Interactive information and coding theory

Mark Braverman*

Abstract. We give a high-level overview of recent developments in interactive information and coding theory. These include developments involving interactive noiseless coding and interactive error-correction.

The overview is primarily focused on developments related to complexity-theoretic applications, although the broader context and agenda are also set out. As the present paper is an extended abstract, the vast majority of proofs and technical details are omitted, and can be found in the respective publications and preprints.

Mathematics Subject Classification (2010). Primary 94A15; Secondary 68Q99.

Keywords. Coding theory, communication complexity, information complexity, interactive computation.

1. Introduction

1.1. A high-level overview of information and coding theory.

We begin with a very high-level overview of information and coding theory. This is an enormous field of study, with subareas dealing with questions ranging from foundations of probability and statistics to applied wireless transmission systems. We will focus only on some of the very basic foundational aspects, which were set forth by Shannon in the late 1940s, or shortly after. The goal will be to try and translate those to interactive communication settings, of the type that is used in theoretical computer science. This program is only very partially complete, but some of the early results are promising. While our overview of information and coding theory in this section focuses on fairly simple facts, we present those in some detail nonetheless, as they will be used as a scaffold for the interactive coding discussion. A thorough introduction into modern information theory is given in [15].

Noiseless coding. Classical information theory studies the setting where one terminal (Alice) wants to transmit information over a channel to another terminal (Bob). Two of the most important original contributions by Shannon are the

*Work supported in part by an NSF CAREER award (CCF-1149888), NSF CCF-0832797, NSF CCF-1215990, a Turing Centenary Fellowship, and a Packard Fellowship in Science and Engineering.

Noiseless Coding (or Source Coding) Theorem and the *Noisy Coding* (or Channel Coding) Theorem. The Source Coding Theorem asserts that the cost of Alice transmitting n i.i.d. copies of a discrete random variable X to Bob *over a noiseless channel* scales as Shannon's entropy $H(X)$ as $n \to \infty$:

$$H(X) = \sum_{x \in \mathrm{supp}(X)} \Pr[X = x] \log \frac{1}{\Pr[X = x]}. \quad (1)$$

If we denote by X^n the concatenation of n independent samples from X, and by $C(Y)$ the (expected) number of bits needed for Alice to transmit a sample of random variable Y to Bob, then the Source Coding Theorem asserts that

$$\lim_{n \to \infty} \frac{C(X^n)}{n} = H(X). \quad (2)$$

This fact can be viewed as the operational definition of entropy, i.e. one that is grounded in reality. Whereas definition (1) may appear artificial, (2) implies that it is the right one, since it connects to the "natural" quantity $C(X^n)$. Another indirect piece of evidence indicating that $H(X)$ is a natural quantity is its additivity property:

$$H(X^n) = n \cdot H(X), \quad (3)$$

and more generally, if XY is the concatenation of random variables X and Y, then $H(XY) = H(X) + H(Y)$ whenever X and Y are independent. Note that it is not hard to see that (3) fails to hold for $C(X)$, making $H(X)$ a "nicer" quantity to deal with than $C(X)$. Huffman coding (11) below blurs the distinction between the two, as they only differ by at most one additive bit, but we will return to it later in the analogous distinction between communication complexity and information complexity.

Noisy coding. So far we assumed a noiseless channel — bits sent over the channel by Alice are received by Bob unaltered. If the channel is *noisy*, that is, messages sent over the channel may get corrupted, then clearly some redundancy in transmission is necessary. Abstractly, the task of *coding* is the task of converting the message being sent into symbols to be transmitted over the channel, in a way that allows the original message to be recovered from what has been transmitted by the channel. The important considerations for how good a code is are the type (and amount) of errors it can withstand and still accomplish the transmission successfully, and the rate by which the error-correcting encoding enlarges the message being transmitted.

Shannon's Noisy-Channel Coding Theorem was first to address the noisy coding scenario theoretically. The most important insight from that theorem is that, at least in the limit, the ability of a channel to conduct information — defined

All logs in this paper are base-2.

In fact, Shannon's Source Coding Theorem asserts that due to concentration the *worst case* communication cost scales as $H(X)$ as well, if we allow negligible error. We ignore this stronger statement at the present level of abstraction.

formally as Shannon's channel capacity — can be decoupled from the content being transmitted over the channel. Informally, for a memoryless channel $\mathcal{C}$ one can define its capacity $\mathsf{cap}(\mathcal{C})$ as "how many bits of information is one utilization of $\mathcal{C}$ (i.e. one transmission over $\mathcal{C}$) worth?". For any X, if we denote by $C_{\mathcal{C}}(X^n)$ the expected number of utilizations of channel $\mathcal{C}$ needed to transmit n independent samples of X (except with negligible error), then

$$\lim_{n\to\infty} \frac{C_{\mathcal{C}}(X^n)}{n} = \frac{H(X)}{\mathsf{cap}(\mathcal{C})}. \tag{4}$$

This means, conveniently, that one can study properties of channels separately from properties of what is being transmitted over the channels. The information-theoretic quantities needed to express $\mathsf{cap}(\mathcal{C})$ are *conditional entropy* and *mutual information*. While these are standard basic notions in information theory, we will define them here, to keep the exposition accessible. For a pair of random variables X and Y, the conditional entropy $H(X|Y)$ can be thought of as the amount of uncertainty remaining in X for someone who knows Y:

$$H(X|Y) := H(XY) - H(Y) = \mathbf{E}_{y\sim Y} H(X|Y = y). \tag{5}$$

In the extreme case where X and Y are independent, we have $H(X|Y) = H(X)$. In the other extreme, when $X = Y$, we have $H(X|X) = 0$. The *mutual information* $I(X;Y)$ between two variables X and Y measures the amount of information revealing Y reveals about X, i.e. the reduction in X's entropy as a result of conditioning on Y. Thus

$$I(X;Y) = H(X) - H(X|Y) = H(X) + H(Y) - H(XY) = I(Y;X). \tag{6}$$

Conditional mutual information is defined similarly to conditional entropy:

$$I(X;Y|Z) := H(X|Z) - H(X|YZ) = I(Y;X|Z). \tag{7}$$

A very important property of conditional mutual information is the *chain rule*:

$$I(XY;Z|W) = I(X;Z|W) + I(Y;Z|WX) = I(Y;Z|W) + I(X;Z|WY). \tag{8}$$

An informal interpretation of (8) is that XY reveal about Z what X reveals about Z, plus what Y reveals about Z to someone who already knows X.

Abstractly, a memoryless channel (i.e. one where each utilization of the channel is independent of other utilization) can be viewed as a set of pairs of variables $(X, \mathcal{C}(X))$ where X is the signal the sender inputs to the channel, and $\mathcal{C}(X)$ is the output of the channel received on input X from the sender. If the channel is noiseless then $\mathcal{C}(X) = X$. Under this notation, the channel capacity of $\mathcal{C}$ is equal to

$$\mathsf{cap}(\mathcal{C}) = \sup_Y I(Y;\mathcal{C}(Y)). \tag{9}$$

In other words, it is the supremum over all input distributions of the amount of information preserved by the channel. The scenario just discussed is obviously

a very simple one, but even in more elaborate settings issues surrounding coding transmissions over a noisy channel (at least when the noise is random) are very well understood. For example, for the *binary symmetric channel* BSC_ε that accepts bits $b \in \{0,1\}$ and outputs b with probability $1-\varepsilon$ and $\bar{b}$ with probability ε, the capacity is

$$\mathsf{cap}(BSC_\varepsilon) = 1 - H(\varepsilon) := 1 - (\varepsilon \log 1/\varepsilon + (1-\varepsilon)\log 1/(1-\varepsilon)). \quad (10)$$

One caveat is that mathematically striking characterizations such as above only become possible *in the limit*, where the size of the message we are trying to transmit over the channel — i.e. the block-length — grows to infinity. What happens for fixed block lengths, which we discuss next, is of course important for both practical and theoretical reasons, and it will be even more so in the interactive regime.

For noiseless coding in the one-way regime, it turns out that while $H(X)$ does not exactly equal the expected number of bits $C(X)$ needed to transmit a *single* sample from X, it is very close to it. For example, the classical Huffman's coding [25] implies that

$$H(X) \leq C(X) < H(X) + 1, \quad (11)$$

where the "hard" direction of (11) is the upper bound. The upper bound showing that $C(X) < H(X) + 1$ is a *compression result*, showing how encode a message with low average information content (i.e. entropy) into a message with a low communication cost (i.e. number of bits in the transmission). Note that this result is much less "clean" than the limit result (2): in the amortized case the equality is exact, while in the one-shot case a gap is created. This gap is inevitable, if only for integrality reasons, but as we will see later, it becomes crucial in the interactive case.

Adversarial noise and list-decoding. So far we only discussed channels affected by randomized errors. A variant of the noisy regime where the situation appears mathematically much less clear is one where the errors on the channel are introduced *adversarially*. For example, an adversarial ε-error rate binary channel receives a string $S \in \{0,1\}^n$ of n bits, and outputs a string S' such that the Hamming distance $d_H(S,S') < \varepsilon n$, i.e. S' differs from S in at most an ε-fraction of positions. A coding scheme for this setting is a pair of encoding and decoding functions $E : \{0,1\}^m \to \{0,1\}^n$ and $D : \{0,1\}^n \to \{0,1\}^m$, respectively, such that for each $X \in \{0,1\}^m$ and each S' with $d_H(E(X),S') < \varepsilon n$, X is recovered correctly from S', i.e. $D(S') = X$. Clearly we want m to be as large as possible as a function of n. It turns out that such an encoding scheme is possible with $m = \Omega_\varepsilon(n)$ for each $\varepsilon < 1/4$ (and $\varepsilon < 1/2$ if the binary alphabet is replaced with an alphabet Σ of size $|\Sigma| = O_\varepsilon(1)$). Unlike the random-noise case the exact optimal *rate* of the code, i.e. the largest achievable ratio of $\frac{m}{n}$ is unknown for the adversarial model. Clearly, the limit cannot exceed $\mathsf{cap}(BSC_\varepsilon)$, but it is bound to be lower, since correcting adversarial errors is much harder than randomized ones. *A priori* it is not even obvious that the adversarial channel capacity is a positive

The $O_\varepsilon(1)$ notation means a function that is bounded by a constant for each fixed ε.

constant when $\varepsilon < 1/4$. Despite much work in the field [45, 38], even the basic binary channel capacity problem remains open, with a notorious gap between the Gilbert-Varshanov lower bound, and the Linear Programming upper bound [44].

A clear limitation of any error-correcting code, even over a large constant-size alphabet Σ, is that no decoding is possible when $\varepsilon \geq 1/2$: for two valid codewords X_1, X_2 and any encoding function E, there is a string S' such that $d_H(E(X_1), S') \leq n/2$ and $d_H(E(X_2), S') \leq n/2$, making decoding S' an impossible task (note that over a large constant-size alphabet, with a high probability one can recover from random errors of rate exceeding $1/2$). It turns out, however, that for any $\varepsilon < 1$, for $|\Sigma| = O_\varepsilon(1)$, it is possible to come up with a constant-rate *list-decoding* scheme: one where the decoding function $D(S')$ outputs a list of size $s = O_\varepsilon(1)$ of possible $X_1, \ldots, X_s$ such that these are the only possible X's satisfying $d_H(E(X), S') < (1 - \varepsilon)n$. List decodable codes, first introduced in the 1950s [16, 47] have played an important role in a number of areas of theoretical computer science, a partial survey of which can be found in [23, 24].

1.2. Interactive computation models in complexity theory. In theoretical computer science interactive communication models are studied within the area of *communication complexity*. While communication complexity can be viewed as a direct extension of the study of non-interactive communication models which were discussed in the previous section, its development has been largely disjoint from the development of information theory, and the areas have not reconnected until fairly recently. This may be partially explained by the combinatorial nature of the tools most prevalent in theoretical computer science.

Communication complexity was introduced by Yao in [48], and is the subject of the text [30]. It has found numerous applications for unconditional lower bounds in a variety of models of computation, including Turing machines, streaming, sketching, data structure lower bounds, and VLSI layout, to name a few. In the basic (two-party) setup, the two parties Alice and Bob are given inputs $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, respectively, and are required to compute a function $F(X, Y)$ of these inputs (i.e. both parties should know the answer in the end of the communication), while communicating over a noiseless binary channel. The parties are computationally unbounded, and their only goal is to minimize the number of bits transmitted in the process of computing $F(X, Y)$. In a typical setup F is a function $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. Examples of functions commonly discussed and used include the Equality function $EQ_n(X, Y) := \mathbf{1}_{X=Y}(X, Y)$, and the Disjointness function

$$Disj_n(X, Y) := \bigwedge_{i=1}^{n} (\neg X_i \vee \neg Y_i). \tag{12}$$

We will return to these functions later in our discussion.

Of course, the (non-interactive) transmission problem can be viewed as just a special case of computing the function $P_x : \mathcal{X} \times \{\perp\} \to \mathcal{X}$, which maps $(X, \perp)$ to X. However, there are two important distinctions between the "flavors" of typical information theory results and communication complexity. Firstly, information

theory is often concerned with coding results where block length — i.e. the number of copies of the communication task to be performed — goes to infinity. Recall, for example, that Shannon's Source Coding Theorem (2) gave Shannon's entropy as a closed-form expression for the amortized transmission cost of sending a growing number of samples X (this is often but not always the case, for example, the Huffman coding (11) result is not of this type). On the other hand, communication complexity more commonly studies the communication cost of computing a single copy of F. Secondly, as in the examples above, communication complexity often studies functions whose output is only a single bit or a small number of bits, thus "counting style" direct lower bound proofs rarely apply. Tools that have been successfully applied in communication complexity over the years include combinatorics, linear algebra, discrepancy theory, and only later classical information theory.

To make our discussion of communication complexity more technical, we will focus on the two-party setting. We briefly discuss the multi-party setting, which also has many important applications, but is generally much less well-understood, in the last section of the paper. The basic notion in communication complexity is that of a *communication protocol*. A communication protocol over a binary channel formalizes a conversation, where each message only depends on the input to the speaker and the conversation so far:

Definition 1.1. A (deterministic) protocol π for $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ is defined as a finite rooted binary tree, whose nodes correspond to partial communication transcripts, such that the two edges coming out of each vertex are labeled with a 0 and 1. Each leaf ℓ is labeled by an output value $f_\ell \in \{0,1\}$. Each internal node v is labeled by a player's name and either by a function $a_v : \mathcal{X} \to \{0,1\}$ or $b_v : \mathcal{Y} \to \{0,1\}$ corresponding to the next message of Alice or Bob, respectively.

The protocol $\pi(X,Y)$ is executed on a pair of inputs (X,Y) by starting from the root of the tree. At each internal node labeled by a_v the protocol follows the child $a_v(X)$ (corresponding to Alice sending a message), and similarly at each internal node labeled by b_v the protocol follows $b_v(Y)$. When a leaf ℓ is reached the protocol outputs f_ℓ.

By a slight abuse of notation, $\pi(X,Y)$ will denote both the transcript and the output of the protocol; which is the case will be clear from the context. The communication cost of a protocol is the depth of the corresponding protocol tree. A protocol *succeeds* on input (X,Y) if $\pi(X,Y) = F(X,Y)$. Its communication cost on this pair of inputs is the depth of the leaf reached by the execution. The *communication complexity* $CC(F)$ of a function F is the lowest attainable communication cost of a protocol that successfully computes F. In the case of deterministic communication we require the protocol to succeed on all inputs.

A deterministic communication protocol π induces a partition of the input space $\mathcal{X} \times \mathcal{Y}$ into sets S_ℓ by the leaf ℓ that $\pi(X,Y)$ reaches. Since at each step the next move of the protocol depends only on either X or Y alone, each S_ℓ is a combinatorial rectangle of the form $S_\ell = S_\ell^{\mathcal{X}} \times S_\ell^{\mathcal{Y}}$. This key combinatorial property is at the heart of many combinatorial communication complexity lower

bounds. To give an example of such a simple combinatorial proof, consider the *rank* bound. Let $N = |\mathcal{X}|$, $M = |\mathcal{Y}|$, and consider the $N \times M$ matrix M_F over $\mathbb{R}$ whose (X,Y)-th entry is $F(X,Y)$. Each protocol π with leaf set $\mathcal{L}$ of size L, induces a partition of $\mathcal{X} \times \mathcal{Y}$ into combinatorial rectangles $\{S_\ell\}_{\ell \in \mathcal{L}}$. Let M_ℓ be the matrix whose entries are equal to $M_{X,Y}$ for $(X,Y) \in S_\ell$ and are 0 elsewhere. Since $\{S_\ell\}_{\ell \in \mathcal{L}}$ is a partition of $\mathcal{X} \times \mathcal{Y}$, we have $M_F = \sum_{\ell \in \mathcal{L}} M_\ell$. Assuming π is always correct, each M_ℓ is *monochromatic*, i.e. either all-0, or all-1 on S_ℓ, depending on the value of f_ℓ. Thus, $\operatorname{rank}(M_\ell) \leq 1$, and

$$\operatorname{rank}(M_F) \leq \sum_{\ell \in \mathcal{L}} \operatorname{rank}(M_\ell) \leq L. \tag{13}$$

In fact, a stronger bound of $L-1$ holds unless M_F is the trivial all-1 matrix. Thus any protocol computing F must have communication cost of at least $\log(\operatorname{rank}(M_F)+1)$, and it follows that the communication complexity of F is at least $\log(\operatorname{rank}(M_F)+1)$. As an example of an application, if $F = EQ_n$ is the Equality function, then $M_{EQ_n} = I_{2^n}$ is the identity matrix, and thus $CC(EQ_n) \geq n+1$. In other words, the trivial protocol where Alice sends Bob her input X (n bits), and Bob responds whether $X = Y$ (1 bit), is optimal.

As in many other areas of theoretical computer science, there is much to be gained from randomization. For example, in practice, the Equality function does not require linear communication as Alice and Bob can just hash their inputs and compare the hash keys. The shorter protocol may return a false positive, but it is correct with high probability, and reduces the communication complexity from $n+1$ to $O(\log n)$.

More generally, a randomized protocol is a protocol that tosses coins (i.e. accesses random bits), and produces the correct answer with high probability. The *distributional setting*, where there is a prior probability distribution μ on the inputs and the players need to output the correct answer with high probability with respect to μ is closely related to the randomized setting, as will be seen below. In the randomized setting there are two possible types of random coins. *Public coins* are generated at random and are accessible to both Alice and Bob at no communication cost. *Private coins* are coins generated privately by Alice and Bob, and are only accessible by the player who generated them. If Alice wants to share her coins with Bob, she needs to use the communication channel. In the context of communication complexity the pubic-coin model is clearly more powerful than the private coin one. Fortunately, the gap between the two is not very large [35], and can be mostly ignored. For convenience reasons, we will focus on the public-coin model.

The definition of a randomized public-coin communication protocol π_R is identical to Definition 1.1, except a public random string R is chosen at the beginning of the execution of the randomized π_R, and all functions at the nodes of π_R may depend on R in addition to the respective input X or Y. We still require the answer f_ℓ to be unequivocally determined by the leaf ℓ alone. The communication cost $|\pi_R|$ of π_R is still its worst-case communication cost (for historic reasons; an average-case notion would also have been meaningful to discuss here).

The randomized communication complexity of F with error $\varepsilon > 0$ is given by

$$R_\varepsilon(F) := \min_{\pi_R : \forall X,Y \Pr_R[\pi_R(X,Y)=F(X,Y)] \geq 1-\varepsilon} |\pi_R|. \quad (14)$$

For a distribution μ on $\mathcal{X} \times \mathcal{Y}$ the *distributional* communication complexity $D_{\mu,\varepsilon}(F)$ is defined as the cost of the best protocol that achieves expected error ε with respect to μ. Note that in this case fixing public randomness R to a uniformly random value does not change (on average) the expected success probability of π_R with respect to μ. Therefore, without loss of generality, we may require π to be deterministic:

$$D_{\mu,\varepsilon}(F) := \min_{\pi : \mu\{X,Y:\ \pi(X,Y)=F(X,Y)\} \geq 1-\varepsilon} |\pi|. \quad (15)$$

It is easy to see that for all μ, $D_{\mu,\varepsilon}(F) \leq R_\varepsilon(F)$. By an elegant minimax argument [49], a partial converse is also true: for each F and ε, there is a distribution against which the distributional communication complexity is as high as the randomized:

$$R_\varepsilon(F) = \max_\mu D_{\mu,\varepsilon}(F). \quad (16)$$

For this reason, we will be able to discuss distributional and randomized communication complexity interchangeably.

How can one prove lower bounds for the randomized setting? This setting is much less restrictive than the deterministic one, making lower bounds more challenging. Given a function F, one can guess the hard distribution μ, and then try to lower bound the distributional communication complexity $D_{\mu,\varepsilon}(F)$ — that is, show that there is no low-communication protocol π that computes F with error $\leq \varepsilon$ with respect to μ. Such a protocol π of cost $k = |\pi|$ still induces a partition $\{S_\ell\}_{\ell \in \mathcal{L}}$ of the inputs according to the leaf they reach, with $L \leq 2^k$ and each S_ℓ a combinatorial rectangle. However, it is no longer the case that when we consider the corresponding submatrix M_ℓ of M_F it must be monochromatic — the output of π is allowed to be wrong on a fraction of S_ℓ, and thus for some inputs the output of π on S_ℓ may disagree with the value of F. Still, it should be true that for *most* leaves the value of F on S_ℓ is strongly biased one way or the other, since the contribution of S_ℓ to the error is

$$e(S_\ell) = \min\left(\mu(S_\ell \cap F^{-1}(0)), \mu(S_\ell \cap F^{-1}(1))\right). \quad (17)$$

In particular, a fruitful lower bound strategy is to show that all "large" rectangles with respect to μ have $e(S_\ell)/\mu(S_\ell) \gg \varepsilon$, and thus there must be many smaller rectangles — giving a lower bound on $L \leq 2^{|\pi|}$. One simple instantiation of this strategy is the *discrepancy* bound: for a distribution μ, the discrepancy $Disc_\mu(F)$ of F with respect to μ is the maximum over all combinatorial rectangles R of

$$Disc_\mu(R,F) := |\mu(F^{-1}(0) \cap R) - \mu(F^{-1}(1) \cap R)|.$$

In other words, if F has low discrepancy with respect to μ then only very small rectangles (as measured by μ) can be unbalanced. With some calculations, it can be shown that for all $\varepsilon > 0$ (see [30] and references therein),

$$D_{\mu,\frac{1}{2}-\varepsilon}(F) \geq \log_2(2\varepsilon/Disc_\mu(F)). \quad (18)$$

Note that (18) not only says that if the discrepancy is low then the communication complexity is high, but also that it remains high even if we are only trying to gain a tiny advantage over random guessing in computing F! An example of a natural function to which the discrepancy method can be applied is the n-bit Inner Product function $IP_n(X,Y) = \langle X,Y\rangle \mod 2$. This simple discrepancy method can be generalized to a richer family of *corruption bounds* that can be viewed as combinatorial generalizations of the discrepancy bound. More on this method can be found in the survey [31].

One of the early successes of applying combinatorial methods in communication complexity was the proof that the *randomized* communication complexity of the set disjointness problem (12) is linear, $R_{1/4}(Disj_n) = \Theta(n)$. The first proof of this fact was given in the 1980s [26], and a much simpler proof was discovered soon after [41]. The proofs exhibit a specific distribution μ of inputs on which the distributional communication complexity $D_{\mu,1/4}(Disj_n)$ is $\Omega(n)$. Note that the uniform distribution would not be a great fit, since uniformly drawn sets are non-disjoint with a very high probability. It turns out that the following family of distributions μ is hard: select each coordinate pair (X_i, Y_i) i.i.d. from a distribution on $\{(0,0),(0,1),(1,0)\}$ (e.g. uniformly). This generates a distribution on pairs of disjoint sets. Now, with probability $1/2$ choose a uniformly random coordinate $i \in_U [n]$ and set $(X_i, Y_i) = (1,1)$. Thus, under μ, X and Y are disjoint with probability $1/2$.

Treating communication complexity as a generalization of one-way communication and applying information-theoretic machinery to it is a very natural approach (perhaps the most natural, given the success of information theory in communication theory). Interestingly, however, this is not how the field has evolved. In fact, the fairly recent survey [31] was able to present the vast majority of communication complexity results to its date without dealing with information theory at all. It is hard to speculate why this might be the case. One possible explanation is that the mathematical machinery needed to tackle the (much more complicated) interactive case from the information-theoretic angle wasn't available until the 1990s; another possible explanation is that linear algebra, linear programming duality, and combinatorics (the main tools in communication complexity lower bounds) are traditionally more central to theoretical computer science research and education than information theory.

A substantial amount of literature exists on communication complexity within the information theory community, see for example [36, 37] and references therein. The flavor of the results is usually different from the ones discussed above. In particular, there is much more focus on bounded-round communication, and significantly less focus on techniques for obtaining specific lower bounds for the communication complexity of specific functions such as the disjointness function. The most relevant work to our current discussion is a relatively recent line of work by Ishwar and Ma, which studied interactive amortized communication and obtained characterizations closely related to the ones discussed below [32, 33].

Within the theoretical computer science literature, in the context of communication complexity, information theoretic tools were explicitly introduced in [13] in

the early 2000s for the simultaneous message model (i.e. 2 non-interactive rounds of communication). Building on this work, [1] developed tools for applying information theoretic reasoning to fully interactive communication, in particular giving an alternative (arguably, more intuitive) proof for the $\Omega(n)$ lower bound on the communication complexity of $Disj_n$. The motivating questions for [13], as well as for subsequent works developing information complexity, were the *direct sum* [17] and *direct product* questions for (randomized) communication complexity.

In general, a direct sum theorem quantifies the cost of solving a problem F^n consisting of n sub-problems in terms of n and the cost of each sub-problem F. The value of such results to lower bounds is clear: a direct sum theorem, together with a lower bound on the (easier-to-reason-about) sub-problem, yields a lower bound on the composite problem (a process also known as hardness amplification). For example, the Karchmer-Wigderson program for boolean formula lower bounds can be completed via a (currently open) direct sum result for a certain communication model [27]. Direct product results further sharpen direct sum theorems by showing a "threshold phenomenon", where solving F^n with insufficient resources is shown to be impossible to achieve except with an exponentially small success probability. Classic results in complexity theory, such as Raz's Parallel Repetition Theorem [39] can be viewed as a direct product result.

In the next section, we will formally introduce information complexity, first as a generalization of Shannon's entropy to interactive tasks. We will then discuss its connections to the direct sum and product questions for randomized communication complexity, and to recent progress towards resolving these questions.

2. Noiseless coding and information complexity

Interactive information complexity. In this section we will work towards developing information complexity as the analogue of Shannon's entropy for interactive computation. It will sometimes be convenient to work with general *interactive two-party tasks* rather than just functions. A task $T(X,Y)$ is any action on inputs (X,Y) that can be performed by a protocol. $T(X,Y)$ can be though of as a set of distributions of outputs that are acceptable given an input (X,Y). Thus "computing $F(X,Y)$ correctly with probability $1-\varepsilon$" is an example of a task, but there are examples of tasks that do not involve function or relation computation, for example "Alice and Bob need to sample strings A and B, respectively, distributed according to $(A,B) \sim \mu_{(X,Y)}$". For the purposes of the discussion, it suffices to think about T as the task of computing a function with some success probability. The communication complexity of a task T is then defined analogously to the communication complexity of functions. It is the least amount of communication needed to successfully perform the task $T(X,Y)$ by a communication protocol $\pi(X,Y)$.

The *information complexity* of a task T is defined as the least amount of information Alice and Bob need to exchange (i.e. reveal to each other) about their inputs to successfully perform T. This amount is expressed using mutual informa-

tion (specifically, conditional mutual information (7)). We start by defining the *information cost* of a protocol π. Given a prior distribution μ on inputs (X, Y) the information cost

$$\mathsf{IC}(\pi, \mu) := I(Y; \Pi|X) + I(X; \Pi|Y), \tag{19}$$

where Π is the random variable representing a realization of the protocol's transcript, including the *public* randomness it used. In other words, (19) represents the sum of the amount of information Alice learns about Y by participating in the protocol and the amount of information Bob learns about X by participating. Note that the prior distribution μ may drastically affect $\mathsf{IC}(\pi, \mu)$. For example, if μ is a singleton distribution supported on one input (x_0, y_0), then $\mathsf{IC}(\pi, \mu) = 0$ for all π, since X and Y are already known to Bob and Alice respectively under the prior distribution μ. Definition (19), which will be justified shortly, generalizes Shannon's entropy in the non-interactive regime. Indeed, in the transmission case, Bob has no input, thus $X \sim \mu$, $Y = \perp$, and Π allows Bob to reconstruct X, thus $\mathsf{IC}(\pi, \mu) = I(X; \Pi) = H(X) - H(X|\Pi) = H(X)$.

The *information complexity* of a task T can now be defined similarly to communication complexity in (15):

$$\mathsf{IC}(T, \mu) := \inf_{\pi \text{ successfully performs } T} \mathsf{IC}(\pi, \mu). \tag{20}$$

One notable distinction between (15) and (20) is that the latter takes an infimum instead of a minimum. This is because while the number of communication protocols of a given communication cost is finite, this is not true about information cost. One can have a sequence $\pi_1, \pi_2, \ldots$ of protocols of ever-increasing communication cost, but whose information complexity $\mathsf{IC}(\pi_n, \mu)$ converges to $\mathsf{IC}(T, \mu)$ in the limit. Moreover, as we will discuss later, this phenomenon is already observed in a task T as simple as computing the conjunction of two bits.

Our discussion of information complexity will be focused on the slightly simpler to reason about *distributional* setting, where inputs are distributed according to some prior μ. In (20), if T is the task of computing a function F with error ε w.r.t. μ, the distribution μ is used twice: first in the definition of "success", and then in measuring the amount of information learned. It turns out that it is possible to define worst-case information complexity [7] as the information complexity with respect to the worst-possible prior distribution in the spirit of the minimax relationship (16). In particular, the direct sum property of information complexity which we will discuss below holds for prior-free information complexity as well.

Information complexity as defined here has been extensively studied in a sequence of recent works [2, 6, 7, 28, 12, 19], and the study is still very much in progress. In particular, it is surprisingly simple to show that information complexity is additive for tasks over independent pairs of inputs. Let T_1 and T_2 be two tasks over pairs of inputs (X_1, Y_1), (X_2, Y_2), and let μ_1, μ_2 be distributions on pairs (X_1, Y_1) and (X_2, Y_2), respectively. Denote by $T_1 \otimes T_2$ to task composed of successfully performing both T_1 and T_2 on the respective inputs (X_1, Y_1) and (X_2, Y_2). Then information complexity is additive over these two tasks:

Theorem 2.1. $\mathsf{IC}(T_1 \otimes T_2, \mu_1 \times \mu_2) = \mathsf{IC}(T_1, \mu_1) + \mathsf{IC}(T_2, \mu_2)$.

Proof. (Sketch, a complete proof of a slightly more general statement can be found in [7]). The "easy" direction of this theorem is the '$\leq$' direction. Take two protocols π_1 and π_2 that perform T_1 and T_2 respectively, and consider the concatenation $\pi = (\pi_1, \pi_2)$ (which clearly performs $T_1 \otimes T_2$). Consider what Alice learns from an execution of π with prior $\mu_1 \times \mu_2$. A straightforward calculation using, for example, repeated application of the chain rule (8) yields

$$I(Y_1Y_2; \Pi_1\Pi_2 | X_1X_2) = I(Y_1; \Pi_1 | X_1) + I(Y_2; \Pi_2 | X_2),$$

and similarly for what Bob learns. Therefore $\mathsf{IC}(\pi, \mu_1 \times \mu_2) = \mathsf{IC}(\pi_1, \mu_1) + \mathsf{IC}(\pi_2, \mu_2)$. By passing to the limit as $\mathsf{IC}(\pi_1, \mu_1) \to \mathsf{IC}(T_1, \mu_1)$ and $\mathsf{IC}(\pi_2, \mu_2) \to \mathsf{IC}(T_2, \mu_2)$ we obtain the '$\leq$' direction.

The '$\geq$' direction is more interesting, even if the proof is not much more complicated. In this direction we are given a protocol π for solving $T_1 \otimes T_2$ with information cost $I = \mathsf{IC}(\pi, \mu_1 \times \mu_2)$, and we need to construct out of it two protocols for T_1 and T_2 of information costs I_1 and I_2 that add up to $I_1 + I_2 \leq I$. We describe the protocol $\pi_1(X_1, Y_1)$ below:

$\pi_{\mathbf{1}}(\mathbf{X_1}, \mathbf{Y_1})$:

- Bob samples a pair $(X_2, Y_2) \sim \mu_2$, and sends X_2 to Alice;
- Alice and Bob execute $\pi((X_1, X_2), (Y_1, Y_2))$, and output the portion relevant to T_1 in the performance of $T_1 \otimes T_2$.

It is not hard to see that the tuple (X_1, Y_1, X_2, Y_2) is distributed according to $\mu_1 \times \mu_2$, and hence by the assumption on π, π_1 successfully performs T_1. Note that there is a slight asymmetry in π_1: X_2 is known to both Alice and Bob while Y_2 is only known to Bob. For the purpose of correctness, the protocol would have worked the same if Bob also sent Y_2 to Alice, but it is not hard to give an example where the information cost of π_1 in that case is too high. The information cost of π is thus given by the sum of what Bob learns about X_1 from π_1 and what Alice learns about Y_1 (note that (X_2, Y_2) are not part of the input):

$$I_1 = I(X_1; \Pi | X_2Y_1Y_2) + I(Y_1; \Pi | X_1X_2).$$

The protocol $\pi_2(X_2, Y_2)$ is defined similarly to π_1 in a skew symmetric way:

$\pi_{\mathbf{2}}(\mathbf{X_2}, \mathbf{Y_2})$:

- Alice samples a pair $(X_1, Y_1) \sim \mu_1$, and sends Y_1 to Bob;
- Alice and Bob execute $\pi((X_1, X_2), (Y_1, Y_2))$, and output the portion relevant to T_2 in the performance of $T_1 \otimes T_2$.

We get that π_2 again successfully performs T_2, and its information cost is:

$$I_2 = I(X_2; \Pi | Y_1Y_2) + I(Y_2; \Pi | X_1X_2Y_1).$$

Putting I_1 and I_2 together we get:

$$\begin{aligned} I_1 + I_2 = I(X_1; \Pi | X_2Y_1Y_2) + I(Y_1; \Pi | X_1X_2) + I(X_2; \Pi | Y_1Y_2) + I(Y_2; \Pi | X_1X_2Y_1) = \\ I(X_2; \Pi | Y_1Y_2) + I(X_1; \Pi | X_2Y_1Y_2) + I(Y_1; \Pi | X_1X_2) + I(Y_2; \Pi | X_1X_2Y_1) = \\ I(X_1X_2; \Pi | Y_1Y_2) + I(Y_1Y_2; \Pi | X_1X_2) = I. \end{aligned}$$

Once again, passing to the limit, gives us the '$\geq$' direction, and completes the proof. □

If we denote an n-time repetition of a task T by $T^{\otimes n}$, then repeatedly applying Theorem 2.1 yields

$$\mathsf{IC}(T^{\otimes n}, \mu^n) = n \cdot \mathsf{IC}(T, \mu). \tag{21}$$

Thus information complexity is additive and has the *direct sum property*: the cost of n copies of T scales as n times the cost of one copy. This fact can be viewed as an extension of the property $H(X^n) = n \cdot H(X)$ to interactive problems, but what does it teach us about communication complexity?

Direct sum and interactive compression. Let us return to the communication complexity setting, fixing T to be the task of computing a function $F(X, Y)$ with some error at most $\varepsilon > 0$ over a distribution μ (the case $\varepsilon = 0$ seems to be different from $\varepsilon > 0$). We will denote by F_ε^n the task of computing n copies of F on independent inputs distributed according to μ^n, with error at most ε *on each copy* (note that computing F correctly with error at most ε on all copies simultaneously is a harder task). The *direct sum* question for communication complexity asks whether

$$D_{\mu^n}(F_\varepsilon^n) = \Omega(n \cdot D_\mu(F_\varepsilon))? \tag{22}$$

While this question remains open, information complexity sheds light on this question by linking it to problems in interactive coding theory. As discussed below, information complexity appears to be the best tool for either proving or disproving (22), as well as for establishing the "right" direct sum theorem in case (22) is false. It is an easy observation that the information cost of a protocol π is always bounded by its length $|\pi|$, and therefore information complexity is always bounded by communication complexity. Therefore, by (21),

$$\frac{1}{n} \cdot D_{\mu^n}(F_\varepsilon^n) \geq \frac{1}{n} \cdot \mathsf{IC}(F_\varepsilon^n, \mu^n) = \mathsf{IC}(F_\varepsilon, \mu). \tag{23}$$

It turns out that the converse is also true in the limit, as $n \to \infty$ [6]:

$$\lim_{n\to\infty} \frac{1}{n} \cdot D_{\mu^n}(F_\varepsilon^n) = \mathsf{IC}(F_\varepsilon, \mu). \tag{24}$$

Equation (24) can be viewed as the interactive version of the Source Coding Theorem (2). In particular, it gives an operational characterization of information complexity exclusively in terms of communication complexity.

A promising attack route (that works to-date followed) on the direct sum question for communication complexity is to try and prove a relationship of the type $\mathsf{IC}(F_\varepsilon, \mu) \gtrsim D_\mu(F_\varepsilon)$ (as discussed above, the converse is trivially true). Indeed, if we could prove that $\mathsf{IC}(F_\varepsilon, \mu) = \Omega(D_\mu(F_\varepsilon))$, by (23) it would imply that $\frac{1}{n} \cdot D_{\mu^n}(F_\varepsilon^n) = \Omega(D_\mu(F_\varepsilon))$ and prove (22).

One equivalent way to interpret the attempts to prove $\mathsf{IC}(F_\varepsilon, \mu) \gtrsim D_\mu(F_\varepsilon)$ is in terms of a search for an interactive analogue of Huffman coding (11) (where

it does hold that $H(X) > C(x) - 1$). (One way) Huffman coding shows how to encode a low-entropy "uninformative" signal into a short one. Its interactive version seeks to simulate a low information cost "uninformative" protocol π with a low communication protocol π'.

Until very recently, we did not know whether such a general compression scheme exists. Just this year, the first example of a relation whose information and communication complexities are exponentially separated was given in a striking work by Ganor, Kol, and Raz [19]. This result, in particular, shows a protocol π for a sampling problem that has information cost I, but which cannot be simulated by a protocol π' with *communication* cost $< 2^{\Omega(I)}$.

Note that (23), which follows from Theorem 2.1, can be further sharpened as follows. If there is a protocol π_n for solving F_ε^n — n copies of F — with communication cost C_n, then there is a protocol π_1 for solving a single copy of F_ε whose communication cost is still at most $C := C_n$, and whose information cost is at most $I \leq C_n/n$. To prove a lower bound on C_n, we can assume that it is "too small", and then show how to convert π_1 into a protocol π' for F_ε that uses $< D_\mu(F_\varepsilon)$ communication. This brings us to the following general interactive coding/compression question:

Problem 2.2. *(Interactive compression problem). Given a protocol π whose communication cost is C and whose information cost is I, what is the smallest amount of communication needed to (approximately) simulate π?*

To prove the strongest possible direct sum theorem we need π' to be compressed all the way down to $O(I)$ bits of communication (the strongest possible interactive compression result), however, partial interactive compression results lead to weaker (but still non-trivial) direct sum theorems. At present, the two strongest compression results, which partially resolve Problem 2.2, compress π to $\tilde{O}(\sqrt{C \cdot I})$ communication [2] and $2^{O(I)}$ communication [7], respectively. Note that these results are incomparable since $C > I$ can be much (e.g. double-exponentially) larger than I.

These result lead to direct sum theorems for randomized communication complexity. As the compression introduces an additional small amount of error, the first result implies for any constant $\rho > 0$:

$$D_{\mu^n}(F_\varepsilon^n) = \tilde{\Omega}(\sqrt{n} \cdot D_\mu(F_{\varepsilon+\rho})), \tag{25}$$

and the second one implies

$$D_{\mu^n}(F_\varepsilon^n) = \Omega(n \cdot \log(D_\mu(F_{\varepsilon+\rho}))). \tag{26}$$

The recent result of Ganor et al. [19] rules out the strongest possible direct sum theorem for relations. Since the hard-to-compress protocol in their example has a very high communication complexity (on the order of 2^{2^I}), it is still possible that any protocol can be compressed to $O(I \cdot \log^{O(1)}(C))$ communication, leading to a

Here, the $\tilde{O}(\cdot)$ notation hides poly-logarithmic factors.

direct sum theorem with $\frac{n}{\log^{O(1)} n}$ instead of just n. We should also note that the direct sum situation with functions (as opposed to relations) remains open.

Why is interactive compression so much harder than non-interactive? The main difference between the interactive and non-interactive compression settings is that in the interactive setting each message of the protocol conveys an average of $I/C \ll 1$ bits of information. There are many ways to compress communication in the relevant setting, but all of them incur an average loss of $\Omega(1)$ bits per round (Huffman coding being one example of this phenomenon). This is prohibitively expensive in the interactive case, if the number of rounds of interaction r is equal to C. Therefore, inevitably, to compress interactive communication one has to compress multiple rounds in one message. This problem disappears when $I \gg r$, and this is what makes the '$\leq$' direction of (24) go through when n is sufficiently large.

Direct product for communication complexity. Next, we turn our attention to the more difficult *direct product* problem for communication complexity. The direct sum question talks about the amount of resources needed to achieve a certain probability of success on n copies of F. What if that amount of resources is not provided? For example, (23) implies that unless $n \cdot \mathsf{IC}(F_\varepsilon, \mu)$ bits of communication is allowed in the computation of F_ε^n, the computation of *some* copy of F will have $< 1 - \varepsilon$ success probability. What does it tell us about the success probability of *all* copies simultaneously? It only tells us that the probability of the protocol succeeding on all copies simultaneously is bounded by $1 - \varepsilon$. This is a very weak bound, since solving the n copies independently leads to a success probability of $(1 - \varepsilon)^n$, which is exponentially small for a constant ε. How can this gap be reconciled? In particular, can one show that Alice and Bob cannot "pool" the errors from all n copies on the same instances, thus keeping the success probability for each coordinate, as well as the global success probability, close to $1-\varepsilon$? The direct product problem precisely addresses this question. Let us denote by $\mathsf{suc}(F, \mu, C)$ as the highest success probability (w.r.t. μ) in computing F that can be attained using communication $\leq C$. Thus $\mathsf{suc}(F, \mu, C) \geq 1-\varepsilon$ is equivalent to $D_\mu(F_\varepsilon) \leq C$. Somewhat informally phrased, the direct product question asks whether

$$\mathsf{suc}(F^n, \mu^n, o(n \cdot C)) < \mathsf{suc}(F, \mu, C)^{\Omega(n)}? \tag{27}$$

As with the direct sum question, the direct product question appears "obvious": one would expect that the best we can do is just execute the best protocol for one copy of F n times independently. This will lead to a success probability of $\leq \mathsf{suc}(F, \mu, o(C))^n$.

A prominent setting within complexity theory where a question similar to the direct product question arose is that of *parallel repetition* for two-prover games [39]. Parallel repetition is used in the context of probabilistically checkable proofs (PCP) and hardness amplification. Hardness amplification is accomplished here by taking a hard task T (e.g. a verification procedure where the success probability of an unauthorized provers is $1-\varepsilon$), and creating a task T^n by taking n independent instances of T. It has been shown [39] that as n grows, the success probability

goes to 0. Unfortunately, it does not go to 0 as $(1-\varepsilon)^n$. Indeed, as shown by a counterexample constructed by Raz [40], the best rate one can hope for is $(1-\varepsilon^2)^n$. The reason for this, pointed out by an earlier example by Feige and Verbitsky [18], is that the answers can be arranged to align errors together, so that when the provers fail, they fail on a lot more than εn coordinates at the same time. This is possible when answers are allowed to be correlated.

The direct product question (27) for communication complexity combines features from the direct sum question (thus hinting that information complexity is to play a role here as well), and from the parallel repetition setup (since we want a success probability dropping exponentially in n). The direct sum discussion already suggests that for $\mathsf{suc}(F,\mu,C) = 1-\varepsilon$, the best scaling of the amount of communication one can hope for is as $n \cdot I$, where $I = \mathsf{IC}(F_\varepsilon,\mu)$. This is because, as $n \to \infty$, the per-copy communication cost of computing F with error ε scales as $n \cdot I$. Thus, if we denote by $\mathsf{suc}^{\mathsf{i}}(F,\mu,I) \geq \mathsf{suc}(F,\mu,I)$ the best success probability one can attain at solving F while incurring an *information cost* of at most I, the direct product question for information asks whether

$$\mathsf{suc}(F^n,\mu^n,o(n\cdot I)) < \mathsf{suc}^{\mathsf{i}}(F,\mu,I)^{\Omega(n)}? \tag{28}$$

Note that the success probability on the left-hand-side is still with respect to communication. A statement such as this with respect to information cost is bound to be false: Information cost being an average-case quantity, one can attain an information-cost I_n protocol by doing nothing with probability $1-\delta$, and incurring an information cost of $I_n/\delta \gg n \cdot I$ with probability δ that can be taken only *polynomially* (and not exponentially) small.

In a sequence of two papers, the second being very recent [11, 12], (28) was shown to be true up to polylogarithmic factors for boolean functions. To simplify parameters, suppose $\mathsf{suc}^{\mathsf{i}}(F,\mu,I) < 2/3$. Then there are constants c_1, c_2 such that

$$\text{if } T\log T < c_1 n\cdot I, \text{ then } \mathsf{suc}(F^n,\mu^n,T) < 2^{-c_2 n}. \tag{29}$$

The proof of (29) is quite involved and combines ideas from the proof of direct sum theorems and of parallel repetition theorems.

Exact communication complexity bounds. One of the great successes of information theory as it applies to (classical, one-way) communication problems is in its ability to give precise answers to fairly complicated asymptotic communication problems, for example ones involving complicated dependencies between terminals or complicated channels. For example, the capacity of the binary symmetric channel $BSC_{0.2}$ is precisely $1-H(0.2) \approx 0.278$, which means that to transmit n bits over such a channel, we will need $\approx 3.596n$ utilizations of the channel (i.e. will need to send $\approx 3.596n$ bits down the channel). Using combinatorial techniques, in most cases, such precision is inaccessible in the two-party setting, since the techniques often lose constant factors by design. In contrast, information complexity extends the precision benefits of one-way information theory to the interactive setting.

We give one specific example of an exact communication complexity bound. Recall that the disjointness problem $Disj_n(X,Y)$ takes two n-bit vectors X,Y and

checks whether there is a location with $X_i = Y_i = 1$. Thus $Disj_n$ is just a disjunction of n independent copies of the two bit $AND(X_i, Y_i)$ function. Using techniques similar to the proof of Theorem 2.1, one can show that the communication complexity of disjointness is tightly linked with the information complexity of AND. Note that disjointness becomes trivial if many coordinates (X_j, Y_j) of the input are $(1, 1)$. However, any distribution of inputs where $\mu((X_j, Y_j) = (1, 1)) \sim 1/n \to 0$ will not be trivial. More formally, denote by 0^+ a function $f(n)$ of n such that $f(n) = o(1)$ and $f(n) \gg 2^{-O(n)}$. For example, one can take $f(n) = 1/n$. Then with some work one shows [9] that

$$R_{0^+}(Disj_n) = \left(\inf_{\mu:\mu(1,1)=0} \mathsf{IC}(AND_0, \mu) \right) \cdot n \pm o(n). \tag{30}$$

Thus, understanding the precise asymptotics of the communication complexity of $Disj_n$ boils down to understanding the (0-error) information complexity of the two-bit AND function. It turns out that one can give an explicit information-theoretically optimal family of protocols for AND, and calculate the quantity in (30) explicitly, obtaining $R_{0^+}(Disj_n) = C_{DISJ} \cdot n \pm o(n)$ where $C_{DISJ} \approx 0.4827$.

Interestingly, even in the case of such a simple function as two-bit AND, the information complexity is not attained by any particular protocol, but rather by an infinite family of communication protocols! Moreover, if we denote by $IC_r(AND_0)$ the information cost of AND where the infimum in (20) is only taken over protocols of length r, then it turns out that $IC_r(AND_0) = IC(AND_0) + \Theta(1/r^2)$, implying that an asymptotically optimal protocol is only achieved with a super-constant number of rounds [9]. We do not yet know how general this $1/r^2$ gap phenomenon is, and which communication tasks admit a minimum in (20).

3. Interactive error-correcting codes

Adversarial error-correction. The discussion so far focused on coding for interactive computing over a noiseless binary channel. In this section we will focus on error-correction problems when the channel contains random or adversarial noise. The first regime we would like to consider is that of adversarial noise. In this regime Alice and Bob are trying to perform a task T over a channel in which an adversary is allowed to corrupt a constant fraction of the messages. Both the regime of a binary channel and that of a channel with constant-size alphabet Σ (i.e. where symbols $\sigma \in \Sigma$ are being transmitted over the channel) are interesting.

The one-way case has been extensively studied for several decades, as discussed in the introduction. If the task T is just a simple transmission task, then the theory of (worst-case) error-correcting codes [34, 44] applies. While there are many open problems in coding theory, the overall picture is fairly well understood. In particular, constructions of "good" positive-rate, constant-distance codes exist (i.e. codes

Note that even when $\mu(1, 1) = 0$ and thus $AND(X, Y) = 0$ on $\mathrm{supp}(\mu)$, the task AND_0 requires the protocol to *always* be correct – even on the $(1, 1)$ input. Otherwise, $\mathsf{IC}(AND_0, \mu)$ would trivially be 0.

that increase communication by a constant factor only, and can tolerate a constant fraction of errors), and there are efficient encoding and decoding constructions.

In the interactive case, the task may include many back-and-forth messages. As a generic task, it is convenient to think about alternating binary pointer jumping (BPJ_n). In this problem the parties are working with a depth-n binary tree. Alice is given a subset T_A of edges on the odd layers of the tree, with exactly one edge coming out of each vertex on odd layers. Similarly, Bob is given a subset T_B of edges on the even layers of the tree. Their goal is to find the unique leaf that is connected to the root by edges from $T_A \cup T_B$. There is an obvious n-bit protocol for finding the leaf, where Alice and Bob alternate. The definition of BJP_n is parallel to the definition of a n-round protocol π as given by Definition 1.1. In this sense, BPJ_n is the generic interactive task, as any interactive protocol can be recast as an instance of BPJ_n.

To continue the comparison with the non-interactive setting, suppose an adversary is allowed to corrupt a δ-fraction of the symbols exchanged by Alice and Bob, for some $\delta > 0$. Can they still compute BPJ_n? Solving BPJ_n efficiently requires a lot of back-and-forth interaction. A naïve approach would be to apply (standard) error-correction to the interactive protocol on a round-by-round basis. This does not work, because the adversary can concentrate all of her errors, for example, on the first round, causing all subsequent communications to be wrong and derailing the protocol's execution. Another obvious solution that dose work is to have Alice send her input T_A to Bob using a standard error-correcting code. Bob then can compute the leaf. This solution works, but causes an exponential blow-up in communication, since T_A takes $\sim 2^n$ bits to describe, while the efficient solution for BPJ_n requires only $O(n)$ communication.

It is not at all clear that a constant-rate error correcting code is possible. Surprisingly, constant-rate error-correcting codes for interactive computing do exist. The first such code was demonstrated in a breakthrough work by Schulman in the 1990s [42], who showed a constant-rate code against an adversary who is allowed to corrupt a constant δ-fraction of the symbols on the channel for $\delta < 1/240$. Schulman introduced a concept of a *tree code*. Variants and extensions of tree codes have been used in all constructions since. The construction opened up opportunities for interactive error-correction, but also left room for improvement, as the error-parameter $\delta < 1/240$ is far from optimal and the error-correction is not efficient in that it requires time exponential in n to compute the encoding/decoding (even though the communication itself is $O(n)$ symbols).

After a gap in progress on interactive error-correction, a substantial amount of progress has been made in the last 5 years [10, 20, 5, 8, 22, 21]. Progress so far has focused on (1) making the tolerable error rate δ as high as possible; (2) making the construction explicit and computationally efficient. This while keeping the rate (i.e. the ratio between the encoding length and the length of the noise-free execution) of the code constant. What remains completely open is the exact coding rate for interactive coding, given a specific value of δ. All we know are characterizations of δ for which various specific types of good codes exist.

Next, let us discuss the error-rate region for which (two-party) interactive error-

correction is possible. Suppose Alice and Bob communicate over a channel which uses an alphabet Σ_2 with $|\Sigma_2| = O(1)$ a large constant that is allowed to depend on δ (the case of a binary noisy channel, $|\Sigma_2| = 2$, is also interesting, with many of the problems still open there). An interactive error-correction scheme π is a protocol of a fixed length $n' = O(n)$ over Σ_2 that solves BPJ_n, even when the channel is affected by a noise of rate δ. In other words, for any inputs (T_A, T_B), any execution transcript Π of π in which a total of at most $\delta \cdot n'$ of the symbols were corrupted results with Alice and Bob outputting the correct leaf

$$BPJ_n(T_A, T_B) = D_A(T_A, \Pi) = D_B(T_B, \Pi), \tag{31}$$

D_A and D_B being the decoding functions for Alice and Bob, respectively. Here D_A and D_B are only allowed to depend on the portions of the transcript Π accessible to Alice and Bob, respectively.

First assume that in π, the player speaking in each round is pre-determined (a single symbol is sent in each round). Such protocols are caller *robust*. Note that without this assumption, it is possible to have a round in which both Alice and Bob (or neither Alice nor Bob) speak, since error may confuse the players as to whose turn it is to speak. In this case further modeling assumptions are needed to specify what happens during these rounds.

In the robust case, note that the adversary knows ahead of time n_A and n_B — the number of rounds Alice and Bob speak, respectively, in π. Here $n_A + n_B = n'$. Assume without loss of generality that $n_A \leq n'/2$. Then, as with the proof that one way error-correcting codes cannot recover from an error rate exceeding $1/2$, by extrapolating between $\pi(T_{A_1}, T_B)$ and $\pi(T_{A_2}, T_B)$, an adversary can corrupt $n_A/2$ rounds of π, and prevent Bob from distinguishing two potential inputs T_{A_1} and T_{A_2} of Alice. If the resulting transcript is Π, as long as $BPJ_n(T_{A_1}, T_B) \neq BPJ_n(T_{A_2}, T_B)$, either $D_B(T_B, \Pi) \neq BPJ_n(T_{A_1}, T_B)$ or $D_B(T_B, \Pi) \neq BPJ_n(T_{A_2}, T_B)$, meaning that π sometimes fails. Thus the adversary can foil the protocol using $n_A/2 \leq n'/4$ errors, so we cannot hope to overcome an error rate of $\delta \geq 1/4$.

It turns out [10] that it is possible to deal with error rates of $\delta = 1/4 - \varepsilon$ using constant-rate codes. As in Schulman's construction, the key technical ingredient of this result is that of a *tree code*. A tree code is a prefix code $C : \{0,1\}^m \to \Sigma_2^m$; in a prefix code the i-th symbol of the codeword $C(S)_i = C_i(S_{[1..i]})$ only depends on the first i symbols of the word being encoded. It is clear that a prefix code cannot have the constant-distance property since, for example $C(0^m)$ and $C(0^{m-1}1)$ cannot differ in more than one symbol. The best property we can hope for is that codewords of length k that deviate after the i-th symbol will differ by close to $(k - i)$ symbols. This is indeed the definition of a tree code: a tree code $C : \{0,1\}^m \to \Sigma_2^m$ is said to have *distance* α if for all i,k, and $w \in \{0,1\}^i$, $w_0, w_1 \in \{0,1\}^{k-i-1}$,

$$d_H(C(w0w_0), C(w1w_1)) \geq \alpha \cdot (k - i). \tag{32}$$

It can be shown [42] that tree codes exist for any constant $\alpha < 1$ (the alphabet Σ_2 may need to be made sufficiently large, with its size increasing as α approaches

1). Note that it is easy to see that a random code will *not* be a tree code with a very high probability. Therefore, even constructing a non-explicit tree codes is not a trivial task. To decode a tree code, the receiver just finds the codeword that is closest to the received word in Hamming distance.

Informally, each symbol sent by the tree code not only encodes the current symbol being sent, but also hashes the entire history of the transmission, ensuring that a mistake introduced by an adversary will be corrected as following rounds arrive. The key useful property of tree codes for the purposes of interactive error-correction codes is the following: Suppose that t rounds ago Alice sent a message z encoded using the tree code, and the adversary managed to keep Bob from receiving it, and instead Bob thinks that $\bar{z}$ was sent t rounds ago. This means that the amount of errors between now and some point $t' \geq t$ rounds ago must be at least $\alpha \cdot t'/2$. In other words, to keep Bob from learning z, the adversary has to introduce many errors in a large stretch that in particular is at least t symbols long.

Next, let us give the intuition for how tree codes can be useful in interactive error correction, following the construction in [10]. Unfortunately, due to space constraints, we will not be able to give a full sketch here. The protocol π will proceed by having Alice and Bob send edges of T_A and T_B, respectively, using a tree code to encode a stream of edges being sent. The parties are trying to build the unique path from the root in $T_A \cup T_B$. At each point in time, one of the parties (say Alice) can extend the path, assuming she correctly decoded the previous edges. By the discussion above about the main property of tree codes, to keep Alice from correctly decoding the previous edges, the adversary will have to use an error rate of at least $\alpha/2$ in Bob's transmissions between the time the previous edge had been sent by Bob, and when it is decoded by Alice. This amounts to an error rate of $\alpha/4$. By choosing $\alpha/4 > \delta$ (which is possible since $\delta < 1/4$) we can guarantee enough rounds in which Alice and Bob will make progress. This outline glosses over how edges are represented, and indeed representing edges so that each only takes $O(1)$ bits which can be encoded using the tree code is the main technical challenge overcome by [10].

As noted earlier, relaxing the robustness assumption requires further modeling assumptions on what happens in rounds where either both Alice and Bob or neither speak. One would expect that by having the party that is being targeted by the adversary speak more, one can improve the error tolerance of the protocol. Indeed, the example showing the $1/4$ limit above could be remedied if the party being targeted by the adversary spoke more than $n'/2$ of the rounds (thus forcing the adversary to expend more of her budget). Under a reasonable model, a recent work [22] shows that the error-tolerance of non-robust protocols can be made $2/7 - \varepsilon > 1/4$, and that this bound is tight.

In the one-way error-correcting coding theory, an important way of going beyond error-rate $1/2$ is using the concept of *list decoding*. A list-decodable code is one where for a corrupt encoded words, there is a (constant-size) list of possible decodings. Over large, constant-size alphabets, list-decodable codes exist for any error rate of $1 - \varepsilon$, where the output list size is $O_\varepsilon(1)$. In the interactive set-

ting, somewhat surprisingly, one can also construct list-decodable error-correcting schemes. In the robust setting, the best error rate attainable by a constant-rate code is $1/2 - \varepsilon$ [8]. This construction uses a generalization of tree codes called list-tree-codes. This generalization has an average-case rather than worst-case coding property, and is instantiated by a random prefix code with a sufficiently large constant $|\Sigma_2|$ with a very high probability. Interestingly, it appears that one needs interactive list-decoding even just to attain optimal error-resistance for unique decoding in some regimes.

One limitation of the constructions above is that they are not explicit. In other words, while we know that they can all be instantiated, often with a random prefix code, no provable explicit constructions of tree codes and list-tree-codes are known. Worse yet, even if one could somehow derandomize these constructions, the brute-force decoding procedures require exponential time. Several recent works developed efficient interactive error-correcting coding schemes. In particular, the very recent work by Ghaffari and Haeupler [21] gives an efficient scheme that achieves the same error-correction guarantees as the best-known non-efficient scheme (see [21] for additional recent history and references). Its only limitation is that it uses randomness for initialization, but it allows this randomness to be accessible by the adversary, so it is not a major limitation since no shared secret between Alice and Bob is needed. Most excitingly, while the scheme has a slightly sub-constant rate, by combining it with the construction of [8] it appears that it can be made constant-rate, thus concluding the quest for efficient interactive error-correcting schemes with optimal error dependence.

All efficient schemes to-date follow a similar paradigm: start with a non-efficient scheme on a very small scale (say, $\log \log n$ rounds). On such a small scale one can just brute-force the search for tree codes, and for efficient encoding-decoding schemes. Next, show how to go from an interactive constant-rate error-correcting scheme of depth k to one of depth, say, 2^k. Note that one will only need to apply such a transition twice to go from depth $\log \log n$ do depth n.

A major gap in our understanding of interactive error-correction is in the *rate* of optimal codes. In other words, for a given error rate $\delta = 1/4 - \varepsilon$, what is the best rate

$$\rho_\delta = \frac{n}{n' \log |\Sigma_2|}$$

one can hope to attain in solving BPJ_n? We do not even know the asymptotics of ρ_δ as δ approaches the boundary points of 0 and 1/4. Perhaps this should not be too disappointing, since parallel questions are open for one-way communication. However, one could hope to resolve these problems in the *random error* model, since there Shannon's classical work does give us precise channel capacity answers. We turn our attention to that regime next.

Random errors and channel capacity. In the random error model, Alice and Bob communicate over a noisy channel $\mathcal{C}$, where the noise is generated randomly. For concreteness, we will focus here on the binary symmetric channel with error ε, BSC_ε, where bits are being transmitted and each bit sent over the channel is independently flipped with probability ε.

As discussed earlier, the channel capacity of BSC_ε is given by (10) and is equal to $1 - H(\varepsilon)$. Informally, this means that the utility of BSC_ε in conducting communication is $1 - H(\varepsilon)$, and that for a growing n, transmitting n random bits over BSC_ε will require

$$n/\mathsf{cap}(BSC_\varepsilon) \pm o(n) = n/(1 - H(\varepsilon)) \pm o(n) \tag{33}$$

utilizations of the channel. How this logic should extend to the interactive case is still up to debate. One natural extension is to consider the pointer jumping problem BPJ_n from before as the standard interactive problem, and to define interactive channel capacity in terms of the number of channel utilizations needed to execute BPJ_n, similarly to

$$\mathsf{icap}_1(\mathcal{C}) := \lim_{n\to\infty} \frac{n}{\#\text{ of utilizations of } \mathcal{C} \text{ needed to perform } BPJ_n \text{ w.h.p.}}. \tag{34}$$

No explicit formulas (or ways of obtaining explicit values) of $\mathsf{icap}_1(BSC_\varepsilon)$ are known. Even establishing directly that $\mathsf{icap}_1(BSC_\varepsilon) > 0$ does not seem completely straightforward, although this fact is a direct consequence of the more general adversarial setting from the previous section. One important recent result by Kol and Raz [29] establishes a gap between icap_1 and Shannon's channel capacity for BSC_ε showing that

$$\mathsf{icap}_1(BSC_\varepsilon) = 1 - \Theta(\sqrt{H(\varepsilon)}) = 1 - \Theta(\sqrt{\varepsilon \log 1/\varepsilon}) < \mathsf{cap}(BSC_\varepsilon) \tag{35}$$

as $\varepsilon \to 0$. This result is quite technical, and underscores the difficulty of the interactive channel capacity question.

One of the nice properties of Shannon's one-way information theory is that the notions of entropy and of channel capacity commute. That is, if we want to transmit a random variable X whose entropy is $H(X) \gg 1$ over a channel $\mathcal{C}$, then the number of channel utilizations needed to transmit X is on average

$$(H(X)/\mathsf{cap}(\mathcal{C}))(1 + o(1)). \tag{36}$$

In the interactive setting, we have established information complexity as the interactive analogue of channel capacity. It is unclear whether there is a way to define interactive channel capacity that makes the interactive analogue of (36) hold. Such an analogue may also help shed light onto the basic structure of interactive communication. The result of [29] implies that such a characterization cannot simultaneously capture interactive and non-interactive tasks, and thus it is bound to be quite complex.

4. Conclusion and discussion

We conclude with some specific open problems and a general discussion. In addition to some of the open questions outlined above, several other questions, which are easy in the non-interactive setting become more challenging when interaction is added to the mix.

Computability of information complexity. The first problem that is (somewhat embarrassingly) open, is computing the information complexity from the truth table of F:

Problem 4.1. *Given the truth table of a function $F : (X, Y) \mapsto \{0, 1\}$, an error parameter $\varepsilon \geq 0$, and a distribution μ of (X, Y), can one give a general procedure for computing the information complexity $\mathsf{IC}(F_\varepsilon, \mu)$?*

We believe the answer to Problem 4.1 to be affirmative. As noted above, the problem is that there might be a sequence of protocols whose information cost decreases as protocol size increases. The $\leq$ direction of (24) gives one way to obtain a decreasing sequence that converges to $\mathsf{IC}(F_\varepsilon, \mu)$ by considering the amortized cost of n copies of F as $n \to \infty$. Unfortunately, for this procedure to compute $\mathsf{IC}(F_\varepsilon, \mu)$, we need to have an effective bound on the sequence's rate of convergence down to $\mathsf{IC}(F_\varepsilon, \mu)$. The work [32] gives a computable characterization of $\mathsf{IC}(F_\varepsilon, \mu)$, but only when one fixes the number of rounds of interaction (back-and-forth messages) in advance. We do not know an effective rate of convergence of the round-restricted information complexity to the unrestricted value.

One can also formulate Problem 4.1 as a continuous dynamic programming problem in the spirit of the Hamilton-Jacobi-Bellman equation [9], but it is not clear how to solve the resulting equation, although it might be doable by better understanding the properties of the function $\mathsf{IC}(F_\varepsilon, \mu)$ when considered as a function on the space of distributions μ.

Multi-party communication. It is a natural and very interesting goal to generalize the discussion above to more than two terminals. There are various models for multi-terminal interactive computation. The main complication stems from the fact that the prior distributions, and the way the inputs to different players are correlated, may be rather sophisticated. One popular model of multi-party computation is that of number-on-forehead (NOF). In the NOF model each party gets to see all inputs but its own and the goal is to compute a function $F(X_1, \ldots, X_k)$ of the inputs [30, 14]. Lower bounds in this model would have profound implications in complexity theory [3]. Multiparty NOF lower bounds are considerably harder than two-party bounds. For example, it is still unknown whether the communication complexity of the 3-party analogue of $Disj_n$ has communication complexity $\Theta(\sqrt{n})$ or $\Theta(n)$ (or something in between) [43].

There are numerous complications in extending notions of information complexity to multi-terminal settings. Apart from sheer technical difficulties, a major obstacle is finding the "right" analogue of public and private randomness. Note that even with three parties we have seven different types of randomness (one "private" for each party, one "public", and three shared between two of the three parties but not the third). Allowing all the different types of randomness leads to another impasse, as in this regime there are information-theoretically secure protocols for multi-party computation [4] which would bring the information complexity of all problems close to 0. This, it turn, limits the usefulness of an information complexity notion based on the amount of information revealed to the parties.

Beyond communication: continuous relaxations for other models of computation. From the viewpoint of theoretical computer science, information complexity can be viewed as the continuous relaxation of communication complexity. Avoiding the "discreteness" of bits and switching to information instead simplified not only the proofs, but the results themselves. For example, the direct sum theorem (Theorem 2.1) is true for information complexity but is not true, at least in full generality, for communication complexity. Thus, this is one more example in the context of complexity theory where a continuous relaxation is easier to deal with. There are many more such examples in the context of algorithms. For example, one of the leading paradigms in approximation algorithms involves relaxing discrete problems into continuous convex optimization programs (for example, linear or semi-definite), and then rounding the resulting fractional solution to obtain an integral one. This allows one to connect the problem of algorithm development with a rich (and deep) theory of continuous analysis and geometry.

In the context of computational complexity, there is still much to be desired in terms of our ability to "de-discretize" computation. The difficulty of dealing with a discrete computation theory has been foreseen by von Neumann as early as 1948 [46] in his Hixon Symposium talk:

> "There exists today a very elaborate system of formal logic, and, specifically, of logic as applied to mathematics. This is a discipline with many good sides, but also with certain serious weaknesses. This is not the occasion to enlarge upon the good sides, which I have certainly no intention to belittle. About the inadequacies, however, this may be said: Everybody who has worked in formal logic will confirm that it is one of the technically most refractory parts of mathematics. The reason for this is that it deals with rigid, all-or-none concepts, and has very little contact with the continuous concept of the real or of the complex number, that is, with mathematical analysis. Yet analysis is the technically most successful and best-elaborated part of mathematics. Thus formal logic is, by the nature of its approach, cut off from the best cultivated portions of mathematics, and forced onto the most difficult part of the mathematical terrain, into combinatorics.
>
> The theory of automata, of the digital, all-or-none type, as discussed up to now, is certainly a chapter in formal logic. It would, therefore, seem that it will have to share this unattractive property of formal logic. It will have to be, from the mathematical point of view, combinatorial rather than analytical."

Over 65 years later, most fundamental problems in the theory of computation, such as the **P** vs. **NP** problem, are wide open, and most unconditional lower bounds are based on diagonalization ideas of Cantor, Gödel and Turing. With some notable exceptions, such as the use of polynomials in circuit complexity lower bounds, Von Neuman's prognostication appears to have withstood the test of time.

Is there a natural continuous relaxation of computational complexity specific enough to deal with its major open problems? And are our mathematical tools

mature enough to pursue one if it exists? In the context of communication, information theory is a great example of a continuous theory that organizes (and greatly simplifies) discrete communication. Communication complexity started out as a discrete theory, but appears to be amenable to continuous treatment, with information complexity being its natural continuous relaxation. It will be very interesting to see whether this push can be extended further into computational complexity.

Acknowledgments. I would like to thank Ankit Garg, Rotem Oshman, Denis Pankratov, and Omri Weinstein for their numerous comments on earlier drafts of this paper.

References

[1] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.

[2] B. Barak, M. Braverman, X. Chen, and A. Rao. How to compress interactive communication. *SIAM Journal on Computing*, 42(3):1327–1363, 2013.

[3] Richard Beigel and Jun Tarui. On acc [circuit complexity]. In *Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on*, pages 783–792. IEEE, 1991.

[4] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 113–131, 1988.

[5] Z. Brakerski and Y.T. Kalai. Efficient interactive coding against adversarial noise. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2012.

[6] M. Braverman and A. Rao. Information equals amortized communication. In *52nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 748–757. IEEE, 2011.

[7] Mark Braverman. Interactive information complexity. In *Proceedings of the 44th symposium on Theory of Computing*, pages 505–524. ACM, 2012.

[8] Mark Braverman and Klim Efremenko. List and unique coding for interactive communication in the presence of adversarial noise. *Electronic Colloquium on Computational Complexity (ECCC)*, 2014.

[9] Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 151–160. ACM, 2013.

[10] Mark Braverman and Anup Rao. Towards coding for maximum errors in interactive communication. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 159–166. ACM, 2011.

[11] Mark Braverman, Anup Rao, Omri Weinstein, and Amir Yehudayoff. Direct products in communication complexity. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 746–755. IEEE, 2013.

[12] Mark Braverman and Omri Weinstein. An interactive information odometer with applications. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2014.

[13] Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In Bob Werner, editor, *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 270–278, Los Alamitos, CA, October 14–17 2001. IEEE Computer Society.

[14] Arkadev Chattopadhyay and Toniann Pitassi. The story of set disjointness. *ACM SIGACT News*, 41(3):59–85, 2010.

[15] Thomas M Cover and Joy A Thomas. *Elements of information theory, 2nd edition*. J. Wiley and Sons, New York, 2006.

[16] Peter Elias. List decoding for noisy channels. 1957.

[17] Tomas Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM Journal on Computing*, 24(4):736–750, 1995.

[18] Uriel Feige and Oleg Verbitsky. Error reduction by parallel repetitiona negative result. *Combinatorica*, 22(4):461–478, 2002.

[19] Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2014.

[20] R. Gelles, A. Moitra, and A. Sahai. Efficient and explicit coding for interactive communication. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 768–777. IEEE, 2011.

[21] Mohsen Ghaffari and Bernhard Haeupler. Optimal error rates for interactive coding ii: Efficiency and list decoding. *arXiv preprint arXiv:1312.1763*, 2013.

[22] Mohsen Ghaffari, Bernhard Haeupler, and Madhu Sudan. Optimal error rates for interactive coding i: Adaptivity and other settings. *arXiv preprint arXiv:1312.1764*, 2013.

[23] Venkatesan Guruswami. *List decoding of error-correcting codes*. Springer, 2004.

[24] Venkatesan Guruswami. Bridging shannon and hamming: List error-correction with optimal rate. In *Proceedings of ICM*, 2010.

[25] David A Huffman et al. A method for the construction of minimum redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[26] Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, November 1992.

[27] Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Computational Complexity*, 5(3/4):191–204, 1995. Prelim version CCC 1991.

[28] Iordanis Kerenidis, Sophie Laplante, Virginie Lerays, Jérémie Roland, and David Xiao. Lower bounds on information complexity via zero-communication protocols and applications. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 500–509. IEEE, 2012.

[29] Gillat Kol and Ran Raz. Interactive channel capacity. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 715–724. ACM, 2013.

[30] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, Cambridge, 1997.

[31] Troy Lee and Adi Shraibman. *Lower bounds in communication complexity*. Now Publishers Inc, 2009.

[32] N. Ma and P. Ishwar. Some results on distributed source coding for interactive function computation. *Information Theory, IEEE Transactions on*, 57(9):6180–6195, 2011.

[33] Nan Ma and P. Ishwar. The infinite-message limit of two-terminal interactive source coding. *Information Theory, IEEE Transactions on*, 59(7):4071–4094, July 2013.

[34] F. J. MacWilliams and N. J. A. Sloane. *The theory of error correcting codes*. North–Holland, New York, 1977.

[35] Ilan Newman. Private vs. common random bits in communication complexity. *Information Processing Letters*, 39(2):67–71, 31 July 1991.

[36] A Orlitsky and A El Gamal. Communication complexity. In *Complexity in information theory*, pages 16–61. Springer, 1988.

[37] Alon Orlitsky and James R Roche. Coding for computing. In *Information Theory, 1995. Proceedings., 1995 IEEE International Symposium on*, page 451. IEEE, 1995.

[38] Vera Pless, Richard A Brualdi, and William Cary Huffman. *Handbook of coding theory*. Elsevier Science Inc., 1998.

[39] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.

[40] Ran Raz. A counterexample to strong parallel repetition. *SIAM Journal on Computing*, 40(3):771–777, 2011.

[41] Alexander Razborov. On the distributed complexity of disjointness. *TCS: Theoretical Computer Science*, 106, 1992.

[42] Leonard J. Schulman. Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6):1745–1756, 1996.

[43] Alexander A Sherstov. Communication lower bounds using directional derivatives. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 921–930. ACM, 2013.

[44] M. Sudan. Algorithmic introduction to coding theory – course notes, 2001. http://people.csail.mit.edu/madhu/FT01/course.html.

[45] Jacobus Hendricus Van Lint. *Introduction to coding theory*, volume 86. Springer, 1982.

[46] J. von Neumann. The general and logical theory of automata. In *John von Neumann, collected works*, chapter 9, pages 288–328. Pergamon Press, 1951.

[47] John M Wozencraft. List decoding. *Quarterly Progress Report*, 48:90–95, 1958.

[48] Andrew C. C. Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 209–213. ACM, 1979.

[49] Andrew C. C. Yao. Lower bounds by probabilistic arguments. In *Foundations of Computer Science, 1983., 24th Annual Symposium on*, pages 420–428. IEEE, 1983.

Department of Computer Science, Princeton University, Princeton, NJ 08544, USA
E-mail: mbraverm@cs.princeton.edu