# Extrema Propagation: Fast Distributed Estimation of Sums and Network Sizes

Carlos Baquero, Paulo Sérgio Almeida, Raquel Menezes, and Paulo Jesus

**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 nonidempotent 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 multipath routing occurs; it is tolerant of message loss; it is fast, as the number of message exchange steps can be made just slightly above the theoretical minimum; and it is fully distributed, with no single point of failure and the result produced at every node.

**Index Terms**—Aggregation, network size estimation, distributed sums, probabilistic estimation, self-configuration.

## 1 Introduction

Aggregation is recognized as an important building block for distributed applications in peer-to-peer, ad-hoc, and sensor network infrastructures [1], [2], [3]. 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 [4], when setting a quorum for voting algorithms [5], 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 [6], [7], 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 nonidempotent functions (e.g., COUNT, SUM, AVG) is problematic and we are restricted to idempotent functions that are duplicate insensitive (e.g., MIN, MAX) [8]. Aggregation functions, that can be made duplicate insensitive, have the advantage of being usable under multipath routing.

This paper presents Extrema Propagation, a technique for distributed estimation of the sum of positive real numbers. It is a probabilistic technique, based on the statistics of extremes, that exchanges duplicate insensitive messages and, thus, can be applied in flood and/or epidemic settings, where multipath routing occurs. It can also be easily adapted to tolerate message loss. The paper expands on our earlier results in [9] and shows a comprehensive presentation of the technique that complements the theoretical analysis of the statistical estimator with important practical concerns: message size reduction by controlling the binary encoding of reals; termination detection (i.e., knowing that the estimation has converged); handling of message loss and variable link latency in asynchronous settings.

Extrema Propagation has some important properties: the precision is controlled by message size, independently of network size; it is fast: the number of message exchange steps can be made just slightly above the theoretical minimum; 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.

The remainder of this paper is organized as follows. Section 2 introduces the basic algorithm for network size estimation. Section 3 introduces and proves the correctness of a maximum likelihood estimator that can be used for both counting and summing of a distributed set of positive values. In Section 4, we describe a compact bit encoding of the real values that fits the requirements of the estimator. Section 5 deals with termination detection in different network topologies. In Section 6, we sketch how the algorithm can be adapted for an asynchronous setting with message loss. Finally, we contrast in more detail our results with the related work, in Section 7, and conclude in Section 8.

- *C. Baquero, P.S. Almeida, and P. Jesus are with the Departamento de Informática, Universidade do Minho, Campus de Gualtar, 4710-057 Braga, Portugal. E-mail: {cbm, psa, pcoj}@di.uminho.pt.*
- *R. Menezes is with the Departamento de Matemática e Aplicações, Universidade do Minho, Campus de Azurem, 4800-058 Guimarães, Portugal. E-mail: rmenezes@math.uminho.pt.*

*Manuscript received 2 Aug. 2010; revised 8 Apr. 2011; accepted 10 July 2011; published online 21 July 2011.*
*Recommended for acceptance by S. Ranka.*
*For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2010-08-0462.*
*Digital Object Identifier no. 10.1109/TPDS.2011.209.*

## 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 neighbor 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 is later addressed in Section 6.

The estimation of network size in all nodes takes at least $D$ (the network diameter) rounds, to allow each node at distance $D$ from some others to become aware of them.

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 both the need for unique identifiers and message sizes which depend on network size [10]. It is based on idempotent operations on numbers, more specifically the minimum function, and the use of statistical inference.

### 2.1 Synopsis of the Estimation Technique

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 percent we need to make $K = 2 + (\frac{1.96}{r})^2$. 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 percent would lead to large messages), but for a fast computation of a useful approximation that can serve as input to some other algorithm (in some cases even 10 percent may be more than enough, only the order of magnitude may be needed).

### 2.2 Basic 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.** Basic Extrema Propagation

```
const K
var n, x[1..K]
Upon: Init
    n ← neighbors(self)
    for all i ∈ 1..K do x[i] ← rExp(1)
    Send x to every p ∈ n
Upon: Receive m_1..m_j from all p ∈ n
    for all l ∈ 1..j do
        x ← pointwisemin(x, m_l)
    end for
    Send x to every p ∈ n
Upon: Query
    return N̂(x)
```

The algorithm consists of a series of rounds toward convergence. In each round every node sends a message containing vector $x$ to its neighbors, collects the corresponding messages from its neighbors, 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). In this first version, we do not deal with termination and assume that a node can be queried at any time, possibly before convergence is reached, i.e., before we have collected the pointwise minimum of every vector in the network. Termination is addressed below.

One important property of the algorithm is that a node sends the same message to all its neighbors. 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 neighbor 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 $X_1, \ldots, X_N$ of random observations from the same arbitrary distribution $F(x)$ [11].

**Proposition 1.** *Given a vector of $K$ minimums $x[1], \ldots, x[K]$, 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., [12]).

**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*

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

**Proposition 2.** *The estimator given by*

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

*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 $Sum$. □

## 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.

It is intuitive to see that, when aiming for a precision of only a few percent, storing each value in the vector 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 an exponential distribution, and the way that they contribute to the sum in the estimator, a range of only nine binary orders of magnitude in the exponent contributes to 99.9 percent of the result.

Table 1 shows the relative cumulative contribution of values from higher to lower exponents occurring in an exponential distribution. The exponents shown, from 3 to $-5$

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 |

would be the ones contributing almost exclusively to the sum, for $N = 1$ (1 node network). The distribution of minimums for an $N$ 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.)

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 discretization of vector $x$ is denoted by $\underline{x}$.

Although $\hat{N}(x)$ was proved to be unbiased, the coarser grain discretization 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 $E[\hat{N}(x)] \approx E[s(K)\hat{N}(\underline{x})]$.

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 configured in the protocol one simply needs to pick the appropriate scale factor for the respective $K$. In short, under binary encoding the estimator for $N$ becomes

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

We can define a metric that indicates the relative error of the estimation. The metric is named $TRE$ (*Theoretical Relative Error*) and is defined as follows:

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

TABLE 2
Scale Factor $s(K)$ and Respective Standard Deviation

| $K$ | *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 |

*5 bits,* $\text{base} = 2$*. Using 50 points and* sample *repetitions per point.*

TABLE 3
Theoretical and Average Observed Relative Errors

| $K$ | $J$ | 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 |

*5 bits,* $\text{base} = 2$*. Using 200 points from* $N = 1$ *to* $N = 2^{20}$ *and* $J$ *samples per point.*

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 $N = 2^{20} \approx 10^6$. Table 3 shows how the values for $TRE$ and the average $ORE$ compare, for different $K \in \{10, 100, 1{,}000, 10{,}000\}$ (with $J$ samples for each $N$). We can conclude that for practical purposes the observed values agree with the theoretical ones.

## 5 Termination Detection

Until now, we have not addressed termination. In the basic algorithm, nodes can be queried at any time, before we have taken into account the vectors from every node in the network. If we query it too soon, messages from distant nodes will not have yet contributed and the estimate will be a number smaller and unrelated to the network size.

As the algorithm collects vectors from all neighbors, at each new round the algorithm takes into account vectors from all nodes one hop further than in the previous round; i.e., the "visibility radius" increases at each round.

The intuition for the termination is as follows: at each round, we collect information from some new nodes (all nodes that entered the expanded visibility radius); as each of these nodes contributes with $K$ random numbers, the probability that no new minimum is obtained at any index in the vector (i.e., that "no news" has arrived) is small; moreover, the probability that such "absence of news" occurs $T$ times in a row is close to zero even for a small $T$ (smaller $T$ for larger $K$).

The termination of the algorithm is based precisely on the detection of $T$ (for some configurable $T$) consecutive rounds where no component changed in the vector stored locally. When that happens the algorithm assumes that all nodes have contributed and the result can be reported. The algorithm including termination detection is shown in Algorithm 2.

**Algorithm 2.** Extrema Propagation with termination detection

**const** $K, T$
**var** $n, x[1..K]$
**var** $oldx[1..K], nonews, converged$
**Upon:** Init
$nonews \leftarrow 0$
$converged \leftarrow False$
$n \leftarrow neighbors(self)$
**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 $p \in n$
$oldx \leftarrow x$
**for all** $l \in 1..j$ **do**
$x \leftarrow pointwisemin(x, m_l)$
**end for**
**if** $oldx \neq x$ **then**
$nonews \leftarrow 0$
**else**
$nonews \leftarrow nonews + 1$
**end if**
**if** $nonews \geq T$ **then**
$converged \leftarrow True$
**end if**
Send $x$ to every $p \in n$
**Upon:** Query & $converged = True$
**return** $\hat{N}(x)$

In order to determine the appropriate value of $T$ to use for termination detection, we have simulated runs of the algorithm for different network settings. Detailed results are depicted in the *supplemental material*, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.209. In short, suitable values for $T$ are a small fraction of the network diameter, which makes the overhead of termination detection a small overhead over the optimal number of rounds.

## 6 Asynchrony and Message Loss

A strong point in our estimation technique is that it is suitable to address scenarios where message loss and arbitrary delays can occur. Contrary to techniques such as [2], 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 neighbors and wait for messages from all neighbors. This means that a single message loss will deadlock the entire system. Some simple modifications to deal with the problem are possible.

- One is the use of a timeout, ONLY-TIMEOUT strategy. The algorithm would wait for messages from all neighbors, but if more than some time elapsed, it would proceed using the messages received so far.
- Another is to wait until a given number of neighbors have responded. However, this must be complemented with a timeout, since it would still deadlock if less than those neighbor messages arrived. This strategy is referred as F-PLUS-TIMEOUT and parametrized by the $F$ ratio of neighbor messages that can be missed and the timeout value.

These strategies can make the algorithm robust to message loss and slow links. In order to evaluate them and determine adequate values for the timeout parameter, we conducted a simulation of the algorithm in an asynchronous setting and considered both fault free and 20 percent message loss scenarios. The setup and details are described in the available online *supplemental material* and we resume here the main findings.

The F-PLUS-TIMEOUT depicted no significant advantage over the simpler ONLY-TIMEOUT strategy. This later strategy was analyzed in detail over the influence of the chosen timeout value. Short timeout values lead to fast convergence at the expense of a high level of message transmissions, while higher timeouts induce slower convergence under message loss. We concluded that, as a rule of thumb, we will be well served by choosing a timeout corresponding to the 98 percent percentile of the CDF of the latency distribution.

## 7 Related Work

Several distributed algorithms to estimate sums and network sizes (i.e., count) can be found in the literature. Some require the use of a specific communication structure, like classic tree-based and cluster-based aggregation techniques, such as TAG [13], [14] and I-LEAG [15], [16]. These hierarchy-based approaches are known to be cheap in terms of message exchange, being commonly used in WSN (Wireless Sensor Networks) due to energy efficiency. However, a single point of failure might greatly impact their accuracy.

Other approaches [17], [18] rely on the existence of a ring, establishing successor relations among nodes. In [17], a network size estimation is produced at each node relying only on the arrival and departure of nodes, without further communication, simply by incrementing/decrementing local estimators. However, this scheme provides coarse estimates, ranging from $N/2$ to $N^2$. In [18], the network size is estimated based on the average distance between consecutive nodes on a ring, executing an averaging process (Push-Pull Gossiping [2]). This approach makes strong assumptions on the ring structure, besides it should inherit the dependability issues of the used push-pull protocol [19]. Here, we are interested in providing an estimate in a robust way, without assuming the existence of any predefined routing structure.

Regarding unstructured aggregation approaches, three main groups of techniques can be identified: sampling, sketching, and averaging. A study comparing a few sampling and averaging approaches is found in [20], but sketching techniques have been left out of it.

### 7.1 Sampling

These kind of algorithms [21], [22], [23], [24], [25] rely on the results obtained from some sampling process to probabilistically estimate the size of the sampled population, generally assuming that nodes possess unique identifiers. These approaches are not accurate, with an approximation

error that depends on the quality of the collected sample and the used estimator. Moreover, sampling is commonly slow, performing random walks and taking several rounds to collect one sample at a single node. For example: in Sample and Collide [21], [22] a single step (random walk) takes $\bar{d}T$ (where $\bar{d}$ is the average connection degree and $T$ is a timer value that must be sufficiently large to provide a good sample quality), and must be repeated until $l$ sample collisions have been observed; the capture-recapture method [23] produces an estimate based on the number of repeated peers in each sample, requiring at least two random walks at a source node, and being the sampling quality highly affected by the network properties (e.g., degree).

Two different sampling algorithms, not based on random walks, are presented in [24] and [25]. Still, both assume that all nodes have unique IDs, and produce the estimate at an initiator node. The first, *Hops Sampling* is a gossip-based technique to sample receipt times, requiring a membership list chosen uniformly at random, and the capability to establish connections between arbitrary nodes. The second, *Interval Density* has lighter requirements. In this case, nodes have randomized IDs mapped into the interval $[0, 1]$, which are collected the initiator. The estimate is produced by determining the number $X$ of IDs that fall in a given subregion $I$ of $[0, 1]$, returning $X/I$. The weakness is that it is difficult to set an adequate $I$ since $N$ is not known and the transmitted data are always a fraction of $N$. For both algorithms the achieved relative accuracy was reported to be 5 percent.

### 7.2 Sketching

The use of idempotent messages for duplicate insensitive aggregations in WSN was presented on [26], [27], [8], and [28]. These papers make use of a sketching technique, referred to as FM sketches, which was developed by Flajolet and Martin [29] to estimate the number of distinct elements in a multiset, and further enhanced in [30] and [31].

FM sketches is a discrete technique that builds on the use of hash functions and bitmaps and that can estimate sums of positive integers. Our approach is more general, building on extreme value statistics and operating in the real domain. 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 [27], [26], and [28] the authors use the nonenhanced 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 [30], one could expect in 5 bits to be able to count up to $2^{32}$ while we are limited to about $2^{23}$. However, the accuracy of FM sketches for small counts is very weak.

The COMP algorithm [32], related to the earlier work on *k-mins sketches* [33], reaches an estimator $\widehat{N}_{SF}$ that is equivalent to a biased version of our exponential estimator for sums, $\widehat{Sum}$. Their estimator is biased and does not converge to $N$ but instead to $\frac{K}{K-1}N$, thus, it is much less accurate for small values of $K$.

The statistics that lead to the estimator in [32] followed a different path and build on asymptotic properties of exponential random variables, being tied to this distribution. In contrast, our results (see Proposition 2) are more general and apply to an arbitrary continuous distribution $F(x)$. In our case, the exponential distribution is one possible instantiation that has the advantage of leading to a simple formula in the derived estimator.

When considering the accuracy of $\widehat{N}_{SF}$, Mosk-Aoyama and Shah derive in [32] (where $\alpha = 2\epsilon$) the expression

$$Pr(|\widehat{N}_{SF} - N| > \alpha N) \leq 2e^{-\frac{\alpha^2 K}{12}}.$$

In order to compare this envelope with our results, we need to analyze

$$Pr(|\widehat{Sum} - N| > \alpha N).$$

We know the variance and the expectation of $\widehat{Sum}$ and that the asymptotic distribution of maximum likelihood estimators can be approximated by a normal distribution. From this, we derive

$$Pr(|\widehat{Sum} - N| > \alpha N) = 2(1 - \phi(\alpha\sqrt{K-2})),$$

where $\phi()$ denotes the cumulative normal distribution. Comparing the two expressions, one can verify that our error bound is tighter and contained in the looser envelope depicted for $\widehat{N}_{SF}$.

### 7.3 Averaging

A different trade-off in network size estimation can be found in averaging techniques [34], [35], [2], [36], [37], [38]. To compute the COUNT function, these approaches start by setting a value $v$ to 1 at a single node and to 0 in all remaining nodes. Then, nodes successively average the values between its neighbors, and all eventually converge to the network wide average of the initial values. When all values converge each node has an estimate of $N$ in $\hat{N} = 1/v$. Message state can be very small, since one needs to encode a small set of reals with high precision (e.g., a single real in the classic Push-Sum and Push-Pull approaches, or one for each adjacent node in the case of Flow Updating). Convergence requires a number of message exchange steps much larger than the network diameter, making this kind of approach slower than Extrema Propagation. However, averaging approaches are well suited for higher precision estimates.

### 7.4 Comparison

We compared Extrema Propagation against other representative algorithms, namely: COMP [32], Push-Vector [34], and Flow Updating [38]. We choose those algorithms, because like Extrema Propagation they operate independently from the network routing structure, and produce a result at all nodes. In a nutshell, the obtained results show that Extrema Propagation should be preferred to obtain fast results with a fair accuracy, outperforming averaging approaches in terms of speed. Our encoding technique allows a noticeable increase of precision (for the same message size) over COMP. The comparison results are discussed in the available online *supplemental material* (Comparison).

## 8 Conclusions

We have introduced Extrema Propagation, a new 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. Termination detection makes the estimate available after a short additional number of communication steps. Being fast, this technique is suitable for dynamic systems, by adding a simple periodic restart mechanism. The evaluation shows that it outperforms averaging approaches in terms of speed, and is more precise than previous sketching approaches.

In the algorithm, a node sends the same message to all its neighbors. 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 only 5 bits are needed for each floating-point number; an estimate with a 4 percent error with 95 percent confidence can be obtained using $K = 2{,}400$, which allows the vector to fit in a 1,500 bytes MTU.

Finally, Extrema Propagation possesses an assortment of interesting properties: it is fully distributed with no single point of failure and with the result produced at every node, it does not require globally unique identifiers and it is suitable to tolerate message loss and slow links.

## References


[1] R. van Renesse, "The Importance of Aggregation," *Proc. Future Directions in Distributed Computing,* pp. 87-92, 2003.

[2] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-Based Aggregation in Large Dynamic Networks," *ACM Trans. Computer System,* vol. 23, no. 3, pp. 219-252, 2005.

[3] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-Based Computation of Aggregate Information," *Proc. IEEE 44th Ann. Symp. Foundations of Computer Science (FOCS),* pp. 482-491, 2003.

[4] I. Stoica, R. Morris, D.R. Karger, M.F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," *Proc. SIGCOMM,* pp. 149-160, 2001.

[5] I. Abraham and D. Malkhi, "Probabilistic Quorums for Dynamic Systems," *Proc. 17th Int'l Symp. Distributed Computing,* pp. 60-74, 2003.

[6] S. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong, "Tag: A Tiny Aggregation Service for Ad-Hoc Sensor Networks," *Proc. Fifth Symp. Operating Systems Design and Implementation (OSDI),* 2002.

[7] J. Li, K.R. Sollins, and D.-Y. Lim, "Implementing Aggregation and Broadcast over Distributed Hash Tables," *Computer Comm. Rev.,* vol. 35, no. 1, pp. 81-92, 2004.

[8] S. Nath, P.B. Gibbons, S. Seshan, and Z.R. Anderson, "Synopsis Diffusion for Robust Aggregation in Sensor Networks," *Proc. Second Int'l Conf. Embedded Networked Sensor Systems (SenSys),* pp. 250-262, 2004.

[9] C. Baquero, P.S. Almeida, and R. Menezes, "Fast Estimation of Aggregates in Unstructured Networks," *Proc. Fifth Int'l Conf. Autonomic and Autonomous Systems (ICAS),* pp. 88-93, http://doi.ieeecomputersociety.org/10.1109/ICAS.2009.31, 2009.

[10] D. Psaltoulis, D. Kostoulas, I. Gupta, K. Birman, and A. Demers, "Practical Algorithms for Size Estimation in Large and Dynamic Groups," technical report, Univ. of Illinois, http://www.cs.cornell.edu/Info/Projects/Spinglass/Pubs.html, 2004.

[11] E.J. Gumbel, *Statistics of Extremes.* Columbia Univ. Press, 1958.

[12] R.V. Hogg and A.F. Craig, *Introduction to Mathematical Statistics,* fifth ed. Prentice-Hall, 1995.

[13] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks," *ACM SIGOPS Operating Systems Rev.,* vol. 36, no. SI, pp. 131-146, Dec. 2002.

[14] S. Madden, R. Szewczyk, M. Franklin, and D. Culler, "Supporting Aggregate Queries over Ad-Hoc Wireless Sensor Networks," *Proc. IEEE Fourth Workshop Mobile Computing Systems and Applications,* pp. 49-58, Mar. 2002.

[15] Y. Birk, I. Keidar, L. Liss, A. Schuster, and R. Wolff, "Veracity Radius: Capturing the Locality of Distributed Computations," *Proc. 25th Ann. ACM Symp. Principles of Distributed Computing (PODC),* July 2006.

[16] Y. Birk, I. Keidar, L. Liss, and A. Schuster, "Efficient Dynamic Aggregation," *Proc. 20th Int'l Symp. DIStributed Computing (DISC),* pp. 90-104, Sept. 2006.

[17] K. Horowitz and D. Malkhi, "Estimating Network Size from Local Information," *Information Processing Letters,* vol. 88, no. 5, pp. 237-243, 2003.

[18] T. Shafaat, A. Ghodsi, and S. Haridi, "A Practical Approach to Network Size Estimation for Structured Overlays," *Proc. Third Int'l Self-Organizing Systems,* pp. 71-83, Dec. 2008.

[19] P. Jesus, C. Baquero, and P.S. Almeida, "Dependability in Aggregation by Averaging," *Simpósio de Informatica (INForum),* Sept. 2009.

[20] E.L. Merrer, A.-M. Kermarrec, and L. Massoulie, "Peer to Peer Size Estimation in Large and Dynamic Networks: A Comparative Study," *Proc. IEEE 15th Int'l Symp. High Performance Distributed Computing,* Jan. 2006.

[21] A. Ganesh, A. Kermarrec, E.L. Merrer, and L. Massoulié, "Peer Counting and Sampling in Overlay Networks Based on Random Walks," *Distributed Computing,* vol. 20, no. 4, pp. 267-278, 2007.

[22] L. Massoulié, E. Merrer, A.-M. Kermarrec, and A. Ganesh, "Peer Counting and Sampling in Overlay Networks: Random Walk Methods," *Proc. 25th Ann. ACM Symp. Principles of Distributed Computing (PODC),* 2006.

[23] S. Mane, S. Mopuru, K. Mehra, and J. Srivastava, "Network Size Estimation in a Peer-to-Peer Network," technical report, Dept. of Computer Science, Univ. of Minnesota, p. 12, Sept. 2005.

[24] D. Kostoulas, D. Psaltoulis, I. Gupta, K. Birman, and A. Demers, "Decentralized Schemes for Size Estimation in Large and Dynamic Groups," *Proc. IEEE Fourth Int'l Symp. Network Computing and Applications,* pp. 41-48, 2005.

[25] D. Kostoulas, D. Psaltoulis, I. Gupta, K.P. Birman, and A.J. Demers, "Active and Passive Techniques for Group Size Estimation in Large-Scale and Dynamic Distributed Systems," *J. Systems and Software,* vol. 80, no. 10, pp. 1639-1658, Jan. 2007.

[26] J. Considine, F. Li, G. Kollios, and J.W. Byers, "Approximate Aggregation Techniques for Sensor Databases," *Proc. 20th Int'l Conf. Data Eng. (ICDE),* pp. 449-460, 2004.

[27] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani, "Estimating Aggregates on a Peer-To-Peer Network," Technical Report TR-2003-24, Stanford Univ., http://dbpubs.stanford.edu/pub/2003-24, 2003.

[28] A. Manjhi, S. Nath, and P. Gibbons, "Tributaries and Deltas: Efficient and Robust Aggregation in Sensor Network Streams," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 287-298, 2005.

[29] P. Flajolet and G.N. Martin, "Probabilistic Counting Algorithms for Data Base Applications," *J. Computer and System Sciences,* vol. 31, no. 2, pp. 182-209, 1985.

[30] M. Durand and P. Flajolet, "Loglog Counting of Large Cardinalities (Extended Abstract)," *Proc. 11th Ann. European Symp. Algorithms,* pp. 605-617, 2003.

[31] P. Flajolet, E. Fusy, O. Gandouet, and F. Meunier, "Hyperloglog: The Analysis of a Near-Optimal Cardinality Estimation Algorithm," *Int'l Conf. Analysis of Algorithms (AofA),* pp. 127-146, June 2007.

[32] D. Mosk-Aoyama and D. Shah, "Computing Separable Functions via Gossip," *Proc. 25th Ann. ACM Symp. Principles of Distributed Computing,* pp. 113-122, July 2006.

[33] E. Cohen, "Size-Estimation Framework with Applications to Transitive Closure and Reachability," *J. Computer and System Sciences*, vol. 55, no. 3, pp. 441-453, 1997.
[34] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-Based Computation of Aggregate Information," *Proc. IEEE 44th Ann. Symp. Foundations of Computer Science*, pp. 482-491, 2003.
[35] M. Jelasity and A. Montresor, "Epidemic-Style Proactive Aggregation in Large Overlay Networks," *Proc. 24th Int'l Conf. Distributed Computing Systems*, pp. 102-109, Jan. 2004.
[36] F. Wuhib, M. Dam, R. Stadler, and A. Clemm, "Robust Monitoring of Network-Wide Aggregates through Gossiping," *Proc. IFIP/IEEE 10th Int'l Symp. Integrated Network Management*, pp. 226-235, May 2007.
[37] P. Jesus, C. Baquero, and P.S. Almeida, "Fault-Tolerant Aggregation by Flow Updating," *Proc. Ninth IFIP Int'l Conf. Distributed Applications and Interoperable Systems (DAIS)*, pp. 73-86, 2009.
[38] P. Jesus, C. Baquero, and P.S. Almeida, "Fault-Tolerant Aggregation for Dynamic Networks," *Proc. IEEE 29th Symp. Reliable Distributed Systems*, pp. 37-43, 2010.

**Carlos Baquero** received the MSc and PhD degrees from the Universidade do Minho in 1994 and 2000, respectively. Currently, he is working as an assistant professor at the Computer Science Department in the Universidade do Minho (Portugal). His research interests are focused on distributed systems, in particular, in causality tracking, peer-to-peer systems, and distributed data aggregation. Recent research is focused on highly dynamic distributed systems, both in Internet P2P settings and in mobile and sensor networks.

**Paulo Sérgio Almeida** received the MSc degree in electrical engineering and computing from the Universidade do Porto in 1994 and the PhD degree in computer science from Imperial College London in 1998. Currently, he is working as an assistant professor at the Computer Science Department in the Universidade do Minho (Portugal). His research interests are focused on distributed systems, in particular, distributed algorithms (namely aggregation) and logical clocks for causality tracking (with applications to optimistic replication).

**Raquel Menezes** received the MSc degree in computer sciences from Minho University in 1996, and the PhD degree in mathematics/statistics from Santiago de Compostela and Lancaster Universities in 2005. Currently, she is working as an assistant professor in the Department of Mathematics and Applications of the Universidade do Minho (Portugal). Her current main interest include spatial statistics, mainly geostatistics, and nonparametric estimation, motivated by environmental and health applications.

**Paulo Jesus** received the BEng degree in systems and informatics in 2001, and the MSc degree in mobile systems in 2007, both from the Universidade do Minho (Portugal). Currently, he is working toward the PhD degree as a student of the MAP-i doctoral program in computer science by the Universities of Minho, Aveiro and Porto (Portugal). He worked during 5 years as a software developer, and taught informatics during 1 year. His research interests include distributed algorithms, fault tolerance, and mobile systems.