When distributed computation is communication expensive

David P. Woodruff · Qin Zhang

Received: 18 November 2013 / Accepted: 26 April 2014 / Published online: 5 May 2014
© Springer-Verlag Berlin Heidelberg 2014

Abstract We consider a number of fundamental statistical and graph problems in the message-passing model, where we have $k$ machines (sites), each holding a piece of data, and the machines want to jointly solve a problem defined on the union of the $k$ data sets. The communication is point-to-point, and the goal is to minimize the total communication among the $k$ machines. This model captures all point-to-point distributed computational models with respect to minimizing communication costs. Our analysis shows that exact computation of many statistical and graph problems in this distributed setting requires a prohibitively large amount of communication, and often one cannot improve upon the communication of the simple protocol in which all machines send their data to a centralized server. Thus, in order to obtain protocols that are communication-efficient, one has to allow approximation, or investigate the distribution or layout of the data sets.

Keywords Multi-party communication complexity · Message-passing · Statistical and graph problems

1 Introduction

Recent years have witnessed a spectacular increase in the amount of data being collected and processed in various applications. In many of these applications, data is often distributed across a group of machines, referred to as sites in this paper, which are connected by a communication network. These sites jointly compute a function defined on the union of the data sets by exchanging messages with each other. For example, consider the following scenarios.

1. We have a set of network routers, each observing a portion of the network, and periodically they want to compute some functions defined on the global network which can be used to determine the overall condition/health of the network. Concrete functions include the number of distinct source IP addresses, the set of most frequent destination IP addresses, etc.

2. The massive social network graphs are usually stored in many sites, and those graphs keep changing. To answer queries such as whether the whole graph is connected, or whether the graph exhibits a particular property (e.g., bipartiteness, cycle-freeness), we have to synthesize data from all the sites.

In distributed computational models for big data, besides traditional measurement like local CPU processing time and the number of disk accesses, we are also interested in minimizing two other objectives, namely, the communication cost and the round complexity. The communication cost, which we shall also refer to as the communication complexity, denotes the total number of bits exchanged in all messages across the sites during a computation. The round complexity refers to the number of communication rounds needed for the computation, given various constraints on what messages can be sent by each site in each round.

The communication cost is a fundamental measure since communication is often the bottleneck of applications (e.g., applications mentioned above), and so it directly relates to
energy consumption, network bandwidth usage, and overall running time. The round complexity is critical when the computation is partitioned into rounds and the initialization of each round requires a large overhead. In this paper we will focus on the communication complexity, and analyze problems in an abstract model called the message-passing model (see the definition in Sect. 1.1) that captures all models for point-to-point distributed computation in terms of their communication costs. In particular, our lower bound results hold even if the communication protocol sends only a single bit in each message, and each site has an unbounded amount of local memory and computational power. Note that this means our lower bounds are as strong as possible, not requiring any assumptions on the local computational power of the machines. We also present several upper bounds, all of which are also locally computationally efficient, and the protocols we present do not need extra memory beyond what is required to accommodate the input. We will briefly discuss the issue of round-efficiency in Sect. 7.

Common sources of massive data include numerical data, e.g., IP streams and logs of queries to a search engine, as well as graph data, e.g., web graphs, social networks, and citation graphs. In this paper we investigate the communication costs for solving several basic statistical and graph problems in the message-passing model. Solving these problems is a minimal requirement of protocols seeking to solve more complicated functions on distributed data.

We show that if we want to solve many of these problems exactly, then there are no better solutions than the almost trivial ones, which are usually quite communication-inefficient. The motivation of this work is thus to deliver the following message to people working on designing protocols for solving problems on distributed systems: for many statistical and graph problems in the distributed setting, if we want efficient communication protocols, then we need to consider the following relaxations to the original problem:

1. Allow for returning an approximate solution. Here, approximation can be defined as follows: for a problem whose output is a single numerical value $x$, allowing an approximation means that the protocol is allowed to return any value $\tilde{x}$ for which $\tilde{x} \in [(1-\varepsilon)x, (1+\varepsilon)x]$, for some small user-specified parameter $\varepsilon > 0$. For a problem whose output is YES or NO, e.g., a problem deciding if a certain property of the input exists or not, we could instead allow the protocol to return YES if the input is close to having the property (under some problem-specific notion of closeness) and NO if the input is far from having that property. For example, in the graph connectivity problem, we return YES if the graph can be made connected by adding a small number of edges, while we return NO if the graph requires adding a large number of edges to be made connected. This latter notion of approximation coincides with the property testing paradigm [16] in the computer science literature.

2. Use well-designed input layouts. Here are two examples:
   (1) All edges from the same node are stored in the same site or on only a few sites. In our lower bounds the edges adjacent to a node are typically stored across many different sites. (2) Each edge is stored on a unique (or a small number) of different sites. Our results in Table 1 show that whether or not the input graph has edges that occur on multiple sites can make a huge difference in the communication costs.

3. Explore prior distributional properties of the input dataset, e.g., if the dataset is skewed, or the underlying graph is sparse or follows a power-law distribution. Instead of developing algorithms targeting the worst-case distributions, as those used in our lower bounds, if one is fortunate enough to have a reasonable model of the underlying distribution of inputs, this can considerably reduce communication costs. An extreme example is that of a graph on $n$ vertices—if the graph is completely random, that is, each pair of nodes are connected by an edge independently with probability $p$, then the $k$ sites can simply compute the total number of edges $m$ to decide whether or not the input graph is connected with high probability. Indeed, by results in random graph theory, if $m \geq 0.51n \log n$ then the graph is connected with very high probability, while if $m \leq 0.49n \log n$ then the graph is disconnected with very high probability [12].

1.1 The message-passing model

In this paper we consider the message-passing model, studied, for example, in [26,30]. In this model we have $k$ sites, e.g., machines, sensors, database servers, etc., which we denote as $P_1, \ldots, P_k$. Each site has some portion of the overall data set, and the sites would like to compute a function defined on the union of the $k$ data sets by exchanging messages. There is a two-way communication channel between all pairs of sites $P_i$ and $P_j$. Note that our lower bounds also hold for topologies in which each site can only talk to a subset of other sites. The communication is point-to-point, that is, if $P_i$ talks to $P_j$, then the other $k-2$ sites do not see the messages exchanged between $P_i$ and $P_j$. At the end of the computation, at least one of the sites should report the correct answer. The goal is to minimize the total number of bits exchanged among the $k$ sites. The computation can be randomized: Each site is allowed to use infinite amount of private random bits, but no public random bits can be used. For the purposes of proving impossibility results, i.e., lower bounds,
Table 1 All results are in terms of number of bits of communication

| Problem          | With duplication | Without duplication |
|------------------|------------------|--------------------|
|                  | LB               | UB                 | LB               | UB               |
| \( F_0 \)        | \( \tilde{\Omega}(kF_0) \) | \( \tilde{O}(k(F_0 + \log n)) \) | – | – |
| \( \ell_\infty \) | \( \tilde{\Omega}(\min(k, \ell_\infty)n) \) | \( \tilde{O}(\min(k, \ell_\infty)n) \) | – | – |
| Degree           | \( \tilde{\Omega}(kd_i) \) | \( O(kd_i \log n) \) | \( \tilde{\Omega}(k) \) | \( O(k \log n) \) |
| Cycle-free       | \( \tilde{\Omega}(kn) \) | \( \tilde{O}(kn) \) | \( \Omega(\min(n, m)) \) | \( \tilde{O}(\min[n, m]) \) |
| Connectivity     | \( \tilde{\Omega}(kn) \) | \( \tilde{O}(kn) \) | \( \tilde{\Omega}(kr) \) | \( \tilde{O}(kr) \) |
| #CC              | \( \tilde{\Omega}(kn) \) | \( \tilde{O}(kn) \) | \( \tilde{\Omega}(kr) \) | \( \tilde{O}(kr) \) |
| Bipartite        | \( \tilde{\Omega}(kn) \) | \( \tilde{O}(kn) \) | \( \tilde{\Omega}(kr) \) | \( \tilde{O}(kr) \) |
| Triangle-free    | \( \tilde{\Omega}(km) \) | \( \tilde{O}(km) \) | \( \Omega(m) \) | \( \tilde{O}(m) \) |

Our lower bounds hold for randomized protocols which succeed with at least a constant probability of 2/3, while all of our upper bounds are deterministic protocols (which always succeed). \( k \) refers to the number of sites, with a typical value ranging from 100 to 10,000 in practice. For \( F_0 \) and \( \ell_\infty \), \( n \) denotes the size of the element universe. For graph problems, \( n \) denotes the number of vertices and \( m \) denotes the number of edges. \( d_i \) is the degree of the queried vertex \( v \). We make the mild assumption that \( \Omega(\log n) \leq k \leq \min(n, m) \). Let \( r = \min[n, m/k] \). Except for the upper bound for cycle-free in the “without duplication” case, for which \( m \geq n \) implies that a cycle necessarily exists (and therefore makes the problem statement vacuous), we assume that \( m \geq n \) in order to avoid a messy and uninteresting case-by-case analysis.

we can allow each site to have an infinite local memory and infinite computational power; note that such an assumption will only make our lower bounds stronger. Further, we do not place any constraints on the format of messages or any ordering requirement on the communication, as long as it is point-to-point.

The message-passing model captures all point-to-point distributed communication models in terms of the communication cost, including the BSP model by Valiant [28], the MRC MapReduce framework proposed by Karloff et al. [21], the generic MapReduce model by Goodrich et al. [17], and the Massively Parallel model by Koutris and Suciu [22].

**Remark 1** We comment that in some settings, where the primary goal is to parallelize a single computation on a big dataset, communication may not be the only bottleneck; CPU time and disk accesses are also important. However, in this paper we are mainly interested in the following setting: The data has already been distributed to the sites, and periodically we would like to compute some function that is defined on the dataset (e.g., queries). In this setting, communication is the most expensive operation, since it directly connects to network bandwidth usage and energy consumption.

1.2 Our results

We investigate lower bounds (impossibility results) and upper bounds (protocols) of the exact computation of the following basic statistical and graph problems in the message-passing model.

1. Statistical problems: computing the number of distinct elements, known as \( F_0 \) in the database literature; and finding the element with the maximum frequency, known as the \( \ell_\infty \) or iceberg query problem. We note that the lower bound for \( \ell_\infty \) also applies to the heavy-hitter problem of finding all elements whose frequencies exceed a certain threshold, as well as many other statistical problems for which we have to compute the elements with the maximum frequency exactly.

2. Graph problems: computing the degree of a vertex; testing cycle-freeness; testing connectivity; computing the number of connected components (#CC); testing bipartiteness; and testing triangles-freeness.

For each graph problem, we study its lower bound and upper bound in two cases: with edge duplication among the different sites and without edge duplication. Our results are summarized in Table 1. Note that all lower bounds are matched by upper bounds up to some logarithmic factors. For convenience, we use \( \tilde{\Omega}(f) \) and \( \tilde{O}(f) \) to denote functions of forms \( f/\log^{O(1)}(f) \) and \( f \cdot \log^{O(1)}(f) \), respectively. That is, we hide logarithmic factors.

We prove most of our lower bound results via reductions from a meta-problem that we call \( \text{THRESH}_\theta \). Its definition is given in Sect. 4.

In Sect. 6.6 we discuss the diameter problem, i.e., the problem of computing the distance of the farthest pair of vertices in a graph. This is a problem that we cannot completely characterize by the technique proposed in this paper. We further show that by allowing an error as small as an additive-2, we can reduce the communication cost of computing the diameter by roughly a \( \sqrt{n} \) factor, compared with the naive algorithm for exact computation. This further supports our claim that even a very slight approximation can result in a dramatic savings in communication.
1.3 Related work

For statistical problems, a number of approximation algorithms have been proposed recently in the distributed streaming model, which can be thought of as a dynamic version of the one-shot distributed computation model considered in this paper: the $k$ local inputs arrive in the streaming fashion and one of the sites has to continuously monitor a function defined on the union of the $k$ local inputs. All protocols in the distributed streaming model are also valid protocols in our one-shot computational model, while our impossibility results in our one-shot computational model also apply to all protocols in the distributed streaming model. Example functions studied in the distributed streaming model include $F_0$ [8], $F_2$ (size of self join) [8,30], quantile and heavy-hitters [19], and the empirical entropy [3]. All of these problems have much lower communication cost if one allows an approximation of the output number $x$ in a range $[(1-\varepsilon)x, (1+\varepsilon)x]$, as mentioned above (the definition as to what $\varepsilon$ is for the various problems differs). These works show that if an approximation is allowed, then all these problems can be solved using only $\tilde{O}(k/\varepsilon^{O(1)})$ bits of communication. A suite of (almost) matching lower bounds for approximate computations was developed in [29,30]. For exact $F_0$ computation, the best previous communication cost lower bound was $\Omega(F_0 + k)$ bits (the $F_0$ term is simply by a reduction from the 2-party set-disjointness problem, and the $k$ term is because every site needs to send at least 1 bit). In this paper we improve the communication cost lower bound to $\tilde{O}(kF_0)$, which is optimal up to a small logarithmic factor.

For graph problems, Ahn, Guha and McGregor [1,2] developed an elegant technique for *sketching* graphs, and showed its applicability to many graph problems including connectivity, bipartiteness, and minimum spanning tree. Each sketching step in these algorithms can be implemented in the message-passing model as follows: each site computes a sketch of its local graph and sends its sketch to $P_1$. The site $P_1$ then combines these $k$ sketches into a sketch of the global graph. The final answer can be obtained based on the global sketch that $P_1$ computes. Most sketches in [1,2] are of size $\tilde{O}(n^{1+\gamma})$ bits (for a small constant $\gamma \geq 0$), and the number of sketching steps varies from 1 to a constant. Thus direct implementations of these algorithms in the message-passing model have communication $\tilde{O}(k \cdot n^{1+\gamma})$ bits. On the lower bound side, it seems not much is known. Phillips et al. [26] proved an $\Omega(kn/\log^2 k)$ bits lower bound for connectivity. Their lower bound proof relies on a well-crafted graph distribution. In this paper we improve their lower bound by a factor of $\log k$. Another difference is that their proof requires the input to have edge duplications, while our lower bound holds even if there are no edge duplications, showing that the problem is hard even if each edge occurs on a single site. Very recently in an unpublished manuscript, Huang et. al. [18] showed that $\Omega(kn)$ bits of communication is necessary in order to even compute a constant factor approximation to the size of the maximum matching of a graph. Their result, however, requires that the entire matching has to be reported, and it is unknown if a similar lower bound applies if one is only interested in estimating the matching size.

Besides statistical and graph problems, Koutris and Suciu [22] studied evaluating conjunctive queries in their massively parallel model. Their lower bounds are restricted to one round of communication, and the message format has to be tuple-based, etc. Some of these assumptions are removed in a recent work by Beame et al. [5]. We stress that in our message-passing model there is no such restriction on the number of rounds and the message format; our lower bounds apply to arbitrary communication protocols. Recently, Daumé III et al. [9,10] and Balcan et al. [4] studied several problems in the setting of distributed learning, in the message-passing model.

1.4 Conventions

Let $[n] = \{1, \ldots, n\}$. All logarithms are base-2. All communication complexities are in terms of bits. We typically use capital letters $X, Y, \ldots$ for sets or random variables, and lower case letters $x, y, \ldots$ for specific values of the random variables $X, Y, \ldots$. We write $X \sim \mu$ to denote a random variable chosen from distribution $\mu$. For convenience we often identify a set $X \subseteq [n]$ with its characteristic vector when there is no ambiguity, i.e., the bit vector which is 1 in the $i$-th bit if and only if element $i$ occurs in the set $X$.

All our upper bound protocols are either deterministic or only use private randomness.

We make a mild assumption that $\Omega(\log n) \leq k \leq \min(n, m)$, where for $F_0$ and $\ell_\infty$, $n$ denotes the size of the element universe; and for graph problems, $n$ denotes the number of vertices and $m$ denotes the number of edges.

1.5 Roadmap

In Sect. 2, we give a case study on the number of distinct elements ($F_0$) problem. In Sect. 3, we include background on communication complexity which is needed for understanding the rest of the paper. In Sect. 4, we introduce the meta-problem $\text{THRESH}_{F_0}$ and study its communication complexity. In Sects. 5 and 6, we show how to prove lower bounds for a set of statistical and graph problems by performing reductions from $\text{THRESH}_{F_0}$. We conclude the paper in Sect. 7.

2 The number of distinct elements: a case study

In this section we give a brief case study on the number of distinct elements ($F_0$) problem, with the purpose of justifying
the statement that approximation is often needed in order to obtain communication-efficient protocols in the distributed setting.

The $F_0$ problem requires computing the number of distinct elements of a data set. It has numerous applications in network traffic monitoring [13], data mining in graph databases [25], data integration [7], etc., and has been extensively studied in the last three decades, mainly in the data stream model. It began with the work of Flajolet and Martin [15] and culminated in an optimal algorithm by Kane et al. [20]. In the streaming setting, we see a stream of elements coming one at a time and the goal is to compute the number of distinct elements in the stream using as little memory as possible. In [14], Flajolet et al. reported that their HyperLogLog algorithm can estimate cardinalities beyond $10^9$ using a memory of only 1.5 KB, and achieve a relative accuracy of 2%, compared with the $10^9$ bytes of memory required if we want to compute $F_0$ exactly.

Similar situations happen in the distributed communication setting, where we have $k$ sites, each holding a set of elements from the universe $[n]$, and the sites want to compute the number of distinct elements of the union of their $k$ data sets. In [8], a $(1+\varepsilon)$-approximation algorithm (protocol) with $O(k\log n + 1/\varepsilon^2 \log 1/\varepsilon)$ bits of communication was given in the distributed streaming model, which is also a protocol in the message-passing model. In a typical setting, we could have $\varepsilon = 0.01$, $n = 10^9$ and $k = 1,000$, in which case the communication cost is about $6.6 \times 10^7$ bits.\(^1\) On the other hand, our result shows that if exact computation is required, then the communication cost among the $k$ sites needs to be at least $\Omega(kF_0/\log k)$ (see Corollary 1), which is already $10^9$ bits even when $F_0 = n/100$.

3 Preliminaries

In this section we introduce some background on communication complexity. We refer the reader to the book by Kushilevitz and Nisan [23] for a more complete treatment.

In the basic two-party communication complexity model, we have two parties, which we denote by Alice and Bob. Alice has an input $x$ and Bob has an input $y$, and they want to jointly compute a function $f(x, y)$ by communicating with each other according to a protocol $\Pi$. Let $\Pi(x, y)$ be the transcript of the protocol, that is, the concatenation of the sequence of messages exchanged by Alice and Bob, given the inputs $x$ and $y$. In this paper when there is no confusion, we abuse notation by using $\Pi$ for both a protocol and its transcript, and we further abbreviate the transcript $\Pi(x, y)$ by $\Pi$.

\(^1\)In the comparison we neglect the constants hidden in the big-$O$ and big-$\Omega$ notation which should be small.

The deterministic communication complexity of a deterministic protocol is defined to be

$$\max\{|\Pi(x, y)| \mid \text{all possible inputs } (x, y)\},$$

where $|\Pi(x, y)|$ is the number of bits in the transcript of the protocol $\Pi$ on inputs $x$ and $y$. The randomized communication complexity of a randomized protocol $\Pi$ is the maximum number of bits in the transcript of the protocol over all possible inputs $x$, $y$, together with all possible random bits used by the players. We say a randomized protocol $\Pi$ computes a function $f$ correctly with error probability $\delta$ if for all input pairs $(x, y)$, it holds that $\Pr[\Pi(x, y) \neq f(x, y)] \leq \delta$, where the probability is taken only over the random bits of the players. The randomized $\delta$-error communication complexity of a function $f$, denoted by $R^\delta(f)$, is the minimum communication complexity of a protocol that computes $f$ with error probability at most $\delta$.

Let $\mu$ be a distribution over the input domain, and let $(X, Y) \sim \mu$. For a deterministic protocol $\Pi$, we say that $\Pi$ computes $f$ with error probability $\delta$ on $\mu$ if $\Pr[\Pi(X, Y) \neq f(X, Y)] \leq \delta$, where the probability is over the choices of $(X, Y) \sim \mu$. The $\delta$-error $\mu$-distributional communication complexity of $f$, denoted by $D^\delta_\mu(f)$, is the minimum worst-case communication complexity of a deterministic protocol that gives the correct answer for $f$ on at least $(1-\delta)$ fraction of all inputs (weighted by $\mu$). We denote $ED^\delta_\mu(f)$ to be the $\delta$-error $\mu$-distributional expected communication complexity, which we define to be the minimum expected cost (rather than the worst-case cost) of a deterministic protocol that gives the correct answer for $f$ on at least $(1-\delta)$ fraction of all inputs (weighted by $\mu$), where the expectation is taken over distribution $\mu$.

We can generalize the two-party communication complexity to the multi-party setting, which is the message-passing model considered in this paper. Here we have $k$ players $P_1, \ldots, P_k$ with $P_j$ having the input $x_j$, and the players want to compute a function $f(x_1, \ldots, x_k)$ of their joint inputs by exchanging messages with each other. The transcript of a protocol always specifies which player speaks next.

At the end of the communication, only one player needs to output the answer.

The following lemma shows that randomized communication complexity is lower bounded by distributional communication complexity under any distribution $\mu$. We include a proof here since the original proof is for the two-party communication setting.

Lemma 1 (Yao’s Lemma [31]) For any function $f$ and any $\delta > 0$, $R^\delta(f) \geq \max_{\mu} D^\delta_\mu(f)$.

Proof The original proof is for two players, though this also holds for $k > 2$ players since for any distribution $\mu$, if $\Pi$ is a $\delta$-error protocol then for all possible inputs $x_1, \ldots, x_k$ to the $k$ players, let $rand$ be the random bits of the players,

$$\Pr_{rand}[\Pi(x_1, \ldots, x_k) = f(x_1, \ldots, x_k)] \geq 1 - \delta.$$
which implies for any distribution $\mu$ on $(x^1, \ldots, x^k)$ that
\[
\Pr_{x^1,\ldots,x^k \sim \mu}[\prod_{i=1}^k (x_i^1, \ldots, x_i^k) = f(x^1, \ldots, x^k)] \geq 1 - \delta,
\]
which implies there is a fixing of the random bits of the players so that
\[
\Pr_{x^1,\ldots,x^k \sim \mu}[\prod_{i=1}^k (x_i^1, \ldots, x_i^k) = f(x^1, \ldots, x^k)] \geq 1 - \delta,
\]
which implies $D^\delta_\mu(f)$ is at most $R^\delta(f)$.

Therefore, one way to prove a lower bound on the randomized communication complexity of $f$ is to first pick a (hard) input distribution $\mu$ for $f$, and then study its distributional communication complexity under $\mu$.

Note that given a 1/3-error randomized protocol for a problem $f$ whose output is 0 or 1, we can always run the protocol $C \log(1/\delta)$ times using independent randomness each time, and then output the majority of the outcomes. By a standard Chernoff bound (see below), the output will be correct with error probability at most $e^{-\kappa C \log(1/\delta)}$ for an absolute constant $\kappa$, which is at most $\delta$ if we choose $C$ to be a sufficiently large constant. Therefore $R^{1/3}(f) = \Omega(R^\delta(f)/\log(1/\delta)) = \Omega(\max_\mu D^\delta_\mu(f)/\log(1/\delta))$ for any $\delta \in (0, 1/3]$. Consequently, to prove a lower bound on $R^{1/3}(f)$ we only need to prove a lower bound on the distributional communication complexity of $f$ with an error probability $\delta \leq 1/3$.

Chernoff bound (c.f. [24]). Let $X_1, \ldots, X_n$ be independent Bernoulli random variables such that $\Pr[X_i = 1] = p_i$. Let $X = \sum_{i \in [n]} X_i$. Let $\mu = \mathbb{E}[X]$. It holds that $\Pr[X \geq (1 + \delta)\mu] \leq e^{-\delta^2\mu/3}$ and $\Pr[X \leq (1 - \delta)\mu] \leq e^{-\delta^2\mu/2}$ for any $\delta \in (0, 1)$.

4 A meta-problem

In this section we discuss a meta-problem $\text{THRESH}^\theta_\delta$ and derive a communication lower bound for it. This meta-problem will be used to derive lower bounds for statistical and graph problems in our applications.

The organization of this section is as follows: We first define the $\text{THRESH}^\theta_\delta$ problem, and then discuss a model called the coordinator model which is basically the same as the message-passing model but is more convenient to work with for proving the lower bound for $\text{THRESH}^\theta_\delta$. Next, we show a slightly improved theorem which was first introduced in [26], and will be used in our lower bound proof for $\text{THRESH}^\theta_\delta$. After that, we introduce two auxiliary problems which will be used to prove the lower bound for $\text{THRESH}^\theta_\delta$ at the end.

The $\text{THRESH}^\theta_\delta$ Problem. In this problem, each site $P_i$ ($i \in [k]$) holds an $r$-bit vector $x_i = \{x_{i,1}, \ldots, x_{i,r}\}$, and the $k$ sites want to compute

\[
\text{THRESH}^\theta_\delta(x_1, \ldots, x_k) = \begin{cases}
0, & \text{if } \sum_{i \in [r]}(\forall j \in [k] \Pi_{i,j}) \leq \theta, \\
1, & \text{if } \sum_{i \in [r]}(\forall j \in [k] \Pi_{i,j}) \geq \theta + 1.
\end{cases}
\]

That is, if we think of the input as a $k \times r$ matrix with $x_1, \ldots, x_k$ as the rows, then in $\text{THRESH}^\theta_\delta$ we want to find out whether the number of columns that contain a 1 is more than $\theta$ for a threshold parameter $\theta$.

The coordinator model. In this model we have an additional site called the coordinator,\(^2\) which has no input (formally, his input is the empty set). We require that the $k$ sites can only talk to the coordinator. The message-passing model can be simulated by the coordinator model since every time a site $P_i$ wants to talk to $P_j$, it can first send the message to the coordinator, and then the coordinator can forward the message to $P_j$. Such a re-routing only increases the communication complexity by a factor of 2 and thus will not affect the asymptotic communication complexity.

Let $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ be an arbitrary function. Let $\mu$ be a probability distribution over $\mathcal{X} \times \mathcal{Y}$. Let $f_{\text{OR}} : \mathcal{X}^k \times \mathcal{Y} \rightarrow \{0, 1\}$ be the function of computing $f(x_1, y) \lor f(x_2, y) \lor \ldots \lor f(x_k, y)$ in the coordinator model, where $P_i$ has input $x_i \in \mathcal{X}$ for each $i \in [k]$, and the coordinator has $y \in \mathcal{Y}$. Given the distribution $\mu$ on $\mathcal{X} \times \mathcal{Y}$, we construct a corresponding distribution $\nu$ on $\mathcal{X}^k \times \mathcal{Y}$: We first pick $(X_1, Y) \sim \mu$, and then pick $X_2, \ldots, X_k$ from the conditional distribution $\mu \mid Y$.

4.1 A meta-theorem

The following theorem was originally proposed in [26]. Here we improve it by a log $k$ factor by a slightly modified analysis, which we include here for completeness.

**Theorem 1** For any function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ and any distribution $\mu$ on $\mathcal{X} \times \mathcal{Y}$ for which $\mu(f^{-1}(1)) \leq 1/k^2$, we have $D^{1/k^3}(f_{\text{OR}}) = \Omega(k \cdot \mathbb{E}_{\mu}[1/100k^2])$.

**Proof** Suppose Alice has $X$ and Bob has $Y$ with $(X, Y) \sim \mu$, and they want to compute $f(X, Y)$. They can use a protocol $P$ for $f_{\text{OR}}$ to compute $f(X, Y)$ as follows. The first step is an input reduction. Alice and Bob first pick a random $I \in [k]$ using shared randomness, which will later be fixed by the protocol to make it deterministic. Alice simulates $P_I$ by assigning it an input $X_I = X$. Bob simulates the coordinator and the remaining $k - 1$ sites. He first assigns $Y$ to the coordinator, and then samples $X_1, \ldots, X_{I-1}, X_{I+1}, \ldots, X_k$ independently according to the conditional distribution $\mu \mid Y$, and assigns $X_i$ to $P_i$ for each $i \in [k] \setminus I$. Now $(X_1, \ldots, X_k, Y) \sim \nu$.\(^2\)

\(^2\)We can also choose, for example, $P_I$ to be the coordinator and avoid the need for an additional site, though having an additional site makes the notation cleaner.
Since \( \mu(f^{-1}(1)) \leq 1/k^2 \), with probability \((1 - 1/k^2)^{k-1} \geq 1 - 1/k\), we have \( f(X_i, Y) = 0 \) for all \( i \in [k] \setminus I \). Consequently,

\[
f^k_{OR}(X_1, \ldots, X_k, Y) = f(X, Y).
\] (1)

We say such an input reduction is good.

Alice and Bob construct a protocol \( \mathcal{P}' \) for \( f \) by independently repeating the input reduction three times, and running \( \mathcal{P} \) on each input reduction. The probability that at least one of the three input reductions is good is at least \( 1 - 1/k^3 \), and Bob can learn which reduction is good without any communication. This is because in the simulation he locally generates all \( X_i \ (i \in [k] \setminus I) \) together with \( Y \). On the other hand, by a union bound, the probability that \( \mathcal{P} \) is correct for all three input reductions is at least \( 1 - 3/k^3 \). Note that if we can compute \( f^k_{OR}(X_1, \ldots, X_k, Y) \) correctly for a good input reduction, then by (1), \( \mathcal{P} \) can also be used to correctly compute \( f(X, Y) \). Therefore \( \mathcal{P} \) can be used to compute \( f(X, Y) \) with probability at least \( 1 - 3/k^3 - 1/k^3 \geq 1 - 1/(100k^2) \).

Since in each input reduction, \( X_1, \ldots, X_k \) are independent and identically distributed, and since \( I \in [k] \) is chosen randomly in the two input reductions, we have that in expectation over the choice of \( I \), the communication between \( P_l \) and the coordinator is at most a \( 2/k \) fraction of the expected total communication of \( \mathcal{P} \) given inputs drawn from \( v \). By linearity of expectation, if the expected communication cost of \( \mathcal{P} \) for solving \( f^k_{OR} \) under input distribution \( v \) with error probability at most \( 1/k^3 \) is \( C \), then the expected communication cost of \( \mathcal{P}' \) for solving \( f \) under input distribution \( \mu \) with error probability at most \( 1/(100k^2) \) is \( O(C/k) \). Finally, by averaging there exists a fixed choice of \( I \in [k] \), so that \( \mathcal{P}' \) is deterministic and for which the expected communication cost of \( \mathcal{P}' \) for solving \( f \) under input distribution \( \mu \) with error probability at most \( 1/(100k^2) \) is \( O(C/k) \). Therefore we have \( D_v^{1/k^3}(f^k_{OR}) = \Omega(k \cdot \mathbb{E}D_\mu^{1/(100k^2)}(f)) \). \( \Box \)

Remark 2 In Theorem 1, the error parameter is inversely polynomial in \( k \), which will cause a loss of a \( \log k \) factor in its applications compared with a constant error. This is inherent if we would like to create a totally symmetric input distribution for \( X_1, \ldots, X_k \). However, in a follow-up work [29], by using an asymmetric distribution we managed to avoid such a loss. We refer readers to Section 4.5 in [29] for details.

4.2 The 2-DISJ’ problem

Now we choose a concrete function \( f \) to be the set-disjointness problem. In this problem we have two parties: Alice has \( x \subseteq [r] \) while Bob has \( y \subseteq [r] \), and the parties want to compute

\[
\text{2-DISJ}'(x, y) = \begin{cases} 1, & \text{if } x \cap y \neq \emptyset, \\ 0, & \text{otherwise}. \end{cases}
\]

Set-disjointness is a classical problem used in proving communication lower bounds. We define an input distribution \( \tau_\beta \) for 2-DISJ’ as follows. Let \( \ell = (r + 1)/4 \). With probability \( \beta \), \( x \) and \( y \) are random subsets of \([r]\) such that \( |x| = |y| = \ell \) and \( |x \cap y| = 1 \). If we choose \( \beta = 1/4 \), \( D_v^{(1/4)100}(2-DISJ') = \Omega(r) \), and one can extend his arguments to any \( \beta \in (0, 1/4) \), and to the expected distributional communication complexity where the expectation is taken over the input distribution.

Theorem 2 ([26], Lemma 2.2) For any \( \beta \in (0, 1/4] \), \( \mathbb{E}D_\tau^{\beta/100}(2-DISJ') = \Omega(r) \), where the expectation is taken over the input distribution.

4.3 The OR-DISJ’ problem

If we choose \( f \) to be 2-DISJ’ and let \( \mu = \tau_\beta \) with \( \beta = 1/k^2 \), then we call \( f^k_{OR} \) in the coordinator model the OR-DISJ’ Problem. By Theorems 1 and 2, we have

Theorem 3 \( D_v^{1/k^3}(OR-DISJ') = \Omega(kr) \).

4.3.1 The complexity of THRESH’_\theta

We prove our lower bound for the setting of the parameter \( \theta = (3r - 1)/4 \). We define the following input distribution \( \zeta \) for \( THRESH'_{(3r-1)/4} \): We choose \( \{X_1, \ldots, X_k, Y\} \sim v \) where \( v \) is the input distribution for OR-DISJ’, and then simply use \( \{X_1, \ldots, X_k\} \) as the input for \( THRESH'_{\zeta} \).

Lemma 2 Under the distribution \( \zeta \), assuming \( k \geq c_k \log r \) for a large enough constant \( c_k \), we have that \( \bigvee_{i \in [k]} X_{i,j} = 1 \) for all \( j \in [r] \setminus Y \) with probability \( 1 - 1/k^{10} \).

Proof For each \( j \in [r] \setminus Y \), we have \( \bigvee_{i \in [k]} X_{i,j} = 1 \) with probability at least \( 1 - (1 - 1/4)^k \). This is because \( \Pr[X_{i,j} = 1] \geq 1/4 \) for each \( j \in [r] \setminus Y \), by our choices of \( X_j \). By a union bound and our assumption \( c_k \log r \leq k \leq r \) for a large enough constant \( c_k \), the probability that \( \bigvee_{i \in [k]} X_{i,j} = 1 \) for all \( j \in [r] \setminus Y \) is at least

\[
\left(1 - (3/4)^k \cdot |Y|\right) = \left(1 - (3/4)^k \cdot (3r - 1)/4\right) \geq 1 - 1/k^{10}.
\] \( \Box \)

Theorem 4 \( D_\zeta^{1/k^4}(THRESH'_{(3r-1)/4}) = \Omega(kr) \), assuming \( c_k \log r \leq k \leq r \) for a large enough constant \( c_k \).
Proof By Lemma 2, it is easy to see that any protocol $P$ that computes $\text{THRESH}_{0}^{(3r-1)/4}$ on input distribution $\xi$ correctly with error probability $1/k^4$ can be used to compute $\text{OR-DISJ}'$ on distribution $\nu$ correctly with error probability $1/k^4 + 1/k^{10} < 1/k^{3}$, since if $(X_1, \ldots, X_k, Y) \sim \nu$, then with probability $1 - 1/k^{10}$, we have

$$\text{OR-DISJ}'(X_1, \ldots, X_k, Y) = (\exists j \in [k], \text{2-DISJ}'(X_j, Y) = 1) = \text{THRESH}_{0}^{(3r-1)/4}(X_1, \ldots, X_k).$$

The theorem follows from Theorem 3.

\[ \square \]

5 Statistical problems

For technical convenience in the reductions, we make the mild assumption that $c_k \log n \leq k \leq n$ where $c_k$ is some large enough constant. For convenience, we will repeatedly ignore an additive $O(1/k^{10})$ error probability introduced in the reductions, since these will not affect the correctness of the reductions, and can be added to the overall error probability by a union bound.

5.1 $F_0$ (\#distinct-elements)

Recall that in the $F_0$ problem, each site $P_i$ has a set $S_i \subseteq [n]$, and the $k$ sites want to compute the number of distinct elements in $\bigcup_{i \in [k]} S_i$.

For the lower bound, we reduce from $\text{THRESH}^n_{(3n-1)/4}$. Given an input $\{X_1, \ldots, X_k\} \sim \xi$ for $\text{THRESH}^n_{(3n-1)/4}$, each site sets $S_i = X_i$. Let $\sigma_F$ be the input distribution of $F_0$ after this reduction.

By Lemma 2 we know that under distribution $\xi$, with probability $1 - 1/k^{10}$, for all $j \in [n]$, $Y$ (recall that $Y$ is the random subset of $[n]$ of size $(n+1)/4$ we used to construct $X_1, \ldots, X_k$ in distribution $\xi$). $V_{j \in [k]} X_{i,j} = 1$. Conditioned on this event, we have

$$\text{THRESH}^{n}_{(3n-1)/4}(X_1, \ldots, X_k) = 1 \iff F_0(\bigcup_{i \in [k]} S_i) > (3n-1)/4.$$

Therefore, by Theorem 4 we have that $D^{1/k^4}_{\sigma_F}(F_0) = \Omega(kn)$. Note that in this reduction, we have to choose $n = \Theta(F_0)$.

Therefore, it makes more sense to write the lower bound as

$$D^{1/k^4}_{\sigma_F}(F_0) = \Omega(kF_0).$$

The following corollary follows from Yao's Lemma (Lemma 1) and the discussion following it.

**Corollary 1** $R^{1/3}(F_0) = \Omega(kF_0/\log k)$.

An almost matching upper bound of $O(k(F_0 \log F_0 + \log n))$ can be obtained as follows: the $k$ sites first compute a $2$-approximation $F_0'$ to $F_0$ using the protocol in [8] (see Sect. 2), which costs $O(k \log n)$ bits. Next, they hash every element to a universe of size $(F_0')^3$, so that there are no collisions among hashed elements with probability at least $1 - 1/F_0$, by a union bound. Finally, all sites send their distinct elements (after hashing) to $P_1$ and then $P_1$ computes the number of distinct elements over the union of the $k$ sets locally. This step costs $O(kF_0 \log F_0)$ bits of communication.

5.2 $\ell_\infty$ (MAX)

In the $\ell_\infty$ problem, each site $P_i$ has a set $S_i \subseteq [n]$, and the $k$ sites want to find an element in $\bigcup_{i \in [k]} S_i$ with the maximum frequency.

For the lower bound, we again reduce from $\text{THRESH}^{n}_{(3n-1)/4}$, Recall that in our hard input distribution for $\text{THRESH}^{n}_{(3n-1)/4}$, there is one special column that contains zero or a single 1. The high level idea is that we try to make this column to have the maximum number of 1’s if originally it contains a single 1, by flipping bits over a random set of rows. Concretely, given an input $\{X_1, \ldots, X_k\} \sim \xi$ for $\text{THRESH}^{n}_{(3n-1)/4}$, the $k$ sites create an input $\{S_1, \ldots, S_k\}$ as follows: first, $P_1$ chooses a set $R \subseteq [k]$ by independently including each $i \in [k]$ with probability $7/8$, and informs all sites $P_i (i \in R)$ by sending each of them a bit. This step costs $O(k)$ bits of communication. Next, for each $i \in R$, $P_i$ flips $X_{i,j}$ for each $j \in [n]$. Finally, each $P_i$ includes $j \in S_i$ if $X_{i,j} = 1$ after the flip and $j \not\in S_i$ if $X_{i,j} = 0$. Let $\sigma_L$ be the input distribution of $\ell_\infty$ after this reduction.

They repeat this input reduction independently $T$ times where $T = c_T \log k$ for a large constant $c_T$, and at each time they run $\ell_\infty(\bigcup_{i \in [k]} S_i)$. Let $R_1, \ldots, R_T$ be the random set $R$ sampled by $P_1$ in the $T$ runs, and let $O_1, \ldots, O_T$ be the outputs of the $T$ runs. They return $\text{THRESH}^{n}_{(3n-1)/4}(X_1, \ldots, X_k) = 1$ if there exists a $t \in [T]$ such that $O_t \geq |R_t| + 1$ and 0 otherwise.

We view an input for $\text{THRESH}^{n}_{(3n-1)/4}$ as a $k \times n$ matrix. The $i$-th row of the matrix is $X_i$. After the bit-flip operations, for each column $j \in [n] \setminus Y$, we have for each $i \in [k]$ that

$$\Pr[X_{i,j} = 1] \leq 7/8 \cdot \left(1 - \frac{(n+1)/4 - 1}{(3n-1)/4}\right) + 1 / 8 \cdot \frac{(n+1)/4}{(3n-1)/4} < 3/4.$$

By a Chernoff bound, for each $j \in [n] \setminus Y$, $\sum_{i \in [k]} X_{i,j} < 13k/16$ with probability $1 - e^{-2\Omega(k)}$. Therefore with probability at least $(1 - e^{-2\Omega(k)} \cdot n) \geq (1 - 1/k^{10})$ (assuming that $c_k \log n \leq k \leq n$ for a large enough constant $c_k$), $\sum_{i \in [k]} X_{i,j} < 13k/16$ holds for all $j \in [n] \setminus Y$.

Now we consider columns in $Y$. We can show again by Chernoff bound that $|R| > 13k/16$ with probability $(1 - 1/k^{10})$ for all columns in $Y$, since each $i \in [k]$ is included into
\[ R \text{ with probability } 7/8, \text{ and after the flips, the probability that } X_{i,j} = 1 \text{ for an } i \text{ when } j \in Y \text{ is negligible. Therefore with probability } (1 - 1/k^{10}), \text{ the column with the maximum number of 1s is in the set } Y, \text{ which we condition on in the rest of the analysis.} \]

In the case when $\text{THRESH}_{(3n-1)/4}^n(X_1, \ldots, X_k) = 1$, then with probability at least $1/8$, there exists a column $j \in Y$ and a row $i \in [k]\backslash R$ for which $X_{i,j} = 1$. If this happens, then for this $j$ we have $\sum_{i \in [k]} X_{i,j} \geq |R| + 1$, or equivalently, $\ell_\infty(\bigcup_{i \in [k]} S_i) \geq |R| + 1$. Otherwise, if $\text{THRESH}_{(3n-1)/4}^n(X_1, \ldots, X_k) = 0$, then $\sum_{i \in [k]} X_{i,j} = |R|$ for all $j \in Y$. Therefore, if $\text{THRESH}_{(3n-1)/4}^n(X_1, \ldots, X_k) = 1$, then the probability that there exists a $t \in [T]$ such that $O_t \geq |R_t| + 1$ is at least $1 - (1 - 1/8)^T > 1 - 1/k^{10}$ (by choosing $c_T$ large enough). Otherwise, if $\text{THRESH}_{(3n-1)/4}^n(X_1, \ldots, X_k) = 0$, then $O_t = |R_t|$ for all $t \in [T]$.

Since our reduction only uses $T \cdot O(k) = O(k \log k)$ extra bits of communication and introduces an extra error of $O(1/k^{10})$, which will not affect the correctness of the reduction. By Theorem 4, we have that $D_{2L}^{1/k} (\ell_\infty) = O(kn)$. Note that in the reduction, we have to assume that $\Theta(\ell_\infty) = \Theta(k)$. In other words, if $\ell_\infty \ll k$ then we choose to have $k' = \Theta(\ell_\infty)$ sites out of the $k$ sites to perform the reduction. Therefore it makes sense to write the lower bound as $D_{2L}^{1/k} (\ell_\infty) = \Omega(\min(\ell_\infty, k)n)$.

The following corollary follows from Yao’s Lemma (Lemma 1) and the discussion following it.

**Corollary 2** $R^{1/3}(\ell_\infty) = \Omega(\min(\ell_\infty, k)n/\log k)$.

A simple protocol that all sites send their elements-counts to the first site solves $\ell_\infty$ with communication $O(\min(k, \ell_\infty)n \log n)$ bits, which is almost optimal in light of our lower bound above.

### 6 Graph problems

In this section we consider graph problems. Let $G = (V, E)$ with $|V| = n$ and $|E| = m$ be an undirected graph. Each site has a subgraph $G_i \subseteq G$, and the $k$ sites want to compute a property of $G$ via a communication protocol. For technical convenience we again assume that $c_k \log n \leq k \leq \min(n, m)$, where $c_k$ is a large enough constant. Except for the upper bound for cycle-freeness in the case without edge duplication, for which $m \geq n$ always causes the graph to not be cycle-free, we assume that $m \geq n$ to avoid an uninteresting case-by-case analysis.

We prove most lower bounds in this section by reductions from the problem $\text{THRESH}_{(3r-1)/4}^n$ for some value $r \leq n^2$. For convenience of presentation, during some reductions we may generate graphs with more than $n$ vertices. This will not affect the order of the lower bounds as long as the number of vertices is $O(n)$ and the number of edges is $O(m)$ (if $m$ appears in the lower bound as a parameter).

The following procedure will be used several times in our reductions. Thus, we present it separately.

**Reconstructing $Y$ from $X_1, X_2, \ldots, X_k$.** Given an input $\{X_1, \ldots, X_k\} \sim \zeta$ for $\text{THRESH}_{(3r-1)/4}^n$ to the $k$ sites, the first site $P_1$ can construct $Y$ correctly with probability $1 - O(1/k^{10})$, using $O(r \log r)$ bits of communication, assuming that $c_k \log r \leq k \leq r$ for a large enough constant $c_k$. We view the input as a $k \times r$ matrix with the $k$ sites’ inputs as rows. For each column $j \in [r]$, $P_1$ randomly selects $c_y \log r$ sites for some large enough constant $c_y$, asks each of them for the $j$-th bit of their input vectors, and then computes the sum of these bits, denoted by $s_j$. Note that if $j \in [r]\backslash Y$, then by a Chernoff bound with probability $1 - e^{-c_y \log r}$ ($k$ is an absolute constant), we have that $s_j \geq c_y \log r/4$. Therefore with probability $1 - e^{-c_y \log r}$, $r \geq 1 - 1/k^{10}$, for all $j \in [r]\backslash Y$, it holds that $s_j \geq c_y \log r/4$. On the other hand, again by a Chernoff bound we have that with probability $1 - 1/k^{10}$, for all $j \in Y$, $s_j \leq c_y \log r/4$. Therefore $P_1$ can reconstruct $Y$ correctly with probability $1 - O(1/k^{10})$. Since the $O(r \log r)$ extra bits of communication and the $O(1/k^{10})$ additional error probability will not affect the correctness of any of our reductions below, we can simply assume that $P_1$ can always reconstruct $Y$ for free.

#### 6.1 Degree

In the degree problem, give a vertex $v \in V$, the $k$ sites want to compute the degree of $v$.

If edge duplication is not allowed, then the degree problem can be solved in $O(k \log n)$ bits of communication: each site sends the number of edges containing the query vertex to the first site $P_1$ and then $P_1$ adds up these numbers. A lower bound of $\Omega(k)$ bits also holds since each site has to speak at least once in our communication model.

When we allow edge duplication, the degree problem is essentially the same as that of $F_0$, by the following reduction. Given an input $\{X_1, \ldots, X_k\} \sim \tau_F$ for $F_0$, the $k$ sites construct a graph $G = (V, E)$ where $V = \{v_1, \ldots, v_n\}$. Each site $P_i$ ($i \in [k]$) does the following: for each element $j \in X_i$ and $j \neq 1$, it creates an edge $(v_1, v_j)$. Let $G$ be the resulting graph. Then $F_0(X_1, \ldots, X_k) = \text{Degree} (v_1) - 1$. Thus from Corollary 1 we get

**Corollary 3** $R^{1/3}(\text{degree}) = \Omega(kd_v/\log k)$.

An $O(kd_v \log n)$ bit upper bound is the following: each site sends all the neighbouring vertices of $v$ to the first site.

#### 6.2 Cycle-freeness

In the cycle-freeness problem, the $k$ sites want to check whether $G$ contains a cycle.
6.2.1 Without edge duplication

If edge duplication is not allowed, then we have the following simple protocol: $P_2, \ldots, P_k$ send the number of their local edges to $P_1$ and $P_1$ computes the total number of edges in the graph $G$, denoted by $m$. If $m \geq n$ then $P_1$ determines immediately that $G$ contains a cycle, since every graph on $n$ vertices having at least $n$ edges must contain a cycle. Otherwise, $m < n$ and $P_2, \ldots, P_k$ send all their edges to $P_1$, who then does a local check. The communication cost of this protocol never exceeds $O(k \log n + \min(m, n) \log n) = O(\min(m, n) \log n)$ bits.

Let $h = \min(m, n)$. An $\Omega(h)$ bit lower bound holds even when $k = 2$, by a reduction from the $2$-DISJ$^h$ problem: suppose $P_1$ has $X$ and $P_2$ has $Y$, where $(X, Y) \sim \tau_{1/4}$ is the hard input distribution for $2$-DISJ$^h$. $P_1$ and $P_2$ construct a graph $G$ on the vertex set $\{s, t, v_1, \ldots, v_h\}$ as follows: for each $i \in X$, $P_1$ creates an edge $(s, v_i)$, and he/she also creates an additional edge $(s, t)$. Similarly, for each $i \in Y$, $P_2$ creates an edge $(v_i, t)$. Let $\sigma_{C_1}$ be the resulting input distribution of $G$. It is easy to see from the reduction that if $X \cap Y \neq \emptyset$, then there is a cycle in the form of $s \rightarrow t \rightarrow v_i \rightarrow s$ for some $i \in [h]$. Otherwise the graph is a forest. Therefore, $2$-DISJ$^h(X, Y) = 1 \iff G$ contains a cycle.

Therefore by Theorem 2 it follows that $D_{\sigma_{C_1}}^{1/400}$ (cycle-freeness) = $\Omega(h)$.

**Corollary 4** $R^{1/3}$ (cycle-freeness without edge duplication) = $\Omega(\min(m, n))$.

6.2.2 With edge duplication

For the lower bound, we reduce from $\text{THRESH}^n_{(3n-1)/4}$. For each $j \in [n]$, $G$ contains a vertex $v_j$. $G$ also contains a special vertex $u$. The total number of vertices in $G$ is $n + 1$.

Given an input $\{X_1, \ldots, X_k\} \sim \zeta$ for $\text{THRESH}^n_{(3n-1)/4}$, the $k$ sites create a graph $G$ for cycle-freeness as follows. Each $P_i$ creates an edge $(u, v_j)$ for each $X_i \in [r]$. In addition, $P_1$ reconstructs $Y$, picks an arbitrary set of $\tilde{Y} \subset [n] \setminus Y$ of size $(n+1)/4$, and creates an arbitrary perfect matching between $\tilde{Y}$ and $Y$. Let $\sigma_{C_2}$ be the resulting input distribution of $G$. By Lemma 2, we have with probability $(1 - 1/k^{10})$ that all pairs $(u, v_j)$ for $j \in [n] \setminus Y$ are connected. It is easy to see from the reduction that if $\text{THRESH}^n_{(3n-1)/4}(X_1, \ldots, X_k) = 1$, then there is a cycle of the form $s \rightarrow v_i \rightarrow v_j \rightarrow s$ for some $i \in [n] \setminus Y$ and $j \in Y$. Otherwise the graph is a forest. Therefore, $\text{THRESH}^n_{(3n-1)/4}(X_1, \ldots, X_k) = 0 \iff G$ is cycle-free.

Thus by Theorem 4, we have that $D_{\sigma_{C_2}}^{1/k^4}$ (cycle-freeness) = $\Omega(kn)$.

**Corollary 5** $R^{1/3}$ (cycle-freeness with edge duplication) = $\Omega((kn/\log k))$.

There is again a trivial $O(kn \log n)$ upper bound. Each site first checks its local graph and reports directly if a cycle is found, otherwise the site sends all its edges (there are no more than $n - 1$ such edges for a cycle-free graph) to $P_1$. Finally $P_1$ checks the cycle-freeness on the union of those edges.

6.3 Connectivity and #CC

In the connectivity problem, the $k$ sites want to check whether $G$ is connected. In the #connected-components (#CC) problem, the $k$ sites want to compute the number of connected components in $G$. Note that solving #CC also solves connectivity, thus we only show the lower bound for connectivity.

6.3.1 Without edge duplication

For the lower bound, we reduce from $\text{THRESH}^n_{(3r-1)/4}$ where $r = \min\{m/k, n\}$. For each $i \in [k]$, $G$ contains a vertex $u_i$; and for each $j \in [r]$, $G$ contains a vertex $v_j$. The total number of vertices in $G$ is at most $n + k \leq 2n$. Given an input $\{X_1, \ldots, X_k\} \sim \zeta$ for $\text{THRESH}^n_{(3r-1)/4}$, the $k$ sites create a graph $G$ for connectivity as follows. Each $P_i$ creates an edge $(u_i, v_j)$ for each $X_{i,j} = 1$. In addition, $P_1$ reconstructs $Y$, and then creates a path containing $\{v_j \mid j \in [r]\}$ and a path containing $\{v_j \mid j \in [r] \setminus Y\}$. See Fig. 1 for an illustration. Let $\sigma_{N_1}$ be the resulting input distribution of $G$. It is easy to see from the reduction that $\text{THRESH}^n_{(3r-1)/4}(X_1, \ldots, X_k) = 1 \iff G$ is connected.

Thus by Theorem 4, we have that $D_{\sigma_{N_1}}^{1/k^4}$ (connectivity) = $\Omega(kr)$.

**Corollary 6** $R^{1/3}$ (connectivity without edge duplication) = $\Omega(kr/\log k)$.

For the upper bound, all connected components (thus #CC and connectivity) can be found by the protocol in which all sites send their local spanning trees to the first site $P_1$ and then $P_1$ does a local computation, which costs $O(kr \log n)$ bits of communication.

6.3.2 With edge duplication

For the lower bound, we use a slightly modified reduction of the one for the without edge duplication case. We reduce from $\text{THRESH}^n_{(3n-1)/4}$. For each $j \in [n]$, $G$ contains a vertex $v_j$. $G$ also contains a special vertex $u$. The total number of vertices in $G$ is $n + 1$. Given an input $\{X_1, \ldots, X_k\} \sim \zeta$ for $\text{THRESH}^n_{(3n-1)/4}$, the $k$ sites create a graph $G$ for connectivity as follows. Each $P_i$ creates an edge $(u, v_j)$ for
each $X_{i,j} = 1$. In addition, $P_1$ reconstructs $Y$, and then creates a path containing $\{v_j \mid j \in Y\}$ and a path containing $\{v_j \mid j \in [n] \setminus Y\}$. The total number of edges is $O(n) = O(m)$. This graph can be seen as the graph we constructed for the without edge duplication case after merging $u_1, \ldots, u_k$ to a single vertex $u$ while maintaining all the adjacent edges. Let $\sigma_{X_2}$ be the resulting input distribution of $G$. The correctness of the reduction is the same as before, that is, $\text{THRESH}_{\text{3r-1/4}}^r(X_1, \ldots, X_k) = 1$ if and only if $G$ is connected. Thus by Theorem 4, we have that $D_{\text{3r-1/4}}^{1/k^4}$ (connectivity) $= \Omega(kn)$. 

**Corollary 7** $R^{1/3}$ (connectivity with edge duplication) $= \Omega(kn / \log k)$. 

The upper bound is the same as the without edge duplication case, and the cost is $O(kn \log n)$ bits (note that since we allow edge duplication here, the total number of edges of the $k$ spanning trees cannot be bounded by $O(m)$).

6.4 Bipartiteness

In the bipartiteness problem, the $k$ sites want to check whether $G$ is bipartite.

6.4.1 Without edge duplication

For the lower bound, we reduce from $\text{THRESH}_{\text{3r-1/4}}^r$ where $r = \min\{m/k, n\}$, Let $\{X_1, \ldots, X_k\} \sim \xi$ be an input for $\text{THRESH}_{\text{3r-1/4}}^r$. The $k$ sites create a graph $G = (V, E)$ with $V = A \cup B \cup C$ where $A = \{a_1, \ldots, a_r\}$, $B = \{b_1, \ldots, b_r\}$ and $C = \{c_1, \ldots, c_k\}$, as follows: each $P_i$ creates an edge $(c_i, b_j)$ for each $X_{i,j} = 1$. In addition, $P_1$ does the following:

1. Creates an edge $(a_i, b_i)$ for each $i \in [r]$.
2. Reconstructs $Y$. For each $i \in [k]$ and $j \in Y$, creates an edge $(c_i, a_j)$.

The total number of vertices of $G$ is $2r + k < 3n$. Let $\sigma_{B_1}$ be the resulting input distribution of $G$. One can see from the reduction that if

$\text{THRESH}_{\text{3r-1/4}}^r(X_1, \ldots, X_k) = 1$,

then there exists at least one triangle in the form of $(a_i, b_i, c_j)$ for some $i \in [r], j \in [k]$. Otherwise if

$\text{THRESH}_{\text{3r-1/4}}^r(X_1, \ldots, X_k) = 0$,

then all edges are between two vertex sets $\{a_i \mid i \in [r] \setminus Y\} \cup C \cup \{b_i \mid i \in Y\}$ and $\{b_i \mid i \in [r] \setminus Y\} \cup \{a_i \mid i \in Y\}$ and consequently $G$ is a bipartite graph. Therefore, $\text{THRESH}_{\text{3r-1/4}}^r(X_1, \ldots, X_k) = 0 \iff G$ is bipartite.

Thus by Theorem 4, we have that $D_{\text{3r-1/4}}^{1/k^4}$ (bipartiteness) $= \Omega(kr)$. 

**Corollary 8** $R^{1/3}$ (bipartiteness without edge duplication) $= \Omega(kr / \log k)$. 

For the upper bound, we can assume that the graph is connected, since otherwise we can first compute all connected components (which costs $O(kn \log n)$ bits of communication as mentioned in Sect. 6.3), and then work on each connected component. The protocol works as follows: the first site $P_1$ chooses an arbitrary vertex $u$ in the graph, and grows a breadth-first-search (BFS) tree rooted at $u$ by communicating with the other $k - 1$ sites. In the first round, $P_1$ asks each site to report the vertices adjacent to $u$ using its local edges. The communication is at most $O(|N(u)| \log n \cdot k)$ bits, where $N(u)$ denotes the set of neighbors of $u$, and $|N(u)|$ denotes the number of (distinct) neighbors of $u$. From this, $P_1$ computes the entire set $N(u)$ of neighbors of $u$, without duplication, and sends it to the other $k - 1$ sites. This also takes $O(|N(u)| \log n \cdot k)$ bits of communication. Now the sites all know $N(u)$, and they can build the first layer of the BFS tree rooted at $u$. Next, $P_1$ picks the first child $v$ (according to an arbitrary but fixed order) of $u$, and repeats this process on $v$. If ever a site finds an odd cycle, it is announced to $P_1$. Notice
that every vertex is sent at most \( k \) times to \( P_1 \), meanwhile the total number of vertices sent is no more than \( O(m) \), so the total communication is \( O(kr \log n) \) bits.

**Remark 3** We notice that if a graph is node-partitioned among the \( k \) sites, that is, each node is stored at one site together with all its adjacent edges, then we can directly implement the algorithm for connectivity in \([1]\) in the message-passing model using \( O(k + n) \) bits of communication. This shows a sharp difference between node-partition and edge-partition for connectivity with respect to the input storage.

**6.4.2 With edge duplication**

We reduce from \( \text{THRESH}^n_{(3n-1)/4} \). The reduction is a simple modification of the one for the without edge duplication case. Given an input \([X_1, \ldots, X_k] \sim \xi\) for \( \text{THRESH}^n_{(3n-1)/4} \), the \( k \) sites create a graph \( G = (V, E) \) with \( V = A \cup B \cup C \) where \( A = \{a_1, \ldots, a_n\} \), \( B = \{b_1, \ldots, b_n\} \) and \( C = \{c\} \), as follows: each \( P_i \) creates an edge \((c, b_j)\) for each \( X_{i,j} = 1 \). In addition, \( P_1 \) creates an edge \((a_i, b_i)\) for each \( i \in [n]\), reconstructs \( Y \), and for each \( j \in Y \) creates an edge \((c, a_j)\). The total number of vertices of \( G \) is \( 2n + 1 \). Let \( \sigma_{B_2} \) be the resulting input distribution of \( G \). The correctness of the reduction is similar as before, that is, \( \text{THRESH}^n_{(3n-1)/4}(X_1, \ldots, X_k) = 0 \) if and only if \( G \) is bipartite. Thus by Theorem 4, we have that \( D_{\sigma_{B_2}}^{1/k^4}(\text{bipartiteness}) = \Omega(kn) \).

**Corollary 9** \( R^{1/3}(\text{bipartiteness with edge duplication}) = \Omega(kn / \log k) \).

The upper bound is the same as the without edge duplication case, and the communication complexity is \( O(kn \log n) \) bits. Note that since we have edge duplication here, the claim that “the total number of vertices sent is no more than \( O(m) \)” does not hold.

**6.5 Triangle-freeness**

In the triangle-freeness problem, the \( k \) sites want to check whether \( G \) contains a triangle.

**6.5.1 Without edge duplication**

An \( O(m \log n) \) upper bound is the following: \( P_2, \ldots, P_k \) send all their edges to \( P_1 \) and then \( P_1 \) does a local check.

There is an \( \Omega(m) \) bit lower bound on the communication which holds even when \( k = 2 \), by a reduction from \( 2\text{-DISJ}^m \). Suppose \( P_1 \) holds \( X \) and \( P_2 \) holds \( Y \), where \([X, Y] \sim \tau_{1/4}\) is the hard input distribution for \( 2\text{-DISJ}^m \). Sites \( P_1 \) and \( P_2 \) construct a graph \( G = (V, E) \) with \( V = A \cup B \cup C \) where \( A = \{a_1, \ldots, a_n\} \), \( B = \{b_1, \ldots, b_n\} \) and \( C = \{c, \ldots, c_n\} \) as follows: for each \( i \in X \), \( P_1 \) creates an edge \((a_i, b_i)\) such that \( (p - 1)n + q = i \) \((p, q \in [n])\) (note that the solution of \((p, q)\) is unique). He/she also creates an edge \((a_i, b_i)\) for all \( i \in [n] \). Similarly, for each \( i \in Y \), \( P_2 \) creates an edge \((b_i, c_i)\) such that \( (p - 1)n + q = i \) \((p, q \in [n])\). The graph \( G \) has \( 3n \) vertices and \( O(m) \) edges. See Fig. 2 for an illustration.

Let \( \sigma_{T_1} \) be the resulting input distribution of \( G \). One can see from the reduction that if \( X \cap Y \neq \emptyset \), then there is a triangle in the form of \((a_p, b_q, c_q)\) for some \( p, q \in [n] \). Otherwise the graph is triangle-free. Therefore, \( 2\text{-DISJ}^m(X, Y) = 0 \iff G \) is triangle free.

Therefore by Theorem 2 we have \( D_{\sigma_{T_1}}^{1/400}(\text{triangle-freeness}) = \Omega(m) \).

---

Fig. 2 Graph \( G \) in the reduction for triangle-freeness

(a\(p, c_q\)) exists if and only if \( i \in X \) where \( i = (p - 1)n + q \).

(b\(p, c_q\)) exists if and only if \( i \in Y \) where \( i = (p - 1)n + q \).
Corollary 10 \( R^{1/3} (\text{triangle-freeness without edge duplication}) = \Omega(m) \).

6.5.2 With edge duplication

We reduce from \( \text{THRESH}^m \) to \( \text{THRESH}^{(3m-1)/4} \), and prove an \( \Omega(km) \) lower bound on the communication cost. The reduction is an extension of the one for the without edge duplication case.

Given an input \( \{X_1, \ldots, X_k\} \sim \xi \) for \( \text{THRESH}^{(3m-1)/4} \), the \( k \) sites create the following input graph \( G = (V, E) \) for triangle-freeness with \( V = A \cup B \cup C \) where \( A = \{a_1, \ldots, a_n\}, B = \{b_1, \ldots, b_n\} \) and \( C = \{c_1, \ldots, c_n\} \). Each site \( P_i \) does the following: for each \( j \in [m] \) such that \( X_{i,j} = 1 \), the site creates an edge \((a_p, c_q)\) such that \((p-1)n + q = j (p, q \in [n])\). In addition, the first site \( P_1 \) also does the following.

1. Creates an edge \((a_t, b_t)\) for each \( t \in [n] \).
2. Reconstructs \( Y \). For each \( j \in Y \), creates an edge \((b_p, c_q)\) such that \((p-1)n + q = j (p, q \in [n])\).

Let \( \sigma_{T_2} \) be the resulting input distribution of \( G \).

As before, it is easy to see that if \( \text{THRESH}^{(3m-1)/4}(X_1, \ldots, X_k) = 1 \), then there is a triangle of the form \((a_{p}, b_{p}, c_{q})\) for some \( p, q \in [n] \). Otherwise the graph is triangle-free. Therefore,

\[ \text{THRESH}^{(3m-1)/4}(X_1, \ldots, X_k) = 0 \iff G \text{ is triangle-free.} \]

By Theorem 4, we have that \( D_{\sigma_{T_2}}^{1/k^4} (\text{triangle-freeness}) = \Omega(km) \).

Corollary 11 \( R^{1/3} (\text{triangle-freeness with edge duplication}) = \Omega(km / \log k) \).

There is a simple protocol with \( O(km \log n) \) bits of communication: each site sends all its edges to \( P_1 \) and the \( P_1 \) does a local check.

We comment that our upper and lower bounds also applies to testing clique-freeness, that is, the \( k \) sites want to check whether \( G \) contains a clique of size \( s \) for a fixed constant \( s \).

6.6 The diameter problem and an approximation algorithm

We would like to mention the diameter problem which cannot be solved by the technique introduced in this paper. In the diameter problem, the \( k \) sites want to compute the diameter of a graph \( G = (V, E) \) in which the edges are distributed amongst the \( k \) sites. We noticed that a lower bound for the \( k \)-site set-disjointness problem, shown in a very recent paper by Braverman et al. [6], can be used to prove a tight lower bound for this problem.\(^3\)

\( ^3 \) We conjectured Theorem 5 in the conference version of this paper.

Theorem 5 The randomized communication complexity of the diameter problem in the message-passing model is \( \Omega(km) \) bits, assuming edge duplication is allowed.

Proof We briefly state the set-disjointness problem and sketch the reduction here. We can view the input of the set-disjointness problem as a \( k \) by \( m \) matrix \( M \). Each cell in the matrix is 0 or 1. Site \( P_i \) holds the \( i \)-th row of \( M \). Their goal is to check if there is a column in the matrix whose cells are all 1. In [6] it was shown that solving \( k \)-site set-disjointness needs at least \( \Omega(mk) \) bits of communication.

The reduction from set-disjointness to diameter works as follows: We create \( n \) nodes \( v_1, \ldots, v_n \). Let \( M(i, j) \) be the cell in the \( i \)-th row and \( j \)-th column of \( M \). For each \( M(i, j) = 0 \), we add an edge \((v_p, v_q)\) where \((p-1)n + q = j (p, q \in [n])\). Now if there is no column whose cells are all 1, we get a clique, that is, the diameter of the graph is 1. Otherwise there is at least one edge missing, thus the diameter will be at least 2. \( \square \)

Note that the naive algorithm in which every site sends all of its edges to the first site will match this lower bound up to a logarithmic factor.

In [11] an algorithm for constructing a graph spanner in the random-access memory (RAM) model with an additive distortion 2 is proposed. A graph spanner with an additive distortion \( d \) preserves all pairwise distances of vertices in the original graph up to an additive error \( d \). In particular, the diameter is preserved up to an additive error \( d \) when saying all pairwise distances of vertices are preserved up to an additive error \( d \). This algorithm can be easily implemented in the message-passing model with a communication complexity of \( O(kn^{3/2} \log^2 n) \) bits.

This fact gives us a hint that in order to prove an \( \tilde{\Omega}(kn^2) \) (in the case when \( m = \Theta(n^2) \) lower bound, we have to explore the difficulty of distinguishing a diameter \( T \) versus \( T + 1 \) for a value \( T \in [1, n - 2] \) (in fact, one can show that \( T \) also needs to be a fixed constant). Now we briefly describe how to implement the algorithm in [11] in the message-passing model.

The algorithm in [11] works as follows (in the RAM model).

1. We pick \( \Theta(\sqrt{n} \log n) \) vertices in the graph uniformly at random, and grow a breadth-first-search (BFS) tree rooted on each of these vertices. We then include all edges of these BFS trees into the spanner.
2. We include all edges incident to vertices whose degrees are no more than \( \sqrt{n} \).

In [11] it was shown that this algorithm computes a spanner with an additive distortion 2 correctly with probability 0.99. Now let us briefly discuss how to implement this algorithm in the message-passing model. For the first step, the
random sampling can be done by $P_1$ locally, and then $P_1$ communicates with the other $k-1$ sites to grow a BFS tree rooted on each of these vertices using the algorithm described in Sect. 6.4. Recall that constructing each of these BFS trees costs $O(kn \log n)$ bits of communication. Thus the total communication needed for the first step is $O(kn^{3/2} \log^2 n)$ bits.

For the second step, the $k$ sites first use the algorithm in [8] to compute the degree of each vertex (this is essentially $P_0$) up to a factor of 2, using $O(kn \log n)$ bits of communication, and then they construct the set $H = \{ v \in V \mid \text{degree}(v) \leq 2 \sqrt{n} \}$. Next, $P_2, \ldots, P_k$ send $P_1$ all edges that are incident to a vertex in $H$. Note that each of $P_i$ ($i = 2, \ldots, k$) will send at most $O(n^{3/2})$ edges to $P_1$. Therefore, the total communication cost of the second step is bounded by $O(kn \log n + kn^{3/2} \log n) = O(kn^{3/2} \log^2 n)$ bits. We conclude with the following theorem for diameter.

**Theorem 6** There exists a randomized protocol that approximates the diameter of a graph up to an additive error of 2 in the message-passing model. The protocol succeeds with probability 0.99 and uses $O(kn^{3/2} \log^2 n)$ bits of communication.

7 Concluding remarks

In this paper we show that exact computation of many basic statistical and graph problems in the message-passing model are necessarily communication-inefficient. An important message we want to deliver through these negative results, which is also the main motivation of this paper, is that a relaxation of the problem, such as an approximation, is necessary in the distributed setting if we want communication-efficient protocols. Besides approximation, the layout and the distribution of the input are also important factors for reducing communication.

An interesting future direction is to further investigate efficient communication protocols for approximately computing statistical and graph problems in the message-passing model, and to explore realistic distributions and layouts of the inputs.

One question which we have not discussed in this paper but is important in practice, is whether we can obtain round-efficient protocols that (almost) match the lower bounds which hold even for round-inefficient protocols? Most simple protocols presented in this paper only need a constant number of rounds, except the ones for bipartiteness and (approximate) diameter, where we need to grow BFS trees which are inherently sequential (require $\Omega(\Delta)$ rounds where $\Delta$ is the diameter of the graph). Using the sketching algorithm in [1], we can obtain a 1-round protocol for bipartiteness that uses $O(kn)$ bits of communication. We do not know whether a round-efficient protocol exists for the additive-2 approximate diameter problem that could (almost) match the $\tilde{O}(kn^{3/2})$ bits upper bound obtained by the round-inefficient protocol in Sect. 6.6.

References

1. Ahn, K.J., Guha, S., McGregor, A.: Analyzing graph structure via linear measurements. In: Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 459–467. SIAM (2012)
2. Ahn, K.J., Guha, S., McGregor, A.: Graph sketches: Sparsification, spanners, and subgraphs. In: Proceedings of ACM Symposium on Principles of Database Systems, pp. 5–14 (2012)
3. Arackaparambil, C., Brody, J., Chakrabarti, A.: Functional monitoring without monotonicity. In: Proceedings of International Colloquium on Automata, Languages, and Programming (2009)
4. Balcan, M.F., Blum, A., Fine, S., Mansour, Y.: Distributed learning, communication complexity and privacy. J. Mach. Learn. Res. Proc. Track, 23, 26.1–26.22 (2012)
5. Beame, P., Koutris, P., Suciu, D.: Communication steps for parallel query processing. In: Proceedings of ACM Symposium on Principles of Database Systems, pp. 273–284 (2013)
6. Braverman, M., Ellen, F., Oshman, R., Pitassi, T., Vaikuntanathan, V.: A tight bound for set disjointness in the message-passing model. In: FOCS, pp. 668–677 (2013)
7. Brown, P., Haas, P.J., Myllymaki, J., Pirahesh, H., Reinwald, B., Sismanis, Y.: Toward automated large-scale information integration and discovery, pp. 161–180. In: Data Management in a Connected World (2005)
8. Cormode, G., Muthukrishnan, S., Yi, K.: Algorithms for distributed functional monitoring. ACM Trans. Algorithms 7(2), 21 (2011)
9. Daumé, III, H.D., Phillips, J.M., Saha, A., Venkatasubramanian, S.: Efficient protocols for distributed classification and optimization. In: Algorithmic Learning Theory, pp. 154–168 (2012)
10. Daumé, III, H.D., Phillips, J.M., Saha, A., Venkatasubramanian, S.: Protocols for learning classifiers on distributed data. J. Mach. Learn. Res. Proc. Track 22, 282–290 (2012)
11. Dor, D., Halperin, S., Zwick, U.: All-pairs almost shortest paths. SIAM J. Comput. 29(5), 1740–1759 (2000)
12. Erdös, P., Rényi, A.: On the evolution of random graphs. In: Publication of the mathematical institute of the hungarian academy of sciences, pp. 17–61 (1960)
13. Estan, C., Varghese, G., Fisk, M.: Bitmap algorithms for counting active flows on high-speed links. IEEE/ACM Trans. Netw. 14(5), 925–937 (Oct. 2006)
14. Flajolet, P., Fusy, É., Gandouet, O., Meunier, F.: Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm. DMTCS Proceedings (1) (2008)
15. Flajolet, P., Martin, G.N.: Probabilistic counting algorithms for data base applications. J. Comput. Syst. Sci. 31(2), 182–209 (1985)
16. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. J. ACM 45(4), 653–730 (1998)
17. Goodrich, M.T., Sitchinava, N., Zhang, Q.: Sorting, searching, and simulation in the mapreduce framework. In: Proceedings of International Symposium on Algorithms and Computation, pp. 374–383 (2011)
18. Huang, Z., Radunović, B., Vojnović, M., Zhang, Q.: The communication complexity of approximate maximum matching in distributed data. Manuscript (2013), http://research.microsoft.com/apps/pubs/default.aspx?id=188946
19. Huang, Z., Yi, K., Zhang, Q.: Randomized algorithms for tracking distributed count, frequencies, and ranks. In: Proceedings of
Distributed computation is communication expensive

20. Kane, D.M., Nelson, J., Woodruff, D.P.: An optimal algorithm for the distinct elements problem. In: Proceedings of ACM Symposium on Principles of Database Systems, pp. 295–306 (2012)

21. Karloff, H.J., Suri, S., Vassilvitskii, S.: A model of computation for mapreduce. In: Proceedings of ACM-SIAM Symposium on Discrete Algorithms, pp. 938–948 (2010)

22. Koutris, P., Suciu, D.: Parallel evaluation of conjunctive queries. In: Proceedings of ACM Symposium on Principles of Database Systems, pp. 223–234 (2011)

23. Kushilevitz, E., Nisan, N.: Communication Complexity. Cambridge University Press, Cambridge (1997)

24. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, Cambridge (1995)

25. Palmer, C.R., Gibbons, P.B., Faloutsos, C.: Anf: a fast and scalable tool for data mining in massive graphs. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 81–90 (2002)

26. Phillips, J.M., Verbin, E., Zhang, Q.: Lower bounds for number-in-hand multiparty communication complexity, made easy. In: Proceedings of ACM-SIAM Symposium on Discrete Algorithms (2012)

27. Razborov, A.A.: On the distributional complexity of disjointness. In: Proceedings of International Colloquium on Automata, Languages, and Programming (1990)

28. Valiant, L.G.: A bridging model for parallel computation. Commun. ACM 33(8), 103–111 (1990)

29. Woodruff, D.P., Zhang, Q.: An optimal lower bound for distinct elements in the message passing model. In SODA, pp. 718–733 (2014)

30. Woodruff, D.P., Zhang, Q.: Tight bounds for distributed functional monitoring. In: Proceedings of ACM Symposium on Theory of Computing (2012)

31. Yao, A.C.: Probabilistic computations: Towards a unified measure of complexity. In: Proceedings of IEEE Symposium on Foundations of Computer Science (1977)