiping algorithm.

1. Introduction

Smart sensors networks, multihop ad-hoc networks and process groups over the Internet are examples of large groups of processes that inherently need to communicate and coordinate to perform higher level tasks. A few thousand sensors might be installed on the wing of an airplane, each detecting the air pressure, temperature, etc. within a few centimeters' radius of its location. A few hundred thousand smart dust computers might be randomly dropped on an inhospitable terrain, each making critical measurements in its own vicinity. These sensors or smart computers will communicate with each other over networks which provide a multihop routing mechanism between any two nodes. The networks can be either fixed (eg., on the airplane wing) or formed on the fly, i.e., ad-hoc (eg., in an inhospitable terrain).

Higher-level coordination activities in such groups are driven by protocols that *aggregate* the individual group members' measurements, or *votes*, into properties. For example, the network of airplane wing sensors might calculate the average temperature of all sensors on the wing, triggering a coolant release at certain sensors if this average temperature is above some threshold. For self-managing applications using such large groups as the ones described above, it is much more important to be able to disseminate answers of queries such as "what is the average temperature measured by all sensors near the leading edge of the right wing ?" than queries such as "what is the temperature measured by sensor # 5634 ?" [6].

In this paper, we tackle the problem of calculating such *global aggregate functions* over the *votes* of individual group members. Formally, our aim is develop a one-shot protocol by which *each member* of a group evaluates an accurate estimate of a global aggregate function $f(v_1, ..v_N)$ where $v_1, ..v_N$ are the individual votes of the group members. In the airplane wing example discussed above, the v_i's would be the individual temperature measurements at wing sensors i, and f would be the average. We focus only on *composable* global functions, i.e., 1) if W_1 and W_2 are two disjoint sets of votes, then the function f satisfies the property $f(W_1, W_2) = g(f(W_1), f(W_2))$ for some known function g; and 2) the byte-size of the function f's output is not much larger than the byte-size of an individual vote. The second assumption is required to keep all network messages small and bounded by a constant size. Average, minimum and maximum are all examples of composable functions.

We assume that group members use an asynchronous communication medium that is unreliable in delivering messages. Although it might be possible to construct a synchronous, fixed and reliable network in some scenarios such as for sensors on an airplane wing, such a network model would not be realistic for settings such as multihop ad-hoc networks, eg., sensors in an inhospitable terrain. In this model, we also assume that each group member has a globally unique identity number or address (which might be imprinted at manufacture-time, or assigned at run time). When the process groups we are talking of lie over the Internet, the routing mechanism could simply be TCP/IP, while it would need to be one of the specialized protocols such as TORA, AODV etc. over a multihop ad-hoc network, where the sensors themselves would act as routers. As in real-life networks, group members are also prone to arbitrary crash and recovery.

One might ask the question: "Why is the above problem of calculating a global aggregate function so difficult over such a communication network ? One could have each member send its individual vote to each other group member and in turn, calculate the aggregate function from the values it receives." While this fully distributed solution works well for small groups, it does not scale to groups of beyond a few hundred members. Centralized solutions, where the aggregate is calculated at a well-known *leader* member, suffer from several problems, most notable being the leader's failure. We discuss disadvantages of these traditional approaches in detail later in the paper.

This problem of accurately calculating aggregate global functions, like many other problems in distributed systems, can be reduced to the Consensus problem, which is typically unsolvable over an unreliable network [7]. It is therefore impossible to have a correct protocol, that is, one that always calculates the exact aggregate function value at all members, even if the member votes are time invariant.

As a result, different protocol solutions to the above problem have to be evaluated and compared based on several metrics. The metrics we consider in this paper are the most basic:

1. protocol message complexity,
2. protocol time complexity, and
3. Completeness of the final result of the protocol.

Completeness is the percentage of member votes included in a final global aggregate evaluation delivered at a random group member. If member votes do not differ vastly in value from each other, Completeness represents the accuracy of the protocol-measured aggregate to the actual aggregate. We will assume in this paper that there is a constant bound on the message size (which is larger than the byte-size of individual votes and any composable function evaluation), and that there is a limit on the network bandwidth utilization at each member. Then, the optimum achievable limits for the above three metrics, by any protocol on a group of N members, are $O(N), O(1)$, and 1.0 respectively.

In this paper, we first discuss the disadvantages of using the fully distributed and centralized solutions for global aggregate function calculation in large groups. We then propose a technique, called the *Grid Box Hierarchy*, that can be used to construct hierarchical algorithms for this problem. We discuss several alternatives for hierarchical global aggregate function calculating protocols using this technique. Finally, we propose a novel solution based on *gossiping* that uses this hierarchical technique and performs quite well under heavy unreliability, and is only slightly sub-optimal in message and time complexity with group size N ($O(Nlog^2N)$ and $O(log^2N)$ respectively[1]). This protocol shows resilience to message failures and member crashes, with a gracefully degrading completeness as these failures increase. The protocol's completeness scales quite well with group size N.

The rest of the paper is organized as follows. Section 2 describes the model and problem statement concisely. Section 3 discusses previous and related work on this problem. In Sections 4-5, we briefly argue why the fully distributed and centralized approaches do not work well for large groups. In Section 6, we discuss the abstract *Grid Box Hierarchy* and several alternatives for hierarchical solutions to calculating a global aggregate function using this technique. We then present the Hierarchical Gossiping protocol with a mathematical analysis. Section 7 presents performance results of running the Hierarchical Gossiping protocol over a simulated lossy unreliable network. We conclude in Section 8.

2. Model and Problem Statement

As mentioned in Section 1, we deal with the abstract notion of large *groups* of members (processes, sensors etc.) communicating with each other over an unreliable asynchronous network such as the Internet or a multihop adhoc network. We assume the presence of an underlying routing mechanism in this network that enables any member to send messages to any other member. Members have globally unique identifiers. Members may arbitrarily suffer crash failures and then recover. Since we are concerned about scalability, we assume that all messages sent over the network are constant size bounded, and that each member has a maximum network bandwidth constraint.

Each member also maintains a *view*, a list of other group members it knows about. We assume henceforth that all members know about each other, although this can be relaxed in our final hierarchical gossiping solution. Our algorithms do not require any failure detection.

The goal of the global aggregation protocol is to have *each group member* calculate a global estimate of a (composable) global aggregate function (as described in Section 1). We also impose a constraint that no member vote

[1] Unless specified otherwise, all logarithms in this paper are to base e

is counted twice in any global aggregate calculation. For example, when the global function is the average of all sensor temperature readings, this *no double counting* constraint would exclude all protocols that might possibly include some sensor's reading twice in calculating the average.

Our discussion considers only one run of the aggregation protocol, but this can be extended to one which periodically calculate the global aggregate. The protocol is assumed to be initiated simultaneously at all members, but our results apply in cases such as a multicast being used for protocol initiation.

The metrics for evaluating different algorithms for this problem are message complexity, time complexity, and completeness of the result at a random group member. With the *no double counting* constraint, completeness is thus the percentage of group member votes taken into account in the final global function value calculated at a random member.

3. Previous and Related Work

The growth of the Internet and the advent of application scenarios for large-scale ad-hoc and sensor networks [6, 11] have fueled research for scalable solutions to several problems that arise in such scenarios [3, 4, 6, 15, 16].

Theoretical results on the global aggregation problem date back to Pease et al [13]. The authors were concerned about Byzantine member failures and did not address message unreliability concerns. Our approach to providing probabilistic guarantees also bears a resemblance to randomized consensus algorithms such as [2]. However, the class of protocols in [2, 13] are inappropriate for large groups as they use several "rounds", with up to $O(N^2)$ message exchanges per round.

The technique of gossiping (or epidemic algorithms) has recently attained popularity in the research community as a technique for solving several large-scale distributed computing problems. Gossip protocols for reliable multicast [3], failure detection [16], resource management [15], etc., scale very well with group size, while being robust to random message losses and process crashes.

Several strategies such as *directed diffusion* [6], *Amorphous Computing Hierarchy* [4] etc. have been proposed for ad-hoc sensor networks coordinating to achieve higher level tasks. However, we believe ours is the first paper to discuss scalable solutions to the global aggregation problem via the gossiping technique. [9] describes adaptive protocols for energy efficient information dissemination in sensor networks, but we are not concerned with minimizing energy consumption at nodes in this paper.

Research in the area of calculating global snapshots focuses on evaluating global predicates for process groups [8], mostly towards distributed computation termination detection and deadlock detection problems. Such algorithms ignore message and member failures and reduce to the unscalable centralized or the fully distributed approaches when applied to calculating a composable global aggregate function at each member.

Random sampling techniques are used in databases for aggregate calculation (e.g., salary averages) without scanning all the input data (votes) [10]. These are potentially applicable to the global aggregation problem [5], but these results are yet unpublished.

Our hierarchical gossiping solution to global aggregation is similar to the philosophy of the Astrolabe project [15]. Astrolabe also uses a hierarchy, and gossiping, to provide scalable management support for distributed systems and applications. However, Astrolabe's hierarchies are fixed depending on the exact network topology, while our technique for constructing hierarchies is more general and can be modified on the fly. Astrolabe focuses on maintaining long-lived management information bases (MIBs) to answer queries regarding aggregate properties at any time, while we focus on a one-shot evaluation of a global aggregate property.

4. Fully Distributed Solution

A naive solution to the aggregation problem is to have each member send its vote to every other group member and calculate the aggregate function based on the votes it has received. While this solution would work well in networks where there is no message loss (such as synchronous networks), the completeness of the calculated estimate is only as good as the network message loss rate for real-life asynchronous fault-prone networks. Moreover, as all real networks constrain the bandwidth per member, the time complexity of this approach varies as $O(N)$. The message complexity is $O(N^2)$, and this is not optimal.

5. Centralized Solutions

Another approach is to have each group member send its vote to a special member (or members) denoted as a *leader* (or committee of leaders), which calculates the global function based on the votes received, and then disseminates this information out to all the group members.

This scheme has an optimal message complexity of $O(N)$, but has a message implosion problem at the leader(s). In the absence of highly available servers as leaders in infrastructure-less ad-hoc sensor networks, a costly leader election protocol would need to be run. The likelihood of a leader failing and losing a substantial part of the votes is significant for large groups since the running time of the algorithm varies linearly with group size, being

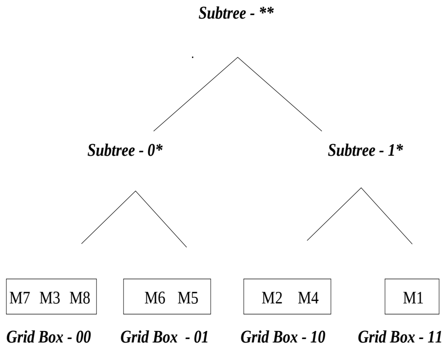

Figure 1. Division of 8 members {M1 ...M8} into 4 grid boxes, and the Grid Box Hierarchy induced therefrom.

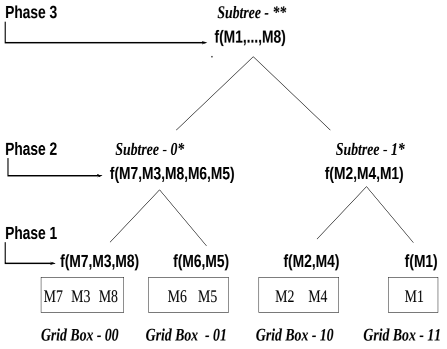

Figure 2. Global aggregate function evaluation on the hierarchy example of figure 1.

$O(N)$ because of the bandwidth constraint on the leader. Having a larger leader committee entails achieving coordination among these leaders, and this is a non-trivial task.

6. Hierarchical Solutions

In this section, we explore several hierarchical approaches to solving this problem. Hierarchical solutions tend to have very good scalability in terms of algorithm complexity but usually provide reduced robustness to (process and message delivery) failures. We first propose a technique, called the *Grid Box Hierarchy*, to construct an abstract hierarchy for large groups. Then, we discuss several ways of robustly calculating global aggregates using this hierarchy.

6.1. A Technique for Building a Hierarchy - The Grid Box Hierarchy

In this section, we present a technique for building an abstract hierarchy, called the *Grid Box Hierarchy*, over a large process group.

The hierarchy is constructed by first dividing the N group members into N/K *grid boxes* with an *average* of K members per grid box. K is a constant integer chosen independent of N, and well-known at all group members. Each grid box is assigned a unique $(log_K N - 1)$-digit address in base K. So, each digit is an integer between 0 and $(K - 1)$ (inclusive of both). Now, for all $1 \leq i \leq log_K N$, *subtrees of height* i in the hierarchy contain the set of grid boxes (actually, the members inside them) whose addresses match in the most significant $(log_K N - i)$ digits. Figure 1 shows a possible division of $N = 8$ members into 4 grid boxes with an average of $K = 2$ members per grid box. The figure also illustrates the hierarchy induced among the members in these grid boxes.

The global aggregate function is then calculated bottom-up in this hierarchy, and in $log_K N$ phases. In the first phase, each group member M_j (or a representative) evaluates an estimate of the function when evaluated over the votes of all members in its grid box. In each subsequent ith ($i > 1$) phase, each member M_j (or a representative) evaluates the value of the aggregate function over the set of votes of members belonging to the same height-i subtree as M_j - this is evaluated from the function evaluations for the child subtrees of height $(i - 1)$ obtained from phase $(i - 1)$, and the composable nature of the aggregate function. For the example of Figure 1, an ideal aggregate function evaluation would proceed as shown in Figure 2.

Two questions arise: 1) what mechanism does one use to build such a hierarchy, and 2) what is the actual protocol used to calculate the global function in the hierarchical manner described above. We answer the first question in the following paragraphs, and explore different solutions to the second question in the following sections.

The easiest way to build the hierarchy described above is to use a well-known hash function H that maps the unique group member identifiers randomly into the interval $[0, 1]$. A member with identifier M_j would then belong to a grid box with address $H(M_j) * \frac{N}{K}$ (written in base-K). Further, notice that any arbitrary group member M_j can calculate the grid box address of any other group member M_l (that is present in its view) - this is simply $H(M_l) * \frac{N}{K}$. Thus, at each phase i of the global function calculation, member M_j would know about all the members in its view that belong to M_j's height-i subtree in that phase.

Such a construction requires that the hash function H and the group size N are well-known at all group members. The former can easily be achieved by statically fixing the hash function at all members. The global knowledge of N is trivial if the maximal group membership is fixed. For a dynamically changing group membership, members need to be periodically informed of changes in the group size. However, an approximate estimate of N at each member usually suffices, and thus these updates can be done rather infrequently.

The above randomized scheme ensures an average K members per grid box, regardless of the actual composition of the group, i.e., the hash function H need not assume either a fixed group membership, or even a fixed set of possible group members. This is a very desirable property.

Further, it is often possible to have the grid division scheme mirror the geographical/network topology location of the group members, without a static knowledge of the group membership. Mathematically, this can be achieved by replacing the random hash function above by a more "topologically aware" function that maps group members that are nearby in the network to the same grid box, while maintaining the average of K members per grid box.

Most sensors (group members) in sensor networks usually know their exact geographical locations either by virtue of having a fixed physical location (e.g., sensors on an airplane wing), or via GPS (Global Positioning System). In the Internet, IP addresses usually reflect the geographical/network locations of group members, eg., CIDR (Classless Interdomain Routing) naming of Class C IP addresses by IANA allocates different subnet headers to addresses in Europe than those in the Americas, and then different subnets inside Europe depending on their location, and so on.

A topologically aware hash function would then (deterministically) map member addresses to grid boxes so that there are an average of K members per grid box, and grid boxes consist of members that are topologically proximate. If members are mobile (e.g., mobile sensors) or membership is dynamic, this might need a priori knowledge of the probability distribution of prospective group members across the network region. Such a division can also be achieved in network routing schemes such as the Landmark Hierarchy [14] that assign hierarchical addresses to machines depending on their network locations.

Using such a topologically aware H would result in a reduction of the load, imposed by the global aggregation protocol (as described above), on links in a sparsely connected network. This is because the ($O(N)$) messages in the initial phases of the protocol would be restricted to travel short distances (hops) in the network, and longer network routes would be taken only by the (much fewer) messages in the latter phases.

As an example, we can adapt the *Grid Location Scheme* described in [12] to construct such a topologically aware hash function for wireless sensor networks or ad-hoc networks. In [12], a wireless sensor network is divided into several closed, disjoint regions. If these closed regions are tailored to have an equal expected number of members, they can be treated as the *grid boxes* in our scheme. H then simply specifies a mapping from member addresses to addresses of these closed disjoint regions (or grid boxes). For example, the region occupied by the eight sensors M1 through M8 in Figure 3 can be divided into four grid boxes with addresses as shown, thus giving us the same Grid Box hierarchy as depicted in Figure 1.

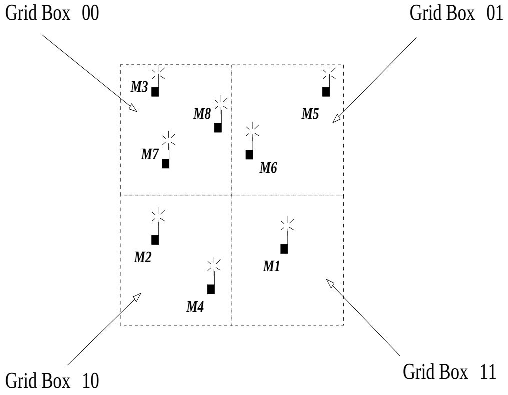

Figure 3. An instance of a grid box division of a network region of eight wireless sensors using a topologically aware hash function. This induces the Grid Box hierarchy shown in Figure 1.

Although H is assumed to be well-known above, it could also be dynamically specified by a multicast initiating the aggregation protocol. Further implementation details of such topologically aware or dynamic hash functions in more general scenarios are beyond the scope of this paper. We believe that this is a significant area for future study.

In the next few sections, we discuss different strategies for using the Grid Box Hierarchy described in Section 6.1 to calculate a composable global aggregate function.

6.2. The Leader Election Approach

This algorithm works by electing a single group member as a leader for every internal node of the tree constructed by the Grid Box Hierarchy. More concretely, each member is initially (before phase 1) a leader of its own height 0 subtree. In phase i ($1 \leq i \leq log_K N$), a leader is elected for each subtree of height i from the leaders of its child subtrees of height-$(i-1)$. This leader calculates the global aggregate function for the set of members in this subtree by obtaining and then composing the global function values calculated by the leaders of its height-$(i-1)$ child subtrees. The algorithm finally terminates in phase $log_K N$ with the entire tree (thus the group) electing one leader who has the aggregate function estimate for the entire group, and subsequently disseminates this to the group via the tree.

If the group membership view is consistent and complete at all group members, this scheme has a time complexity of

$O(logN)$, and an optimal message complexity of $O(N)$, since K is chosen fixed independent of N. However, this scheme is not fault-tolerant to failures of members, particularly leaders in latter phases. Failure of a member elected as the leader of a subtree of height i would result in the exclusion of the votes of an expected K^i members from the final global estimate. The completeness of the algorithm is also not masked from the network message loss rate.

Another scheme elects a committee of K' (instead of just one) leaders in each protocol phase (conceptually, at each internal tree node). Such a scheme is $(K'-1)$-fault-tolerant at each subtree root. This may appear to be sufficient, but phase i requires that knowledge of each height-$(i-1)$ subtree's leader committee be disseminated among all members of its sibling height-$(i-1)$ subtrees - this typically takes at least $O(logN)$ time because of the constant member bandwidth constraint. Thus, K' needs to be $O(logN)$ to survive the possibility of all the leaders in the committee failing before the dissemination completes. Such a dissemination can be avoided by having views consistent and complete at all members, but this approach requires the use of accurate failure detectors.

In conclusion, using leader election appears to be either inadequate or require unrealistic assumptions for a one-shot global aggregation algorithm using the Grid Box hierarchy. Random message delivery failures and process crashes can arbitrarily affect the completeness of the aggregate value.

6.3. The Gossiping Approach

In this section, we present an algorithm for evaluating a composable global aggregate function over member votes. This algorithm uses the Grid Box Hierarchy, but avoids leader election. We present a discussion and analysis of the algorithm, showing that it is only poly-logarithmically sub-optimal in time and message complexity. We then present simulation results in Section 7 to evaluate the resilience of the completeness of the protocol to message losses and member failures.

Informally, the algorithm at each member consists of $log_K N$ *phases*, phase i calculating the aggregate function for the set of votes in the subtree of height i (in the Grid Box Hierarchy) that the member belongs to. As discussed in Section 6.1, in each of subtree of each phase, the component votes of child members (phase 1) or aggregates of child subtrees (higher phases) are required for this calculation. These component aggregates or votes are obtained by gossiping rather than by electing a leader and sending votes to it. Gossiping lends itself to robustness against random message and process failures, while scaling very well with group size. Our algorithm inherits these characteristics - its completeness characteristics scale well with increasing group size, while the time and message complexity are only poly-logarithmically sub-optimal for any group size.

Our scheme does not require complete or consistent views at any group member - however, we will assume these in order to simplify the analysis.

The algorithm is started simultaneously at each group member. Each group member M_j executes the following three steps in different protocol phases:

I. Phase 1: (a) M_j starts out in phase $i = 1$, where it *gossips*, within its own grid box, about individual votes that it knows of and that belong to members in its own grid box - this of course includes M_j's own vote. M_j does so by periodically (once every *gossip round*) 1) randomly selecting a few *gossipees* only from among other members in its own grid box, and 2) sending (gossiping to) them one randomly selected known vote along with the identifier of the member whose vote it is. In turn, M_j knows about the vote of another member in its own grid box when it first receives the same by a gossip message from another member.
(b) After $KlogN$ gossip rounds, M_j applies the aggregate function to the known votes of members in its grid box, and bumps itself up to phase 2.

II. Phase i **(**$2 \leq i \leq log_K N - 1$**): (a)** In every gossip round in phase i, M_j chooses a few gossipees randomly from the set of all members in the same subtree of height i as itself, i.e., the set of members whose grid box addresses agree with M_j's in the most significant $(log_K N - i)$ digits. M_j then sends these gossipees a randomly selected aggregate value from among the known (at M_j) aggregates for the height-$(i-1)$ child subtrees of M_j's height-i subtree. Note that there can be at most K such values in phase-i at any member M_j, and M_j already knows about the aggregate value for its own height-$(i-1)$ subtree immediately after phase $(i-1)$ concludes. M_j knows about the aggregate values of another sibling height-$(i-1)$ subtree when it first receives the same by a gossip message from another member in phase i.
(b) Finally in this phase, when M_j has either managed to obtain the values of the function evaluation for all its $(K-1)$ sibling subtrees, or has faced a time-out ($KlogN$ gossip rounds), it evaluates the aggregate function for the subtree of height-i from these values and the composable nature of the aggregate function. M_j then bumps itself up to phase $(i+1)$.

III. Final Phase: When M_j finds itself in phase $i = (log_K N + 1)$, it has an estimate of the global aggregate function evaluated over the entire group's votes. The protocol then terminates at M_j.

If the first phase of this protocol is started almost simultaneously at all members (eg., through a multicast), executing the above steps has the effect of having each group member calculate aggregate function values for larger and larger sets of group members as it moves through the protocol phases, until (after phase $log_K N$) it has an estimate of the global aggregate function for the entire group.

The protocol itself does not require synchronized member clocks, although the clock drifts need to be much smaller than the protocol running time. To simplify the analysis of the above protocol, we assume below that all members have synchronized clocks and start the first phase simultaneously. We also assume that a member stays in each phase i for exactly $KlogN$ gossip rounds, rather than bumping itself up to phase $(i+1)$ when it has managed to obtain estimates of the aggregate function evaluations of all its sibling subtrees of height $(i-1)$ (as specified in step 2(b) above). This has the effect of having all the group members move together synchronously from phase to phase. We relax this second synchrony assumption in Section 7 when we measure the performance of this protocol through simulations.

Time Complexity: Each phase of this algorithm lasts for $KlogN$ gossip rounds. Since K is fixed independent of N, and there are $log_K N$ phases, the time complexity of this algorithm is $O(log^2 N)$.

Message complexity: Each member gossips at a constant rate in each gossip round. Hence, the message complexity of this algorithm is $O(Nlog^2 N)$.

These values are only poly-logarithmically sub-optimal. In fact, in most real-life networks, $logN$ and $log^2 N$ can be considered to be constants (since N is usually bounded from above) - this algorithm would then have optimal time and message complexities for all practical purposes.

Completeness: We now analyze the probabilistic Completeness guarantees of this algorithm using the deterministic methodology of analyzing epidemic processes [1]. We assume that the hash function H is fair, i.e., it maps any given member to each grid box with probability $\frac{K}{N}$.

In each phase, each member M_j gossips about several different aggregate "values". In phase 1, these (on an average K) values are the individual votes of members in the same grid box as M_j. In each subsequent phase i, these values are the K aggregate function evaluations for the height-$(i-1)$ child subtrees of M_j's height-i subtree. The propagation of each of these values can be modeled as a deterministic epidemic [1] among the members of the respective grid box or subtree.

Let b be the average number of members an arbitrary group member M_j successfully (in spite of failures) gossips with in each gossip round. Notice that the value of b depends not only on the number of members M_j *chooses* to gossip with in each round, but also on the reduction induced by the message loss and member failure rates.

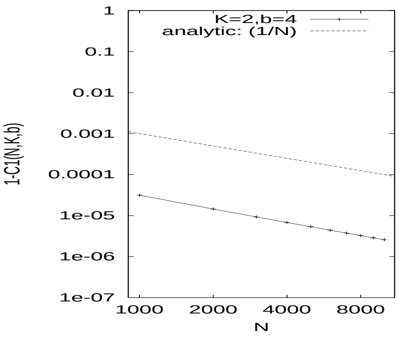

Figure 4. Variation of -log(incompleteness) vs. log(N).

Bailey [1] analyzes the spread of an infection within a group with m members and one initial infective. Once infected, a member (randomly) chooses b other groups members in every gossip round, and infects them (unless they are already infected). The relation between the number of yet non-infected members x (initially $(m-1)$) and number of rounds t is

$$\frac{dx}{dt} = \frac{b}{m} \cdot x \cdot (m-x) \Rightarrow x = \frac{m}{1+me^{-bt}}$$

In our protocol, in any phase $i > 1$ at a member M_j, after $KlogN$ gossip rounds of the above protocol, each of the height-$(i-1)$ child subtrees' aggregate values is received (via gossip) at M_j with a probability $C_i(N,K,b)$ that can be bounded from below from (1) as:

$$C_i(N,K,b) \geq \frac{1}{1+N\cdot e^{-K\cdot b\cdot \frac{logN}{K}}} \simeq \left[1-\frac{1}{N^{b-1}}\right]$$

This analysis of course does not apply to the first phase of the protocol since a grid box can have anywhere between 0 and N members, and that many values have to be gossiped about during the first phase. However, we can express the expected completeness of the first phase in any grid box from (1) as: $C_1(N,K,b) =$

$$\sum_{i=0}^{N} \left(\begin{array}{c} N \\ i \end{array} \right) \cdot \left(\frac{K}{N}\right)^{i} \cdot \left(1-\frac{K}{N}\right)^{N-i} \cdot \frac{1}{1+i\cdot e^{\frac{-K\cdot b\cdot logN}{i}}}$$

Evaluating $C_1(N,K,b)$ exactly is beyond the scope of this paper. Instead, here, we will use a pragmatic approach that combines simulation and reasoning to bound the completeness of our protocol's first phase.

Figure 4 shows that at $K = 2$ and $b = 4$, $-log(1 - C_1(N,K,b))$ varies linearly with $log(N)$ (since both

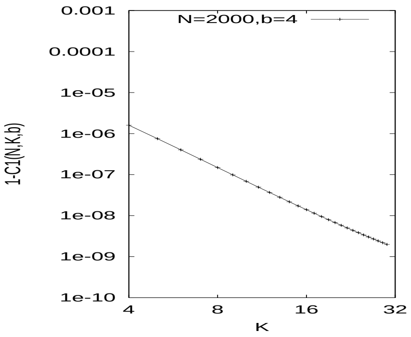

Figure 5. Variation of -log(incompleteness) vs. log(K).

axes are logarithmic). From this curve, we obtain that $C_1(N, K = 2, b = 4) \geq 1 - \frac{1}{N}$, which is a very pessimistic lower bound. Further, $C_1(N, K, b)$ is monotonically increasing with both b and K. The former is verified by observation. Figure 5 shows the variation of $(1 - C_1(N = 2000, K, b = 4))]$ with K, both axes being logarithmic. Evidently, the completeness is monotonically increasing with K. The same trend was also observed for other values of N and b. Thus,

Postulate 1: For $K \geq 2, b \geq 4$, the completeness of the first phase in any grid box with an average of K members can be lower bounded by $[1 - \frac{1}{N}]$. □

Theorem 1: For $K \geq 2, b \geq 4$ and large N, the expected completeness of the Hierarchical Gossiping protocol can be lower bounded by $[1 - \frac{1}{N}]$.

Proof: The expected completeness of the protocol = the probability that a random group member vote is included in the final aggregate function obtained at member M_j =

$$\begin{aligned} = & \quad \Pi_{i=1}^{log_K N} C_i(N, K, b) \\ \geq & \quad \left(1 - \tfrac{1}{N}\right) \cdot \left(1 - \tfrac{1}{N^{b-1}}\right)^{log_K N - 1} \\ \simeq & \quad [1 - \tfrac{1}{N} - \tfrac{log_K N}{N^{b-1}}] \qquad \text{(since } N \gg 1\text{)} \\ \simeq & \quad [1 - \tfrac{1}{N}] \qquad \text{(since } N \gg 1\text{)} \end{aligned}$$

□

This is indeed a satisfactory, although pessimistic, lower bound on the protocol's completeness, for very reasonable assumptions on the protocol parameters (K, b). However, this analysis does not reflect the effect of members executing protocol phases asynchronously, or of having $b < 4$ (i.e., gossiping at low rates), or of explicit message and member failures, on the completeness of the protocol. In the next section, we present performance results of running our protocol over a simulated lossy network with fail-prone machines (members). These experiments seek to better quantify the effect of varying group size, message delivery and member failure rates, and gossip rates, on the completeness of the protocol.

7. Simulation Results

In this section, we present performance results of the Hierarchical Gossiping Approach to calculate aggregate global composable functions in large groups.

The analysis of Section 6.3 showed that the protocol's completeness is satisfactory for fairly high rates of gossip ($b \geq 4$). In this section, we investigate the effect of low gossip rates on the completeness of our protocol. These simulations demonstrate the effect of varying group size, message delivery and member failure rates, and gossip rates on the completeness probability guaranteed. They also account for the effect of asynchrony among the different members as to which phase they are in. This contrasts with the simplistic assumption made in the earlier analysis that all members proceed together from phase to phase in the protocol.

Figures 6-11 show the effect of the different protocol parameters and network characteristics on the completeness achieved by the Hierarchical gossiping protocol. Each point in these plots is the average of several runs of the protocol in a group with N (initial) members, communicating over a lossy asynchronous network with independent unicast (point-to-point) message loss probability $ucastl$. Members were prone to crashes (without recovery) in every gossip round with probability pf. A gossip round at a member consisted of attempts to gossip with M randomly selected members. The number of gossip rounds per protocol phase was $\lfloor C \cdot log_M N \rfloor$. The hash function H used was a fair one, and not topologically aware. The protocol was started simultaneously at all group members, but thereafter, members proceeded asynchronously from phase to phase in the protocol (as described in step 2(b) in Section 6.3). Unless otherwise stated, the default parameters used in the Figures 6-11 were $N = 200, ucastl = 0.25, pf = 0.001, K = 4, M = 2, C = 1.0$. The metric measured on the y-axis was the protocol's average measured $Incompleteness = 1 - completeness$. Figures 6-10 plot the incompleteness on a logarithmic scale.

Figure 6 plots the variation of $Incompleteness$ versus the group size N. As N is increased, the number of protocol phases and the duration of each phase also rise. This curve shows that the average completeness guarantees of the algorithm improve slightly as N is increased into the 1000's. Notice that the result of theorem 1 does not apply here (since the parameter b evaluates to about 0.75), yet the completeness guaranteed improves with the number of group members varying into the 1000's.

Figure 7 shows that the incompleteness falls exponentially, with increasing network message reliability. Figure 8

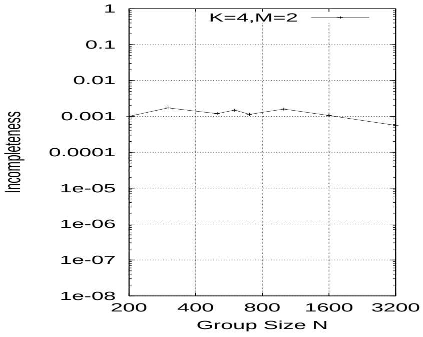

Figure 6. *Scalability 1:* Even at low gossip rates (where Theorem 1 does not apply), the protocol's completeness scales well at high values of group size N.

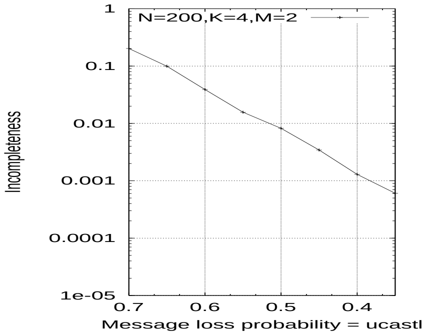

Figure 7. *Fault-tolerance 1:* The protocol's incompleteness falls exponentially fast with decreasing unicast message loss probability.

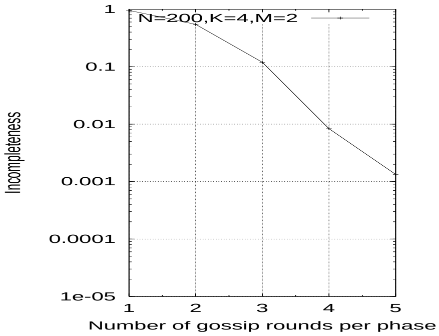

Figure 8. *Effect of gossip rate:* The protocol's incompleteness falls exponentially with increasing gossip rate / gossip round length.

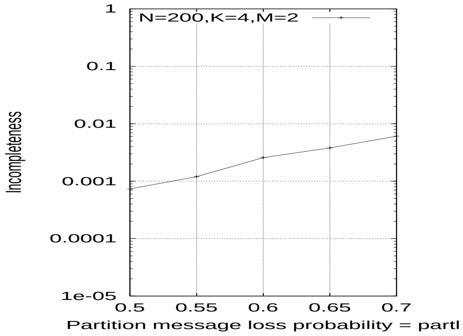

Figure 9. *Fault-tolerance 2:* The protocol's incompleteness is degrades gracefully due to the effect of soft network partitions induced by correlated message losses.

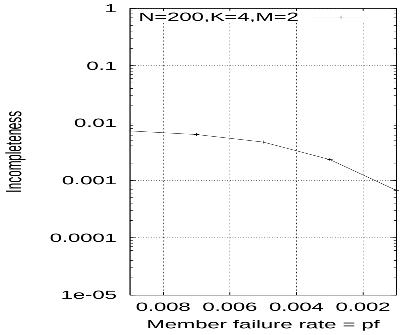

Figure 10. *Fault-tolerance 3:* The protocol's incompleteness falls exponentially fast with decreasing member failure rate / round.

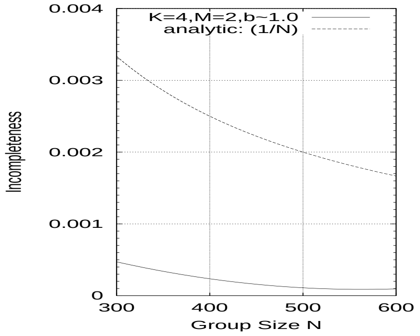

Figure 11. *Scalability 2:* The protocol's incompleteness falls with N, and is upper bounded by $\frac{1}{N}$.

shows the effect of the duration of a phase in the protocol (in number of gossip rounds) on the average completeness guaranteed. Since $M = 2$ is fixed, increasing the phase duration amounts to increasing the gossip rate. The incompleteness falls exponentially with increasing duration of a protocol phase, and thus with the rate of gossip.

Figure 9 shows the result of an experiment where the group with N members was partitioned into two halves, with messages across the partition being subjected to dropping independently with probability $partl$. Messages within each partition were dropped independently with probability $ucastl$. This experiment was conducted to measure the performance of the Hierarchical gossiping protocol in the presence of a network partition, the most major symptom of congestion and *correlated* message delivery failures in wide area networks. Figure 9 shows that the protocol's completeness degrades gracefully as the partition/correlated failure rate becomes worse.

Figure 10 demonstrates that the protocol's incompleteness falls very quickly (faster than exponential) with falling member failure rate. Finally, Figure 11 compares the average incompleteness guaranteed by a run of the protocol with the limit imposed in Theorem 1. Values of $C = 1.4, ucastl = pf = 0.0$ were used, so that b evaluated to about 1.0. Although this does not satisfy the conditions for Theorem 1, Figure 11 shows that the incompleteness is bounded by $\frac{1}{N}$. This reflects the pessimism of the bound imposed by Theorem 1, and with Figure 8, suggests that a more rigorous analysis of our protocol will show an exponential variation of incompleteness with the gossip rate.

8. Conclusion

In this paper, we have discussed several solutions to the problem of scalably and accurately calculating global (composable) aggregate functions in large groups, targeting large-scale sensor networks, ad-hoc networks and process groups over the Internet. We have argued why traditional approaches to solving this problem do not scale in large groups, and do not perform well over fault-prone networks. We have then proposed a technique to construct abstract hierarchies over such large process groups, and proposed a solution that uses gossiping within this hierarchy to evaluate composable global aggregate functions in the group. Our mathematical analysis and simulation results show that the proposed Hierarchical Gossiping protocol is poly-logarithmically sub-optimal in time and message complexity and guarantees good completeness probability, i.e., probability of a member vote being included in the final global estimate. These completeness guarantees are fairly robust to random message losses, crashes of group members, and even correlated message failures in the network. Besides, the completeness guarantees improve with increasing group size, even at low rates of gossiping.

Acknowledgments: We wish to thank Al Demers, Johannes Gehrke, Kate Jenkins, Jon Kleinberg and the reviewers for their suggestions on the topic and the paper.

References

[1] N. T. J. Bailey. *Epidemic Theory of Infectious Diseases and its Applications*. Hafner Press, second edition, 1975.

[2] Z. Bar-Joseph and M. Ben-Or. A tight lower bound for Randomized Consensus. In *Proc. ACM Symp. Principles of Distributed Computing*, pages 193–199, 1998.

[3] K. P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Trans. Computer Systems*, 17(2):41–88, May 1999.

[4] R. W. D. Coore, R. Nagpal. Paradigms for structure in an Amorphous Computer. A.I. Memo 1614, Massachusetts Institute of Technology, October 1997.

[5] A. Demers, J. Gehrke, and K. Jenkins. Private communication. 2000.

[6] D. Estrin, R. Govindan, J. Heidermann, and S. Kumar. Next century challenges: scalable coordination in sensor networks. In *Proc. 6th Intnl Conf. Mobile Computing and Networking*, pages 263–270, Aug 2000.

[7] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed Consensus with one faulty process. *Journ. ACM*, 32(2):374–382, Apr 1985.

[8] V. K. Garg. Methods for observing global properties in distributed systems. *IEEE Concurrency*, 5(4):69–77, Oct-Dec 1997.

[9] W. R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proc. 5th Intnl. Conf.Mobile Computing and Networking*, pages 174–185, 1999.

[10] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. In *Proc. ACM SIGMOD Intnl. Conf. on Management of Data*, pages 171–182, May 1997.

[11] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: mobile networking for smart dust. In *Proc. 5th Intnl. Conf. Mobile Computing and Networking*, pages 271–28, Aug 2000.

[12] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, and R. Morris. A scalable location service for geographic adhoc routing. In *Proc. 6th Intnl. Conf. Mobile Computing and Networking*, pages 120–130, Aug 2000.

[13] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journ. ACM*, 27(2):228–234, Apr 1980.

[14] P. F. Tsuchiya. The Landmark hierarchy: a new hierarchy for routing in very large networks. In *Proc. Symp. Communications Architectures and Protocols*, pages 35–42, Aug 1988.

[15] R. van Renesse. Scalable management with Astrolabe. Technical report, Dept. of Computer Science, Cornell University, 2000.

[16] R. van Renesse, Y. Minsky, and M. Hayden. A gossip-style failure detection service. In *Proc. Middleware '98 (IFIP)*, pages 55–70, Sep 1998.