Beyond Set Disjointness: The Communication Complexity of Finding the Intersection

Joshua Brody
Swarthmore College
brody@cs.swarthmore.edu

Amit Chakrabarti
Dartmouth College
ac@cs.dartmouth.edu

Ranganath Kondapally
Dartmouth College
rangak@cs.dartmouth.edu

David P. Woodruff
IBM Almaden
dpwoodru@us.ibm.com

Grigory Yaroslavtsev
Brown University, ICERM
grigory@grigory.us

ABSTRACT

We consider the following fundamental communication problem - there is data that is distributed among servers, and the servers want to compute the intersection of their data sets, e.g., the common records in a relational database. They want to do this with as little communication and as few messages (rounds) as possible. They are willing to use randomization, and fail with a tiny probability. Given a protocol for computing the intersection, it can also be used to compute the exact Jaccard similarity, the rarity, the number of distinct elements, and joins between databases. Computing the intersection is at least as hard as the set disjointness problem, which asks whether the intersection is empty.

Formally, in the two-server setting, the players hold subsets $S, T \subseteq [n]$. In many realistic scenarios, the sizes of S and T are significantly smaller than n, so we impose the constraint that $|S|, |T| \leq k$. We study the minimum number of bits the parties need to communicate in order to compute the intersection set $S \cap T$, given a certain number r of messages that are allowed to be exchanged. While $O(k \log(n/k))$ bits is achieved trivially and deterministically with a single message, we ask what is possible with more than one message and with randomization. We give a smooth communication/round tradeoff which shows that with $O(\log^* k)$ rounds, $O(k)$ bits of communication is possible, which improves upon the trivial protocol by an order of magnitude. This is in contrast to other basic problems such as computing the union or symmetric difference, for which $\Omega(k \log(n/k))$ bits of communication is required for any number of rounds. For two players, known lower bounds for the easier problem of set disjointness imply our algorithms are optimal up to constant factors in communication and number of rounds. We extend our protocols to m-player protocols, obtaining an optimal $O(mk)$ bits of communication with a similarly small number of rounds.

PODC'14, July 15–18, 2014, Paris, France.
Copyright 2014 ACM 978-1-4503-2944-6/14/07 ...$15.00.
http://dx.doi.org/10.1145/2611462.2611501 .

1. INTRODUCTION

Communication complexity [27] quantifies the communication necessary for two or more players to compute a function, where each player holds only a portion of the function's input. This model is widely studied, with applications in circuit complexity [24], combinatorial auctions [20], compressed sensing [2], data streams [1], and many other areas. We refer the reader to the book by Kushilevitz and Nisan [18] for a thorough treatment of the subject, which we only briefly describe here.

For randomized protocols, there are two well-studied and closely-related models. In the common random string model the players share an infinite string of independent unbiased coin tosses, and the players are otherwise deterministic. The correctness requirement is that for every input pair x, y, the output of Alice and Bob is equal to $f(x, y)$ with probability at least $1 - \delta$, for some specified $\delta > 0$, where the probability is taken over the shared random string. We let $R_\delta(f)$ be the minimum, over protocols in the common random string model satisfying the correctness protocol for f, of the maximum number of bits exchanged by the protocol over all inputs and shared random strings. For brevity, we let $R(f) = R_{1/3}(f)$. We note that a $2/3$ success probability can be amplified to $1 - \epsilon$ for an arbitrarily small constant $\epsilon > 0$ by incurring a constant factor overhead in communication.

In the private random string model, the players do not share a random string, but rather are allowed to be randomized using private randomness. By a result of Newman [19], any problem that can be solved in the common random string model can be solved in the private random string model, adding only $O(\log \log T)$ to the communication complexity, where T is the number of different inputs to the players. One unfortunate aspect of this reduction is that it is non-constructive in the sense that for each input length n, the protocol used is either hardwired an advice string that depends on n, or the players must search for the advice string, which doesn't require communication, but can result in unnecessary computation. We give our upper bounds in the common random string model, but describe how to translate them into constructive protocols in the private random string model, preserving optimality.

Besides the total communication, another well-studied resource is the total number of messages exchanged between the two players, known as the round complexity. In certain applications a server may not always be online, resulting in a significant delay between messages. There may also be undesirable overhead in the transmission of each message. Thus, it is important to not only achieve optimal communication, but also an optimal number of rounds for a given amount of communication. We use $D^{(r)}(f)$ and $R^{(r)}(f)$ to denote the deterministic and randomized communication complexity (in the common random string model) for protocols restricted to using at most r rounds.

One of the most well-studied problems in communication complexity is the disjointness function $\mathsf{DISJ}_k^n(S,T)$. In this problem, Alice has an input set $S \subseteq [n]$ of size at most k, Bob has an input set $T \subseteq [n]$ of size at most k, and $\mathsf{DISJ}_k^n(S,T) = 1$ iff $|S \cap T| = 0$. Håstad and Wigderson [15] showed that $R(\mathsf{DISJ}_k^n) = \Theta(k)$. The lower bound of $\Omega(k)$ follows by taking known lower bounds for set disjointness without a cardinality restriction on S and T, due to Kalysundaram and Schnitger [16], simplified by Razborov [25] and Bar-Yossef et al. [3], and combining them with a padding argument. The upper bound of $O(k)$ is due to a protocol given by Håstad and Wigderson, which they also remark was known and used many years ago [22].

In this paper we are interested in a seemingly much harder problem than the disjointness function. Namely, we are interested in recovering the *entire set intersection* $S \cap T$, rather than only deciding if $|S \cap T| = 0$. We call this problem the INT^k problem. Computing the intersection or the size of the intersection of two sets is a fundamental problem in computer science, with applications to distributed databases, including computing joins, finding duplicates, measuring Jaccard similarity, and computing rarity [9]; for more details on these applications, see below. We note that a recent work of Pagh et al. [21] studies approximating the size of the set intersection in the 1-way communication model, while we seek to recover the actual intersection and allow 2-way communication.

By the lower bound for the disjointness function, we have that $R(\mathrm{INT}^k) = \Omega(k)$, which holds for any number of rounds. Also, Alice and Bob can deterministically exchange their inputs using only $O(k \log(n/k))$ bits of communication, so $D^{(1)}(\mathrm{INT}^k) = O(k \log(n/k))$. They can also first hash the elements in their sets to $O(\log k)$-bit strings, and exchange the hashed values, from which they can decide which elements are in the intersection with probability $1 - 1/k^C$, for an arbitrarily large constant $C > 0$. This means that $R^{(1)}(\mathrm{INT}^k) = O(k \log k)$, which is optimal since $R^{(1)}\mathsf{DISJ}_k^n = \Omega(k \log k)$ [8, 7].

This was extended in [26] to interpolate between the one-round and unbounded-round situations, giving an r-round upper bound of $O(k \log^{(r)} k)$. Both [15] and [26] work by interpreting the public coin as a sequence of sets and having Alice or Bob send the index of the first set in this sequence containing her or his set. If $S \cap T = \emptyset$, then if the sets in the public coin were uniformly random and Alice sends the index of a set Z to Bob, w.h.p. $|Z \cap T| \approx |T|/2$, and so in $O(\log k)$ rounds they can solve k-DISJ. In [26] the public coin is instead interpreted as a list of random sparse sets, so now if $S \subseteq Z$, $|Z \cap T| \ll |T|/2$, and so in fewer rounds they can solve k-DISJ, at the cost of larger communication per round. These protocols seem specific to k-DISJ, and we do not know how to adapt them to the intersection problem, the main difficulty being in handling large $|S \cap T|$.

A somewhat related problem is that of computing k copies of the equality function EQ_k^n. In this problem, Alice has k strings $x^1, \ldots, x^k \in \{0,1\}^n$, Bob has k strings $y^1, \ldots, y^k \in \{0,1\}^n$ and they wish to compute $\mathsf{EQ}_k^n(x^1, \ldots, x^k, y^1, \ldots, y^k)$, a length-$k$ bit vector, where the i-th bit is 1 iff $x^i = y^i$. Feder, Kushilevitz, Naor, and Nisan [12] show that $R(\mathsf{EQ}_k^n) = \Theta(k)$. One unfortunate aspect of their protocol is that the number of rounds they achieve is $\Omega(\sqrt{k})$, as their protocol seems to be inherently sequential.

We observe in Section 3.1 below that by hashing into buckets, given a protocol for EQ_k^n, we can build a protocol for the INT^k problem. Plugging in the protocol of Feder et al., we obtain a randomized protocol for INT^k with the optimal $O(k)$ bits of communication in Theorem 3.1. However, the round complexity is $O(\sqrt{k})$. Another way of obtaining the optimal $O(k)$ bits of communication is to use a technique of Braverman and Rao to compress a protocol to its so-called internal information cost [5]. For the INT^k problem, the internal information cost is $O(k)$, and so this results in a protocol with the optimal $O(k)$ bits of communication, with a much smaller $O(\log k)$ number of rounds. It may seem plausible that one can combine the hashing technique we use in Section 3.1 together with $O(k)$ invocations of the recent round-optimal protocols for EQ^n [6], each with error probability $O(1/k)$. However, with such low error probability one invocation of the protocol of [6] requires $\Omega(\log k)$ communication for any number of rounds, even though the expected communication for the simpler task of verifying that two unequal inputs are indeed not equal with error probability $O(1/k)$, can be smaller.

Our Results: In this paper we give a new randomized protocol for INT^k which achieves the optimal $O(k)$ bits of communication, and simultaneously achieves $O(\log^* k)$ number of rounds, where $\log^* k$ is the iterated logarithm function, that is the number of times the logarithm function must be iteratively applied before the result is at most 1. Our number of rounds provides a significant improvement on the earlier $O(\log k)$ rounds needed to achieve the optimal $O(k)$ bits of communication given in previous work.

We also provide a more refined tradeoff, showing that with $O(r)$ rounds, one can achieve communication $O(k \log^{(r)} k)$, where $\log^{(r)} k$ is the function obtained by iteratively applying the logarithm function r times (e.g., $\log^{(0)} k = k, \log^{(1)} k = \log k$, $\log^{(2)} k = \log \log k$, etc.). Our protocols work in the common random string model, but can be turned into constructive protocols (i.e., without using Newman's theorem) in the private random string model, incurring an additive $O(\log \log n)$ bits of communication with no increase in the number of rounds.

Next we extend this to the setting in which there are m players in the private messages model [4, 23] and give a protocol with $O(k \log^{(r)} k)$ average communication per player, expected number of rounds $O(r \cdot \max(1, \frac{\log m}{k}))$, and error probability $1 - 1/2^k$. We give a similar guarantee with a worst-case communication bound ber player.

Our protocols for two players are communication-optimal, up to a constant factor in the number of rounds r, in light of a recent $\Omega(k \log^{(r)} k)$ communication lower bound for the DISJ_k^n problem [26]. For m players and $O(\log^* k \cdot \max(1, \frac{\log m}{k}))$ rounds, our $O(mk)$ communication is also optimal up to constant factors [4, 23].

Since EQ_k^n is also a special case of INT^k (Fact 2.1), we also significantly improve the round complexity of Feder et al. [12].

Applications: Set-intersection and list-intersection are very basic problems in databases, occurring in data mining applications, text analytics, and evaluation of conjunctive queries. They are also key operations in enterprise and web search. We refer the reader to a recent sample of database theory using the set-intersection primitive [10, 28, 13, 11]. While these papers focus on the computational costs of set-intersection, given the emergence of cloud computing and distributed databases, the communication cost is just as important. A quite basic problem, such as computing the join of two databases held by different servers, requires computing an intersection, which one would like to do with as little communication and as few messages as possible.

Prior to our work, it was not even known how to compute the size $|S \cap T|$ of the intersection with $O(k)$ communication and fewer than $O(\log k)$ rounds. Given our upper bound for set intersection, we significantly improve the communication/round tradeoffs for computing $|S \cap T|$. Since communicating $|S|$ and $|T|$ can be done in one-round, this gives the first protocol for computing the size $|S \cup T|$ of the union with our communication/round tradeoff. This in turn gives the first protocol for computing the exact Jaccard similarity $\frac{|S \cap T|}{|S \cup T|}$, exact Hamming distance, exact number of distinct elements, and exact 1-rarity and 2-rarity [9].

Our Technique: Our upper bound uses hashing and verification. First consider the following toy protocol: there is a hash function $h : [n] \to [k/\log k]$ that the two players share. For each $i \in [k/\log k]$, the players run a set intersection protocol on $S_i = \{x \in S \mid h(x) = i\}$ and $T_i = \{y \in T \mid h(y) = i\}$. To do so, note that with high probability, simultaneously for all $i \in [k/\log k]$, $|S_i| = O(\log k)$ and $|T_i| = O(\log k)$. Alice and Bob now agree on a hash function $g_i : [n] \to [\log^3 k]$. If Alice sends $g_i(x)$ to Bob for each $x \in S_i$, then Bob can compute $g_i(y)$ for each $y \in T_i$ and check if $g_i(y)$ is in the list of hashed elements that Alice sent. Bob can similary send the $g_i(y)$ values to Alice. Both parties therefore obtain candidate sets $I_A \subseteq S_i$ and $I_B \subseteq T_i$, respectively, for the intersection $S_i \cap T_i$. The communication for a given $i \in [k/\log k]$ is $O(\log k \log\log k)$ and the correctness probability is $1 - \frac{1}{\Omega(\log k)}$. An important observation now is that I_A and I_B contain $S_i \cap T_i$ with probability 1. Therefore, if $I_A = I_B$, then in fact $I_A = I_B = S_i \cap T_i$. By spending an additional $O(\log k)$ bits of communication, Alice and Bob can run an equality test on I_A and I_B, which one should think of as a "verification test", which succeeds with probability $1 - \frac{1}{k^C}$, for an arbitrarily large constant $C > 0$. Whenever the equality test succeeds, Alice and Bob can conclude $I_A = I_B = S_i \cap T_i$, since all such equality tests simultaneously succeed with very high probability. For the values of $i \in [k/\log k]$ for which the corresponding equality test detects that $I_A \neq I_B$, then the players re-run the set intersection protocol on S_i and T_i. The expected number of re-runs for each $i \in [k/\log k]$ is less than 1, and so the overall expected communication is at most $2k/\log k \cdot O(\log k \log\log k) = O(k \log\log k)$, which can be made worst-case by terminating the protocol if it consumes more than a constant factor times its expected communication cost.

To improve the communication futher, we instead hash into k buckets using a hash function h, and build a "verification tree" with these k buckets as the leaves. The tree has r levels, where r is the number of rounds we seek to achieve. For $2 \le h \le r$, the nodes with height h have $\log^{(r-h)} k / \log^{(r-h+1)} k$ children, while the nodes with height 1 (the parents of the leaves) have $\log^{(r-1)} k$ children. For a given $i \in [k]$, define S_i and T_i as before. For each $i \in [k]$, we run a set intersection protocol on S_i and T_i, now with only constant expected communication. For a node with height 1, we have a candidate set intersection for each of its $\log^{(r-1)} k$ children. We concatenate these $\log^{(r-1)} k$ candidate intersections as strings, and verify they are equal with a single equality test. If the equality test succeeds, then we proceed to the next level in the tree. At a node v in a given level of the tree, we perform a single equality test on all candidate intersections of leaves in the subtree $T(v)$ rooted at v. If the equality test fails at v, we re-run the set intersection protocol at all leaves in $T(v)$. By carefully choosing the correctness probabilty of the equality tests run at different levels in the tree, we are able to inductively show the expected communication until the root succeeds is $O(k)$, and the number of rounds is $O(r)$. Detailed description of the protocol and analysis is given in Section 3.3, which gives our main result:

THEOREM 1.1. *For $r > 0$ there exists an $6r$-round communication protocol for INT^k with total expected communication $O(k \log^{(r)} k)$ and success probability $1 - 1/poly(k)$.*

It remains open whether there exists an r-round protocol with communication $O(k \log^{(r)} k)$.

2. DEFINITIONS AND PRELIMINARIES

We will use the following definition of the iterated logarithm functions $\log^{(i)} z$. Let $\log^{(0)} z = z$ and for an integer $i \ge 1$ let $\log^{(i)} z = \log\left(\log^{(i-1)} z\right)$.

Let EQ^n denote the communication problem of solving EQUALITY on binary strings of length n. Let EQ^n_k denote the communication problem, corresponding to k independent instances of EQ^n. Let INT^k denote the communication problem of computing the intersection of two sets $S, T \subseteq [n]$, such that $|S|, |T| \le k$.

A simple reduction from EQ^n_k to INT^k can be given as follows. For an instance $(x^1, \ldots, x^k, y^1, \ldots, y^k)$ of EQ^n_k an instance of INT^k is constructed by creating two sets of pairs $(1, x^1), \ldots (k, x^k)$ and $(1, y_1), \ldots (k, y^k)$. The size of the intersection between these two sets is exactly equal to the number of equal (x^i, y^i) pairs. This fact for DISJ^n_k can be also found in [6].

FACT 2.1 ([6]). *If there exists a protocol Π for INT^k, where the sets are drawn from a universe of size $N \ge k^c$ for $c > 2$ then there exists a protocol Π' for EQ^n_k with the same communication complexity and success probability for $n = \lfloor \log(\frac{N}{k}) \rfloor$.*

We will use the following fact about collision probability of a randomly chosen hash function.

FACT 2.2. *For any set $S \subseteq [n]$ of size $|S| \ge 2$ and $i \ge 0$ a random hash function $h\colon [n] \to [t]$, where $t = O(|S|^{i+2})$ has no collisions with probability at least $1 - 1/|S|^i$, namely for all $x, y \in S$ such that $x \neq y$ it holds that $h(x) \neq h(y)$. Moreover, a random hash function satisfying such guarantee can be constructed using only $O(\log n)$ random bits.*

3. TWO-PARTY SET INTERSECTION

In this section we give upper bounds in both private and public randomness model. In the private random string model, the players do not share a random string, but rather are allowed to be randomized using private randomness. By a result of Newman [19], any problem that can be solved in the common random string model can be solved in the private random string model, adding only $O(\log\log T)$ to the communication complexity, where T is the number of different inputs to the players. One unfortunate aspect of this reduction is that it is non-constructive in the sense that for each input length n, the protocol used is either hardwired an advice string that depends on n, or the players must search for the advice string, which doesn't require communication, but can result in unnecessary computation. We give our upper bounds in the common random string model, but describe how to translate them into constructive protocols in the private random string model, preserving optimality.

We start by describing a simple protocol with linear communication in Section 3.1 and then show how to achieve an optimum round vs. communication trade-off in Section 3.2 and Section 3.3.

3.1 $O(\sqrt{k})$-round protocol

THEOREM 3.1. *There exists an $O(\sqrt{k})$-round constructive randomized protocol for INT^k with success probability $1 - 1/poly(k)$. In the model of shared randomness the total expected communication is $O(k)$ and in the model of private randomness it is $O(k + \log\log n)$*

PROOF. Let $N = k^c$ for a constant $c > 2$. First, the parties pick a random hash function $H\colon [n] \to [N]$, which gives no collisions on the elements in $S \cup T$ with probability at least $1 - 1/\Omega(k^{c-2})$. Thus, for the rest of the analysis we can assume $S, T \subseteq [N]$.

The parties pick a random hash function $h\colon [N] \to [k]$. For a set $U \subseteq [N]$ we use notation $U_i = h^{-1}(i) \cap U$ for the preimage of i in U. Using preimages S_i and T_i the parties construct a collection of instances of EQUALITY, which contains an instance of EQUALITY(s,t) for every $(s,t) \in S_i \times T_i$ for every $i \in [k]$.

Formally, for two sets of instances of a communication problem C, say $C_1 = C(x_1, y_1), \ldots, C(x_i, y_i)$ and $C_2 = C(x'_1, y'_1), \ldots, C(x'_j, y'_j)$ let's denote their concatenation, which corresponds to solving C_1 and C_2 simultaneously as

$$C_1 \sqcup C_2 = (x_1, y_1), \ldots, (x_i, y_i), (x'_1, y'_1), \ldots (x'_j, y'_j).$$

Let's denote as $E_i = \bigsqcup_{(s,t)\in(S_i \times T_i)} \mathrm{Eq}(s,t)$ the collection of instances of equality corresponding to hash value i. The collection of all instances constructed by the parties is $E = \bigsqcup_{i=1}^{k} E_i$.

The expected number of instances $\mathbb{E}[|E|]$ is given as:

$$\mathbb{E}[|E|] = \mathbb{E}\left[\sum_{i=1}^{k} |S_i||T_i|\right] = \sum_{i=1}^{k} \mathbb{E}[|S_i||T_i|]$$

$$\leq \sum_{i=1}^{k} \mathbb{E}[|(S \cup T)_i|^2] = \sum_{i=1}^{k} Var[|(S \cup T)_i|] + \mathbb{E}[|(S \cup T)_i|]^2 \tag{1}$$

Given that for a set Z, the random variable $|Z_i|$ is distributed according to a binomial distribution $B(|Z|, 1/k)$, for each i we have $Var[|\,(S \cup T)_i\,|] \leq 2k \cdot (1/k)(1 - 1/k) \leq 2$ and $\mathbb{E}[|\,(S \cup T)_i\,|] \leq 2$ so $\mathbb{E}[|E|] \leq 6k$.

We use the following result of [12]:

THEOREM 3.2 ([12]). *There exists aconstructive randomized protocol for EQ_k^n with $O(\sqrt{k})$ rounds, which has success probability $2^{-\Omega(\sqrt{k})}$. In the public randomness model the expected total communication is $O(k)$ and in the private randomness model it is $O(k + \log n)$.*

In the shared randomness model the result now follows immediately. In the private randomness model the parties need to construct two random hash functions H and h, using Fact 2.2 with only $O(\log n) + O(\log k) = O(\log n)$ random bits. These bits are exchanged through the channel in the first round of the protocol and are added to the total communication, bringing it down to $O(k + \log n)$. To further reduce the communication we can use the hashing scheme of Fredman, Komlos and Szemeredi [14] as the first step of the protocol. In [14] it is shown that mapping elements $[n]$ by taking a remainder modulo a random prime $q = \tilde{O}(k^2 \log n)$ gives no collisions on a subset of size $O(k)$ with probability $1 - 1/poly(k)$. Applying this result to $S \cup T$ we can reduce the length of strings in the instances of equality down to $O(\log k + \log\log n)$. Thus, we can now specify the pairwise independent hash function using only $O(\log k + \log\log n)$ random bits. See Appendix A.1.1 in [17] for a detailed discussion.

3.2 Auxiliary protocols

We first describe auxiliary protocols BASIC-INTERSECTION (Lemma 3.3) and EQUALITY (Fact 3.5) that we use as building blocks in our main algorithm in Section 3.3. For a two-party communication protocol $\mathcal{P}$ we denote the output of the protocol for the first party as $\mathcal{P}_A(x,y)$ and for the second party as $\mathcal{P}_B(x,y)$.

LEMMA 3.3 (PROTOCOL BASIC-INTERSECTION(S,T)). *There exists a randomized protocol $\mathcal{P}$ (with shared randomness), such that for any $S, T \subset [n]$ and an integer $i \geq 1$, the sets $S' = \mathcal{P}_A(S,T)$ and $T' = \mathcal{P}_B(S,T)$ satisfy the following properties:*

1. *$S' \subseteq S, T' \subseteq T$.*
2. *If $S \cap T = \emptyset$ then $S' \cap T' = \emptyset$ with probability 1.*
3. *If $S \cap T \neq \emptyset$ then $(S \cap T) \subseteq (S' \cap T')$. Also, with probability $1 - 1/N^i$ it holds that $S' = T' = (S \cap T)$.*

The total communication in the protocol is

$$O\left(i \cdot (|S| + |T|)\log(|S| + |T|)\right)$$

and the protocol can be executed in 4 rounds.

Note that Lemma 3.3 guarantees that $S' \cap T'$ is always a superset of the intersection. Also, if the sets S' and T' are equal then each of them is exactly the intersection of S and T.

PROOF. The parties first exchange the sizes of their sets $|S|$ and $|T|$ and determine $m = |S| + |T|$. Using shared randomness they pick a random hash function $h\colon [n] \to [t]$, where $t = \Theta(m^{i+2})$. They exchange sets $h(S)$ and $h(T)$ using total communication $O(i \cdot N \log N)$. The outcome of the protocol is $\mathcal{P}_A(S,T) = h^{-1}(h(T)) \cap S$ and $\mathcal{P}_B(S,T) =$

$h^{-1}(h(S)) \cap T$. Since exchanging the sizes of the sets takes two rounds and another two rounds are required to exchange $h(S)$ and $h(T)$, the total number of rounds of communication is 4.

By construction we have $S' = h^{-1}(h(T)) \cap S \subseteq S$ and similarly $T' \subseteq T$ so the first property holds. If $S \cap T = \emptyset$ then $S' \cap T' = (h^{-1}(h(T)) \cap S) \cap (h^{-1}(h(S)) \cap T) \subseteq (S \cap T) = \emptyset$ and the second property holds. Because $S \subseteq h^{-1}(h(S))$ and $T \subseteq h^{-1}(h(T))$ we have

$$S \cap T \subseteq (h^{-1}(h(T)) \cap S) \cap (h^{-1}(h(S)) \cap T) = S' \cap T',$$

the first part of the third property. Moreover, if the hash function h has no collisions among $S \cup T$ then

$$S' = h^{-1}(h(T)) \cap S = T \cap S$$

and

$$T' = h^{-1}(h(S)) \cap T = S \cap T.$$

The proof is completed using the analysis of collision probability given by Fact 2.2. ∎

We have the following corollary.

Corollary 3.4. *If for the protocol $\mathcal{P}$ in Lemma 3.3 it holds that $\mathcal{P}_A(S,T) = \mathcal{P}_B(S,T)$ then*

$$\mathcal{P}_A(S,T) = \mathcal{P}_B(S,T) = S \cap T.$$

In our main protocol in Section 3.3 we will use an Eq_n test with the following guarantees to verify correctness of the protocol Basic-Intersection. The following guarantee is achieved by a protocol, which uses a random hash function h into k bits.

Fact 3.5. *There exists a randomized (with shared randomness) protocol $\mathcal{P}$ for Eq_n with the following properties.*

1. *If $x = y$ then $\mathcal{P}_A(x,y) = \mathcal{P}_B(x,y) = 1$ with probability 1.*
2. *If $x \neq y$ then $\mathcal{P}_A(x,y) = \mathcal{P}_B(x,y) = 0$ with probability at least $1 - 1/2^k$.*

The total communication in the protocol is $O(k)$ and it can be executed in two rounds.

3.3 Main protocol

Theorem 3.6 (Restatement of Theorem 1.1). *For every integer $r > 0$ there exists an $6r$-round constructive randomized communication protocol (with shared randomness) for INT^k with total expected communication $O(k \log^{(r)} k)$ and success probability $1 - 1/poly(k)$.*

Proof. For $r = 1$ the parties use shared randomness to pick a hash function $h\colon [n] \to [N]$ for $N = k^c$, where $c > 2$. Then each of the parties uses $ck \log k$ bits to exchange $h(S)$ and $h(T)$ respectively. By Fact 2.2 the probability that h has a collision on a set $S \cup T$ is at most $1 - 1/\Theta(k^{c-2})$.

For $r > 1$ consider a tree $\mathcal{T}$ of depth r with the set of nodes at the i-th level for $0 \le i \le r$ denoted as L_i (these are the nodes at distance i from the leaves). Let the degree at the i-th level for $2 \le i \le r$ be equal to $d_i = \log^{(r-i)} k / \log^{(r-i+1)} k$ and the degree at the first level is $d_1 = \log^{(r-1)} k$. Note that this guarantees that the total number of leaves in the tree is k. For a node $v \in \mathcal{T}$, let $c(v)$ denote the set of children of v. For a node $v \in \mathcal{T}$, let $\mathcal{C}(v)$ denote the set of all leaves in the subtree of v. Note that for a node $v \in L_i$ the number of such leaves is $|\mathcal{C}(v)| = \log^{(r-i)} k$.

Definition 3.1 (Set assignment). *A set assignment $\mathcal{A}$ to the leaves of $\mathcal{T}$ is a vector $\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_k)$, consisting of k sets. We say that the set $\mathcal{A}_\ell$ is assigned to a corresponding leaf ℓ in $\mathcal{T}$.*

Every set assignment to the leaves of $\mathcal{T}$ naturally induces a set assignment on all internal vertices of $\mathcal{T}$. Let $\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_k)$ be a set assignment for the leaves of $\mathcal{T}$. For every internal node $v \in \mathcal{T}$ we denote an assignment induced at this vertex by $\mathcal{A}$ as $\mathcal{A}_v = \cup_{i \in \mathcal{C}(v)} \mathcal{A}_i$.

Now we describe the protocol used by the parties. First, Alice and Bob use shared randomness to pick a hash function $h\colon [n] \to [k]$. Using this hash function they define initial assignments of sets S^{-1} and T^{-1} respectively as follows. For a leaf $\ell \in [k]$ of $\mathcal{T}$, let $S^{-1}_\ell = h^{-1}(h(\ell)) \cap S$ and $T^{-1}_\ell = h^{-1}(h(\ell)) \cap T$.

Then the protocol proceeds in r stages. In stage i for $0 \le i < r$ the parties construct new assignments to the leaves of $\mathcal{T}$, which induce new assignments on the internal nodes. We will show that after r stages the parties obtain an assignment to the leaves, such that with high probability the set induced by this assignment in the root of $\mathcal{T}$ is exactly $S \cap T$. We use notation S^i and T^i respectively for the i-th assignment that the parties make to the leaves of the tree. The description of the i-th stage is given as Algorithm 1. This completes the description of the protocol.

```
Input: Sets $S, T \in [k]^k$, assignments $S^{i-1}, T^{i-1}$.

1: For every $v \in L_i$ run the protocol
   EQUALITY$(S^{i-1}_v, T^{i-1}_v)$ with success probability
   $1 - 1/(\log^{(r-i-1)} k)^4$.
2: Let $\mathcal{F}$ be the set of vertices for which the equality
   protocol above returns $S^{i-1}_v \neq T^{i-1}_v$. We call these
   vertices failed.
3: For every $v \in \mathcal{F}$ and every leaf $u \in \mathcal{C}(v)$ run
   BASIC-INTERSECTION$(S^{i-1}_u, T^{i-1}_u)$ with success
   probability $1 - 1/(\log^{(r-i-1)} k)^4$ and assign $S^i_u =$
   $\mathcal{P}_A(S^{i-1}_u, T^{i-1}_u)$ and $T^i_u = \mathcal{P}_B(S^{i-1}_u, T^{i-1}_u)$ respectively.
4: For every $v \notin \mathcal{F}$ and every leaf $u \in \mathcal{C}(v)$ assign
   $S^i_u = S^{i-1}_u$ and $T^i_u = T^{i-1}_u$.
```

Algorithm 1: Protocol for INT^k. Stage i.

In the rest of the proof we first analyze the correctness probability of the protocol above (the key lemma is Lemma 3.7) and then total communication (Lemma 3.10). The proof of Theorem 1.1 is completed by observing that the protocol can be executed in $O(r)$ rounds.

Lemma 3.7. *After stage i for every leaf $u \in \mathcal{T}$ it holds that $S^i_u = T^i_u$ with probability at least $1 - 1/(\log^{(r-i-1)} k)^4$, taken over all the randomness of the protocol.*

Proof. If u is in the subtree of a node v, which is not *failed* at level i then we know that $S_v = T_v$ and thus $S_u = T_u$ for each $u \in \mathcal{C}(v)$ with probability at least $1 - 1/(\log^{(r-i-1)} k)^4$ by the guarantee of the Equality(S_v, T_v) test. Otherwise, u is in the subtree of a failed node v at level i. In this case the claim follows because we run Basic-Intersection protocol for this leaf with success probability at least $1 - 1/(\log^{(r-i-1)} k)^4$. ∎

We call a node $v \in L_i$ *correct* if after stage i it holds that $S^i_v = T^i_v$.

COROLLARY 3.8. *Every node $v \in L_i$ is correct with probability at least $1 - 1/(\log^{(r-i-1)} k)^3$. In particular, the root of the tree is correct with probability at least $1 - 1/k^3$.*

PROOF. From Lemma 3.7 applied to the level i it follows that after the execution of stage i for every leaf $u \in \mathcal{C}(v)$ it holds that $S_u^i = T_u^i$ with probability at least $1 - 1/(\log^{(r-i-1)} k)^4$. Hence, by a union bound over all $\log^{(r-i)} k$ such leaves with probability at least

$$1 - \log^{(r-i)} k/(\log^{(r-i-1)} k)^4 \geq 1 - 1/(\log^{(r-i-1)} k)^3$$

we have $S_v^i = T_v^i$. ▮

The correctness proof of the protocol now follows from Corollary 3.8 together with the following invariant applied to the root of the tree after round $r-1$.

PROPOSITION 3.9. *If for a node $v \in \mathcal{T}$ Alice and Bob assign S_v^i and T_v^i to it respectively then if $S_v^i = T_v^i$ then $S_v^i = T_v^i = S_v \cap T_v$.*

PROOF. Note that this invariant is maintained by BASIC-INTERSECTION (Corollary 3.4). During the execution of the protocol the sets S_v' and T_v' only change when we apply BASIC-INTERSECTION to the leaves in $\mathcal{T}$. Clearly, if the invariant is maintained for all leaves then it is also maintained for all internal nodes as well. ▮

Now we analyze the total communication in the protocol. For a leaf $u \in \mathcal{T}$ let n_u denote the expected number of times the BASIC-INTERSECTION protocol was run on the sets assigned to u.

LEMMA 3.10. *For every leaf $u \in \mathcal{T}$ it holds that $\mathbb{E}[n_u] = O(1)$.*

PROOF. For a leaf u let's denote it's unique predecessor in level i as $p_i(u)$. Formally, $p_i(u) = v$ if and only if $v \in L_i$ and u is in the subtree of v. We can express $\mathbb{E}[n_u]$ as:

$$\mathbb{E}[n_u] = \sum_{i=0}^{r-1} \Pr[p_i(u) \text{ is failed}] \cdot (4 \log^{(r-i)} k)$$

$$\leq \sum_{i=0}^{r-1} d_i \cdot \Pr\left[v \text{ is an incorrect child of } p_i(u)\right] (4 \log^{(r-i)} k),$$

$$\leq \sum_{i=0}^{r-1} \frac{\log^{(r-i)} k}{\log^{(r-i+1)} k} \cdot \frac{1}{(\log^{(r-i)} k)^3} \cdot (4 \log^{(r-i)} k) = O(1)$$

where the first inequality holds by a union bound and the second by Corollary 3.8. ▮

The total expected communication in the protocol can be expressed as the sum of the total communication for EQUALITY and BASIC-INTERSECTION. The total communication for EQUALITY is:

$$\sum_{i=0}^{r-1} |L_i| (4 \log^{(r-i)} k)$$
$$= O(k \log^{(r)} k) + \sum_{i=1}^{r-1} (k/\log^{(r-i)} k) \cdot (4 \log^{(r-i)} k)$$
$$= O(k \log^{(r)} k) + O(rk)$$
$$= O(k \log^{(r)} k).$$

The expected total communication for BASIC-INTERSECTION is by Lemma 3.3 equal to:

$$\mathbb{E}\left[\sum_{i=1}^{k} (|S_i| + |T_i|) \log(|S_i| + |T_i|) \cdot n_i\right] =$$
$$\sum_{i=1}^{k} \mathbb{E}\left[(|S_i| + |T_i|) \log(|S_i| + |T_i|)\right] \mathbb{E}[n_i],$$

where the equality follows from the independence of the random variables. Because for every i we have $\mathbb{E}[n_i] = O(1)$ by Lemma 3.10, to complete the proof it is sufficient to show that $\mathbb{E}[(|S_i| + |T_i|) \log(|S_i| + |T_i|)] = O(1)$ and thus the total communication for BASIC-INTERSECTION is $O(k)$. We have $\mathbb{E}[(|S_i| + |T_i|) \log(|S_i| + |T_i|)] \leq \mathbb{E}[(|S_i| + |T_i|)^2]$, where the right-hand side is constant by the same argument as used to bound each term in (1). Finally, the bound on the number of rounds of communication follows from the fact the communication in each of the r stages for the EQUALITY tests can be done in parallel in two rounds (Fact 3.5). After in four more rounds we can perform all BASIC-INTERSECTION protocols in parallel (Lemma 3.3). This gives $6r$ rounds of communication. ▮

4. MULTI-PARTY SET INTERSECTION IN THE MESSAGE PASSING MODEL

In the multi-party case we have m players, each holding a set $S_i \subseteq [n]$ such that $|S_i| \leq k$. The goal of the parties is to output a set $S = \bigcap_{i=1}^m S_i$. We allow arbitrary communication between the parties (i.e. any player i can send a message to any player j). In each round of the protocol the parties first perform some local computation and then can exchange messages. This is known as the message passing model (see e.g. [4]). We consider two optimization goals: minimizing the total communication (or equivalently average communication per player) and minimizing the worst-case communication per player. In both cases we keep the number of rounds as small as possible.

First, observe that we can amplify the success probability of the two-party protocol in Theorem 1.1 to be $1 - 1/2^k$ while keeping the expected total communication $O(k \log^{(r)} k)$ and only incurring a penalty in the number of rounds: the protocol will have expected $O(r)$ rounds instead of worst-case $6r$ rounds. This follows by repeating the protocol if it hasn't succeeded. The latter condition can be checked by exchanging k-bit equality checks after the protocol terminates. With a total of $O(1)$ expected repetitions this gives expected $O(r)$ number of rounds and success probability which is only limited by the equality checks and is thus $1 - 1/2^k$ by Fact 3.5.

Using this observation we obtain a protocol with the following guarantee for the average-case multi-party setting.

COROLLARY 4.1. *(Average-case) For every $r > 0$ there exists a multi-party protocol in the message passing model with expected average communication per player $O(k \log^{(r)} k)$, expected number of rounds $O\left(r \cdot \max(1, \frac{\log m}{k})\right)$ and error probability $1 - 1/2^k$.*

PROOF. First, the set of m players is partitioned into groups of size at most 2^k. Consider one such group, which consists of players holding sets $S_1, \ldots, S_{2^k}$. The player holding S_1 is chosen as a coordinator. Within the group all players execute the modified version of the two-party protocol described above with the coordinator, who computes sets $T_i = S_1 \cap S_i$ for each $2 \le i \le 2^k$. This step is repeated until the coordinator succeeds in verifying that $\bigcap_{i=2}^{2^k} T_i = \bigcap_{i=1}^{2^k} S_i$ with probability at least $1 - 1/2^k$. This is done by using a $2k$-bit equality check with each of the players. By Fact 3.5 the equality check succeeds with probability $1 - 1/2^{2k}$ and hence by a union bound over the 2^k players in the group the desired success probability follows. Once all $m' = \lceil m/2^k \rceil$ coordinators succeed in verifying their sets the protocol is executed recursively among them for their respective sets.

The number of active players decreases exponentially between the levels and thus the total communication is dominated by the first level. The first level has average complexity $O(k \log^{(r)} k)$ per player and expected $O(r)$ rounds using the same reasoning as for the case of two-parties discussed above. The total number of levels of recursion is $\max(1, \log_{2^k} m) = \max\left(1, \frac{\log m}{k}\right)$, which gives the claimed bound on the total number of rounds. ∎

Taking $r = \log^* k$ in Corollary 4.1 we get average communication $O(k)$ per player, which matches the lower bounds of [23, 4] who show that average communication $\Omega(k)$ is necessary for solving SET INTERSECTION and SET DISJOINTNESS in the message passing model.

In the protocol from Corollary 4.1 every coordinator has to perform $O(2^k k \log^{(r)} k)$ communication per level. By increasing the number of rounds we can amortize this cost among the players.

COROLLARY 4.2. *(Worst-case) For every $r > 0$ there exists a multi-party protocol in the message passing model with worst-case communication $O\left(k^2 \log^{(r)} k \cdot \max\left(1, \frac{\log m}{k}\right)\right)$ per player, expected number of rounds $O\left(r \cdot k \cdot \max(1, \frac{\log m}{k})\right)$ and error probability $1 - 1/2^k$.*

PROOF. The protocol is executed recursively in $\max\left(1, \frac{\log m}{k}\right)$ levels and in each level the players are assigned to groups of size at most 2^k as in Corollary 4.1. Consider one such group. Instead of using a coordinator in each level the players are assigned to the leaves of a complete binary tree of depth k. They run the two-party protocol recursively in pairs. This gives expected number of rounds $O(rk)$ per level and the bound on the number of rounds follows. When the two-party protocol is executed for the top two nodes in the tree (the children of the root) the parties also perform a k-bit equality check in order to certify the correctness of the result with probability $1 - 1/2^k$. If this check fails then the entire computation in the tree is repeated, which gives $O(1)$ repetitions in expectation using the same reasoning as before. Finally, adding up over all nodes on a path of length k the worst-case communication per level is $O(k^2 \log^{(r)} k)$ which gives the bound on the desired worst-case communication per player. ∎

5. REFERENCES

[1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

[2] K. D. Ba, P. Indyk, E. Price, and D. P. Woodruff. Lower bounds for sparse recovery. In *SODA*, pages 1190–1197, 2010.

[3] Z. Bar-Yossef, T. S. Jayram, R. Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.*, 68(4):702–732, 2004.

[4] M. Braverman, F. Ellen, R. Oshman, T. Pitassi, and V. Vaikuntanathan. A tight bound for set disjointness in the message-passing model. In *FOCS*, pages 668–677, 2013.

[5] M. Braverman and A. Rao. Information equals amortized communication. In *FOCS*, pages 748–757, 2011.

[6] J. Brody, A. Chakrabarti, R. Kondapally, D. P. Woodruff, and G. Yaroslavtsev. Certifying equality with limited interaction. *Manuscript.*

[7] H. Buhrman, D. García-Soriano, A. Matsliah, and R. de Wolf. The non-adaptive query complexity of testing k-parities. *CoRR*, abs/1209.3849, 2012.

[8] A. Dasgupta, R. Kumar, and D. Sivakumar. Sparse and lopsided set disjointness via information theory. In *APPROX-RANDOM*, pages 517–528, 2012.

[9] M. Datar and S. Muthukrishnan. Estimating rarity and similarity over data stream windows. In *ESA*, pages 323–334, 2002.

[10] B. Ding and A. C. König. Fast set intersection in memory. *PVLDB*, 4(4):255–266, 2011.

[11] B. Ding, H. Wang, R. Jin, J. Han, and Z. Wang. Optimizing index for taxonomy keyword search. In *SIGMOD Conference*, pages 493–504, 2012.

[12] T. Feder, E. Kushilevitz, M. Naor, and N. Nisan. Amortized communication complexity. *SIAM J. Comput.*, 24(4):736–750, 1995.

[13] M. Fontoura, M. Gurevich, V. Josifovski, and S. Vassilvitskii. Efficiently encoding term co-occurrences in inverted indexes. In *CIKM*, pages 307–316, 2011.

[14] M. L. Fredman, J. Komlós, and E. Szemerédi. Storing a sparse table with 0(1) worst case access time. *J. ACM*, 31(3):538–544, 1984.

[15] J. Håstad and A. Wigderson. The randomized communication complexity of set disjointness. *Theory of Computing*, 3(1):211–219, 2007.

[16] B. Kalyanasundaram and G. Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Discrete Math.*, 5(4):545–557, 1992.

[17] D. M. Kane, J. Nelson, and D. P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *SODA*, pages 1161–1178, 2010.

[18] E. Kushilevitz and N. Nisan. *Communication complexity.* Cambridge University Press, 1997.

[19] I. Newman. Private vs. common random bits in communication complexity. *Inf. Process. Lett.*, 39(2):67–71, 1991.

[20] N. Nisan and I. Segal. The communication requirements of efficient allocations and supporting prices. *J. Economic Theory*, 129(1):192–224, 2006.

[21] R. Pagh, M. Stockel, and D. P. Woodruff. Is min-wise hashing optimal for summarizing set intersection? In *PODS*, 2014.

[22] I. Parnafes, R. Raz, and A. Wigderson. Direct product results and the gcd problem, in old and new communication models. In *STOC*, pages 363–372, 1997.

[23] J. M. Phillips, E. Verbin, and Q. Zhang. Lower bounds for number-in-hand multiparty communication complexity, made easy. In *SODA*, pages 486–501, 2012.

[24] R. Raz and A. Wigderson. Monotone circuits for matching require linear depth. *J. ACM*, 39(3):736–744, 1992.

[25] A. A. Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992.

[26] M. Saglam and G. Tardos. On the communication complexity of sparse set disjointness and exists-equal problems. In *FOCS*, pages 678–687, 2013.

[27] A. C.-C. Yao. Some complexity questions related to distributive computing (preliminary report). In *STOC*, pages 209–213, 1979.

[28] J. Zhou, Z. Bao, W. Wang, T. W. Ling, Z. Chen, X. Lin, and J. Guo. Fast slca and elca computation for xml keyword queries based on set intersection. In *ICDE*, pages 905–916, 2012.