Gossip protocols for renaming and sorting

George Giakkoupis, Anne-Marie Kermarrec, Philipp Woelfel

Gossip Protocols for Renaming and Sorting

George Giakkoupis[1]*, Anne-Marie Kermarrec[1], and Philipp Woelfel[2]**

[1] INRIA Rennes – Bretagne Atlantic, France
{george.giakkoupis,anne-marie.kermarrec}@inria.fr
[2] Department of Computer Science, University of Calgary, Canada
woelfel@ucalgary.ca

Abstract. We devise efficient gossip-based protocols for some fundamental distributed tasks. The protocols assume an n-node network supporting point-to-point communication, and in every round, each node exchanges information of size $O(\log n)$ bits with (at most) one other node. We first consider the *renaming* problem, that is, to assign distinct IDs from a small ID space to all nodes of the network. We propose a renaming protocol that divides the ID space among nodes using a natural push or pull approach, achieving logarithmic round complexity with ID space $\{1, \ldots, (1+\epsilon)n\}$, for any fixed $\epsilon > 0$. A variant of this protocol solves the *tight* renaming problem, where each node obtains a unique ID in $\{1, \ldots, n\}$, in $O(\log^2 n)$ rounds.
Next we study the following *sorting* problem. Nodes have consecutive IDs 1 up to n, and they receive numerical values as inputs. They then have to exchange those inputs so that in the end the input of rank k is located at the node with ID k. Jelasity and Kermarrec [20] suggested a simple and natural protocol, where nodes exchange values with peers chosen uniformly at random, but it is not hard to see that this protocol requires $\Omega(n)$ rounds. We prove that the same protocol works in $O(\log^2 n)$ rounds if peers are chosen according to a non-uniform power law distribution.

Keywords: renaming, sorting, gossip protocols, epidemic protocols, distributed algorithms, randomized algorithms, network algorithms

1 Introduction

Today's highly distributed systems are based on networks of massive scale. Such networks often suffer from link and node failures, and from limited computational capabilities of their nodes. For example, peer-to-peer and mobile ad-hoc networks are inherently highly dynamic, with nodes joining and leaving the system frequently; or sensor networks are often used in harsh environments leading to communication disruptions, and their nodes have little computational power.

* This work was funded in part by INRIA Associate Team RADCON and ERC Starting Grant GOSSPLE 204742.
** This research was supported in part by a Discovery Grant and the Canada Research Chair Program of the Natural Sciences and Engineering Research Council of Canada (NSERC), and in part by the HP Innovation Research Program.

Gossip (or *epidemic*) protocols have emerged as an important communication paradigm for these networks. In gossip protocols nodes repeatedly contact random neighbors and exchange small amounts of information in order to distribute and gather information. Such protocols are usually simple, scalable, and fault-tolerant. They generally offer small communication overhead and modest demands on the nodes' storage space and computational power. Even though they only provide probabilistic guarantees, the probability of failure typically converges quickly to 0 with the time the protocol is run.

The classical problem solved with gossip protocols is *rumor spreading* [10] in the random phone-call model [22]. In this model, nodes exchange information in synchronous parallel communication rounds, using either push, pull, or push-pull communication with peers chosen uniformly at random among all nodes (or just among the node's neighbors, if the network topology is not a complete graph). Such rumor spreading protocols have been shown to be very efficient, requiring only a logarithmic number of rounds for the complete graph and various other topologies [22, 19, 14, 18, 7, 12, 13].

Later, gossip protocols have been used to solve node aggregation problems [6, 23, 28, 8]. Here, the goal is to compute the value $f(x_1, \ldots, x_n)$ of some aggregation function f (e.g., sum, average, or extrema), where x_i is an input to the i-th node. Most gossip protocols for aggregation need only poly-logarithmic many rounds in the complete graph before nodes know the value of the aggregation function (with sufficient accuracy) with high probability. In the design of gossip protocols it is often assumed that any given node can in each round exchange information with a peer selected uniformly at random from all nodes, independently of the network topology. In practice [17], this is usually realized by a peer-sampling service [21], which can be singled out from the application.

In the present paper, we study practical and fundamental problems that cannot be expressed by aggregation functions. First, we study the problem of *renaming*. Here, every node must obtain a unique ID from an *ID space* $\{1, \ldots, m\}$ of size $m \geq n$. The renaming problem has been studied extensively in the distributed computing literature, especially in the areas of shared memory and message passing (see, e.g., [2] and references therein). Many distributed tasks can only be solved if the participants have unique IDs, and often the complexity of algorithms depends on the size of the domain from which those IDs are chosen. For example, an algorithm to construct overlay networks in peer-to-peer networks proposed by Angluin et al. [3] has expected round complexity $O(W \log n)$, where W is the bit-length of node IDs. Another application is the unique assignment of a small number of resources (e.g., servers or printers) to processors (nodes). Nodes can also use their IDs as "tags" to mark their presence in some data structure (e.g., a priority queue), so that a node can distinguish whether itself or some other node has placed the tag [4]. We solve both, the *loose renaming* problem, where $m = (1 + \epsilon)n$ for some constant $\epsilon > 0$, and the *tight renaming* problem, where $m = n$, with simple protocols that have respectively $O(\log n)$ and $O(\log^2 n)$ round complexity with high probability, and logarithmic message-size complexity. Both protocols assume that each node can contact a

uniform random node in a round. The tight renaming protocol assumes further that a node can contact an arbitrary node directly, if it knows its network address (see Section 1.1). Note that non-gossip based algorithms, e.g., algorithms based on leader election protocols, can be used to solve tight renaming in $O(\log n)$ time. But contrary to our gossip based solution, such algorithms require "exact" communication, and tolerate no or almost no transmission faults.

Then we consider the problem of *sorting* n input values $x_1, \ldots, x_n$, each one given to a distinct node. Here we assume that the n nodes have consecutive IDs in $\{1, \ldots, n\}$. Nodes must exchange their input values in multiple communication rounds, such that in the end the value of rank k is located at the node with ID k. Jelasity and Kermarrec [20] proposed the following simple gossip protocol for this problem: In each round, a node contacts a peer chosen uniformly at random, and both nodes exchange their values, if they are out of order with respect to their IDs. However, this protocol may need in expectation $\Omega(n)$ rounds until all input values are sorted. For example, suppose node 1 holds value 2 and node 2 holds value 1, and each node $i \geq 3$ holds value i. Then it takes $\Omega(n)$ rounds in expectation before nodes 1 and 2 contact each other and resolve their inversion. (There are other input instances for which it takes up to $\Omega(n \log n)$ rounds with high probability before all input values are sorted.) We show that the round complexity drops to $O(\log^2 n)$, if peers are not chosen *uniformly* at random, but rather from a power law distribution: A node with ID x chooses a peer with ID y with a probability inversely proportional to $|x - y|$. (A similar distribution for sampling peers is used in Kleinberg's small-world graph routing scheme [26, 25], and also in the spatial gossip algorithms proposed by Kempe et al. [24].)

Our protocols for renaming and sorting are very simple and natural, however, their analysis is non-trivial and is based on potential function arguments. Further, the protocols can tolerate random transmission faults, similar to the standard rumor spreading protocols [15]. I.e., if communication channels fail to be established between parties independently with a probability of q, then the round complexity increases only by a factor of at most $1/(1 - q)$, which is the expected number of trials before a connection is established.

1.1 Model and Practical Considerations

We assume that the network supports the abstraction of point-to-point communication. That is, each node has a unique network *address* from some arbitrary domain, and node u can contact any other node v, if u knows v's address. Nodes do not know the addresses of other nodes in advance, but they can find out during the course of the protocol. When two nodes have established a communication channel, both can reliably exchange information for one round.

We assume further that the abstraction of a *random peer-sampling service* is supported. Each time this service is invoked it returns a node chosen independently and uniformly at random among all nodes. In a large-scale dynamic system it is unrealistic that nodes maintain complete tables of network addresses of peers, from which they can sample at random. To overcome this obstacle, various distributed designs of peer-sampling services have been proposed and stud-

ied experimentally by the systems community (see, e.g., [21]). The use of such services has become a standard practice in the implementation of gossip-based systems [17]. This service is often implemented by building and maintaining a random overlay network, that changes over time by having nodes exchange random fractions of their list of neighbors with other (randomly selected) neighbors. For related theoretical results on this problem see, e.g., [16, 9].

Our loose renaming algorithm relies only on the assumption that in each round a node can contact some uniformly random node. The tight renaming algorithm has the additional requirement that a node can contact a node by its address. Initially, nodes do not know the address of any other node, but a node can add its own address to a message (or the address of another node it knows of), thus allowing the recipients of that message to contact the node directly in future rounds. We stress that addresses may come from an arbitrary large space that may be much larger than n, thus they cannot be used themselves as IDs.

For the sorting algorithm we assume that nodes already have IDs 1 up to n. Similar to the loose renaming algorithm, the sorting algorithm does not use network addresses directly. However, it requires a non-uniform peer-sampling service, which allows each node with ID i to choose a random node with ID j according to a probability distribution that depends on $|i - j|$. Precisely, the probability of choosing j needs to be inversely proportional to $|i-j|$. A DHT-like overlay network can be used to provide this service: By overlaying the network with a Chord topology [29], peer-sampling with the required power-law distribution can be achieved in such a way that it does not increase the overall asymptotic round complexity of our sorting protocol.[3] If the non-uniform peer-sampling abstraction is provided by other means, then no overlay network is required for the sorting protocol.

In order to solve the sorting problem, one could also follow a different approach that is not gossip-based: One can construct a (perfect) Chord overlay on top of the network, and then implement a sorting network, where each comparator is replaced with a link between two peers in the network. If one uses a Bitonic sorter [5], the comparators correspond to Chord links, and thus no lookups in Chord are necessary. This would yield a sorting algorithm with the same round complexity as ours. (One could even use an AKS [1] sorting network to obtain, with some additional tricks, a round complexity of $O(\log n)$, but AKS networks are considered impractical due to the extremely large constant factors [27].[4]) Most sorting networks, however, provide no inherent fault-tolerance (with the exception of the AKS sorting network). Our gossip-based algorithm is naturally fault-tolerant in the sense that it still works without an increase in the

[3] This requires nodes to sample multiple peers at the beginning of the protocol and leads to a poly-logarithmic increase in the message size complexity. The details can be found in Appendix A.

[4] In our analysis of the sorting algorithm we have not tried to optimize the constant multiplicative factor in front of $\log^2 n$. This analysis gives an upper bound of roughly 100 on this constant, and a more careful analysis yields a bound of roughly 25. We believe, however, that the actual value is much smaller.

asymptotic round complexity, if any two peers fail to establish a communication with constant probability. By repeating comparators (cf. [30]) one can also make sorting networks fault-tolerant, but the repetition of comparators increases the depth of the sorting network (and thus the round complexity in our application) by a factor of $\Omega(\log n)$ in order to allow for a constant failure probability for each communication/comparator. Note also that we only need to use an overlay network to provide the non-uniform peer sampling service, while such an overlay seems inherent for a sorting network based approach.

To bound the message-size complexity of our protocols, we assume that each network address is a W-bit string, where $W = O(\log n)$. If W is super-logarithmic, then the complexity increases by an additive term of $O(W)$.

We present our protocols in terms of synchronous rounds. The synchrony assumption is not really necessary for the definition of the protocols. Instead, nodes may simply follow their own clocks in deciding when to initiate connections. We expect that the running time of our protocols should not be affected, as long as (most) nodes take steps at roughly the same rate, e.g., in the standard asynchronous model where each node takes steps at times determined by a poisson process with a fixed rate for all nodes.

2 Renaming Protocols

2.1 Loose Renaming

We present an algorithm that assigns IDs to n nodes from the integer interval $[1..(1+\epsilon)n]$, for some $\epsilon > 0$; ϵ can be a function of n, but the running time increases linearly with $1/\epsilon$. At any time, each node stores zero or more IDs, and each ID is stored at exactly one node. If node u has one or more IDs at a given time, then one of them is permanently stored by u, and is the ID assigned to u by the algorithm, while the remaining IDs, if any, are u's *free IDs*. The free IDs of a node are *consecutive*, and thus they can be stored using at most $2\log n$ bits. We present two versions on the algorithm: a *pull* algorithm, and a *push* algorithm.

In round 0, a *starting* node sends the ID interval $[1..(1+\epsilon)n]$ to itself.[5] If node u receives interval $[a..b]$ in round $t \geq 0$, and it has not received any IDs prior to that, then ID a is assigned to u. Further, if $a \neq b$ then the interval $[a+1..b]$ of remaining IDs will be the free IDs of u for the next round.

In the *pull* version of the algorithm, in every round $t \geq 1$, each node u that has no free IDs (u may or may not have been assigned an ID yet) sends a request to a random node v. If v has an interval $[a..b]$ of free IDs, then it chooses an arbitrary node u' among the nodes from which it received requests in round t, and sends to u' half of $[a..b]$, precisely, the interval $[\lceil (a+b)/2 \rceil..b]$. If $a \neq b$ then v is left with the interval $[a..\lceil (a+b)/2 \rceil - 1]$ of free IDs, while if $a = b$ (i.e., u had only one free ID) then v has no free IDs in the next round.

[5] The starting node can be chosen randomly via a gossip-based sampling procedure and the network size n can also be estimated via gossip (see, e.g., [23]).

The *push* algorithm is symmetric: In round $t \geq 1$, each node u that has at least one free ID sends half of its interval $[a..b]$ of free IDs, i.e., $[\lceil (a+b)/2 \rceil..b]$, to a randomly chosen node v. If v has no free IDs at the time, then it accepts (an arbitrary) one of the ID intervals it receives in round t, and rejects the remaining ones; if v already has some free IDs then it rejects any ID intervals it receives. If the interval that u sent is rejected, then u keeps the whole interval $[a..b]$ of free IDs, thus no IDs are 'lost'.

From the analysis of the pull protocol presented below, it follows that a node which has been assigned an ID but has no free IDs may as well stop sending requests after the first t_1 rounds, for some $t_1 = \Theta(\log n)$, without affecting the performance guarantees of the protocol. Then, only nodes with no assigned IDs continue to send requests. This offers a natural stopping condition for the protocol. The push algorithm, on the other hand, does not have a natural way to determine when nodes that have free IDs should stop trying to push those IDs. A drawback of pull is that nodes must be notified when the protocol starts so that they can begin to send pull request.

Theorem 1. *The loose renaming protocol described above for distributing a set of $(1+\epsilon)n$ IDs to n nodes guarantees that all nodes acquire IDs after at most $O\big(\frac{(1+\epsilon)n}{\epsilon n+1} \cdot \log n\big)$ rounds with probability $1-n^{-\beta}$ for any $\epsilon \geq 0$ and any fixed $\beta > 0$.*

Proof. We prove the theorem for the pull algorithm. The proof for push is almost the same and is omitted. We start with an overview of the proof. We define a potential function Φ_t, which measures the unbalance in the distribution of free IDs among nodes, and we show that Φ_t drops by a constant factor per round on average, as long as most nodes have 0 or 1 IDs. On the other hand, when most nodes have 2 or more IDs, we observe that the number of nodes with 0 IDs decreases by a constant factor on average per round. We combine these two results to show that w.h.p. in $O(\log n)$ rounds either all nodes have acquired IDs or the free IDs are fairly balanced among nodes. In the latter case we bound the additional number of steps until all nodes obtain IDs, by looking at a single node and bounding the steps until it contacts some node that has free IDs.

Next we give the detailed proof. Let $X_{u,t}$ denote the number of IDs that node u has after round t (including its assigned ID). Let $X_t = \{X_{u,t}\}_u$ be the vector of all $X_{u,t}$ for a given round t. Let $N_t^k = |\{u\colon X_{u,t} = k\}|$ be the number of nodes that have exactly k IDs after round t, and let $N_t^{\geq k} = |\{u\colon X_{u,t} \geq k\}|$ and $N_t^{\leq k} = |\{u\colon X_{u,t} \leq k\}|$.

We define the potential $\Phi_{u,t}$ of node u after round t, as $\Phi_{u,t} = (X_{u,t}-2)^2$ if $X_{u,t} \geq 3$, and $\Phi_{u,t} = 0$ if $X_{u,t} \leq 2$. The (total) potential after round t is then

$$\Phi_t = \sum_u \Phi_{u,t} = \sum_{u\colon X_{u,t} \geq 3} (X_{u,t}-2)^2.$$

The next lemma bounds the expected potential difference in a single round.

Lemma 1. $\mathbf{E}[\Phi_{t+1} \mid X_t] \leq \Phi_t \left(1 - \frac{N_t^{\leq 1}}{4n}\right)$.

Proof. Fix X_t, and let u be a node with $X_{u,t} \geq 3$. Suppose that u receives a request in round $t+1$ to share its $X_{u,t}-1$ free IDs, and thus sends $\lceil (X_{u,t}-1)/2 \rceil$ of them to some node v with $X_{v,t} \in \{0,1\}$. We show that

$$\Phi_{u,t+1} + \Phi_{v,t+1} \leq \Phi_{u,t}/2. \tag{1}$$

We have

$$X_{u,t+1} = X_{u,t} - \lceil (X_{u,t}-1)/2 \rceil = \begin{cases} X_{u,t}/2 + 1/2, & \text{if } X_{u,t} \text{ is odd;} \\ X_{u,t}/2, & \text{if } X_{u,t} \text{ is even; and} \end{cases}$$

$$X_{v,t+1} = X_{v,t} + \lceil (X_{u,t}-1)/2 \rceil \leq 1 + \lceil (X_{u,t}-1)/2 \rceil = \begin{cases} X_{u,t}/2 + 1/2, & \text{if } X_{u,t} \text{ odd;} \\ X_{u,t}/2 + 1, & \text{if } X_{u,t} \text{ even.} \end{cases}$$

It follows that if $X_{u,t}$ is odd (recall also that $X_{u,t} \geq 3$), then

$$\begin{aligned} \Phi_{u,t+1} + \Phi_{v,t+1} &\leq (X_{u,t}/2 + 1/2 - 2)^2 + (X_{u,t}/2 + 1/2 - 2)^2 \\ &= (X_{u,t}-3)^2/2 \leq (X_{u,t}-2)^2/2 = \Phi_{u,t}/2; \end{aligned}$$

and, similarly, if $X_{u,t}$ is even (and thus $X_{u,t} \geq 4$) then

$$\begin{aligned} \Phi_{u,t+1} + \Phi_{v,t+1} &\leq (X_{u,t}/2 - 2)^2 + (X_{u,t}/2 + 1 - 2)^2 \\ &= (X_{u,t}-4)^2/4 + (X_{u,t}-2)^2/4 \leq (X_{u,t}-2)^2/2 = \Phi_{u,t}/2. \end{aligned}$$

Thus, in both cases, Eq. (1) holds. We can now bound the total potential, Φ_{t+1}. From (1), if a node u with $X_{u,t} \geq 3$ shares its free IDs with some node v then $\Phi_{u,t+1} + \Phi_{v,t+1} \leq \Phi_{u,t}/2$, while if u does not share its free IDs then $\Phi_{u,t+1} = \Phi_{u,t}$. Further, all other nodes have zero potential. Therefore, if Y_u is a 0/1 random variable with $Y_u = 1$ iff u shares its free IDs in round $t+1$, we have

$$\Phi_{t+1} \leq \sum_{u\colon X_{u,t} \geq 3} \Big(Y_u \Phi_{u,t}/2 + (1-Y_u)\Phi_{u,t} \Big) = \sum_{u\colon X_{u,t} \geq 3} (1 - Y_u/2)\Phi_{u,t}.$$

Taking the expectation (recall that we have fixed X_t), yields

$$\mathbf{E}[\Phi_{t+1}] \leq \sum_{u\colon X_{u,t} \geq 3} (1 - \mathbf{E}[Y_u]/2)\Phi_{u,t}. \tag{2}$$

Since $\mathbf{E}[Y_u]$ is the probability that u receives a request in round $t+1$ from at least one of the $N_t^{\leq 1}$ nodes v with $X_{v,t} \leq 1$, we have

$$1 - \mathbf{E}[Y_u] = (1-1/n)^{N_t^{\leq 1}} \leq e^{-N_t^{\leq 1}/n} \leq 1 - N_t^{\leq 1}/n + (N_t^{\leq 1}/n)^2/2 \leq 1 - N_t^{\leq 1}/(2n).$$

Thus, $\mathbf{E}[Y_u] \geq N_t^{\leq 1}/(2n)$. Applying this to (2) completes the proof of Lemma 1. □

Next we bound the expected drop in a round of the number N_t^0 of nodes that have no IDs.

Lemma 2. $\mathbf{E}[N^0_{t+1} \mid X_t] \leq N^0_t \left(1 - \frac{N^{\geq 2}_t}{en}\right)$.

Proof. Fix X_t, and suppose that $X_{v,t} = 0$ for some node v. In order to have $X_{v,t+1} > 0$ it suffices that v sends its request in round $t+1$ to some node u with $X_{u,t+1} \geq 2$, and u does not receive a request from any other node. The probability that v sends its request to some u with $X_{u,t+1} \geq 2$ is $N^{\geq 2}_t/n$; and the probability that no node sends a request to the same node as v is

$$(1-1/n)^{N^{\leq 1}_t - 1} \geq (1-1/n)^{n-1} \geq 1/e.$$

Thus, the probability of $X_{v,t+1} > 0$ is at least $N^{\geq 2}_t/(en)$. From the linearity of expectation then we get $\mathbf{E}[N^0_t - N^0_{t+1}] \geq N^0_t N^{\geq 2}_t/(en)$, which proves Lemma 2. □

Consider now the product $Z_t := \Phi_t N^0_t$. From Lemma 1 and the fact that $N^0_{t+1} \leq N^0_t$, it follows

$$\mathbf{E}[Z_{t+1} \mid X_t] \leq N^0_t \cdot \mathbf{E}[\Phi_{t+1} \mid X_t] \leq N^0_t \Phi_t \left(1 - \frac{N^{\leq 1}_t}{4n}\right),$$

and similarly, from Lemma 2 and the fact that $\Phi_{t+1} \leq \Phi_t$,

$$\mathbf{E}[Z_{t+1} \mid X_t] \leq \Phi_t \cdot \mathbf{E}[N^0_{t+1} \mid X_t] \leq \Phi_t N^0_t \left(1 - \frac{N^{\geq 2}_t}{en}\right).$$

Thus,

$$\mathbf{E}[Z_{t+1} \mid X_t] \leq Z_t \left(1 - \max\left\{\frac{N^{\leq 1}_t}{4n}, \frac{N^{\geq 2}_t}{en}\right\}\right),$$

and since $N^{\leq 1}_t + N^{\geq 2}_t = n$, we can easily compute that $\max\left\{\frac{N^{\leq 1}_t}{4n}, \frac{N^{\geq 2}_t}{en}\right\} \geq \frac{1}{e+4} \geq \frac{1}{7}$. Therefore, we have that $\mathbf{E}[Z_{t+1} \mid X_t] \leq (6/7)Z_t$. It follows $\mathbf{E}[Z_t] \leq (6/7)^t Z_0 \leq (6/7)^t n^3$. For

$$t_1 = (\beta+3)\log_{7/6} n + \log_{7/6} 2 = O(\log n), \tag{3}$$

we obtain then that $\mathbf{E}[Z_{t_1}] \leq n^{-\beta}/2$. And by Markov's Inequality, $\Pr(Z_{t_1} > 0) = \Pr(Z_{t_1} \geq 1) \leq n^{-\beta}/2$. Thus, we have that $N^0_{t_1} = 0$ or $\Phi_{t_1} = 0$, with probability at least $1 - n^{-\beta}/2$.

Suppose first that $\epsilon > 1$. Then $\Phi_{t_1} > 0$, for otherwise, no node has more than two IDs after round t_1, which is not possible as there are $(1+\epsilon)n > 2n$ IDs in total. It follows that $N^0_{t_1} = 0$ with probability $1 - n^{-\beta}/2$, and thus all nodes obtain IDs in $t_1 = O(\log n) = O\Big(\frac{(1+\epsilon)n}{\epsilon n+1} \cdot \log n\Big)$ rounds; this proves the theorem.

For the remainder of the proof we assume that $\epsilon \leq 1$. Suppose that $\Phi_{t_1} = 0$. We will compute a t_2 such that $N^0_{t_1+t_2} = 0$ with probability $1 - n^{-\beta}/2$. Since $\Phi_{t_1} = 0$, no node has more that 2 IDs after round t_1. It follows that $N^2_t = \epsilon n + N^0_t$, for all $t \geq t_1$. If $X_{v,t} = 0$ for some node v and round $t \geq t_1$, then the probability of $X_{v,t+1} > 0$ is bounded from below by the probability of the event that in

round $t+1$, v sends a request to one of the $N_t^2 \geq \epsilon n+1$ nodes with free IDs, and this node does not receive any other request. This probability is at least

$$\tfrac{\epsilon n+1}{n} \cdot \left(1-\tfrac{1}{n}\right)^{N_t^{\leq 1}-1} \geq \tfrac{\epsilon n+1}{n} \cdot \left(1-\tfrac{1}{n}\right)^{n-1} \geq \tfrac{\epsilon n+1}{en}.$$

It follows that for any node v, the probability of $X_{v,t_1+k}=0$ is at most $\left(1-\tfrac{\epsilon n+1}{en}\right)^k \leq e^{-k(\epsilon n+1)/(en)}$. For

$$t_2 = ((\beta+1)\ln n+1)en/(\epsilon n+1) = O(n\log n/(\epsilon n+1)),$$

we obtain then that $X_{v,t_1+t_2}=0$ with probability at most $n^{-\beta}/(2n)$. Hence, by the union bound, we have that $X_{v,t_1+t_2}\neq 0$ for *some* v (i.e., $N^0_{t_1+t_2}\neq 0$) with probability at most $n^{-\beta}/2$. This probability is conditional on $\Phi_{t_1}=0$, i.e., formally, $\Pr(N^0_{v,t_1+t_2}\neq 0 \mid \Phi_{t_1}=0)\leq n^{-\beta}/2$. It follows

$$\Pr(N^0_{v,t_1+t_2}\neq 0 \wedge \Phi_{t_1}=0) = \Pr(N^0_{v,t_1+t_2}\neq 0 \mid \Phi_{t_1}=0)\cdot\Pr(\Phi_{t_1}=0)\leq n^{-\beta}/2.$$

And since we showed earlier that $\Pr(\Phi_{t_1}N^0_{t_1}\neq 0)\leq n^{-\beta}/2$, we get

$$\Pr(N^0_{v,t_1+t_2}\neq 0) = \Pr(N^0_{v,t_1+t_2}\neq 0\wedge\Phi_{t_1}=0)+\Pr(N^0_{v,t_1+t_2}\Phi_{t_1}\neq 0)\leq n^{-\beta}. \quad (4)$$

Finally, observing that $t_1+t_2 = O\left(\log n+\tfrac{n\log n}{\epsilon n+1}\right) = O\left(\tfrac{(1+\epsilon)n}{\epsilon n+1}\cdot\log n\right)$, completes the proof of Theorem 1. □

2.2 Tight Renaming

The previous protocol cannot be used to solve efficiently tight renaming, in which the size of the ID space is exactly n: If there are just n IDs, then once there are only few nodes left that have not received an ID, there are also only few nodes that still have a non-empty interval of free IDs. Then it takes a long time, until a node that needs an ID contacts one with a free ID. We solve the tight renaming problem by adding a second phase to the loose renaming algorithm. In this phase, any node that has not been assigned an ID yet, periodically broadcasts (via rumor spreading) "requests" for an ID to the network; the requests contain the network address of the node. When requests of different nodes "meet" at some node, only one of them (the most recent one) survives. Thus, not all requests reach all nodes, but each node receives at least some requests. This approach ensures that message sizes and the information that each node stores is just $O(\log n)$ bits. Nodes that receive requests in this second phase and have free IDs respond by sending to the requesting node some of their free IDs. (They can do so, as the request message contains the address of the requesting node.)

More precisely, in the first phase, nodes run the algorithm described in the previous section for $t_1=\Theta(\log n)$ rounds.[6] In the second phase, a node u that has not acquired an ID yet, generates a request every $O(\log n)$ rounds and sends this

[6] This is the same t_1 as that defined in Eq. (3).

request to itself. The request contains u's network address and an age counter (which increases in each round). Each node keeps only the most recently generated request it has received, choosing arbitrarily among requests with the same age. In every round, each node holding a request sends a copy of it to a randomly chosen node. When a node v that has free IDs receives a request generated by node u, it responds by sending to u directly half of its interval of free IDs (similar to the loose renaming algorithm). Node u accepts the interval if it has not already acquired an ID from some other node that also responded to its requests, or rejects the offer otherwise.

We stress that is not required for different nodes to generate their requests in the same round, or with the same frequency. The only requirement is that each node generates a new request every $O(\log n)$ rounds for as long as it has no IDs.

Theorem 2. *The tight renaming protocol described above for distributing IDs to n nodes guarantees that all nodes acquire IDs after at most $O(\log^2 n)$ rounds with probability $1 - n^{-\beta}$ for any fixed $\beta > 0$.*

Proof. We will use the same notation as in the proof of Theorem 1, namely, $X_{u,t}$, X_t, N_t^k, and Φ_t. Recall that for tight renaming the ID space has size exactly n.

We have shown in the proof of Theorem 1 that with probability at least $1 - n^{-\beta}/2$, we have $N_{t_1}^0 = 0$ or $\Phi_{t_1} = 0$.

Suppose that $\Phi_{t_1} = 0$, and thus no node has more than 2 IDs after round t_1. We will lift this assumption only at the end of the proof. Suppose that node u has no ID yet after round $t \geq t_1$, and it sends a request in round $t+1$. We show that with some constant probability, either u acquires an ID by round $t + \log n$, or the number of nodes with no IDs drops by a constant factor by that time.

Lemma 3. *Let $t \geq t_1$. If a node u with $X_{u,t} = 0$ sends a request in round $t+1$, then with some probability $p = \Omega(1)$ we have $X_{u,t'} \neq 0$ or $N_{t'}^0 \leq N_t^0/2$, for $t' = t + \log(n/N_t^0) + 1$.*

Proof. Fix X_t. Let A_i, for $i \geq 0$, denote the set of nodes that have received u's request and still have it at the end of round $t+i$. Recall that nodes keep only the most recently generated request they have received. Let B_i be the set of nodes which, at the end of round $t+i$, have a request generated after round t by a node other than u. Further, let $a_i = |A_i|$ and $b_i = |B_i|$. Then,

$$a_i \leq 2^{i-1} \quad \text{and} \quad b_i \leq (N_t^0 - 1)2^{i-1}.$$

Next we show for $i = \log(n/N_t^0)$ that $a_i = \Omega(2^i) = \Omega(n/N_t^0)$ with constant probability. Further, we show that if $a_i = \Omega(n/N_t^0)$ and $N_{t+i}^0 \geq N_t^0/2$ and also u has still no ID after round $t+i$, then in the next round u acquires an ID with probability $\Omega(1)$. The claim then follows.

To show the lower bound on a_i, we first bound $\mathbf{E}[a_i]$. Given a_i and b_i, we bound the conditional expectation of a_{i+1} as follows: The expected number of nodes $v \notin A_i \cup B_i$ that receive u's request (and possibly other requests) in round $t+i+1$ is at least $a_i(n - 2a_i - b_i)/n$ (we subtract $2a_i$ instead of a_i to account for collisions). The probability that a given one of these node does not receive

a request pushed by a node in B_i in this round is $(1-1/n)^{b_i} \geq (1-b_i/n)$. Combining these yields

$$\begin{aligned}\mathbf{E}[a_{i+1} \mid a_i, b_i] &\geq (a_i + a_i(n-2a_i-b_i)/n) \cdot (1-b_i/n)\\ &= 2a_i(1-a_i/n-b_i/2n)\cdot(1-b_i/n)\\ &\geq 2a_i(1-a_i/n-3b_i/2n)\\ &\geq 2a_i(1-3N_t^0 2^{i-2}/n),\end{aligned}$$

where for the last relation we used the upper bounds for a_i and b_i we mentioned earlier. Applying the above inequality repeatedly and using that $a_1 = 1$ gives

$$\mathbf{E}[a_i] \geq 2^{i-1}\left(1 - 3N_t^0 \sum_{j=1}^{i-1} 2^{j-2}/n\right) \geq 2^{i-1}\left(1 - 3N_t^0 2^{i-2}/n\right).$$

For $i^* = \log(n/N_t^0)$ we get $\mathbf{E}[a_{i^*}] \geq 2^{i^*-1}(1-3/4) = 2^{i^*-3}$, and by Markov's Inequality,

$$\Pr(a_{i^*} \leq 2^{i^*-4}) = \Pr(2^{i^*-1} - a_{i^*} \geq 2^{i^*-1} - 2^{i^*-4}) \leq \frac{2^{i^*-1} - 2^{i^*-3}}{2^{i^*-1} - 2^{i^*-4}} = 6/7.$$

Next suppose that $a_{i^*} \geq 2^{i^*-4} = n/(2^4 N_t^0)$ and $N_{t+i^*}^0 \geq N_t^0/2$. The conditional probability of $X_{t+i^*+1} \neq 0$ is lower-bounded by the probability that some node from A_{i^*} chooses some node v with free IDs (there are $N_{t+i^*}^2 = N_{t+i^*}^0 \geq N_t^0/2$ such nodes) and at the same time no other node chooses v. Thus, this probability is at least

$$\left(1 - \left(1 - (N_t^0/2)/n\right)^{n/(2^4 N_t^0)}\right)(1-1/n)^{n-1} \geq \left(1 - e^{1/2^5}\right)(1/e) \geq (1/2^6)(1/e).$$

We can now use the above bounds to prove the lemma. Define the events: $\mathcal{X} = (X_{u,t'} \neq 0)$, $\mathcal{N} = (N_{t+i^*}^0 < N_t^0/2)$, and $\mathcal{A} = (a_{i^*} \geq 2^{i^*-4})$. We have shown that $\Pr(\mathcal{A}) \geq 1 - 6/7$, and $\Pr(\mathcal{X} \mid \neg\mathcal{N} \wedge \mathcal{A}) \geq (1/2^6)(1/e)$. The probability we want to lower-bound is

$$\begin{aligned}\Pr(\mathcal{X} \vee \mathcal{N}) &= \Pr(\mathcal{N}) + \Pr(\mathcal{X} \wedge \neg\mathcal{N})\\ &\geq \Pr(\mathcal{N} \wedge \mathcal{A}) + \Pr(\mathcal{X} \wedge \neg\mathcal{N} \wedge \mathcal{A})\\ &= 1 \cdot \Pr(\mathcal{N} \wedge \mathcal{A}) + \Pr(\mathcal{X} \mid \neg\mathcal{N} \wedge \mathcal{A}) \cdot \Pr(\neg\mathcal{N} \wedge \mathcal{A})\\ &\geq \Pr(\mathcal{X} \mid \neg\mathcal{N} \wedge \mathcal{A}) \cdot \left(\Pr(\mathcal{N} \wedge \mathcal{A}) + \Pr(\neg\mathcal{N} \wedge \mathcal{A})\right)\\ &= \Pr(\mathcal{X} \mid \neg\mathcal{N} \wedge \mathcal{A}) \cdot \Pr(\mathcal{A})\\ &\geq (1/2^6)(1/e)(1-6/7).\end{aligned}$$

This completes the proof of Lemma 3. □

We can now finish the proof of the theorem as follows. Assume that $X_{u,t_1} = 0$, and let $r_0 < r_1 < \ldots$ be the rounds after which u is supposed to send requests (if it has not yet an ID by that round, i.e., $X_{u,r_i} = 0$). W.l.o.g. we assume

$r_{i+1} - r_i \geq \log n$, otherwise we can achieve that by omitting some of these times. Further, from the algorithm we have $r_{i+1} - r_i = O(\log n)$.

Define the random variables Y_i, $i \geq 0$, such that $Y_i = N^0_{r_i}$ if $X_{u,r_i} = 0$, and $Y_i = 0$ if $X_{u,r_i} \neq 0$. Note that $Y_i \neq 0$ iff $X_{u,r_i} = 0$. From Lemma 3 it follows

$$\mathbf{E}[Y_{i+1} \mid Y_i] \leq pY_i/2 + (1-p)Y_i = (1 - p/2)Y_i;$$

and thus $\mathbf{E}[Y_i] \leq (1-p/2)^i N^0_{r_0}$. Choosing $i^* = (2/p)\big((\beta+1)\ln n + 1\big)$ gives $\mathbf{E}[Y_{i^*}] \leq n^{-\beta-1}/2$, and from Markov's Inequality, $\Pr(Y_{i^*} \neq 0) = \Pr(Y_{i^*} \geq 1) \leq n^{-\beta-1}/2$. It follows that for

$$t^* = r_{i^*} = t_1 + O(i^* \log n) = t_1 + O(\log^2 n) = O(\log^2 n)$$

we have $\Pr(X_{u,t^*} = 0) \leq n^{-\beta-1}/2$, as we observed earlier that $Y_i \neq 0$ iff $X_{u,r_i} = 0$. From this and the union bound over all u, it follows that $\Pr(N^0_{t^*} \neq 0) \leq n^{-\beta}/2$. Recall that we have assumed $\Phi_{t_1} = 0$. But since $\Pr(\Phi_{t_1} N^0_{t_1} \neq 0) \leq n^{-\beta}/2$ as we saw at the beginning, we can obtain similar to Eq. (4) that the unconditional probability that $N^0_{t^*} \neq 0$ is bounded by $n^{-\beta}$. This completes the proof of Theorem 2. □

3 Sorting Protocol

For the sorting problem we assume that nodes have consecutive IDs, $1, \ldots, n$, and each node has an input value from some totally ordered domain. W.l.o.g. we assume that the input values are numbers, and nodes have distinct inputs. We will say 'node i' to refer to the node with ID i. The goal is to redistribute the values to nodes (one value per node) so that for each i, node i stores the value of rank i, that is, the i-th smallest one among the input values.

In every round of the protocol, each node chooses to be *active* independently with probability 1/2. Each active node i picks a node at random, choosing node j with probability proportional to $1/|i-j|$. If a non-active node j is contacted by one or more active nodes, then it chooses one of them, say node i, and the two nodes compare their values. Let X_i and X_j be the values of i and j respectively, at the time. If $(i-j)(X_i - X_j) < 0$ then the two nodes swap their values; otherwise, the do nothing. If an active node is contacted by another active node, it does not respond to it.

Theorem 3. *The sorting protocol described above sorts the inputs of all n nodes in $O(\log^2 n)$ rounds with probability $1 - n^{-\beta}$ for any fixed $\beta > 0$.*

Proof. The proof uses a potential function argument. For each node i, we consider the distance between i and the node that should have the value stored by node i. We claim that the sum of the squares of these distances drops by a factor of $1 - \Omega(1/\log n)$ in expectation in each round; and thus it becomes zero after $O(\log^2 n)$ rounds.

For each node i, let $X_{i,t}$ be the value that node i has after round t, and let $R_{i,t} = \mathrm{rank}(X_{i,t})$ be the rank of that value. Hence, $R_{i,t}$ is equal to the ID of the

node at which value $X_{i,t}$ should be stored eventually. Further, let $d_{i,t} = |R_{i,t} - i|$ be the distance between nodes $R_{i,t}$ and i. We define the potential $\Psi_{i,t}$ of node i after round t to be $\Psi_{i,t} = d_{i,t}^2$. The (total) potential after round t is then

$$\Psi_t = \sum_i \Psi_{i,t} = \sum_i d_{i,t}^2.$$

Lemma 4. $\mathbf{E}[\Psi_{t+1} \mid \Psi_t] \leq (1 - c/\ln n)\,\Psi_t$, *for some constant* $c > 0$

Proof. The drop in the potential when two nodes i and j swap their values in round $t+1$ is

$$\begin{aligned}\Psi_{i,t} + \Psi_{j,t} - \Psi_{i,t+1} - \Psi_{j,t+1} &= (R_{i,t} - i)^2 + (R_{j,t} - j)^2 - (R_{i,t} - j)^2 - (R_{j,t} - i)^2 \\ &= 2(i-j)(R_{j,t} - R_{i,t}) \\ &= 2|i-j| \cdot |R_{j,t} - R_{i,t}|,\end{aligned}$$

where the last equality holds because $2(i-j)(R_{j,t} - R_{i,t}) > 0$, as nodes i and j swap values only if $(i-j)(X_{i,t} - X_{j,t}) < 0$, and the differences $R_{j,t} - R_{i,t}$ and $X_{j,t} - X_{i,t}$ have the same sign.

Below we assume w.l.o.g. that $i \leq R_{i,t}$. Consider the two sets of nodes $U = [(i + d_{i,t}/3)..n]$ and $W = [1..(R_{i,t} - d_{i,t}/3)]$. The intersection of the two sets has size $|U \cap W| = d_{i,t}/3$. It follows that there are at least $d_{i,t}/3$ nodes $j \in U$ for which $R_{j,t} \in W$. Fix one of these nodes j. If node i is active in round $t+1$, which happens with probability $1/2$, then the probability that i chooses j is $1/(|i-j| \cdot \nu_i)$, where ν_i is the normalizing factor $\sum_{1 \leq k \leq n,\, k \neq i} (1/|i-k|)$, which is in the range $\ln n < \nu_i < 2 \ln n$. Thus, i chooses j with probability $1/(|i-j| \cdot 2\nu_i) \geq 1/(|i-j| \cdot 4 \ln n)$. Further, the probability that node j is not active and not chosen by any other node $k \neq i$ in the round is

$$\frac{1}{2} \prod_{1 \leq k \leq n,\, k \neq i,j} \big(1 - 1/(|i-k| \cdot 2\nu_j)\big) \geq \frac{1}{2}\left(1 - \sum_{1 \leq k \leq n,\, k \neq i,j} 1/(|i-k| \cdot 2\nu_j)\right) \geq 1/4.$$

From all the above it follows that the expected drop in the potential as a result of the likelihood of i choosing j and swapping values with it is at least

$$2|i-j| \cdot |R_{j,t} - R_{i,t}| \cdot \big(1/(|i-j| \cdot 4 \ln n)\big)(1/4) = |R_{j,t} - R_{i,t}|/(8 \ln n).$$

We saw earlier that there are at least $|U \cap W| = d_{i,t}/3$ such nodes j, and for each we have $R_{i,t} - R_{j,t} \geq d_{i,t}/3$ since $R_{j,t} \in W$. It follows that the expected decrease in the potential as a result of i choosing and swapping values with *some* inactive node in round $t+1$ it at least $(d_{i,t}/3)(d_{i,t}/3)/(8 \ln n) = \Psi_{i,t}/(72 \ln n)$. Therefore, the total expected potential difference is $\mathbf{E}[\Psi_t - \Psi_{t+1} \mid \Psi_t] \geq \sum_i \Psi_{i,t}/(72 \ln n) = \Psi_t/(72 \ln n)$. This completes the proof of Lemma 4. □

Applying Lemma 4 repeatedly and using that $\Psi_0 < n^3$, we obtain for $t^* = (\beta+3)(\ln n)^2/c$,

$$\mathbf{E}[\Psi_{t^*}] \leq (1 - c/\ln n)^{t^*} \Psi_0 \leq e^{-ct^*/\ln n} \Psi_0 \leq e^{-(\beta+3)\ln n} n^3 = n^{-\beta}.$$

Markov's Inequality then yields $\Pr(\Psi_{t^*} > 0) = \Pr(\Psi_{t^*} \geq 1) \leq n^{-\beta}/1$, and thus $\Pr(\Psi_{t^*} = 0) \geq 1 - n^{-\beta}$. Since $\Psi_{t^*} = 0$ implies that sorting is completed in at most t^* rounds, Theorem 3 follows. □

4 Conclusion

We presented and analyzed gossip-based protocols for two fundamental tasks, renaming and sorting. The protocols are simple and natural, and they are fault-tolerant in the sense that they still succeed even if a (random) constant fraction of the communication channels fail to get established. For our sorting protocol it is necessary to use *non-uniform* peer-sampling in order to achieve polylogarithmic round complexity. A DHT-like overlay network can be used to implement this service, but we suggest that further research on non-uniform peer-sampling should be pursued.

The probability distribution that we chose for the peer-sampling in our sorting algorithm is the same power law distribution as the one used in Kleinberg's small world graph model [26, 25]. There, the distribution determines additional edges (long range contacts) to augment the ring network, in order to achieve decentralized greedy routing in $O(\log^2 n)$ expected time. It is known that no other distance-based probability distribution for those augmentations can achieve faster greedy routing time [11]. Since sorting is intuitively harder than routing, it seems unlikely that a faster sorting algorithm can be obtained by a change in the probability distribution of the peer-sampling mechanism.

References

1. M. Ajtai, J. Komlós, and E. Szemerédi. An O($n \log n$) sorting network. In *Proc. 15th STOC*, pages 1–9, 1983.
2. D. Alistarh, J. Aspnes, S. Gilbert, and R. Guerraoui. The complexity of renaming. In *Proc. 52nd FOCS*, pages 718–727, 2011.
3. D. Angluin, J. Aspnes, J. Chen, Y. Wu, and Y. Yin. Fast construction of overlay networks. In *Proc. 17th SPAA*, pages 145–154, 2005.
4. H. Attiya, A. Bar-Noy, D. Dolev, D. Peleg, and R. Reischuk. Renaming in an asynchronous environment. *J. ACM*, 37(3):524–548, 1990.
5. K. E. Batcher. Sorting networks and their applications. In *AFIPS Spring Joint Computing Conference*, pages 307–314, 1968.
6. M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani. Estimating Aggregates on a Peer-to-Peer Network. Technical report, Stanford University, 2003.
7. P. Berenbrink, R. Elsässer, and T. Friedetzky. Efficient randomized broadcasting in random regular networks with applications in peer-to-peer systems. In *Proc. 27th PODC*, pages 155–164, 2008.
8. S. P. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508–2530, 2006.
9. C. Cooper, M. E. Dyer, and A. J. Handley. The flip markov chain and a randomising P2P protocol. In *Proc. 28th PODC*, pages 141–150, 2009.

10. A. J. Demers, D. H. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. E. Sturgis, D. C. Swinehart, and D. B. Terry. Epidemic algorithms for replicated database maintenance. In *Proc. 6th PODC*, pages 1–12, 1987.
11. M. Dietzfelbinger and P. Woelfel. Tight lower bounds for greedy routing in uniform small world rings. In *Proc. 41st STOC*, pages 591–600, 2009.
12. B. Doerr and M. Fouz. Asymptotically optimal randomized rumor spreading. In *Proc. 38th ICALP*, pages 502–513, 2011.
13. B. Doerr, M. Fouz, and T. Friedrich. Social networks spread rumors in sublogarithmic time. In *Proc. 43rd STOC*, pages 21–30, 2011.
14. B. Doerr, T. Friedrich, and T. Sauerwald. Quasirandom rumor spreading: Expanders, push vs. pull, and robustness. In *Proc. 36th ICALP*, pages 366–377, 2009.
15. R. Elsässer and T. Sauerwald. On the runtime and robustness of randomized broadcasting. *Theor. Comput. Sci.*, 410(36):3414–3427, 2009.
16. T. Feder, A. Guetz, M. Mihail, and A. Saberi. A local switch Markov chain on given degree graphs with application in connectivity of peer-to-peer networks. In *Proc. 47th FOCS*, pages 69–76, 2006.
17. P. Felber. Epidemic algorithms: A "systems" perspective. *Talk at the Dagstuhl Seminar 13042, Epidemic Algorithms and Processes: From Theory to Practice*, February 2013. (Slides available online at: `http://www.dagstuhl.de/mat/Files/13/13042/13042.FelberPascal.Slides.pdf`.
18. N. Fountoulakis, A. Huber, and K. Panagiotou. Reliable broadcasting in random networks and the effect of density. In *Proc. 29th INFOCOM*, pages 2552–2560, 2010.
19. G. Giakkoupis. Tight bounds for rumor spreading in graphs of a given conductance. In *Proc. 28th STACS*, pages 57–68, 2011.
20. M. Jelasity and A.-M. Kermarrec. Ordered slicing of very large-scale overlay networks. In *Peer-to-Peer Computing*, pages 117–124, 2006.
21. M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. Gossip-based peer sampling. *ACM Trans. Comput. Syst.*, 25(3):8, Aug. 2007.
22. R. M. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking. Randomized rumor spreading. In *Proc. 41st FOCS*, pages 565–574, 2000.
23. D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *Proc. 44th FOCS*, pages 482–491, 2003.
24. D. Kempe, J. M. Kleinberg, and A. J. Demers. Spatial gossip and resource location protocols. *J. ACM*, 51(6):943–967, 2004.
25. J. M. Kleinberg. Navigation in a small-world. *Nature*, page 845, 2000.
26. J. M. Kleinberg. The small-world phenomenon: An algorithm perspective. In *Proc. 32nd STOC*, pages 163–170, 2000.
27. D. E. Knuth. *The Art of Computer Programming, Vol 3, Sorting and Searching*. Addison-Wesley, Reading, USA, 2 edition, 1998.
28. D. Mosk-Aoyama and D. Shah. Fast distributed algorithms for computing separable functions. *IEEE Trans. Inf. Theory*, 54(7):2997–3007, 2008.
29. I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Netw.*, 11(1):17–32, 2003.
30. A. C.-C. Yao and F. F. Yao. On fault-tolerant networks for sorting. *SIAM J. Comput.*, 14(1):120–128, 1985.

APPENDIX

A Peer-Sampling with Chord

We explain now how a peer-sampling service based on Chord can be used with our sorting algorithm without increasing its $O(\log^2 n)$ round complexity, and by just increasing the message-size complexity from $O(\log n)$ to $O(\log^2 n)$. Each node chooses the $\Theta(\log^2 n)$ IDs to use in the sorting algorithm in advance, and sends parallel routing requests via Chord to find the addresses of the nodes with those IDs. Routing in Chord takes $O(\log n)$ steps, however, we now have to route many messages in parallel, and each node can communicate with just one (Chord) neighbor in each round. We employ the following ideas: (1) when a node contacts a neighbor in Chord, it forwards to it *all* the requests that should be routed through that node; and (2) in each round a node chooses which one of its $\log n$ neighbors to call uniformly at random, and with probability $1/2$ it does not call any. A node responds to a call only if it does not itself call a neighbor in that round (if it receives more than one calls it responds to a randomly chosen one). It follows that a given request is forwarded to the next hop after $\Theta(\log n)$ rounds in expectation, and it reaches the destination after $O(\log^2 n)$ rounds both in expectation and w.h.p.

For the message sizes a bound of $O(\log^3 n)$ bits is obtained as follows. From the symmetry of the Chord topology and the fact that all nodes use the same sampling distribution that depends only on the distance, we can show that the total number of requests that must be routed through a given edge is $O(\log^2 n)$ in expectation and also w.h.p.; thus at most $O(\log^3 n)$ bits are sent through that edge. To show the stronger $O(\log^2 n)$ bound we must use a more careful argument, which takes into account the specific sampling distribution used.