Constant-Rate Interactive Coding Is Impossible, Even In Constant-Degree Networks

Ran Gelles*[1] and Yael T. Kalai[2]

1 Faculty of Engineering, Bar-Ilan University, Ramat-Gan, Israel
ran.gelles@biu.ac.il

2 Microsoft Research, Boston, USA
yael@microsoft.com

Abstract

Multiparty interactive coding allows a network of n parties to perform distributed computations when the communication channels suffer from noise. Previous results (Rajagopalan and Schulman, STOC '94) obtained a multiparty interactive coding protocol, resilient to random noise, with a blowup of $O(\log(\Delta + 1))$ for networks whose topology has a maximal degree Δ. Vitally, the communication model in their work forces all the parties to send one message at every round of the protocol, even if they have nothing to send.

We re-examine the question of multiparty interactive coding, lifting the requirement that forces all the parties to communicate at each and every round. We use the recently developed information-theoretic machinery of Braverman et al. (STOC '16) to show that if the network's topology is a cycle, then there is a specific "cycle task" for which any coding scheme has a communication blowup of $\Omega(\log n)$. This is quite surprising since the cycle has a maximal degree of $\Delta = 2$, implying a coding with a *constant blowup* when all parties are forced to speak at all rounds.

We complement our lower bound with a matching coding scheme for the "cycle task" that has a communication blowup of $\Theta(\log n)$. This makes our lower bound for the cycle task tight.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity, E.4 Coding and Information Theory

Keywords and phrases Interactive Communication, Coding, Stochastic Noise, Communication Complexity

Digital Object Identifier 10.4230/LIPIcs.ITCS.2017.21

1 Introduction

In multiparty interactive communication, n parties, connected via some arbitrary network $G = (V, E)$, try to compute some function f of their private inputs by communicating messages over the network. *Coding for interactive communication* asks for coding schemes that succeed to compute any such function even when the communication may be noisy. A fundamental question in this field is finding the maximal *rate* such coding schemes can achieve[1], that is, what is the minimal amount of redundancy coding schemes must add in order to successfully compute any function f despite the noise.

* Part of this work was done while the author was at Princeton University. Supported in part by NSF grant CCF-1149888.

[1] The rate of a coding scheme is the ratio between the communication of a protocol that performs over a noiseless network, to the communication of the coding scheme for the same task, over the noisy network.

The work of Rajagopalan and Schulman [13] gave an initial answer to this question, assuming stochastic noise (e.g., when each bit is being flipped independently with some fixed probability $\varepsilon < 1/2$): Let Δ be the maximal degree in G, then any (noiseless) protocol χ can be simulated with high probability over the noisy network by a protocol χ' with communication complexity $\mathsf{CC}(\chi') = \mathsf{CC}(\chi) \cdot O(\log(\Delta + 1))$. That is, for constant-degree networks such as the line or the cycle, the rate, $\mathsf{CC}(\chi)/\mathsf{CC}(\chi')$, is a constant bounded away from 0 while for highly-connected graphs such as the star or the complete graph, the rate goes to zero when n tends to infinity, i.e., the rate is $\Theta(1/\log n)$. The work of Alon et al. [2] shows that coding schemes with constant (non-zero) rate also exist for the complete graph, and other highly-connected graphs, hinting that it may be possible to achieve a constant rate coding scheme for any network G. This hope was terminated by Braverman et al. [3], showing that a rate of $\Theta(\log \log n/ \log n)$ is maximal for a specific task over the star network.

All the above works assume that the communication over the network is performed in rounds, where at every round *all the parties speak*, that is, $2|E|$ symbols are being communicated—one symbol over each channel of the network. A natural question to ask is: Why is such an assumption justifiable? One interpretation is that these previous works try to optimize the *round complexity*, as opposed to the communication complexity, and hence the assumption that all parties send a message to all other connected parties in each round.

In this work, our goal is to optimize the *communication complexity* (as opposed to round complexity), and we ask whether similar bounds on the rate follow if we don't force all parties to speak at every round.

Surprisingly, we show that the rate of coding schemes when G is a cycle (assuming channels with large alphabets) is at most $O(1/\log n)$. This corresponds to a lower bound of $\Omega(\log n)$ on the communication blowup. Informally, our main theorem is the following:

▶ **Theorem 1** (main, informal). *Let G be the cycle graph with n parties. Then, for any constant $\varepsilon < 1/2$ there exists a task whose communication complexity over the noiseless G is d while any coding scheme over any noisy graph G' (with noise parameter ε) that succeeds with high probability has communication complexity $\Omega(d \log n)$.*

The above theorem is quite surprising in light of the result of Rajagopalan and Schulman [13]: the maximal degree in the cycle graph is $\Delta = 2$, therefore the coding scheme of [13] (in the "everybody speaks" model) has a rate of $\Theta(1)$, regardless of the size of the network! In hindsight, the reason for this discrepancy is simple: the fact that everybody speaks in the model of [13] implies an inherent blowup in the communication of $O(n)$, which allows the parties to overcome errors. Indeed, assume that the "relevant" information for computing the function f progresses along the cycle: first p_1 sends a message to p_2 (while all the other parties have nothing to send in the meantime), only then p_2 has a message to send to p_3 and so on. While the "relevant" information is limited to a single edge on the network at any round, the fact that all the parties *must* speak at every round multiplies the effective communication by n both for the noiseless and noisy protocols, hence, it cancels out in the rate. On the other hand, this superfluous redundancy gives the parties the opportunity to correct previous errors in rounds where they are supposed to be idling if we weren't to force all the parties to speak at every round, and charge the parties according to the communication that actually happened.

For our lower bound we don't restrict the topology G' of the noisy graph, and allow any party to communicate with any other party (since anyways we count the actual communication, allowing the coding scheme to utilize any underlying graph just makes our lower bound stronger). Our lower bound actually works when the noise erases symbols instead of corrupting symbols (again, making the result stronger). The only "restrictive" assumption we have on the coding scheme is a fixed speaking order, independent of the inputs and the noise; see the "Communication Model" subsection below for a discussion regarding this assumption.

1.1 The Cycle Task

The noiseless task we use for Theorem 1 is an analog of the "pointer jumping" task over a cycle (see formal description in Section 2.5). Every party begins with a 2^n-ary tree of depth d, where each edge is labeled by a single bit. Each party begins at the root of its own tree, and the goal is to travel down the tree until it reaches a leaf.

It is most convenient to describe this task via the protocol that solves it. The parties are activated in a cyclic order (first p_1, then p_2, etc.). When p_i is activated, it receives a message of the form $(b_1, \dots, b_n)$ from p_{i-1}, corresponding to the labels of the edges traversed by the parties in the previous n rounds (padding with zeros as necessary in the first $n-1$ rounds). Upon receiving this message $\ell = (b_1, \dots, b_n)$ from p_{i-1}, p_i moves down from its current node to its ℓ-th child. Denoting by b the label of the edge it just took, p_i communicates to p_{i+1} the string $(b_2, \dots, b_n, b)$. This process continues until all parties reach a leaf at depth d in their input tree. The output is the path each party took along its tree.

In addition to the lower bound on the communication blowup, we show a coding scheme that successfully computes the cycle task over a noisy network with rate $\Theta(1/\log n)$, matching the rate of our lower bound for the cycle task (up to a constant).

▶ **Theorem 2** (upper bound, informal). *For any constant $\varepsilon < 1/2$, there exists a coding scheme that solves the cycle task of depth d over noisy channels with large alphabet and error parameter ε. The coding scheme obtains a rate of $\Theta(1/\log n)$ and a success probability of $1 - 2^{-\Omega(d \log n)}$.*

1.2 Communication Model

For our communication model, we assume that protocols have a *fixed* order of speaking. That is, we can assume that the protocol works in rounds so that the party that speaks at round i is determined in advance, independently of inputs and noise. This assumption is not without loss of generality, but we claim here that lifting this assumptions trivializes the model.

A completely unrestricted model would let the parties determine, at any round, whether they speak or not (cf. adaptive protocols for the two party case [1]; see also [8]). Such a model trivializes coding in the multiparty scenario, as parties can "encode" information via the path that the message is sent through: say p_1 wants to send a single bit to p_2. If the bit is 0, then p_1 sends the message directly. If it is 1, he can send the bit through p_n (who will relay it to p_2). Now, even if noise occurs[2], p_2 can figure out the bit in certainty by the identity of the sender.

Another model, which is not completely unrestricted but still trivializes coding in our scenario, is described in [11]. There, parties are allowed to decide whether to send a message or not (and to whom) according to the transcript so far. On its surface, this restriction avoids the "path encoding" described above, as parties are not allowed to change the delivery path according to their inputs. Nevertheless, such a model still enables error correction via "path selection", since the transcript still depends on the inputs. To give a simple example, assume a noiseless protocol in which the parties speak in order (p_1 sends a bit to p_2, then p_2 sends a bit to p_3, and so on). Such a protocol can be easily simulated over a noisy network in the [11] model: After p_i sends a bit to p_{i+1} the latter sends the bit back either directly (if it was a 0), or through p_{i-1} (if it was a 1); note that this decision is made as a function of the observed (possibly noisy) transcript, and thus it is allowed in that model. Now p_1 knows if its

[2] As long as we do not allow a stronger type of noise, i.e., insertions and deletions, see [4].

original bit reached p_{i+1} correctly or not and either retransmits the bit, or sends a message to p_{i+2} (who forwards it to p_{i+1}) to indicate that the bit was transferred correctly, and the simulation can move on to simulating the next bit of the noiseless protocol. In other words, this model reduces bit flips into erasures, and performing error correction from erasures with rate $1 - \varepsilon$ is fairly simple if the model allows changing the order of the speaking according to the observed noise.

To conclude, we show that there is a strong relation between the order of speaking and the obtained coding rate. On one hand, allowing the order of speaking to change adaptively, allows trivial coding schemes. On the other hand, fixing the order of speaking allows us to show an $\Omega(\log n)$ lower bound on the blowup for the cycle task. It is however possible that worse rates are possible for other tasks. In fact, we conjecture that the blowup can get as high as $\Omega(n)$ in specific situations, as a function of the "mismatch" between the order of speaking in the noiseless protocol and the coding scheme.

▶ **Conjecture 3.** *There exists a topology G and a noiseless protocol χ with a fixed order of speaking for which any coding scheme χ' with fixed order of speaking has rate at most $O(1/n)$.*

Our findings are reminiscent of the two-party case: if the simulation has a fixed order, the order of speaking in the original scheme determines the maximal rate of the coding; specifically, it is conjectured that there exists a protocol whose simulation has rate bounded away from 1. On the other hand, if the simulation is allowed to be adaptive, better rates (that approach 1) can be achieved. See discussion in [12, 9].

1.3 On Binary vs. Large Alphabet

While our main result (Theorem 1) assumes that the parties communicate symbols from a large alphabet, we also obtain a lower-bound for the case where the parties communicate bits, i.e., use a binary alphabet. Typically, constructing coding schemes over the binary alphabet is harder than constructing such schemes over a large alphabet. However, our result is a lower-bound rather than a coding scheme, and it is not necessarily so that the binary-case is stronger (nor is more difficult to obtain).

Nevertheless, the setting of binary channels and the setting of large-alphabet channels seem *incomparable*, since the alphabet applies both to the original (noiseless) protocol and to the coded (noisy) protocol. We elaborate on this in Section 5.

We extend our lower bound result also to the case where the noiseless protocol and the coding scheme are binary. Specifically, we show a lower bound of $\tilde{\Omega}(\log n)$ on the blowup of the communication for binary coding scheme over the star network (where the $\tilde{\Omega}$ notation means neglecting $\log \log n$ terms). Informally, the theorem is the following.

▶ **Theorem 4** (binary case, informal). *Let G be the star graph with n parties. Then, for any constant $\varepsilon < 1/2$ there exists a task whose communication complexity over the noiseless G is d while any coding scheme (with fixed order) over any noisy graph G' (with noise parameter ε) that succeeds with high probability has communication complexity $\tilde{\Omega}(d \log n)$.*

We stress that the above theorem is incomparable to the result of [3]: in our model parties may speak in an arbitrary (but fixed) order and are not forced to speak at every round. The task in consideration is the generalized jumping pointer described in [3]. The proof of Theorem 4 follows by combining the techniques developed in this paper for the cycle task with the techniques of [3] in quite a straightforward way, and we omit the details here.

1.4 Overview of our Techniques

The proof of our lower bound uses techniques from [3] for bounding the progress of a coding scheme χ' in simulating a noiseless protocol χ. As in [3], we use the notion of *cutoff* (Definition 11) that measures for any partial transcript of χ', how many cycles of the noiseless protocol χ are still not-simulated: when the cutoff is k, then the last $d-k$ cycles of χ are not simulated by the given transcript. More accurately (but still very informally) the transcript gives very little of information about the labels $\{b_i\}$ of the last $d-k$ cycles.

We show that any coding scheme that solves the cycle task with high probability must produce transcripts whose cutoff is $\approx d$, in expectation. Then, we show that for any segment in which the coding scheme communicates $O(n \log n)$ symbols, the cutoff advances by at most $O(1)$ cycles in expectation. Namely, let π be some fixed previous communication (including erasures), and let Π^{new} be the random variable describing the next $O(n \log n)$ symbols communicated by the coding scheme χ' (including erasures), then

$$\mathbb{E}[\mathsf{cutoff}(\pi \circ \Pi^{new}) \mid \mathsf{cutoff}(\pi) = k] \leq k + O(1).$$

In order for χ' to achieve an expected cutoff of $\approx d$, which is crucial for being correct with high probability, the coding scheme must communicate at least $\Omega(dn \log n)$ symbols, yielding a rate of $O(1/\log n)$.

The reason for the restricted progress in the cutoff is that many parties do not send any useful information in the segment Π^{new}, and that the next "move" (in the input tree) of each party depends on the moves of *all* the parties in the previous cycle. This means that most parties are missing a lot of crucial information in order to advance more than a constant number of levels in their input tree. Bounding the exact information sent by the parties (and thus the expected increase in the cutoff) is performed via the machinery of [3].

Showing that many parties give no information in any segment of $O(n \log n)$ rounds in our setting is a main technical difference from [3]. In the model of [3] all parties speak at every round, thus when the coding scheme communicates $O(n \log n)$ symbols we know that this communication is evenly spread—every party communicates exactly $O(\log n)$ symbols. In our setting, it is possible that the communication is evenly spread, but it is also possible that all $O(n \log n)$ symbols are communicated by a single party (or any other pattern in between). In the latter case, even if the noise targets the single party that speaks, that party could still convey $O(n \log n)$ bits of information by encoding its message via a standard error-correction code. Nevertheless, we show that there is a large set of parties that do not communicate any information in the new segment: either they don't speak at all, or they speak very little and their entire communication is completely erased by the noise. Furthermore, previous communication of these parties contains very little information on their labels in the last $d-k$ cycles to begin with.

The existence of this set of "erased" parties implies that the non-erased parties in this segment don't know how to proceed in their input tree, and their communication in that segment is "irrelevant" to the progress of the protocol, even if it is not erased by the noise. Indeed, assume a party's current node in its input tree is given, and assume that the party doesn't know which of its children it should go to next. The best that a party can do is to send all the labels below its current node. However, due to the fact that each node has 2^n children, that party cannot communicate more than $O(1)$ levels below its current node even if it gets to speak all the $O(n \log n)$ symbols in the next segment.

Naturally, the actual proof is more complex, since the party has some prior information about the children it should go (due to communication in previous rounds). This means that the children are not equiprobable and the party can communicate more information about (the labels of) more probable children. Still, since the arity of the input tree is so large and since the information on the next children it should take is rather little, the party will be able to communicate information on the labels of only $O(1)$ levels below its current node (in expectation).

1.5 Other Related Work

The field of coding for interactive communication was initiated by Schulman [14, 15] who formalized the question for the two-party case and developed basic techniques used for solving this task, either when the noise is stochastic (where each bit is flipped with some constant probability) or adversarial (where any subset of up to ε-fraction of the bits can be flipped). Later works in the two-party setting improve on the computational efficiency, success probability, and achievable rate of coding schemes. We refer the reader to [6] for a survey on interactive coding.

As mentioned above, the interactive coding in the multiparty case was initiated by Rajagopalan and Schulman [13] for the random noise case. Efficiency for this setting is obtained by Gelles, Moitra and Sahai [7]. The works of Alon et al. [2] and of Braverman et al. [3] identify the maximal rate obtainable over the complete graph and the star (and provide efficient schemes that obtain such a rate). The case of adversarial noise is considered by Jain, Kalai and Lewko [11] providing a scheme for topologies that have a star as a subgraph, that withstands $O(1/n)$-fraction of adversarial noise and blows up the communication by only a constant. The work of Hoza and Schulman [10] provides a coding scheme for any topology $G = (V, E)$ that withstands $O(1/n)$-fraction of noise and obtains a rate of $\Theta(n/|E| \log n)$.

2 Preliminaries: Notations, Model, Coding Schemes

2.1 Notations and Basic Properties

For $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, 2, \ldots, n\}$. The log() function is taken to base 2.

▶ **Definition 5.** The Hamming distance $\Delta(\sigma, \sigma')$ of two strings $\sigma = \sigma_1 \ldots \sigma_m$ and $\sigma' = \sigma'_1 \ldots \sigma'_m$ of length m over an alphabet S, is the number of positions i such that $\sigma_i \neq \sigma'_i$.

Given any tree $\mathcal{T}$ of depth N, we denote its first k levels by $\mathcal{T}^{\leq k}$ and its $N-k$ last levels by $\mathcal{T}^{>k}$. Given a path $z = (e_1, e_2, \ldots)$, we denote by $\mathcal{T}[z]$ the subtree of $\mathcal{T}$ rooted at the end of the path that begins at the root of $\mathcal{T}$ and follows the edge-sequence z. The above notation composes for sets of trees, e.g., if $\vec{\mathcal{T}} = (\mathcal{T}_1, \mathcal{T}_2, ...)$ is an array of trees and $\vec{z} = (z_1, z_2, \ldots)$ is an array of paths, then we let $\vec{\mathcal{T}}^{\leq k}$ denote the array $(\mathcal{T}_1^{\leq k}, \mathcal{T}_2^{\leq k}, ...)$ and $\vec{\mathcal{T}}[\vec{z}]$ the array $(\mathcal{T}_1[z_1], \mathcal{T}_2[z_2], ...)$, etc.

As a rule, we use small letters to denote specific values (e.g., the input x_i given to party i), and capital letters to denote the corresponding random variables (i.e., X_i for the random variable describing the input of the i-th party, when the inputs are drawn from some given distribution).

2.2 Multiparty Interactive Communication and Protocols

We assume an undirected network $G = (V, E)$ of $n = |V|$ parties, $p_1, \ldots, p_n$, where p_i is connected to p_j if and only if $(i, j) \in E$. Each party is given an input x_i, and is assumed to output $f_i(x_1, \ldots, x_n)$ at the end of the protocol.

A *protocol* dictates to each party what is the next symbol to send and over which channel, given the party's input, the round number, and all the symbols that the party has received so far. After a fixed and predetermined number of rounds, the protocol terminates and each party outputs a value as a function of its input and observed transcript. We assume that the order of speaking is *fixed* and is independent of the party's inputs and the noise. That is, it is determined in advance which channel is utilized at each round.

2.3 Noisy and Noiseless Networks

For the noiseless network, we focus on the *cycle network*. In the cycle, each party p_i is connected to p_{i-1} and p_{i+1} (all indices are modulo n).

For showing lower bounds over the noisy network we allow the parties to utilize the complete graph, avoiding any limitation on the protocol (since limiting the connectivity may harm the rate artificially). For our upper bound (coding scheme) the underlying topology is still the complete graph, however, the specific scheme we show does not need to communicate over all possible links—it communicates only over the cycle subgraph.

In a noisy network, each channel is assumed to suffer from random noise. For our lower bound we will assume each channel is a large-alphabet *erasure channel* EC_ε with erasure probability ε.

▶ **Definition 6.** For $\varepsilon \in [0,1]$ and a finite set Σ, the erasure channel over alphabet Σ is a random function $\mathsf{EC}_\varepsilon : \Sigma \to \Sigma \cup \{\perp\}$ which turns each input symbol into an erasure mark $\perp$ with probability ε, or otherwise keeps the symbol intact. When a channel is accessed multiple times, each instance is independent.

When considering upper bounds (coding schemes), channels with random noise are too weak (i.e., they can be reduced to erasure channels with high probability). Therefore, for our scheme we will assume a stronger type of noise we name semi-adversarial. Here, the transmissions that will be corrupted are determined in a random manner, however the received symbol of a corrupted transmission is determined *adversarially*; see discussion in Section 3.

▶ **Definition 7.** For $\varepsilon \in [0,1]$ and a finite set Σ, the semi-adversarial noisy channel over alphabet Σ is a random function $\mathsf{SAC}_\varepsilon : \Sigma \to \Sigma$ which corrupts any input symbol with probability ε, independently per instance. Once a symbol is corrupted, it may turn into any symbol in Σ, determined adversarially by the channel (possibly, all the corrupted symbols are chosen in a dependent manner).

2.4 Communication Complexity

For any protocol χ communicating symbols from an alphabet Σ, denote by $|\chi|$ the maximum number of symbols communicated by any execution of χ. Since we assume the order of speaking is fixed regardless of the inputs (and noise), each execution of χ has exactly $|\chi|$ number of symbols communicated. The communication complexity of χ is given by $\mathsf{CC}(\chi) = |\chi| \cdot \log|\Sigma|$.

2.5 The Cycle Task

In this section we define the cycle task and discuss a simple protocol that solves it over the noiseless cycle network.

Recall we have n parties $\{p_1, \dots, p_n\}$ where each p_i receives the input x_i. We assume each input x_i is a labeled $|\Sigma|$-ary tree of depth d, where $\Sigma = \{0,1\}^n$ and each edge in the tree is labeled by a single bit.

The output of p_i is a simple root-to-leaf path (of length d) denoted by path_i, and the complete task output is denoted by $\mathsf{path} = (\mathsf{path}_1, \dots, \mathsf{path}_n)$. We define the output in an inductive manner. For $i \in [n]$ and $j \in [d]$, let $\mathsf{path}_i(j) \in \{0,1\}^n$ denote the (index of the) j-th edge of path_i. Moreover, let $b_i(j) \in \{0,1\}$ denote the label of the edge that corresponds to $\mathsf{path}_i(j)$. For the induction basis, assume $b_i(j) = 0$ for all $i \in [n]$ and $j \leq 0$.

For $j \geq 1$, and for $i \in [n]$ we define $\mathsf{path}_i(j)$ as a function of $\{\mathsf{path}_{i'}(j')\}_{(j',i')<(j,i)}$, where $(x,y) < (u,v)$ holds if $x < u$ or if both $x = u$ and $y < v$; note that this implies a total order on pairs (j,i). The value of $\mathsf{path}_i(j)$ is given by the labels $b_{i'}(j')$ for the $n-1$ pairs (j',i') preceding (j,i) according to the total order we defined. Namely,

$$\mathsf{path}_i(j) = (b_{i+1}(j-1), b_{i+2}(j-1) \dots, b_n(j-1), b_1(j), \dots, b_{i-2}(j), b_{i-1}(j)).$$

Note that the cycle task can be solved by a simple protocol as described in Section 1. The protocol works in "cycles" where each such cycle means repeating the following process for $p_1, p_2, \dots, p_n$ in order. During the j-th cycle p_i sends to p_{i+1} the value of $\mathsf{path}_i(j)$ along with the label $b_i(j)$ of the edge it just took. Now p_{i+1} can infer the value of $\mathsf{path}_{i+1}(j)$, and obtain the bit $b_{i+1}(j)$ labeling that edge in its input x_{i+1}. It follows that after d such "cycles" all parties reach a leaf at level d in their input, and can output path_i. Assuming the parties communicate symbols from Σ, the protocol communicates dn symbols[3] and has a communication complexity of dn^2 bits. It can be verified that the communication complexity of solving the cycle task is $\Theta(dn^2)$.

For our lower bound, we assume the inputs $X = (X_1, \dots, X_n)$ are sampled so that each label is uniform in $\{0,1\}$. We are looking for coding schemes that solve the above task with high probability over the inputs X, the noise and the randomness of the coding scheme.

3 Upper Bound: A Coding Scheme For The Cycle Task With Blowup $\Theta(\log n)$

Before showing a lower bound of $\Omega(\log n)$ on the communication blowup of the cycle task over noisy networks, let us provide a sketch for a coding scheme that achieves a communication blowup of $\Theta(\log n)$, rendering our lower bound tight for the cycle task. The key idea is that repeating each symbol for $\Theta(\log n)$ times reduces the error probability to polynomially small in the number of parties. Then, the event of an error is so rare that standard coding techniques (that recover from small amount of errors) succeed with overwhelming probability.

When considering random noise over large alphabet, notice that the analog of the binary-symmetric-channel—a channel that uniformly picks the corrupted symbol—is too weak. Indeed, the parties could use only a small fraction of the symbol space in order to "catch" errors with high probability, thus essentially reducing the noise model into the case of erasures, while keeping the asymptotic rate the same up to a constant (see, for instance, the blueberry code technique in [5]).

Hence, our upper bound is defined in the somewhat stronger noise-model, which we call *semi-adversarial*, formally defined in Definition 7. In this noise model, each symbol is corrupted with probability ε, independently across different symbols. However, once a symbol is corrupted, the output symbol of the channel is chosen *adversarially*, in a worst case manner.

[3] In fact, it is enough to use $\Sigma = \{0,1\}^{n-1}$. We will neglect this issue as it doesn't change the asymptotic behaviour of the communication complexity, nor the asymptotic rate of related coding schemes.

3.1 Coding Scheme For The Cycle Task

The construction of our coding utilizes a primitive known as tree codes (see [15]; also see [6]).

▶ **Definition 8.** A β-ary tree code of depth γ, distance α and alphabet σ is a prefix code $TC : [\beta]^{\leq\gamma} \to \sigma^{\leq\gamma}$ that satisfies the following. For any two strings $x, y \in [\beta]^{\ell}$ of the same length $\ell \leq \gamma$ whose first difference is at the i-th coordinate,

$$\Delta(TC(x), TC(y)) \geq \alpha(\ell - i + 1),$$

where $\Delta(\cdot,\cdot)$ is the Hamming distance.

Schulman [15] showed that infinite-depth tree codes exist, and described the tradeoff between their distance and arity to their alphabet size.

▶ **Lemma 9** ([15]). *For any fixed $\beta \in \mathbb{N}$ and $\alpha \in (0,1)$, there exists a finite alphabet σ of size $|\sigma| = \beta^{O(1/(1-\alpha))}$ which suffices to construct a β-ary tree code with distance α and any depth.*

Our coding scheme, denoted by χ', uses tools from [13], and adapts them to our communication-model in which the parties are not forced to speak at every round. Let χ be the noiseless protocol for the cycle task described in Section 2.5. Our coding scheme χ' simulates χ step by step, sending each symbol that χ sends using two levels of coding (tree code and repetition code). While the decoding cannot guarantee that a party correctly decodes *all* the symbols sent to him so far, the symbols that were sent earlier in the protocol will be decoded correctly with an increasing probability. The party can then verify that the symbols he has already sent during previous rounds are consistent with his current understanding of the decoded incoming transmissions. In case they are not, the party will transmit a special $\mathcal{B}$ symbol whose meaning at the recipient is to "delete" the last (non-$\mathcal{B}$) symbol it has received. By sending multiple $\mathcal{B}$ symbols, the party can delete any incorrect suffix of his outgoing transmissions, until they become consistent with his (current view of his) incoming transmissions.

In the coding scheme χ' the parties communicate over channels with alphabet of size $(|\Sigma| + 1)$ that corresponds to all the symbols of χ and the additional "back" symbol $\mathcal{B}$. We assume a tree code with input alphabet $O(\Sigma)$ (specifically, a $(|\Sigma| + 1)$-ary tree), distance $\alpha > \varepsilon$, and output alphabet of size $|\Sigma'| = |\Sigma|^{O_\varepsilon(1)}$. Such a tree code exists due to Lemma 9. The coding scheme is described in Figure 1.

▶ Remark. In Figure 1, "sending a symbol" means sending $k = O_\varepsilon(\log n)$ repetitions of the same symbol using a repetition code with failure probability at most n^{-10}.

▶ **Theorem 10.** *For any $\epsilon < 1/2$, the coding scheme χ' has rate $\Theta_\varepsilon\left(\frac{1}{\log n}\right)$ and success probability $\geq 1 - 2^{-\Omega_\varepsilon(d \log n)}$, assuming the communication is over a SAC_ε network.*

4 The Lower Bound

In this section we give an outline of the proof of our lower bound. The complete description as well as detailed proofs are deferred to the full paper. Following [3], we define the notion of *cutoff* which measures the progress in simulating the cycle task. We show that the cutoff of a simulation is correlated with the length of the correct simulated output, in the sense that if the cutoff is k, it is improbable that the simulation gives an output whose correct prefix is of length more than k. Hence, if a simulation is correct with high probability, the implied cutoff must be high (i.e., around d).

The coding scheme χ'.

1. Repeat the following for $d' = 100d$ times.
2. For $i = 1$ to n, perform the following for p_i:
   - **a.** Let $y \in (\Sigma')^{\leq d'}$ be all the received communication from p_{i-1} in all the previous rounds.
     - **i.** Decode y via the tree code to obtain $x \in (\Sigma \cup \{\mathcal{B}\})^{\leq d'}$, i.e., set $x = TC^{-1}(y)$.
     - **ii.** Parse x to obtain $x' = \mathsf{Parse}(x)$.

       The function $\mathsf{Parse}(x)$ is defined in the following manner: Process x symbol-by-symbol in order. When processing a symbol from Σ, copy it to the output register. When processing a $\mathcal{B}$ symbol, delete the last non-deleted symbol in the output register. For instance, the string '$abd\mathcal{B}ccc\mathcal{B}d\mathcal{B}\mathcal{B}d$' is parsed to the string '$abcd$'.
   - **b.** Let $z \in (\Sigma \cup \{\mathcal{B}\})^{\leq d'}$ be all the symbols communicated by p_i so far during the protocol (before the tree-code encoding); let $z' = \mathsf{Parse}(z)$. p_i checks the consistency of its parsed incoming string x' and its parsed outgoing transmissions z'. The consistency is checked according to what p_i should have communicated over the noiseless χ, given the communication x'.

     If all its (parsed) outgoing messages z' are consistent with the (parsed) incoming messages, the next symbol to be sent, σ, is determined according to χ (if χ has already terminated, set $\sigma = 0$).

     If p_i finds an inconsistency, the next symbol to be sent is $\sigma = \mathcal{B}$.
   - **c.** p_i encodes the next symbol using the tree code, that is, it sends to p_{i+1} the last symbol of $TC(z \circ \sigma)$.

Figure 1 The coding scheme χ' for the Cycle Task.

Recall that x_i is the input of the i-th party, and X_i is the random variable describing it; similarly, π is used to describe a specific (observed) transcript while Π is the corresponding random variable. Also recall that the output of the i-th party is path_i describing the root-to-leaf path that the party traversed along x_i. Finally, recall that we denote by $\mathsf{path}_i(k)$ the first k edges in path_i and by $x_i[\mathsf{path}_i(k)]$ the subtree of x_i rooted at the end of $\mathsf{path}_i(k)$.

▶ **Definition 11** (Cutoff). For any transcript π, and any input $x = (x_1, \ldots, x_n)$, the *cutoff of the protocol*, denoted by $\mathsf{cutoff}(\pi, x)$, is the minimal number k, such that

$$\sum_{i=1}^{n} I(X_i[\mathsf{path}_i(k)] \mid \Pi = \pi, \mathsf{PATH}(k) = \mathsf{path}(k)) \leq 0.01n. \tag{1}$$

We note that if $\mathsf{cutoff}(\pi, x) = k$ then for any x' such that $x'^{\leq k} = x^{\leq k}$, it holds that $\mathsf{cutoff}(\pi, x') = k$. Furthermore, the cutoff is only a function of the path up to level k, that is, if $\mathsf{cutoff}(\pi, x) = k$ then for any input x' that has the same $\mathsf{path}(k)$ it holds that $\mathsf{cutoff}(\pi, x') = k$; This property allows us to abuse notation and write $\mathsf{cutoff}(\pi, \mathsf{path}(k)) = k$, when the path is fixed but we do not care about the specific input.

The following proposition shows that in order for a protocol to output the correct value with high probability, the cutoff (given the complete transcript) must be $\approx d$. Hence, protocols that succeed with high probability must produce transcripts whose cutoff is large in expectation.

▶ **Proposition 12.** *Fix a protocol that solves the cycle task of depth d over a network with n parties (with large enough n), that succeeds with probability at least $1/5$ on average, i.e., a protocol for which $\Pr_{X,\Pi}[\textit{correct output}] \geq 1/5$. Then,*

$$\mathbb{E}_{X,\Pi}[\mathsf{cutoff}(\Pi, X)] \geq \frac{d}{10}.$$

Our main theorem shows that in order to obtain a coding with such a high cutoff (which is required for high success probability) a communication blowup of $\Omega(\log n)$ is necessary.

▶ **Theorem 13.** *For any $\varepsilon \in (0, 1)$ there exists a constant $c = c(\varepsilon)$ such that for large enough n, any protocol that solves the cycle task of depth d over a network with n parties communicating less than $cd \cdot n \log n$ symbols assuming each communication channel is an EC_ε, has a success probability at most $1/5$.*

The main idea is to show that $O(n \log n)$ symbols sent by the simulation can increase the cutoff by at most $O(1)$, in expectation. That is, $O(\log n)$ cycles of the simulation are required in order to advance $O(1)$ cycles of the original protocol, giving a rate of $O(1/\log n)$.

Assume that given the (partial) observed transcript π and some path $\mathsf{path}(\ell)$, the cutoff of the coding scheme is ℓ, that is, $\mathsf{cutoff}(\pi, \mathsf{path}(\ell)) = \ell$. Then, assume we let the coding scheme communicate another $\delta \cdot n \log n$ symbols for some parameter $\delta = \delta(\varepsilon)$ we set later. We denote these new observed (potentially erased) symbols by Π^{new}; This is a random variable that depends on the noise and the randomness of the protocol. The claim is that the new cutoff (i.e., with respect to $\pi \circ \Pi^{new}$), is bounded by $\ell + O(1)$ in expectation.

▶ **Proposition 14.** *For any $\ell \leq d$, any $\mathsf{path}(\ell)$ and any transcript π,*

$$\mathbb{E}[\mathsf{cutoff}(\pi \circ \Pi^{new}, X) \mid \Pi = \pi, \mathsf{PATH}(\ell) = \mathsf{path}(\ell), \mathsf{cutoff}(\pi, \mathsf{path}(\ell)) = \ell] \leq \ell + 500.$$

With the above proposition, the proof of the main theorem is immediate.

Proof of Theorem 13. Assume χ is a coding scheme that succeeds with probability at least $1/5$. Proposition 12 claims that the expected cutoff at the end of the protocol χ is at least $d/10$.

On the other hand, assume toward contradiction that χ communicates less than $c \cdot d \cdot n \log n$ symbols. Split χ's transcript into segments of $\delta \cdot n \log n$ transmissions each. Using Proposition 14, the cutoff at the end of χ is bounded in expectation by

$$cd \cdot n \log n \cdot \frac{1}{\delta n \log n} \cdot 500 \leq \frac{500c}{\delta} d.$$

By choosing, say, $c < \delta/5000$, we get that the expected cutoff at the end of χ is strictly less than $d/10$, contradicting Proposition 12. ◀

The proof of Proposition 14 is rather involved and the details are deferred to the full version of this work.

5 Discussion: On the Rate vs. the Channel's Alphabet

In this section we discuss the effect of the channel's alphabet size on the obtainable rate. We can consider four independent settings: binary/large alphabet at the original (noiseless) scheme vs. binary/large alphabet at the coding scheme. For any $n \in \mathbb{N}$ and for $orig, code \in \{b, l\}$ let $c_{orig,code}(n)$ be the infimum over all possible n-party functions f of the maximal rate

Table 1 The relations between maximal rates of coding schemes with {binary, large}-alphabet, given the noiseless protocol uses {binary, large}-alphabet.

	Coding Scheme χ'	
Noiseless Scheme χ	binary alphabet	large alphabet
binary alphabet	c_{bb}	$c_{bl} \geq \frac{c_{bb}}{\log\lvert\Sigma\rvert}$
large alphabet	$\Omega(c_{ll}) \leq c_{lb}$ $c_{bb} \leq c_{lb}$	$c_{ll} \geq c_{bl}$ $c_{ll} \geq \frac{c_{lb}}{\log\lvert\Sigma\rvert}$

obtainable when the original protocol χ for f is binary ($orig = b$) or with a large alphabet ($orig = l$) and the coding schemes χ' for f is binary or with a large alphabet ($code = b$ or $code = l$, respectively),

$$c_{orig,code}(n) = \inf_f \frac{\min_\chi \mathsf{CC}(\chi)}{\min_{\chi'} \mathsf{CC}(\chi')}.$$

The capacity of each setting—the maximal achievable rate in each setting—is defined to be the limit inferior of the above quantities when n tends to infinity,

$$c_{orig,code} = \liminf_{n\to\infty} c_{orig,code}(n).$$

We now explore relations between the four capacities. See Table 1 for a summary of the relations between the capacities of the different settings.

Any binary coding can be simulated by a large-alphabet coding by incurring a blowup of $\log|\Sigma|$, thus trivial relations are $c_{bl} \geq c_{bb}/\log|\Sigma|$ and $c_{ll} \geq c_{lb}/\log|\Sigma|$.

When the original protocol uses large alphabet, a large-alphabet coding can be reduced to a binary one by translating each symbol to a sequence of bits encoded with a standard error-correction code (so that the probability for the entire sequence to be decoded incorrectly is below ε; this can be done with a constant overhead). Thus $\Omega(c_{ll}) \leq c_{lb}$.

To see that $c_{lb} \geq c_{bb}$, note that we can convert the original large-alphabet protocol (that determines c_{lb}) into a binary one with the same communication complexity; this converted protocol may not be the hardest one for coding with a binary simulation, thus the rate we can achieve when coding it may be larger than the rate for the "worst" binary protocol, which determines c_{bb}. A similar reasoning yields $c_{ll} \geq c_{bl}$.

The above relations still allow c_{bb} to be either larger or smaller than c_{ll}, and their specific relation (as well as their feasibility with respect to a given underlying topology) remains an interesting open question.

References

1 Shweta Agrawal, Ran Gelles, and Amit Sahai. Adaptive protocols for interactive communication. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 595–599, 2016. doi:10.1109/ISIT.2016.7541368.

2 Noga Alon, Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Reliable communication over highly connected noisy networks. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, PODC '16, pages 165–173, 2016. doi:10.1145/2933057.2933085.

3 Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Constant-rate coding for multiparty interactive communication is impossible. In *Proceedings of the 48th*

Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, pages 999–1010, 2016. `doi:10.1145/2897518.2897563`.

4 Mark Braverman, Ran Gelles, Jieming Mao, and Rafail Ostrovsky. Coding for interactive communication correcting insertions and deletions. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 61:1–61:14, 2016. `doi:10.4230/LIPIcs.ICALP.2016.61`.

5 Matthew Franklin, Ran Gelles, Rafail Ostrovsky, and Leonard J. Schulman. Optimal coding for streaming authentication and interactive communication. *Information Theory, IEEE Transactions on*, 61(1):133–145, Jan 2015. `doi:10.1109/TIT.2014.2367094`.

6 Ran Gelles. Coding for interactive communication: A survey, 2015. URL: `http://www.eng.biu.ac.il/~gellesr/survey.pdf`.

7 Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient coding for interactive communication. *Information Theory, IEEE Transactions on*, 60(3):1899–1913, March 2014. `doi:10.1109/TIT.2013.2294186`.

8 Mohsen Ghaffari, Bernhard Haeupler, and Madhu Sudan. Optimal error rates for interactive coding I: Adaptivity and other settings. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC '14, pages 794–803, 2014. `doi:10.1145/2591796.2591872`.

9 Bernhard Haeupler. Interactive Channel Capacity Revisited. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, FOCS '14, pages 226–235, 2014. `doi:10.1109/FOCS.2014.32`.

10 William M. Hoza and Leonard J. Schulman. The adversarial noise threshold for distributed protocols. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 240–258, 2016. `doi:10.1137/1.9781611974331.ch18`.

11 Abhishek Jain, Yael Tauman Kalai, and Allison Lewko. Interactive coding for multiparty protocols. In *Proceedings of the 6th Conference on Innovations in Theoretical Computer Science, ITCS '15*, pages 1–10, 2015. `doi:10.1145/2688073.2688109`.

12 Gillat Kol and Ran Raz. Interactive channel capacity. In *STOC '13: Proceedings of the 45th annual ACM Symposium on theory of computing*, pages 715–724, 2013. `doi:10.1145/2488608.2488699`.

13 Sridhar Rajagopalan and Leonard Schulman. A coding theorem for distributed computation. In *STOC '94: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 790–799, 1994. `doi:10.1145/195058.195462`.

14 Leonard J. Schulman. Communication on noisy channels: a coding theorem for computation. *Foundations of Computer Science, Annual IEEE Symposium on*, pages 724–733, 1992. `doi:10.1109/SFCS.1992.267778`.

15 Leonard J. Schulman. Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6):1745–1756, 1996. `doi:10.1109/18.556671`.