Fast Estimation of Aggregates in Unstructured Networks

Carlos Baquero, Paulo Sérgio Almeida
DI/CCTC
Universidade do Minho
Braga, Portugal
{*cbm,psa*}*@di.uminho.pt*

Raquel Menezes
DMCT
Universidade do Minho
Guimarães, Portugal
rmenezes@dmct.uminho.pt

Abstract

Aggregation of data values plays an important role on distributed computations, in particular over peer-to-peer and sensor networks, as it can provide a summary of some global system property and direct the actions of self-adaptive distributed algorithms. Examples include using estimates of the network size to dimension distributed hash tables or estimates of the average system load to direct load-balancing.

Distributed aggregation using non-idempotent functions, like sums, is not trivial as it is not easy to prevent a given value from being accounted for multiple times; this is especially the case if no centralized algorithms or global identifiers can be used.

This paper introduces Extrema Propagation, a probabilistic technique for distributed estimation of the sum of positive real numbers. The technique relies on the exchange of duplicate insensitive messages and can be applied in flood and/or epidemic settings, where multi-path routing occurs; it is tolerant of message loss; it is fast, as the number of message exchange steps equals the diameter; and it is fully distributed, with no single point of failure and the result produced at every node.

1. Introduction

Aggregation is recognized as an important building block for distributed applications in peer-to-peer or sensor network infrastructures [18], [10], [11]. Aggregating data values can provide a summary of some global system property and play an important role in directing the actions of self-adaptive distributed algorithms.

Examples can be found in the use of estimates of the network size to direct the dimensioning of distributed hash table structures [17], when setting a quorum for voting algorithms [1], when estimates of the average system load are needed to direct local load-balancing decisions or when an estimate of the total disk space in the network is required in a P2P sharing system.

Distributed computation of aggregation functions in a network is not trivial. Unlike aggregation in a tree [13], [12], where each value is guaranteed to contribute only once, in a graph it is not easy to prevent a given value from being accounted for multiple times; this is especially the case if no centralized algorithms or global identifiers can be used. Thus, calculating general non-idempotent functions (e.g. COUNT, SUM, AVG) is problematic and we are restricted to idempotent functions that are duplicate insensitive (e.g. MIN, MAX) [15]. Aggregation functions that can be made duplicate insensitive have the advantage of being usable under multi-path routing.

This paper introduces Extrema Propagation, a technique for distributed estimation of the sum of positive real numbers. It is a probabilistic technique that exchanges duplicate insensitive messages and thus can be applied in flood and/or epidemic settings, where multi-path routing occurs. It can also be easily adapted to tolerate message loss.

Our results are similar to those in [14] as they improve the sketching techniques in [7], [6] for sums of positive integers, by allowing summing of positive reals. Contrary to [14], whose estimator is biased and fixed to an exponential distribution, we will show that the technique is generalized to other distributions and will introduce an exponential estimator that is unbiased and has tighter error bounds.

Extrema Propagation has some important properties: the precision is controlled by message size, independently of network size; it is fully distributed, with no single point of failure, and the result is produced at every node. As a special important case (and for presentation purposes) we show how this technique can be applied to network size estimation.

2. Network Size Estimation

In order to simplify the description we concentrate on a specific counting problem: *How many nodes are present in a given network?* Moreover, we aim for a distributed assessment of such estimate and to have it available at every node after a short number of message exchange steps.

Our assumptions are: (1) Each node can communicate with a set of neighbour nodes; (2) Each node has access to a random number generator. We also make use of some assumptions that, although not necessary for this class of algorithms, simplify the presentation and analysis: (a)

messages are not lost or corrupted; (b) the network is static and represented by a connected graph; (c) connections are bidirectional (the graph is undirected). Message loss can easily be tolerated since it only delays the availability of the estimation.

2.1. Minimum Number of Steps Towards Estimation

Our technique avoids the construction of a tree, and works directly on an unstructured network where each node only needs to know its neighbours. Let D be the network diameter: the maximum length of the shortest paths between two nodes in the graph. Tree construction over an unstructured network would require a number of message exchange steps proportional to D (between $D/2$ and D, depending on the node chosen for the tree root). Subsequent aggregation along the tree would again require an identical number of message exchange steps. Moreover, such procedure would not tolerate link failures and the calculated result would be available at just a single node (the tree root, a single point of failure); dissemination to other nodes would require further message exchange steps. Therefore, making the result available at every node would take at least $3D/2$ steps in the best case.

A tree based algorithm cannot reach the theoretical lower bound for the number of steps, which can be trivially shown to be D, since obtaining at every node an estimate of the number of nodes needs at least D message exchange steps. Enough steps to exchange information among the two nodes furthest apart.Thus, the fastest estimations cannot be done in less than D steps (to be available at every node; it can be made earlier at some nodes, and the average number of rounds across the network can be less than D).

2.2. Synopsis of the Estimation Technique

Our approach to estimation is based on finding an idempotent message structure that allows the counting of nodes. One trivial approach would be the use of one unique identifier per node (an additional assumption) and a protocol that collects the set of all identifiers, aggregating by set union. Such a protocol would provide an estimate in D steps, but creates messages that are linear with the network size.

Our technique avoids the need for unique identifiers and aggregate sizes which depend on network size [16]. It is based on idempotent operations on numbers, more specifically the minimum function, and the use of statistical inference.

The insight to our approach is the following: if we generate a random real number in each node using a known probability distribution (e.g. Gaussian or exponential), and aggregate across all nodes using the minimum function, the resulting value has a new distribution which depends on the number of nodes.

The basic idea is then to generate a vector of random numbers at each node, aggregate each component across the network using the pointwise minimum, and then use the resulting vector as a sample from which to infer the number of nodes (by a maximum likelihood estimator).

We will show that if a vector of K numbers is generated per node, it is possible to provide an estimate $\widehat{N}$ of the network size N with a standard deviation of $N/\sqrt{K-2}$. This means that the relative accuracy can be chosen independently of the network size, and is determined by K.

If we want to enforce a maximum relative error $r = |\widehat{N} - N|/N$ with a confidence of 95% we need to make $K = 2 + \left(\frac{1.96}{r}\right)^2$. For example, for an error $r = 10\%$, one needs to make $K = 387$.

The focus of our technique is not accuracy but speed: we do not aim for very low errors (e.g. 1% would lead to large messages) but for a fast computation of an useful approximation that can serve as input to some other algorithm (in some cases even 10% may be more than enough, only the order of magnitude may be needed).

2.3. Extrema Propagation

The basic algorithm that every node runs is shown in Algorithm 1. Each node maintains a vector x of K numbers, initialized using function $rExp(1)$, which returns a random number with an exponential distribution of rate parameter 1.

Algorithm 1 Extrema Propagation

const K
var $n, x[1..K]$
Upon: Init
  $n \leftarrow neighbours(self)$
  for all $i \in 1..K$ **do** $x[i] \leftarrow rExp(1)$
  Send x to every $p \in n$
Upon: Receive $m_1..m_j$ from all $p \in n$
  for all $l \in 1..j$ **do**
    $x \leftarrow pointwisemin(x, m_l)$
  end for
  Send x to every $p \in n$
Upon: Query
  return $\hat{N}(x)$

The algorithm consists of a series of rounds towards convergence. In each round every node sends a message containing vector x to its neighbours, collects the corresponding messages from its neighbours and computes the pointwise minimum of x and all corresponding vectors received, updating x with the result. Each node uses function $\hat{N}(x)$, which takes as parameter the vector of K aggregated minimums, and returns an estimation of the number of participants (network size).

One important property of the algorithm is that a node sends the same message to all its neighbours. This means

that broadcast facilities can be explored if available on the underlying network protocols. This is relevant, for example in sensor networks, where broadcast fits naturally and, due to sharing in the physical medium, a unicast has the same cost as a broadcast; algorithms that need to send a different message to each neighbour are at a significant disadvantage.

3. Estimation Function

We first introduce the maximum likelihood estimator $\widehat{N}_F$ used to estimate the unknown parameter N. We then proceed with the theoretical study of its main properties, namely bias and variance. The likelihood function is obtained from the extreme value theory, which is a branch of statistics dealing with the extreme deviations from the median of probability distributions. The following results deal with deviations imposed by the minimum function, but similar results can be easily derived for the maximum.

Let $F_{min}(x) = 1 - (1 - F(x))^N$ be the limiting distribution for the minimum of a large collection $X_1, ..., X_N$ of random observations from the same arbitrary distribution $F(x)$ [8].

Proposition 1: Given a vector of K minimums $x[1], ..., x[K]$, which are observed values from $F_{min}(x)$ distribution, then the maximum likelihood estimator for the unknown parameter N is

$$\widehat{N}_F = -\frac{K}{\sum_{i=1}^{K} \log\{1 - F(x[i])\}}. \quad (1)$$

Proof. The limiting density for the minimum is $f_{min}(x) = \frac{d}{dx}F_{min}(x) = Nf(x)(1 - F(x))^{N-1}$, where $f(x) = \frac{d}{dx}F(x)$. According to the likelihood method, we wish to maximize the function $L(N) = \prod_{i=1}^{K} f_{min}(x[i])$, or equivalently, to maximize $\log L(N)$ where $\log L(N) = K \log N + \sum_{i=1}^{K} \log f(x[i]) + (N-1)\sum_{i=1}^{K} \log\{1 - F(x[i])\}$. From $\frac{d}{dN} \log L(N) = 0$ one concludes that

$$N = -\frac{K}{\sum_{i=1}^{K} \log\{1 - F(x[i])\}}.$$

□

We now concentrate on the special case of using the exponential distribution for $F(x)$ as it will lead to a simple estimator. We will also derive an unbiased estimator for this distribution. (The generic estimator $\hat{N}_F$ above is not necessarily unbiased.) We denote the exponential distribution with rate 1 by $Exp(1)$.

Now, $F(x) = 1 - e^{-x}$, $x \geq 0$ and the corresponding estimator for N becomes

$$\widehat{N}_{Exp} = \frac{K}{\sum_{i=1}^{K} x[i]}.$$

Moreover, $F_{min}(x) = 1 - e^{-Nx}$, $x \geq 0$, is an exponential distribution with rate N, denoted by $Exp(N)$.

In order to correct the bias in $\widehat{N}_{Exp}$ there is a need for an auxiliary lemma, which follows from a straightforward application of Mathematical Statistics (see e.g. [9]).

Lemma 1: If $X_1, \ldots, X_k$ are independent random variables (r.v.'s) from distribution $Exp(N)$, then

a) $\sum_{i=1}^{K} X_i$ is a r.v. from a gamma distribution with shape and scale parameters equal to K and N, respectively.

b) Furthermore, the next expectation and variance hold:

$$\mathrm{E}\left[\frac{1}{\sum_{i=1}^{K} X_i}\right] = \frac{N}{K-1}$$

and

$$\mathrm{Var}\left[\frac{1}{\sum_{i=1}^{K} X_i}\right] = \frac{N^2}{(K-1)(K-2)} - \frac{N^2}{(K-1)^2}.$$

It is now possible to introduce an unbiased estimator for N.

Proposition 2: The estimator given by

$$\widehat{N} = \frac{K-1}{K}\ \widehat{N}_{Exp} = \frac{K-1}{\sum_{i=1}^{K} x[i]} \quad (2)$$

is unbiased.

Proof. We need to prove that the expectation $\mathrm{E}[\widehat{N}]$ is equal to N. Let X_i be the r.v. related to the observed value $x[i]$. First, by Lemma 1, one has

$$\mathrm{E}[\widehat{N}_{Exp}] = \mathrm{E}\left[\frac{K}{\sum_{i=0}^{K} X_i}\right] = K\ \frac{N}{K-1}$$

and

$$\mathrm{E}[\widehat{N}] = \mathrm{E}\left[\frac{K-1}{K}\ \widehat{N}_{Exp}\right] = N.$$

□

Proposition 3: The variance of $\widehat{N}$ is given by

$$Var[\widehat{N}] = \frac{N^2}{K-2}.$$

Proof. This proof is again straightforward from the application of Lemma 1

$$\mathrm{Var}[\widehat{N}] = (K-1)^2\ \mathrm{Var}\left[\frac{1}{\sum_{i=1}^{K} X_i}\right] = \frac{N^2}{K-2}.$$

□

We now generalize this result so that one can estimate a sum of positive reals. This new estimator can be applied to a broad class of aggregations that can be expressed by operations on sums, e.g. AVG. Here the variance is determined by the magnitude of the sum that is to be estimated.

Proposition 4: For $1 \leq i \leq N$, let X_i be independent r.v.'s from distribution $Exp(\lambda_i)$ with $\lambda_i > 0$, and $\mathrm{minimum}(X_1, \ldots, X_N)$ a new r.v. from distribution $Exp(\sum_{i=1}^{N} \lambda_i)$. Given a set of K minimums $x[1], \ldots, x[K]$,

which are observed values from $Exp(\sum_{i=1}^{N} \lambda_i)$, then an unbiased estimator for $Sum = \sum_{i=1}^{N} \lambda_i$ is

$$\widehat{Sum} = \frac{K-1}{\sum_{i=1}^{K} x[i]}$$

with

$$Var[\widehat{Sum}] = \frac{Sum^2}{K-2}.$$

Proof. The proof is straightforward from the proofs of Propositions 2 and 3, renaming N to Sum. □

For presentation purposes, the next section will concentrate on the practical properties and application of the estimator for network size, $\hat{N}$. Nevertheless, most of the analysis is also applicable to the more generic $\widehat{Sum}$ estimator and to other unbiased estimators that can be derived from other probability distributions, e.g., uniform, Gaussian, etc.

4. Binary Encoding

In some application contexts, e.g. mobile ad-hoc networks and sensor networks, message size has an important practical impact both in speed and energy consumption.

Although precision is dictated by the choice of K, there are some relevant design decisions in the floating point encoding of the numbers in the vector. It is intuitive to see that, when aiming for a precision of only a few percent, storing each value naively as a float or double would probably be using a much higher precision than needed. Therefore we tried encoding values with less precision.

After numerically studying several combinations of bit allocations in a binary mantissa and exponent encoding we have concluded that it is appropriate to store only the exponent. Moreover, looking at values that occur in a exponential distribution, and the way that they contribute to the sum in the estimator, even though there can be more than 20 binary orders of magnitudes in the values that occur, a range of only 9 values in the exponent contributes to 99.9% of the result.

Table 1 shows the relative cumulative contribution of values from higher to lower exponents occurring in a exponential distribution. The exponents shown, from 3 to -5 would be the ones contributing almost exclusively to the sum, for $N = 1$ (1 node network). The distribution of minimums for a N node network is also exponential, but with the range of meaningful values scaled by $1/N$. For a given maximum N, we must use a range of exponents that is $9 + \log_2(N)$. This leads to using 5 bits for storing the exponent, to account for possibly large networks up to about 8 million nodes: 5 bits gives a range of 32 for the exponent; this means networks up to $2^{32-9} = 2^{23}$ nodes. (Using 4 bits would only allow up to $2^{16-9} = 2^7 = 128$ nodes.)

Table 1. Relative cumulative contribution.

exponent	contribution (%)
3	0.350
2	10.26
1	42.64
0	74.99
-1	91.54
-2	97.53
-3	99.33
-4	99.82
-5	99.95

Table 2. Scale factor $s(K)$ and respective standard deviation. 5 bits, base=2. Using 50 points and *sample* repetitions per point.

K	*sample*	$s(K)$	$sd[s(K)]$
10000	100	0.7212	0.0007
1000	1000	0.7212	0.0008
100	10000	0.7208	0.0008
10	100000	0.7161	0.0008

A given real value v in vector x is encoded by the integer $\text{floor}(\log_2 v)$, and when reconstructed becomes $\underline{v} = 2^{\text{floor}(\log_2 v)}$. Likewise, the base 2 discretisation of vector x is denoted by $\underline{x}$.

Although $\hat{N}(x)$ was proved to be unbiased, the coarser grain discretisation due to encoding introduces a bias in $\hat{N}(\underline{x})$. This bias can be corrected as it is possible to calculate a scale factor $s(K)$ such that $E[\hat{N}(x)] \approx E[s(K)\hat{N}(\underline{x})]$.

We considered the possibility of choosing higher bases for encoding, with the intent of reducing the number of bits needed to encode the same range. We can observe that in order to reduce one bit we need to square the base. However the bias correcting scale factor for other bases $b > 2$ shows a non negligible dependence on N, with a periodic oscillation on $\log_b N$.

Calculation of the base 2 scale parameter $s(K)$, was performed numerically and is depicted in Table 2. This value shows a slight dependence on K. This is due to a small change in the shape of the distribution of $\hat{N}$ for small values of K, since the r.v. $\hat{N}$ follows a Gamma distribution with shape parameter K.

Since K is known and configured in the protocol, and the relative periodic oscillations on N are less than 0.001 for base 2, one simply needs to pick the appropriate scale factor for the used K. In short, under binary encoding the estimator for N becomes:

$$s(K)\frac{K-1}{\sum_{i=1}^{K} \underline{x[i]}},$$

where $s(K)$ is the scale parameter for a given K as depicted in Table 2.

From the results in Section 3 we can define a metric that indicates the relative error of the estimation. The metric is

Table 3. Theoretical and average observed relative errors. 5 bits, base=2. Using 200 points from $N = 1$ to $N = 2^{20}$ and J samples per point.

K	J	TRE	ORE
10000	10	0.0100	0.0098
1000	100	0.0316	0.0328
100	1000	0.1010	0.1047
10	10000	0.3535	0.3651

named TRE (*Theoretical Relative Error*) and is defined as follows:

$$TRE = \frac{\sqrt{Var[\hat{N}]}}{N} = \frac{1}{\sqrt{K-2}}.$$

This metric indicates how the estimation deviates from N as a proportion of N.

In order to numerically measure the quality of the estimator after encoded and scale corrected, we define the following metric, named ORE (*Observed Relative Error*)

$$ORE = \frac{\sqrt{\frac{\sum_{i=1}^{J}\left(\hat{N}_i - N\right)^2}{J}}}{N},$$

where $\hat{N}_i$, for $i = 1..J$, is a set of observations of the estimate of a given N. Both metrics are defined in terms of the MSE (*Mean Square Error*), since *Relative Error* can be seen as $\frac{\sqrt{MSE}}{N}$.

To obtain an average observed relative error over different network sizes, we use 200 values of N ranging from $N = 1$ to $N = 2^{20} \approx 10^6$; this gives us 10 values for each cyclic oscillation. Table 3 shows how the values for TRE and the average ORE compare, for different $K \in \{10, 100, 1000, 10000\}$ (with J samples for each N). We can conclude that for practical purposes the observed values agree with the theoretical ones.

5. Message Loss and Slow Links

A strong point in our estimation technique is that it is suitable to address scenarios where message loss can occur. Contrary to techniques such as [10], which cannot afford to lose messages, in ours the knowledge in each message is made obsolete by subsequent ones: if a message from A to B containing vector x is lost, a subsequent message will have content y, where $y \leq x$ (in pointwise order).

This means that our algorithm can be easily modified to deal with message loss. The algorithms presented send a message to all neighbours and wait for messages from all neighbours. This means that a single message loss will deadlock the entire system. Some simple modifications to deal with the problem are possible:

- A possibility is the use of a timeout. Normally the algorithm would wait for messages from all neighbours, but if more than some time elapsed, it would proceed using the messages received so far.
- Another possibility is to design the algorithm to cope with the failure of F messages, for small F like 1 or 2, and make it wait from messages from all neighbours minus F.

The second variation is interesting in another point: it would make the algorithm robust to slow links. Waiting from all minus e.g. 1 neighbour means that if the last message would take much more time to arrive it would not slow down the starting of the next round. The vector in these late messages could be accounted for (in the subsequent round) in computing x, so that we do not ignore a node whose messages are consistently the last one to arrive. The possible increase in the number of rounds would be balanced by faster rounds.

In these variations each message would be tagged with a round number, to distinguish messages from the current round from older rounds. A combination of these possibilities would be to wait for all neighbours until a timeout and then wait for all still missing minus F.

6. Related Work

The use of idempotent messages for duplicate insensitive aggregations in sensor networks was presented on [5], [3], [15]. These papers make use of a sketching technique developed by Flajolet and Martin in [7] and recently enhanced in [6]. The technique, referred to as FM sketches, was developed to estimate the number of distinct elements in a multiset.

Our approach, building on extreme value statistics, operates in the real domain and can estimate sums of positive reals. FM sketches, builds on the use of hash functions and bitmaps and is a discrete technique than can estimate sums of positive integers. It follows that FM sketches are less general.

Although intrinsically different the two techniques have important similarities. If K is the number of units dedicated to the estimation, both estimate with a relative standard error of roughly $O(1/\sqrt{K})$.

When considering the effect of binary encoding, we observe that in [3], [5] the authors use the non enhanced FM sketches and thus would only be able to encode in 5 bits a network size up to 2^5. For practical uses they would need at least 16 bits per unit. Considering the enhanced version of FM sketches in [6], one could expect in 5 bits to be able to count up to 2^{32} while we are limited to about 2^{23}. However it is not clear whether this version would be adequate to estimate both small and large values of N, since the technique was developed for large cardinalities.

The work on Separable Functions [14] was developed in parallel with our work [2] and reaches an estimator which is biased and does not converge to N but instead to $\frac{K}{K-1}N$, thus, it is less correct for small values of K. Both approaches are related to earlier work on k-mins sketches [4], that estimates the size of reachability sets in graphs.

7. Conclusions

We have introduced Extrema Propagation, a practical approach to distributed aggregation, based on the use of the statistical theory of extreme values. The resulting unbiased estimators for exponential distributions lead to very simple algorithms and efficient implementations. Being able to estimate sums of positive reals, we are more expressive than most previous approaches: our technique encompasses summing naturals and counting, constituting an important building block for the construction of aggregate functions.

The technique is fast: all nodes have correct estimates after, at most, a number of communication steps equal to the network diameter, and in this sense we operate at the theoretical minimum.

In the algorithm a node sends the same message to all its neighbours. This means that broadcast facilities can be explored if available on the underlying network protocols. This is relevant, for example in sensor networks where, due to sharing in the physical medium, a unicast has the same cost as a broadcast.

Useful estimates can be obtained using short messages; we have shown that, in this context, only 5 bits are needed to represent a floating-point number; an estimate with a 10% error with 95% confidence can be obtained using around 240 bytes for the vector sent in a message.

Finally, Extrema Propagation possesses some interesting properties: it is fully distributed with no single point of failure and with the result produced at every node, it does not require system-wide identifiers, and it is suitable to tolerate message loss and slow links (by slight changes to the algorithm).

References

[1] Ittai Abraham and Dahlia Malkhi. Probabilistic quorums for dynamic systems. In Faith E. Fich, editor, *DISC*, volume 2848 of *Lecture Notes in Computer Science*, pages 60–74. Springer, 2003.

[2] Carlos Baquero, Paulo Sérgio Almeida, and Raquel Menezes. Extrema propagation: Fast distributed estimation of sums and network sizes. Technical report, Universidade do Minho, May 2006.

[3] Mayank Bawa, Hector Garcia-Molina, Aristides Gionis, and Rajeev Motwani. Estimating aggregates on a peer-to-peer network. Technical Report TR-2003-24, Stanford University, 2003.

[4] Cohen. Size-estimation framework with applications to transitive closure and reachability. *JCSS: Journal of Computer and System Sciences*, 55, 1997.

[5] Jeffrey Considine, Feifei Li, George Kollios, and John W. Byers. Approximate aggregation techniques for sensor databases. In *ICDE*, pages 449–460. IEEE Computer Society, 2004.

[6] Marianne Durand and Philippe Flajolet. Loglog counting of large cardinalities (extended abstract). In Giuseppe Di Battista and Uri Zwick, editors, *ESA*, volume 2832 of *Lecture Notes in Computer Science*, pages 605–617. Springer, 2003.

[7] Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, 1985.

[8] E. J. Gumbel. *Statistics of Extremes*. Columbia University Press, 1958.

[9] R. V. Hogg and A. F. Craig. *Introduction to Mathematical Statistics*. Prentice-Hall, Upper Saddle River, New Jersey, 5th edition, 1995.

[10] Márk Jelasity, Alberto Montresor, and Özalp Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.*, 23(3):219–252, 2005.

[11] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *FOCS*, pages 482–491. IEEE Computer Society, 2003.

[12] Ji Li, Karen R. Sollins, and Dah-Yoh Lim. Implementing aggregation and broadcast over distributed hash tables. *Computer Communication Review*, 35(1):81–92, 2004.

[13] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. Tag: A tiny aggregation service for ad-hoc sensor networks. In *OSDI*, 2002.

[14] Damon Mosk-Aoyama and Devavrat Shah. Computing separable functions via gossip. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 113–122, July 2006.

[15] Suman Nath, Phillip B. Gibbons, Srinivasan Seshan, and Zachary R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In John A. Stankovic, Anish Arora, and Ramesh Govindan, editors, *SenSys*, pages 250–262. ACM, 2004.

[16] D. Psaltoulis, D. Kostoulas, I. Gupta, K. Birman, and A. Demers. Practical algorithms for size estimation in large and dynamic groups. Technical report, University of Illinois, Urbana-Champaign, 2004.

[17] Ion Stoica, Robert Morris, David R. Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM*, pages 149–160, 2001.

[18] Robbert van Renesse. The importance of aggregation. In André Schiper, Alexander A. Shvartsman, Hakim Weatherspoon, and Ben Y. Zhao, editors, *Future Directions in Distributed Computing*, volume 2584 of *Lecture Notes in Computer Science*, pages 87–92. Springer, 2003.