Optimal Short-Circuit Resilient Formulas

Mark Braverman
Department of Computer Science, Princeton University, USA
mbraverm@cs.princeton.edu

Klim Efremenko
Computer Science Department, Ben-Gurion University, Beer Sheba, Israel
klimefrem@gmail.com

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

Michael A. Yitayew
Department of Computer Science, Princeton University, USA

Abstract

We consider fault-tolerant boolean formulas in which the output of a faulty gate is short-circuited to one of the gate's inputs. A recent result by Kalai et al. [FOCS 2012] converts any boolean formula into a resilient formula of polynomial size that works correctly if less than a fraction 1/6 of the gates (on every input-to-output path) are faulty. We improve the result of Kalai et al., and show how to efficiently fortify any boolean formula against a fraction 1/5 of short-circuit gates per path, with only a polynomial blowup in size. We additionally show that it is impossible to obtain formulas with higher resilience and sub-exponential growth in size.

Towards our results, we consider interactive coding schemes when noiseless feedback is present; these produce resilient boolean formulas via a Karchmer-Wigderson relation. We develop a coding scheme that resists up to a fraction 1/5 of corrupted transmissions *in each direction of the interactive channel*. We further show that such a level of noise is maximal for coding schemes with sub-exponential blowup in communication. Our coding scheme takes a surprising inspiration from Blockchain technology.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography; Theory of computation → Interactive computation; Theory of computation → Design and analysis of algorithms

Keywords and phrases Circuit Complexity, Noise-Resilient Circuits, Interactive Coding, Coding Theory, Karchmer-Wigderson Games

Digital Object Identifier 10.4230/LIPIcs.CCC.2019.10

Related Version A full version of the paper is available at `https://arxiv.org/abs/1807.05014`.

Funding *Mark Braverman*: Supported in part by an NSF CAREER award (CCF-1149888), NSF CCF-1525342, a Packard Fellowship in Science and Engineering, and the Simons Collaboration on Algorithms and Geometry.
Klim Efremenko: Supported in part by the Israel Science Foundation (ISF) through grant No. 1456/18.
Ran Gelles: Supported in part by the Israel Science Foundation (ISF) through grant No. 1078/17.

Acknowledgements The authors would like to thank Raghuvansh Saxena and the anonymous reviewer for spotting an error in a preliminary version of this manuscript.

34th Computational Complexity Conference (CCC 2019).
Editor: Amir Shpilka; Article No. 10; pp. 10:1–10:22

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Kleitman, Leighton and Ma [21] asked the following question: assume you wish to build a logic circuit C from AND and OR gates, however, due to some confusion, some small amount of AND gates were placed in the box of the OR gates (and vice versa), and there is no way to distinguish between the two types of gates just by looking at them. Can you construct a "resilient" logic circuit C' that computes the same functionality of C, even if some (small amount) of the AND gates are replaced with OR gates (and vice versa)?

The above toy question is a special case of a more general type of noise (faulty gates) known as *short-circuit* noise. In this model, a faulty gate "short-circuits" one of its input-legs to the output-leg. That is, the output of the gate is determined by the value of one of its input-legs. The specific input that is connected to the output is determined by an all-powerful adversary, possibly as a function of the input to the circuit. This model is equivalent to a setting in which a faulty gate can be replaced with an arbitrary function g, as long as it holds that $g(0,0) = 0$ and $g(1,1) = 1$. Note that this type of noise is different from the so-called von Neumann noise model for circuits [29], in which the noise flips the value of each wire in the circuit independently with probability p. See [21, 19] and references therein for a comparison between these two separate models.

The first solution to the above question – constructing circuits that are resilient to short-circuit faults – was provided by Kleitman et al. [21]. They show that for any number e, a circuit of size $|C|$ gates can be transformed into a "resilient" circuit of size $|C'|$ that behaves correctly even if up to e of its gates are faulty (short-circuited), and it holds that $|C'| \leq O(e \cdot |C| + e^{\log 3})$.

Further progress was made by Kalai, Lewko, and Rao [19] showing, for any constant $\varepsilon > 0$, how to convert any formula[1] F of size $|F|$ into a resilient formula F' of size $|F'| = \text{poly}_\varepsilon(|F|)$ such that F' computes the same function that F computes, as long as at most $(\frac{1}{6} - \varepsilon)$-fraction of the gates in *any input-to-output path* in F' suffer from short-circuit noise. Kalai et al. explicitly leave open the question of finding the *optimal* fraction of faulty gates for a resilient formula F'.[2]

We make further progress on the above open question and show that $\frac{1}{5}$ is a tight bound on the tolerable fraction of faulty gates per input-to-output path, conditioned that the increase in the size of the formula is sub-exponential. Namely, we show how to convert any formula to a resilient version that tolerates up to a fraction $\frac{1}{5} - \varepsilon$ of short-circuit gates per path,

▶ **Theorem 1** (Main, informal). *For any $\varepsilon > 0$, any formula F can be efficiently converted into a formula F' of size $|F'| = \text{poly}_\varepsilon(|F|)$ that computes the same function as F even when up to $\frac{1}{5} - \varepsilon$ of the gates in any of its input-to-output paths are short-circuited.*

We also show that our bound is tight. Namely, for an arbitrary formula F, it is impossible to make a resilient version (of sub-exponential size in $|F|$) that tolerates a fraction $\frac{1}{5}$ (or more) of short-circuit gates per path.

▶ **Theorem 2** (Converse). *There exists a formula F for computing some function f, such that no formula F' of size $|F'| = o(\exp(|F|))$ that computes f is resilient to a fraction $\frac{1}{5}$ of short-circuit noise in any of its input-to-output paths.*

---

[1] A formula is a circuit in which each gate has fan-out 1.

[2] For instance, it is clear that if all the gates in an input-to-output path can be short-circuited (i.e., the fraction of noise is 1), then the adversary has full control on the output of the circuit. Hence, the optimal noise rate for formulas lies within the range $[\frac{1}{6}, 1]$.

Similar to the work of Kalai et al. [19], a major ingredient in our result is a transformation, known as the Karchmer-Wigderson transformation [20] (hereinafter, the KW-transformation), between a formula that computes a boolean function f, and a two-party interactive communication protocol for a task related to f which we denote the KW-game for f, or KW_f for short. Similarly, a reverse KW-transformation converts protocols back to formulas; see below for more details on the KW-transformation. The work of Kalai et al. adapts the KW-transformation to a noisy setting in which the formula may suffer from short-circuit noise, and the protocol may suffer from channel noise. The "attack plan" in [19] for making a given formula F resilient to short-circuit noise is (i) apply the KW-transformation to obtain an interactive protocol π; (ii) Convert π to a noise-resilient protocol π' that tolerates up to δ-fraction of noise; (iii) apply the (reverse) KW-transformation on π' to obtain a formula F'. The analysis of [19] shows that the obtained F' is resilient to $\delta/2$ fraction of noise in any of its input-to-output paths.

The interactive protocols π, π' are defined in a setting where the parties have access to a noiseless feedback channel – the sender learns whether or not its transmission arrived correctly at the other side. Building upon recent progress in the field of coding for interactive protocols (see, e.g., [11]), Kalai et al. [19] construct a coding scheme for interactive protocols (with noiseless feedback) that features resilience of $\delta = \frac{1}{3} - \varepsilon$ for any $\varepsilon > 0$; this gives their result. Note that a resilience of $\delta = 1/3$ is maximal for interactive protocols in that setting [8], suggesting that new techniques must be introduced in order to improve the result by [19].

The loss in resilience witnessed in step (iii) stems from the fact that short-circuit noise affects formulas in a "one-sided" manner: a short-circuit of an AND gate can only turn the output from 0 to 1. A short-circuit in an OR gate can only turn the output from 1 to 0. The noisy AND gates are thus decoupled from the noisy OR gates: if the output of the circuit is 0, any amount of short-circuited OR gates will keep the output 0 and if the output is 1 any amount of short-circuited AND gates will keep the output 1.

Informally speaking, this decoupling reduces by half the resilience of circuits generated by the KW-transformation. Assume the formula F' obtained from the above process is resilient to δ'-fraction of noise. Then F' is correct if on a specific input-to-output path (a) at most δ'-fraction of the AND gates are short-circuited, but also if (b) at most δ'-fraction of the OR gates are short-circuited. Since the noise is decoupled, from (a) and (b) we get that F outputs the correct value even when $2\delta'$-fraction of the gates on that input-to-output path are noisy. Yet, the resilience of F' originates in the resilience of π' (step (iii) above). The KW-transformation limits the resilience of F' by the resilience of π', i.e., $2\delta' \leq \delta$, leading to a factor 2 loss.

We revisit the above line of thought and take a more careful noise analysis. Instead of bounding the *total* fraction of noise by some δ, we consider the case where the noise from Alice to Bob is bounded by some α while the noise in the other direction is bounded by β. A similar approach used by Braverman and Efremenko [6], yields interactive protocols (without noiseless feedback) with maximal resilience. In more detail, assume that the protocol π communicates n symbols overall. We define an (α, β)-corruption as any noise that corrupts up to αn symbols sent by Alice and up to βn symbols sent by Bob. We emphasize that the noise *fraction* on Alice transmissions is higher than α, since Alice speaks less than n symbols overall; the global noise fraction in this case is $\alpha + \beta$.

This distinction may be delicate but is instrumental. The KW-transformation translates a protocol of length n that is resilient to (α, β)-corruptions into a formula which is resilient to up to αn short-circuited AND gates *in addition to* up to βn short-circuited OR gates. When $\alpha = \beta$ the obtained formula is resilient to up to α-fraction of short-circuited gates in any input-to-output path, avoiding the factor 2 loss in resilience.

1.1 Techniques overview

Achievability: Coding schemes for noisy channels with noiseless feedback

We obtain resilient formulas by employing the approach of [19] described above. In order to increase the noise resilience to its optimal level, we develop a novel coding scheme which is resilient to $(1/5-\varepsilon, 1/5-\varepsilon)$-corruptions, assuming noiseless feedback.

The mechanism of our coding scheme resembles, in a sense, the *Blockchain technology* [23]. Given a protocol π_0 that assumes reliable channels, the parties simulate π_0 message by message. These messages may arrive at the other side correctly or not, however, a noiseless feedback channel allows each party to learn which of its messages made it through. With this knowledge, the party tries to create a "chain" of correct messages. Each message contains a pointer to the last message that was not corrupted by the channel. As time goes by, the chain grows and grows, and indicates the entire correct communication of that party. An appealing feature of this mechanism is the fact that whenever a transmission arrives correctly at the other side, the receiver learns *all* the correct transmissions so far. On the other hand, the receiver never knows whether a single received transmission (and the chain linked to it) is indeed correct.

The adversarial noise may corrupt up to $(1/5-\varepsilon)n$ of the messages sent by each party. We think of the adversary as trying to construct a different, corrupt, chain. Due to its limited budget, at the end of the coding scheme one of two things may happen. Either it is the case that the correct chain is the longest, or it is the case where the longest chain contains in its prefix a sufficient amount of uncorrupted transmissions.

Indeed, if the adversary tries to create its own chain, its length is bounded by $(1/5-\varepsilon)n$, while the correct chain is of length $2n/5$ at the least.[3] On the other hand, the adversary can create a longer chain which forks off the correct chain. As a simple example, consider the case where a party sends $\approx 2n/5$ messages which go through uncorrupted. Now the adversary starts corrupting the transmissions and extends the correct chain with $(1/5-\varepsilon)n$ corrupt messages.[4] The corrupt forked chain is of length $2n/5+(1/5-\varepsilon)n$ and may be longer than the correct chain. However, in this case, the information contained in the *uncorrupted* prefix of the corrupt forked chain is sufficient to simulate the entire transcript of π_0.

Another essential part of our coding scheme is its ability to alter the order of speaking according to the observed noise.[5] Most previous work usually follows the succeeding intuition. If party's transmissions were corrupted, then the information contained in these transmissions still needs to reach the other side. Therefore, the coding scheme should allow that party to speak more times. In this work we take the opposite direction – the more a party is corrupted at the first part of the protocol, the *less* it speaks in the later part. The intuition here is that if the adversary has already wasted its budget on the party, it cannot corrupt much of the sequential transmissions and we can reduce their amount. A resembling approach appears in [1].

---

[3] The order of speaking in the coding scheme depends on the noise and is not alternating. Therefore, it is not necessary that a party speak half of the times. See discussion below.

[4] This attack assumes that there are $n/5$ additional rounds where the same party speak. This assumption is usually false and serves only for this intuitive (yet unrealistic) example.

[5] Protocols that change their length or order of speaking as a function of the observed noise are called *adaptive* [15, 1]. Since these decisions are noise-dependent, the parties may disagree on the identity of the speaker in each round, e.g., both parties may decide to speak in a given round, etc. We emphasize that due to the noiseless feedback there is always a consensus regarding whose turn it is to speak next. Hence, while our scheme has a non-predetermined order of speaking, the scheme is *non-adaptive* by the terminology of [8]; see discussion in [8] and in Section 6 of [11].

One hurdle we face in constructing our coding scheme comes from the need to communicate pointers to previous messages using a small (constant-size) alphabet. Towards this end, we first show a coding scheme that works with a large alphabet that is capable of pointing back to any previous transmission. Next, we employ a variable-length coding, replacing each pointer with a large number of messages over a constant-size alphabet. We prove that this coding does not harm the resilience, leading to a coding scheme with a constant-size alphabet and optimal resilience to $(1/5-\varepsilon, 1/5-\varepsilon)$-corruptions.

Converse: Impossibility Bound

The converse proof consists of two parts. First, we show that for certain functions, any protocol resilient to $(1/5, 1/5)$-corruptions must have an exponential blowup in the communication. In the second part, we show a (noisy) KW-transformation from formulas to protocols. Together, we obtain an upper bound on the noise of formulas. Indeed, assuming that there is a "shallow" formula that is resilient to $(1/5, 1/5)$-corruptions, converting it into a protocol yields a "short" protocol with resilience to $(1/5, 1/5)$-corruptions. The existence of such a protocol contradicts the bound of the first part.

The bound on the resilience of protocols follows a natural technique of confusing a party between two possible inputs. We demonstrate that a $(1/5, 1/5)$-corruption suffices in making one party (say, Alice) observe exactly the same transcript whether Bob holds y or y'. Choosing x, y, y' such that the output of the protocol differs between (x, y) and (x, y'), leads to Alice erring on at least one of the two instances.

This idea *does not* work if the protocol is allowed to communicate a lot of information. To illustrate this point, assume $f : \Sigma^n \times \Sigma^n \to \Sigma^z$ defined over a channel with alphabet Σ. Consider a protocol where the parties send their inputs to the other side encoded via a standard Shannon error-correcting code of length $n' = O(n)$ symbols, with distance $1-\varepsilon$ for some small constant $\varepsilon > 0$. The protocol communicates $2n'$ symbols overall, and a valid $(1/5, 1/5)$-corruption may corrupt up to $2n'/5$ symbols *of each one of the codewords*. However, this does not suffice to invalidate the decoding of either of the codewords, since an error correcting code with distance ≈ 1 is capable of correcting up to $\approx n'/2$ corrupted symbols.

On the other hand, once we limit the communication of the protocol, even moderately, to around n symbols, the above encoding is not applicable anymore. Quite informally, our lower bound follows the intuition described below. We show the existence of a function f such that for any protocol that computes f in r rounds (where r is restricted as mentioned above), the following properties hold for one of the parties (stated below, without loss of generality, for Alice). There are inputs x, x', y, y' such that (1) $f(x, y) \neq f(x', y) \neq f(x', y')$ and (2) Alice speaks at most $r/5$ times during the first $2r/5$ rounds. Further, (3) when Alice holds x, the protocol communicates exactly the same messages during its first $2r/5$ rounds, whether Bob holds y or y' (assuming no channel noise is present).

When we bound the protocol to these conditions, a $(1/5, 1/5)$-corruption is strong enough to make the transcript identical from Alice's point of view on (x', y) and (x', y'), implying the protocol cannot be resilient to such an attack. In more details, we now describe an attack and assume Bob speaks at most $2r/5$ times beyond round number $2r/5$, given the attack. [If Bob speaks more, then an equivalent attack will be able to confuse Bob rather than Alice.] The attack changes the first $2r/5$ rounds as if Alice holds x rather than x'; this amounts to corrupting at most $r/5$ transmissions by Alice due to property (2). Bob behaves the same regardless of its input due to property (3). From round $2r/5$ and beyond, the attack corrupts Bob's messages so that the next $r/5$ symbols Bob sends are consistent

with y and the following $r/5$ symbols Bob communicates are consistent with y'. Since Bob speaks less than $2r/5$ times (given the above noise), the attack corrupts at most $r/5$ of Bob's transmissions after round $2r/5$.

Unfortunately, while the above shows that some functions f cannot be computed in a resilient manner, this argument cannot be applied towards a lower bound on resilient formulas. The reason is that the KW_f task is not *a function*, but rather *a relation* – multiple outputs may be valid for a single input. The attack on protocols described earlier shows that a $(1/5, 1/5)$-corruption drives the protocol to produce a different output than in the noiseless instance. However, it is possible that a resilient protocol gives a different *but correct* output.

Therefore, we need to extend the above argument so it applies to computations of arbitrary relations. Specifically, we consider the parity function on n bits and its related KW-game. We show the existence of inputs that satisfy conditions (2) and (3) above while requiring that the outputs of different inputs be disjoint. I.e., any possible output of (x', y) is invalid for (x, y) and for (x', y').

The last part of the converse proof requires developing a KW-transformation from formulas to protocols, in a *noise-resilience preserving* manner. Let us begin with some background on the (standard) KW-transformation. The KW game (or rather a slight adaptation we need for our purposes) is as follows. For a boolean function f on $\{0,1\}^n$, Alice gets an input x such that $f(x) = 0$ and Bob gets an input y such that $f(y) = 1$, their goal is to output a literal function $\ell(z)$ (i.e. one of the $2n$ functions of the form $\ell(z) = z_i$ or $\ell(z) = \neg z_i$) such that $\ell(x) = 0$ and $\ell(y) = 1$.

Let F be a boolean formula for f, consisting of $\vee$ and $\wedge$ gates, and where all the negations are pushed to the input layer (i.e. F is a monotone formula of the literals z_i, $\neg z_i$). The conversion of F to a protocol π for the KW_f game is as follows. View the formula as the protocol tree, with the literals at the bottom of the tree being the output literal function. Assign each $\wedge$ node to Alice, and each $\vee$ node to Bob.

The invariant maintained throughout the execution of the protocol is that if the protocol reaches a node v, then the value of v in F is 0 when evaluated on x, and 1 when evaluated on y. Each time when the protocol is at node v and it is Alice's turn to speak (thus v is an $\wedge$ gate in F), Alice sends the identity of a child which evaluates to 0 on x. Note that assuming the invariant holds for v, Alice can send the identity of such a child (since one of the inputs to an AND gate which outputs a 0 also evaluates to 0), while this child must evaluate to 1 on y assuming v evaluates to 1 on y. By maintaining this invariant, Alice and Bob arrive at the bottom, where they reach a literal evaluating to 0 on x and 1 on y. Note that there is some room for arbitrary decision making: if more than one child of v evaluates to 0 on x, Alice is free to choose any such child – the protocol will be valid for any such choice.

In this work we extend the above standard KW-transformation to the noisy-regime. Namely, we wish to convert a *resilient* formula into an interactive protocol π *while keeping the protocol resilient* to a similar level of channel noise. We note that the extension we need is completely different from previous uses of the KW-transformation. Indeed, for the achievability bound, a KW-transformation is used both in steps (i) and (iii) in the above outline of [19]. However, the instance used in step (i) assumes there is no noise, while the instance in step (iii) works in the other direction, i.e., it transforms (resilient) protocols to (resilient) formulas.

Similar to the standard transformation, our noisy KW-transformation starts by constructing a protocol tree based on the formula's structure, where every $\wedge$-gate is assigned to Alice and any $\vee$-gate to Bob. The main difference is in the decision making of how to proceeds when reaching a node v. The goal is to keep the invariant that the gate v in F evaluates to 0 on x and to 1 on y, *even when noise is present.*

When only one of v's descendants evaluates to 0 on x in F, Alice has no choice but to choose that child. However, when more than a single descendant evaluates to 0 on x, Alice's decision is less obvious. Moreover, this decision may affect the resilience of the protocol – it is possible that noise causes one of the descendants evaluate to 1 on that given x.

We observe, however, that one of v's children evaluates to 0 on x given *all the noise patterns* F is resilient against. The other children may still evaluate to 1 sometimes, as a function of the specific noise. Once we identify this special child that always evaluates to 0, Alice can safely choose it and maintain the invariant (and the correctness of the protocol), regardless of future noise. Giving some more details, we prove that if such a special child did not exist and all descendants could evaluate to both 0 and 1 as a function of the noise, then we could construct a noise E^* that would make all descendants evaluate to 1 on x simultanously. Hence, assuming the noise is E^*, the node v would evaluate to 1 on x, and consequently $F(x) = 1$. At the same time, we show that F is resilient to the noise E^*, so $F(x) = 0$ assuming the noise is E^*, and we reach a contradiction.

1.2 Other related work

The field of interactive coding schemes [11] started with the seminal line of work by Schulman [26, 25, 27]. Commonly, the goal is to compile interactive protocols into a noise-resilient version that has (1) good noise resilience; (2) good rate; and (3) good probability of success. Computational efficiency is another desired goal. Numerous works achieve these goals, either fully or partially [5, 13, 3, 9, 6, 14, 22, 16, 10], where the exact parameters depend on the communication and noise model.

Most related to this work are coding schemes in the setting where a noiseless feedback channel is present. Pankratov [24] gave the first interactive coding scheme that assumes noiseless feedback. The scheme of [24] aims to maximize the rate of the scheme assuming all communication passes over a binary symmetric channel (BSC) with flipping parameter ε (i.e., a channel that communicates bits, where every bit is flipped with probability ε, independently of other bits). Pankratov's scheme achieves a rate of $1 - O(\sqrt{\varepsilon})$ when $\varepsilon \to 0$. Gelles and Haeupler [12] improved the rate in that setting to $1 - O(\varepsilon \log 1/\varepsilon)$, which is the current state of the art. For the regime of large noise, Efremenko, Gelles, and Haeupler [8] provided coding schemes with maximal noise resilience, assuming noiseless feedback. They showed that the maximal resilience depends on the channel's alphabet size and on whether or not the order of speaking is noise-dependent. Specifically, they developed coding schemes with a noise-independent order of speaking and a constant rate that are resilient to $1/4 - \varepsilon$ and $1/6 - \varepsilon$ fraction of noise with a ternary and binary alphabet, respectively. When the order of speaking may depend on the noise, the resilience increases to $1/3 - \varepsilon$ for any alphabet size. They show that these noise levels are optimal and that no general coding scheme can resist higher levels of noise.

There has been tremendous work on coding for noisy channels with noiseless feedback in the one-way (non-interactive) communication setting, starting with the work of Shannon, Horstein, and Berlekamp [28, 18, 2]. It is known that the presence of feedback does not change the channel's capacity, however, it improves the error exponent. The maximal noise-resilience in this setting is also known. Recently, Haeupler, Kamath, and Velingker [17] considered deterministic and randomized codes that assume a partial presence of feedback.

2 Preliminaries

Notations

For integers $i \leq j$ we denote by $[i, j]$ the set $\{i, i+1, \ldots, j\}$ any by $[i]$ the set $\{1, \ldots, i\}$. For a string $s \in \Sigma^*$ and two indices $x, y \in \{1, \ldots, |s|\}$, $x < y$ we let $s[x, y] = s_x s_{x+1} \cdots s_y$. We will treat $\emptyset$ as the empty word, i.e., for any $a \in \Sigma^*$ we have $a \circ \emptyset = \emptyset \circ a = a$. For bits $a, b \in \{0, 1\}$ we let $a \oplus b = a + b \mod 2$, and $\bar{b} = 1 - b$. All logarithms are taken to base 2, unless the base is explicitly written.

Interactive Protocols

In the interactive setting we have two parties, Alice and Bob, which receive private inputs $x \in X$ and $y \in Y$, respectively. Their goal is to compute some predefined function $f(x, y) : X \times Y \to Z$ by sending messages to each other. A *Protocol* describes for each party the next message to send, given its input and the communication received so far. We assume the parties send symbols from a fixed alphabet Σ. The protocol also determines when the communication ends and the output value (as a function of the input and received communication).

Formally, an interactive protocol π can be seen as a $|\Sigma|$-ary tree (also referred to as the *protocol tree*), where each node v is assigned either to Alice or to Bob. For any v node assigned to Alice there exists a mapping $a_v : X \to \Sigma$ that maps the next symbol Alice should send, given her input. Similarly, for each one of Bob's nodes we set a mapping $b_v : Y \to \Sigma$. Each leaf is labeled with an element of Z. The output of the protocol on input (x, y) is the element at the leaf reached by starting at the root node, and traversing down the tree where at each internal node v owned by Alice (resp., Bob), if $a_v(x) = i$ (resp., $b_v(y) = i$) the protocol advances to the i-th child of v. We conveniently denote Alice's nodes by the set V_a and Bob's nodes by the set V_b. We may assume that all the nodes in a given protocol tree are reachable by some input $(x, y) \in X \times Y$ (otherwise, we can prune that branch without affecting the behaviour of the protocol). Note that the order of speaking in π is not necessarily alternating and it is possible the same party is the sender in consecutive rounds. For any given transcript T, we denote $\pi(\cdot \mid T)$ the instance of π assuming the history T. Specifically, assuming Alice is the sender in the next round (assuming the history so far is T), then the next communicated symbol is $\pi(x \mid T)$.

The length of a protocol, denoted $|\pi|$, is the length of the longest root-to-leaf path in the protocol tree, or equivalently, it is the maximal number of symbols the protocol communicates in any possible instantiation. In the following we assume that all instances have the same length $|\pi|$. The communication complexity of the protocol is

$$CC(\pi) = |\pi| \log |\Sigma|.$$

When Σ is constant (independent of the input size), we have $CC(\pi) = O(|\pi|)$.

Transmission Noise with Feedback

We will assume the communication channel may be noisy, that is, the received symbol may mismatch with the sent symbol. All the protocols considered in this work assume the setting of *noiseless feedback*: the sender always learns the symbol that the other side received (whether corrupted or not). The receiver, however, does not know whether the symbol it received is indeed the one sent to him.

A noise pattern is defined as $E \in \{0, 1, \ldots, |\Sigma| - 1, *\}^{|V_a| \cup |V_b|}$. For any node v, E_v denotes the symbol that the receiver gets for the transmission that is done when the protocol reaches the node v. Specifically, say v is an Alice-owned node, then if $E_v = *$, Bob receives the symbol sent by Alice; otherwise, $E_v \neq *$, Bob receives the symbol E_v. Note that due to the feedback, Alice learns that her transmission was corrupted as well as the symbol that Bob received, and the protocol descends to the node dictated by E_v. We denote by π_E the protocol π when the noise is dictated by E; we sometimes write π_0 for a run of the protocol with no transmission noise, i.e., with the pattern $E = *^{|V_a| \cup |V_b|}$.

We say that a protocol is *resilient* to a noise pattern E if for any $(x, y) \in X \times Y$ it holds that π_E outputs the same value as π_0. While it is common to limit the noise to a constant fraction of the transmissions, in this work we take a more careful look at the noise, and consider the exact way it affects the transmissions of each party.

▶ **Definition 3.** *An (α, β)-corruption, is a noise pattern that changes at most $\alpha|\pi|$ symbols sent by Alice and at most $\beta|\pi|$ symbols sent by Bob. Note that the effective (combined) noise rate is $(\alpha + \beta)$.*

3 Resilience to $(1/5, 1/5)$-Corruptions is Impossible

In this section we prove that no coding scheme with constant overhead can be resilient to a $(1/5, 1/5)$-corruption. To this end we show a specific $(1/5, 1/5)$-corruption that confuses any protocol for a specific function f that is "hard" to compute in linear communication. Our results *does not* apply to coding schemes with vanishing rates. In fact, if the communication is exponentially large, coding schemes with resilience higher than $1/5$ exist.[6]

Normally, we discuss the case where protocols compute a *function* $f : X \times Y \to Z$. While our converse bound on the resilience of interactive protocols works for some hard function (e.g., the pointer jumping), such a proof does not suffice towards our converse on the resilience of boolean circuits (Theorem 2). The reason is that the conversion between formulas to protocols does not yield a protocol that computes a function but rather a protocol that computes a *relation*. Recall that for any given function f and any input (x, y) such that $f(x) = 0$ and $f(y) = 1$, the KW-game for f, KW_f, outputs an index $i \in [n]$ for which $x_i \neq y_i$ (see Section 5.1 for a formal definition). However, multiple such indices may exist and each such index is a valid output.

Let X, Y, Z be finite set and $R \subseteq X \times Y \times Z$ be a ternary relation. For any $(x, y) \in X \times Y$ and a given relation R let $R(x, y) = \{z \mid (x, y, z) \in R\}$ be the set of all z that satisfy the relation for x, y. Given such a relation, a protocol that *computes the relation* is the following two-party task. Alice is given $x \in X$ and Bob is given $y \in Y$. The parties need to agree on some $z \in R(x, y)$. We say that (x, y) is a *valid input* for R, if $x \in X$ and $y \in Y$. We assume that for any valid input, $|R(x, y)| > 0$.

We now show an explicit relation for which no protocol (of "short"' length) is resilient to $(1/5, 1/5)$ corruptions. Specifically, in the rest of this section we consider the binary parity function on n bits, $par : \{0, 1\}^n \to \{0, 1\}$, defined for any $x \in \{0, 1\}^n$ by

$$par(x) = x_1 \oplus \cdots \oplus x_n.$$

[6] For instance, consider the scheme in which each party sends its input to the other side encoded via a standard (Shannon) error-correcting code with distance ≈ 1. This trivial protocol is resilient to $(1/4 - \varepsilon, 1/4 - \varepsilon)$-corruption, yet its rate is 0.

Let $X = \{x \in \{0,1\}^n \mid par(x) = 0\}$ and $Y = \{y \in \{0,1\}^n \mid par(y) = 1\}$. We let $KW_{par} \subseteq X \times Y \times [n]$ be the KW-game for the parity function, defined by

$$KW_{par} = \{(x, y, z) \mid par(x) = 0 \wedge par(y) = 1 \wedge x_z \neq y_z\}.$$

We will need the following technical claim.

▷ Claim 4. Let $Y \subseteq \{0,1\}^n$. If $|Y| \geq 2^{n/2} + 1$ then there exist two elements $y_1, y_2 \in Y$ such that $\langle y_1, y_2 \rangle = 1$. Furthermore, if $|Y| \geq 2^{(n+1)/2} + 1$ then there exist two elements $y_1, y_2 \in Y$ such that $\langle y_1, y_2 \rangle = 0$.

Proof. Consider the linear space $L = \text{span}\{Y\}$. By a counting argument, it holds that $\dim(L) > n/2$. Let $L^\perp$ be the space orthogonal to L. Then $\dim(L) + \dim(L^\perp) = n$ and $\dim(L^\perp) < n/2$. Hence, $|L^\perp| < 2^{n/2}$ and thus $Y \not\subset L^\perp$. Let $y_1 \in Y \setminus L^\perp$. Since $y_1 \notin L^\perp$ it means that there must exist some element in Y, say y_2, for which $\langle y_1, y_2 \rangle \neq 0$. It follows that these two elements satisfy the claim, $\langle y_1, y_2 \rangle = 1$.

For the second part of the claim, let us construct $\tilde{Y} = \{(y, 1) \in \{0,1\}^{n+1} : y \in Y\}$; this is merely the set Y with an additional coordinate which is always set to one (over a space of dimension $n + 1$). Note that $|Y'| = |Y| \geq 2^{(n+1)/2} + 1$ and we can use the first part of this claim to show that there exist two elements $(y_1, 1), (y_2, 1) \in \tilde{Y}$ such that $\langle (y_1, 1), (y_2, 1) \rangle = 1$, therefore, $\langle y_1, y_2 \rangle = 0$. ◁

▶ **Lemma 5.** *Let π be an interactive protocol for KW_{par} (with inputs of n bits) of length $|\pi| = r$ defined over a communication channel with alphabet Σ and noiseless feedback. Without loss of generality, let Alice be the party who speaks less in the first $2r/5$ rounds of π. Additionally, assume $n/3 > 2r \log |\Sigma|/5 + 1$.*

Then, there exist inputs $x, x' \in X$, $y, y' \in Y$ for which:

(1) *$\pi(x, y)$ and $\pi(x, y')$ agree on the first $2r/5$ rounds.*

(2) *During the first $2r/5$ rounds of the execution $\pi(x, y)$ Alice speaks fewer times than Bob.*

(3) *$KW_{par}(x', y) \cap KW_{par}(x', y') = \emptyset$ and $KW_{par}(x', y) \cap KW_{par}(x, y) = \emptyset$.*

Note that the above lemma assumes Alice is the party that speaks fewer times in the first $2r/5$ rounds of π when averaging on all possible inputs $(x, y) \in X \times Y$; otherwise, a symmetric lemma holds for Bob.

Proof. Let x be an input for Alice such that on most y's Alice speaks fewer times in the first $2n/5$ rounds of $\pi(x, y)$. Such an input must exist by our choice of Alice. Let

$$Y' = \left\{ y \in Y \;\middle|\; CC_A^{\leq 2r/5}(\pi(x, y)) \leq CC_B^{\leq 2r/5}(\pi(x, y)) \right\}$$

be the set of all inputs for Bob, where Alice speaks fewer times in the first $2r/5$ rounds of π assuming Alice holds the above x. By the choice of x, it holds that $|Y'| \geq 2^n/2$.

Consider the set of transcript prefixes of length $2r/5$ generated by π when Alice holds the above x and Bob holds some input from the set Y',

$$T_x = \{t[1, 2r/5] \mid t = \pi(x, y), y \in Y'\}.$$

Note that there are at most $(2|\Sigma|)^{2r/5}$ different prefixes of length $2r/5$ over Σ with an arbitrary order of speaking. Since we assumed $n/3 > 2r \log |\Sigma|/5 + 1$, we have for large enough n,

$$|Y'| \geq 2^{n-1} \geq (2^{(n+1)/2} + 1)2^{n/3} \geq (2^{(n+1)/2} + 1)2^{2r \log |\Sigma|/5+1} \geq \Upsilon |T_x|,$$

with $\Upsilon = 2^{(n+1)/2} + 1$. Using a pigeon-hole principle, there must be $y^1, y^2, \ldots, y^\Upsilon \in Y'$ such that $\{\pi(x, y^i)\}_{i=1}^{\Upsilon}$ agree on the first $2r/5$ rounds of the protocol – they have an identical order of speaking and they communicate the same information.

Next consider the set $\{\overline{x} \oplus y^i\}_{i=1}^{\Upsilon}$. Claim 4 guarantees that there exist two elements in that set such that $\langle \overline{x} \oplus y^i, \overline{x} \oplus y^j \rangle = par(\overline{x})$; these y^i, y^j will be our y, y'.

Note that Properties (1) and (2) of the lemma are satisfied by the above x, y, y'. We are left to show an input x' for Alice that satisfies property (3).

Based on the above x, y, y' we construct x' in the following manner. For any $i \in [n]$ set

$$x'_i = \begin{cases} y_i & y_i = y'_i \\ \overline{x_i} & y_i \neq y'_i \end{cases}.$$

The above x' is constructed such that outputs given by KW_{par} are disjoint if we change only the input of Alice or only the input of Bob. Formally,

▷ Claim 6. The following claims hold for the above x, x', y, y'

a. $par(x') = 0$

b. $KW_{par}(x', y) \cap KW_{par}(x', y') = \emptyset$

c. $KW_{par}(x', y) \cap KW_{par}(x, y) = \emptyset$ and $KW_{par}(x', y') \cap KW_{par}(x, y') = \emptyset$

Proof.

a. It is easy to check that $x'_i = ((\overline{x_i} \oplus y_i) \cdot (\overline{x_i} \oplus y'_i)) \oplus \overline{x_i}$. Therefore,

$$\begin{aligned} par(x') &= \bigoplus_{i=1}^{n} x'_i = \bigoplus_{i=1}^{n} (((\overline{x_i} \oplus y_i) \cdot (\overline{x_i} \oplus y'_i)) \oplus \overline{x_i}) \\ &= \langle \overline{x} \oplus y, \overline{x} \oplus y' \rangle \oplus par(\overline{x}). \end{aligned}$$

Since we picked y, y' for which $\langle \overline{x} \oplus y, \overline{x} \oplus y' \rangle = par(\overline{x})$, we conclude that $par(x') = 0$.

b. Assume towards contradiction that $i \in KW_{par}(x', y) \cap KW_{par}(x', y')$, i.e., $x'_i \neq y_i$ as well as $x'_i \neq y'_i$. However, x'_i, y_i, y'_i are all bits and these two inequalities imply $y_i = y'_i$. But then, $x'_i = y_i$ by the way we construct x', which is a contradiction.

c. Assume towards contradiction that $i \in KW_{par}(x', y) \cap KW_{par}(x, y)$. That is, $x'_i \neq y_i$ and $x_i \neq y_i$ which means that $x'_i = x_i$. On the other hand, by the construction of x', either $x'_i \neq x_i$ or $x'_i = y_i$. Both options lead to a contradiction. The proof of the second part is identical.

◁

The first claim proves that x' is a valid input, i.e., $x' \in X$. The other claims prove property (3) of the lemma and conclude its proof. ◀

Our main result in this section is the following Theorem, proving that no protocol for the KW_{par} can be resilient to a $(1/5, 1/5)$-corruption if its communication is bounded. This will imply that any coding scheme that is resilient to $(1/5, 1/5)$-corruption must have rate 0. Specifically, it cannot produce a protocol with a constant overhead with respect to the optimal protocol that computes KW_{par} over reliable channels.

▶ **Theorem 7.** *Any interactive protocol π that computes the relation KW_{par} by at most $|\pi| < \frac{5}{6}\frac{n-3}{\log|\Sigma|}$ rounds over a noisy channel with alphabet Σ and noiseless feedback, is not resilient to $(1/5, 1/5)$-corruptions.*

Proof. Let π be a protocol with $r < \frac{5}{6}\frac{n-3}{\log|\Sigma|}$ rounds communicating symbols from the alphabet Σ. Via Lemma 5, let $x_0, x_1 \in X$ and $y_0, y_1 \in Y$ be inputs that satisfy

(1) $\pi(x_0, y_0)$ and $\pi(x_0, y_1)$ agree on the first $2r/5$ rounds
(2) During the first $2r/5$ bits of the protocol $\pi(x_0, y_0)$ Alice speaks less than Bob.
(3) $KW_{par}(x_1, y_0) \cap KW_{par}(x_1, y_1) = \emptyset$ and $KW_{par}(x_1, y_0) \cap KW_{par}(x_0, y_0) = \emptyset$

We now generate a simulated transcript T and show that T is consistent with a $(1/5, 1/5)$-corruption of $\pi(x_1, y_0)$. Additionally, it is either the case that T is consistent with a $(1/5, 1/5)$-corruption of $\pi(x_1, y_1)$ or it is consistent with a $(1/5, 1/5)$-corruption of $\pi(x_0, y_0)$. In the first case, Alice is unable to distinguish the case where Bob holds y_0 and y_1; in the second, Bob cannot tell if Alice holds x_0 or x_1. The outputs for different inputs are distinct by property (3). Thus the confused party is bound to err on at least one of them.

Note that the simulated transcript T contains messages *received* by the two parties, which may be noisy. Due to the feedback, both parties learn T. Additionally, the order of speaking in π is entirely determined by (prefixes of) T. Specifically, if two different instances of π have the same received transcript by round j, the party to speak in round $j + 1$ is identical in both instances.

The string T is obtained in the following manner:
1. Run $\pi(x_0, y_0)$ for $2r/5$ rounds. Let T_1 be the generated transcript.
2. Run $\pi(x_1, y_0 \mid T_1)$ until Bob transmits $r/5$ additional symbols (unless π terminates beforehand). Let T_2 be the generated transcript.
3. (if $|T_1T_2| < r$) Run $\pi(x_1, y_1 \mid T_1T_2)$ until Bob transmits $r/5$ additional symbols (unless π terminates beforehand).
4. (if $|T_1T_2T_3| < r$), let T_4 describe $\pi(x_1, y_0 \mid T_1T_2T_3)$ until it terminates.
5. Set $T = T_1T_2T_3T_4$.

In case the above algorithm didn't execute Step i, for $i \in \{3, 4\}$, assume $T_i = \emptyset$.

We now show that T corresponds to a $(1/5, 1/5)$-corrupted execution of π for two different valid inputs with disjoint outputs. We consider two cases: (i) when Step 3 halts since T reached its maximal size of r symbols (i.e., when $T_4 = \emptyset$), and (ii) when Step 3 halts since Bob transmitted $r/5$ symbols in this step ($T_4 \neq \emptyset$).

case (i) $T_4 = \emptyset$. In this case we show that a $(1/5, 1/5)$-corruption suffices to make the executions of $\pi(x_1, y_0)$ and $\pi(x_1, y_1)$ look the same from Alice's point of view.

Let Π be the transcript of a noisy execution of $\pi(x_1, y_0)$ (defined shortly) and split Π into three parts $\Pi = \Pi_1\Pi_2\Pi_3$ that correspond in length to T_1, T_2, T_3. The noise changes all Alice transmissions in Π_1 so that they correspond to Alice's symbols in T_1; the noise changes all Bob's transmissions in Π_3 so that they correspond to Bob's transmissions in T_3. It is easy to verify that the obtained transcript Π of received messages is exactly T. Furthermore, the first part changes at most $r/5$ transmissions by Alice, since by property (2) Alice speaks fewer times in the first $2r/5$ of the instance $\pi(x_0, y_0)$. The second part changes at most $r/5$ transmissions of Bob since T_3 halts before Bob communicates additional $r/5$ transmissions. Hence the noise described above is a valid $(1/5, 1/5)$-corruption.

On the other hand, and abusing notations, consider a (noisy) instance of $\pi(x_1, y_1)$ and let $\Pi = \Pi_1\Pi_2\Pi_3$ be the received messages transcript split to parts that corresponds in length to T_1, T_2, T_3, assuming the following noise. Again, the noise changes all Alice's transmissions in Π_1 to be the corresponding symbols received in T_1. This makes the $2r/5$ first rounds of the received transcript look as an instance $\pi(x_0, y_1)$. By Property (1), these transmissions agree with the first $2r/5$ transmissions in the noiseless instance $\pi(x_0, y_0)$; hence, the corrupted Π_1 equals T_1. Next, the noise changes Bob's transmissions in Π_2 to correspond to T_2. The obtained transcript Π is then exactly T. Again, T_1 contains at

most $2r/5$ of Alice's transmissions, and T_2 contains at most $r/5$ transmissions of Bob by their definition. Hence, this is a valid $(1/5, 1/5)$-corruption.

We conclude by recalling that $KW_{par}(x_1, y_0) \cap KW_{par}(x_1, y_1) = \emptyset$, then Alice must be wrong on at least one of the above executions, since her view in both executions is the same. Note that above proof holds even when $T_3 = \emptyset$.

case (ii) $T_4 \neq \emptyset$. In this case we show a $(1/5, 1/5)$-corruptions that makes the executions of $\pi(x_0, y_0)$ and $\pi(x_1, y_0)$ look the same from Bob's point of view. We point out that Alice speaks at most $r/5$ times after Step 1. Indeed, Step 1 contains $2r/5$ rounds, and Steps 2–3 contain $2r/5$ rounds where Bob speaks, hence Alice may speak in at most another $r/5$ times after Step 1.

Let Π be the transcript of a noisy execution of $\pi(x_0, y_0)$ where the noise is defined below. Split Π into 4 parts $\Pi = \Pi_1\Pi_2\Pi_3\Pi_4$ that correspond in length to T_1, T_2, T_3, T_4. The noise changes all Alice's transmissions in $\Pi_2\Pi_3\Pi_4$ so that they match the corresponding symbols of T_2, T_3, T_4. As mentioned, this corrupts at most $r/5$ symbols. Additionally, the noise changes Bob's transmissions in Π_3 to correspond to T_3; this by definition entails $r/5$ corruptions of Bob's transmissions. The obtained transcript Π is exactly T.

On the other hand, and abusing notations again, consider a noisy execution of $\pi(x_1, y_0)$ denoted by $\Pi = \Pi_1\Pi_2\Pi_3\Pi_4$. Here the noise is defined as follows. The noise changes all Alice's transmissions in Π_1 to match the corresponding symbols of T_1. As before, the noise changes Bob's transmissions in Π_3 to match T_3. Now it holds that $\Pi = T$, while the noise corrupted at most $r/5$ of each party's transmissions.

We conclude by recalling that $KW_{par}(x_0, y_0) \cap KW_{par}(x_1, y_0) = \emptyset$. Thus, Bob must be wrong on at least one of the above executions, since his view in both executions is exactly the same. ◀

Note that KW_{par} has a protocol of length $O(\log n)$ assuming reliable channels.[7] Theorem 7 leads to the following conclusion.

▶ **Corollary 8.** *There exists an interactive protocol π_0 defined over a noiseless channel with feedback such that any protocol π that computes the same functionality as π_0 and is resilient to $(1/5, 1/5)$-corruptions (assuming noiseless feedback) must incur with an exponential blowup in the communication.*

As a consequence, any coding scheme that compiles any protocol into a $(1/5, 1/5)$-resilient version, must have rate zero.

4 A Coding Scheme with Large Alphabet

In this section we construct a coding scheme for interactive protocols assuming a noiseless feedback. We show that for any constant $\varepsilon > 0$, any protocol π_0 defined over noiseless channels (with noiseless feedback) can be simulated by a protocol $\pi = \pi_\varepsilon$ defined over noisy channels (with noiseless feedback) such that (1) $CC(\pi)/CC(\pi_0) = O_\varepsilon(1)$, and (2) π is resilient to $(1/5 - \varepsilon, 1/5 - \varepsilon)$-corruptions. The protocol π in this section communicates symbols from a large alphabet of polynomial size in π_0. In the full version [7] we show how to reduce the size of the alphabet.

On a high level, the coding scheme π simulates π_0 step by step. The availability of a noiseless feedback channel allows a party to notice when the channel alters a transmission sent by that party. The next time that party speaks, it will re-transmit its message and

[7] This can easily be seen, e.g., by considering a formula that computes the parity of n bits, and applying the Karchmer-Wigderson transformation [20].

"link" the new transmission to its latest uncorrupted transmission. That is, each message carries a "link" – a pointer to a previous message sent by the same party. By following the links, the receiver learns the "chain" of uncorrupted transmissions; the party considers all "off-chain" transmissions as corrupted.

The Parse procedure (in Algorithm 1) parses all the transmissions received so far and outputs the "current chain": the (rounds of the) transmissions linked by the latest *received* transmission. Note that once a new transmission arrives, the current chain possibly changes. Moreover, upon reception of a corrupt transmission, a corrupt chain may be retrieved.

The TempTranscript procedure determines the partial simulated transcript of π_0 according to messages received in π so far, i.e., according to the current chains. Again, the scheme considers only transmissions that are on-chain and ignore all off-chain transmissions. The partial simulated transcript is defined as the concatenation of all the messages that (a) were received uncorrupted and (b) that were generated according to the correct information.

To clarify this issue, consider round i where, without loss of generality, Alice sends the message m_i. The latter property means that the last transmission *received* by Alice prior to round i, must be uncorrupted. This ensures that Alice learns which transmissions (so far) are correct and which are not, *in both directions*. It follows that Alice has full information about the on-going simulation of π_0. In particular, she can generate the correct m_i that extends the simulation of π_0 by one symbol. The former property ensures that m_i itself, the correct extension of the simulation of π_0, indeed arrives uncorrupted at the other side.

In each round of the protocol, the parties construct the partial transcript implied by messages received so far. If the received transmission is uncorrupt, the TempTranscript procedure retrieves the correct implied transcript (i.e., the implied transcript is indeed a prefix of the transcript of π_0). Then, the parties simulate the next rounds of π_0 assuming the implied partial transcript. As long as there is no noise in two alternating rounds, the next transmission extends the simulation of π_0 by one symbol. Otherwise, the sent symbol may be wrong, however, it will be ignored in future rounds once the chains indicate that this transmission was generated due to false information. Finally, at the end of the protocol, the parties output the transcript implied by the *longest* chain. The main part of this section is proving that the longest chain indeed implies a complete and correct simulation of π_0.

An important property of the coding scheme is its adaptive order of speaking. The first $2n/5$ rounds are alternating. On later rounds the order of speaking is determined according to observed noise: the more corrupted transmissions a party has, the *less* the party gets to speak. In particular, the protocol is split into *epochs* of 2 or 3 rounds each. In the first two rounds of an epoch, the order is fixed: Alice speaks in the first round and Bob speaks in the second. Then, the parties estimate the noise each party suffered so far (namely, the length of their current chain) and decide whether or not the epoch has a third round as well as who gets to speak in that extra round. For Alice to be the speaker in the third epoch-round, her *current* chain must be of length less than $n/5$ while Bob's current chain must be longer than $n/5$; Bob gets to speak if his chain is of length less than $n/5$ while Alice's chain is longer than $n/5$. In all other cases, the epoch contains only two rounds. We emphasize that due to the noiseless feedback, both parties agree on the received symbols (in both sides), which implies they agree on the current chains in both side, and thus, on the order of speaking in every epoch. The Next procedure, which determines the next speaker according to the current received transcript, captures the above idea.

The coding scheme is depicted in Algorithm 1. In the full version of this paper [7] we analyze Algorithm 1 and prove the following.

▶ **Theorem 9.** *For any $\varepsilon > 0$ and any binary alternating protocol π_0, Algorithm 1 correctly simulates π_0 over a noisy channel with noiseless feedback and is resilient to any $(1/5 - \varepsilon, 1/5 - \varepsilon)$-corruption.*

In the full version we also show how to reduce the channel's alphabet so it becomes independent of n, and depends only on the noise parameter ε. This is done via quite standard techniques of variable-length coding, cf. [5, 6, 4].

Algorithm 1 (**Part I**): A coding scheme against $(1/5, 1/5)$-corruptions assuming noiseless feedback (Large Alphabet; Alice's side).

Input: A binary alternating protocol π_0 with feedback; noise parameter $1/5 - \epsilon$. Alice's input for π_0 is x. Let $\Sigma = [n] \times \{0, 1, \emptyset\}$.

Throughout the protocol, maintain S_A, R_A, R_B, the sent, received by Alice, and received by Bob (as indicated by the feedback) symbols communicated up to the current round, respectively.
for $i = 1$ to $n = |\pi_0|/\varepsilon$ **do**
  $p_{\text{next}} = \textsc{Next}(R_A, R_B)$ ▷ Determine the next party to speak
  if p_{next} =Alice **then**
    $T \leftarrow \textsc{TempTranscript}(S_A, R_A, R_B)$
    The next symbol $\sigma = (link, b)$ to be communicated is:
      $link$ is the latest *non-corrupted* round $link < i$ where Alice is the speaker
        (0 if no such round exists).
      $b = \pi_0(x \mid T)$ if Alice is the sender in π_0, otherwise (or if π_0 has completed) $b = \emptyset$.
  else
    (receive a symbol from Bob)
  end if
end for

$j \leftarrow \arg\max |\mathsf{Parse}(R_B^{\le j})|$
$j' \leftarrow \arg\max |\mathsf{Parse}(R_A^{\le j'})|$
Output $\textsc{TempTranscript}(S_A, R_A^{\le j'}, R_B^{\le j})$

5 Applications for Circuits with Short-Circuit Noise

In this section we show that the KW-transformation between formulas and protocols (and vice versa) extends to the noisy setting in a manner that preserves noise-resilience. Applying the results of Section 4 onto the realm of boolean formulas gives a construction of formulas resilient to an optimal level of a fraction $(1/5 - \varepsilon)$ of short-circuit gates in any input-to-output path. Additionally, the results of Section 3 imply that noise-resilience of $1/5$ is maximal for formulas (assuming polynomial overhead).

In the following subsections, we show how to convert between formulas and protocols without affecting the noise-resilience. If we start with a formula that is resilient to (α, β)-corruptions, our transformation yields a protocol resilient to (α, β)-corruptions (Proposition 20). Moreover, given a protocol resilient to (α, β)-corruptions, the transformation yields a formula which is resilient to a similar level of noise (Proposition 23). Most proofs are deferred to the full version of this work.

Algorithm 1 (Part II): the Parse, Next, and TempTranscript Procedures.

procedure PARSE($m_1, \ldots, m_t$)
  $Chain \leftarrow \emptyset$; $j \leftarrow t$;
  while $j > 0$ **do**
    $Chain \leftarrow Chain \cup \{j\}$
    $j \leftarrow m_j.link$
  return $Chain$
end procedure

procedure TEMPTRANSCRIPT(S_A, R_A, R_B) ▷ Procedure for Alice; Bob's algorithm is symmetric
  Set $G_B = \text{PARSE}(R_A)$
  Set G_A as all the rounds in which outgoing transmissions are not corrupted (as learnt by R_B, S_A)
  For any $i < n$, if $\mathsf{Prev}(i), i \in G_A \cup G_B \cup \{0\}$ add i to *GoodChain*
  Set T as the concatenation of all $\{b_i\}_{i \in GoodChain}$, where $\sigma_i = (link_i, b_i)$ is the symbol received in round i.
  return T
end procedure

procedure NEXT(R_A, R_B)
  $i \leftarrow |R_A| + |R_B| + 1$ ▷ We are at round i
  $j \leftarrow 1$; $\mathsf{SkipCnt}_A \leftarrow 0$, $\mathsf{SkipCnt}_B \leftarrow 0$
  loop
    if $i = j$ **then return** Alice ▷ The speakers in the first two rounds of each epoch are fixed
    if $i = j + 1$ **then return** Bob
    ▷ Update the Skip counters
    if $|\text{PARSE}(R_A^{<j+2})| \leq n/5$ **then** $\mathsf{SkipCnt}_B \leftarrow \mathsf{SkipCnt}_B + 1$
    if $|\text{PARSE}(R_B^{<j+2})| \leq n/5$ **then** $\mathsf{SkipCnt}_A \leftarrow \mathsf{SkipCnt}_A + 1$

    if $|\mathsf{Parse}(R_B^{<j+2})| \leq n/5 < |\mathsf{Parse}(R_A^{<j+2})|$ **then** ▷ The epoch contains a 3rd round whenever one skip counter increases but the other does not
      if $i = j + 2$ **then return** Bob
      else $j \leftarrow j + 3$
    else if $|\mathsf{Parse}(R_A^{<j+2})| \leq n/5 < |\mathsf{Parse}(R_B^{<j+2})|$ **then**
      if $i = j + 2$ **then return** Alice
      else $j \leftarrow j + 3$
    otherwise
      $j \leftarrow j + 2$ ▷ An epoch with only 2 rounds
  end loop
end procedure

Note: $R_A^{\leq j}$ is the prefix of R_A as received by the j-th round of the protocol (incl. j) and $R_A^{<j}$ excluding round j. $R_B^{\leq j}$ and $R_B^{<j}$ are similarly defined.

5.1 Preliminaries

Formulas

A formula $F(z)$ over n-bit inputs $z \in \{0,1\}^n$ is a k-ary tree where each node is a $\{\wedge, \vee\}$ gate with fan-in k and fan-out 1. [While our results apply to any k, in this section we will usually assume $k = 2$ for simplicity.] Each leaf is a literal (either z_i or $\neg z_i$). The value of a node v given the input $z \in \{0,1\}$, denoted $v(z) \in \{0,1\}$, is computed in a recursive manner: the value of a leaf is the value of the literal (given the specific input z); the value of an $\wedge$ gate is the boolean AND of the values of its k descendants, $v_0, \cdots, v_{k-1}$, that is $v(z) = v_0(z) \wedge \cdots \wedge v_{k-1}(z)$. The value of an OR gate is $v(z) = v_0(z) \vee \cdots \vee v_{k-1}(z)$. The output of the formula on z, $F(z)$, is the value of the root node. We say that F computes the function $f : \{0,1\}^n \to \{0,1\}$ if for any $z \in \{0,1\}^n$ it holds that $F(z) = f(z)$.

The depth of a formula, denoted $\text{depth}(F)$, is the longest root-to-leaf path in it. The size of a formula, denoted $|F|$, is the number of nodes it contains. We denote by $V_\wedge$ the set of all the $\wedge$ nodes, and by $V_\vee$ the set of all the $\vee$ nodes.

Karchmer-Wigderson Games

For any boolean function $f : \{0,1\}^n \to \{0,1\}$, the *Karchmer-Wigderson game* is the following interactive task. Alice is given an input $x \in f^{-1}(0)$ and Bob gets $y \in f^{-1}(1)$. Their task is to find an index $i \in [n]$ such that $x_i \neq y_i$. We are guaranteed that such an index exists since $f(x) = 0$ while $f(y) = 1$. We denote the above task by KW_f.

Karchmer and Wigderson [20] proved the following relation between formulas and protocols.

▶ **Theorem 10** ([20]). *For any function $f : \{0,1\}^n \to \{0,1\}$, the depth of the optimal formula for f equals the length of the optimal interactive protocol for KW_f.*

▶ Remark 11. In the above, formulas are assumed to have fan-in 2 and protocols are assumed to communicate bits. However, the same reasoning and conversion applies also for a more general case, where each $\wedge$, $\vee$ gate has fan-in k, and the protocol sends symbols from alphabet of size $|\Sigma| = k$.

Furthermore, while our claims below are stated and proved assuming fan-in 2, all our claims apply to any arbitrary fan-in k.

Short-Circuit Noise

A short circuit noise replaces the value of a specific node with the value of one of its descendants. A noise pattern $E \in \{0, 1, \ldots, k-1, *\}^{|V_\wedge| \cup |V_\vee|}$ defines for each node whether it is short-circuited and to which input. Specifically, if for some node v, $E_v = *$ then the gate is not corrupted and it behaves as defined above. Otherwise, the value of the node is the value of its E_v-th descendant, $v(z) = v_{E_v}(z)$. We denote by F_E the formula with short circuit pattern E; we sometime write F for the formula with no short-circuit noise, i.e. with the noise pattern $E = *^{|V_\wedge| \cup |V_\vee|}$.

We say that a circuit is resilient to a noise pattern E if for any $z \in \{0,1\}^n$ it holds that $F(z) = F_E(z)$.

▶ **Definition 12.** *We say that F is resilient to δ-fraction of noise if it is resilient to all noise patterns E in which the fraction of corrupted gates in any input-to-output path in F is at most δ.*

We can also be more delicate and distinguish between noise in $\wedge$-gates and $\vee$-gates.

▶ **Definition 13.** *An (α, β)-corruption of short-circuit errors, is a noise pattern on a formula F of depth n that changes at most αn $\wedge$-gates and at most βn $\vee$-gates in any input-to-output path in F.*

The following is immediate by definition.

▷ Claim 14. If, for some $\delta > 0$, the formula F is resilient to any (δ, δ)-corruption of short-circuit errors, then F is also resilient to δ-fraction of noise.

On its surface, the other direction does not necessarily hold: (δ, δ)-corruption may corrupt up to a fraction 2δ of the gates in each path, hence resilience to δ-fraction appears to be insufficient to resist all (δ, δ)-corruptions. Nevertheless, we argue that these two notions are indeed equivalent. The reason is that a short-circuit in an $\wedge$-gate can only turn the output from 0 to 1. A short-circuit in an $\vee$-gate can only turn the output from 1 to 0. Then, if a formula evaluates to 1 on some input, the output remains 1 regardless of any amount of short-circuited $\wedge$-gates. If the output is 0, it remains so regardless of any number of short-circuited $\vee$-gates. This observation was already made by Kalai et al. [19].

▶ **Lemma 15** ([19, Claim 7])**.** *Let F be a formula, z an input and E any error pattern. Let $E_\wedge$ be the error pattern induced by E on the $\wedge$ gates alone (no errors in $\vee$ gates); Let $E_\vee$ be the error pattern induced by E on the $\vee$ gates alone. It holds that if $F_{E_\wedge}(z) = 0$, then $F_E(z) = 0$ and if $F_{E_\vee}(z) = 1$ then $F_E(z) = 1$.*

The above lemma then implies that resilience to δ-fraction of noise corresponds to resilience to the same fraction of noise in both type of gates.

▶ **Lemma 16.** *If, for some $\delta > 0$, the formula F is resilient to a fraction δ of short-circuit noise, then F is also resilient to any (δ, δ)-corruption.*

5.2 From Formulas to Protocols

We begin with a KW-transformation for noisy formulas, given a *specific* noise pattern.

▶ **Definition 17** (Noisy KW-transformation)**.** *For any formula $F(z)$ and any noise pattern E for F, the noisy transformation of F_E yields an interactive protocol π^{F_E} defined as follows over the domain $F_E^{-1}(0) \times F_E^{-1}(1)$.*

- *The formula-tree is converted into a protocol tree, where every $\wedge$ gate becomes a node where Alice speaks and every $\vee$ gate becomes a node where Bob speaks.*
- *For a node v, the mapping $a_v(z)$ for $z \in F_E^{-1}(0)$ and the mapping $b_v(z)$ for $z \in F_E^{-1}(1)$ are set as follows. Consider the evaluation of the formula F_E on z.*
  - *If v is an $\wedge$ gate, write $v(z) = v_0(z) \wedge v_1(z)$ where v_0 and v_1 are v's left and right descendants in F, respectively. For any $z \in F_E^{-1}(0)$, if $v_0(z) = 0$ we set $a_v(z) = 0$; otherwise we set $a_v(z) = 1$.*
  - *For an $\vee$ gate and $z \in F_E^{-1}(1)$ denote $v(z) = v_0(z) \vee v_1(z)$, and set $b_v(z) = 0$ if $v_0(z) = 1$; otherwise $b_v(z) = 1$.*
- *A leaf of F marked with the literal z_i or $\neg z_i$ becomes a leaf (output) of the protocol with the same literal.*

▶ Remark 18. In the above definition, we assume that if both $v_0(z) = 0$ and $v_1(z) = 0$ (for $z \in F^{-1}(0)$), the protocol continues to the left child. This choice is arbitrary, and any other choice is valid and gives an alternative protocol which still satisfies Proposition 19 below.

For instance, we can have non-intersecting sets Z_0 and Z_1 that determine the inputs z for which we take the left or right child, respectively (assuming both subformulas evaluate to 0 exactly on $Z_0 \cup Z_1$).

Following the mapping between formulas and protocols, [19] made the observations that a short-circuit error in a formula translates to channel noise in the equivalent KW protocol, assuming both parties learn the noise, i.e., assuming noiseless feedback. We will sometimes abuse notations and identify a short-circuit noise pattern with a transmission noise pattern for a formula F and a protocol π that share the same underlying tree structure. Furthermore, we will denote the two different objects with the same identifier E.

Next, we claim that performing the protocol π^{F_E} assuming the channel noise E computes the KW game of the noisy formula F_E.

▶ **Proposition 19.** *Assume that $F_E(z)$ computes the function $f(z)$. Then, $\pi_E^{F_E}$ computes KW_f.*

The proof goes by induction, similar to the original KW proof. The short-circuit noise forces some gate's output to be the output of a specific sub-formula. At the same time, channel noise causes the protocol to proceed to the corresponding sub-protocol. Together, we show that the following invariant holds, given the noise E: for any reachable node v, $v(x) = 0$ and $v(y) = 1$. This implies that once a leaf is reached, Alice and Bob disagree on its value, hence it is a valid output for the KW-game.

The above suggests that, for a given noise E, we can construct a protocol resilient to E. The next proposition proves that this can be extended to a family of noise patterns. This yields our main proposition for converting formulas to protocols in a noise-preserving way.

▶ **Proposition 20.** *Let F be a (complete) formula that computes the function f and is resilient to (α, β)-corruption of short-circuit gates in every input-to-output path. Then, a noisy KW-transformation yields a protocol π (over channels with feedback) that solves KW_f and is resilient to (α, β)-corruptions.*

The conversion from resilient formulas into resilient protocols of Proposition 20 implies an upper bound on the maximal resilience of formulas, and proves Theorem 2.

▶ **Theorem 21.** *There exists a function $f : \{0,1\}^n \to Z$ such that no formula F that computes f with fan-in k and depth less than $r < \frac{5}{6}\frac{n-3}{\log|k|}$, is resilient to a fraction $1/5$ of short-circuit noise.*

Proof. For $z \in \{0,1\}^n$, let $par(z) = z_1 \oplus \cdots \oplus z_n$ be the parity function.

Let F be a formula that computes $par(z)$ with AND/OR gates of fan-in k and $depth(F) < \frac{5}{6}\frac{n-3}{\log k}$. Assume that F is resilient to a fraction $1/5$ of short-circuit noise. Lemma 16 shows that F is also resilient to $(1/5, 1/5)$-corruptions of short-circuits. Moreover, assume that the formula's underlying graph is a complete k-ary tree.[8] Then, using Proposition 20 we obtain an interactive protocol π for KW_{par} of length $|\pi| = depth(F) < \frac{5}{6}\frac{n-3}{\log k}$ that communicates symbol from alphabet of size $|\Sigma| = k$, and is resilient to $(1/5, 1/5)$-corruptions. This contradicts Theorem 7. ◀

[8] If F has a node that has missing children, we can duplicate one of its children to obtain a complete graph. This clearly does not change the functionality of F, nor its resilience.

Note that computing the parity of n bits can be done with a formula of depth $O(\log n)$. However, the above theorem shows that any resilient formula for the parity function will have an exponential blow-up in depth, and thus exponential blow-up in size.

▶ **Corollary 22.** *There is no coding scheme that converts any formula F of size s into a formula F' of size $o(\exp(s))$, such that F' computes the same function as F and is resilient to $1/5$-fraction of short-circuit gates on every input to output path.*

5.3 From Protocols to Formulas

Here we would like to prove that a resilient protocol implies a resilient formula.

▶ **Proposition 23.** *Let π be a protocol that solves KW_f for some function f and is resilient to (α, β)-corruption. The KW-transformation on the reachable protocol tree of π yields a formula F that computes f and is resilient to (α, β)-corruption of short-circuit noise in any of its input-to-output paths.*

The above proposition is in fact a reformulation of a result by Kalai, Lewko, and Rao [19], implied by Lemma 15 and by Lemma 8 in [19].

Using our coding scheme that is resilient to $(1/5 - \varepsilon, 1/5 - \varepsilon)$-corruptions we get that we can fortify any formula F so it becomes resilient to $(1/5 - \varepsilon)$-fraction of short-circuit noise, with only polynomial growth in size.

▶ **Theorem 24.** *For any $\varepsilon > 0$, any formula F of depth n and fan-in 2 that computes a function f can be efficiently converted into a formulas F' that computes f even up to $1/5 - \varepsilon$ of the gates in any of its input-to-output path are short-circuited. F' has a constant fan-in $O_\varepsilon(1)$ and depth $O(n/\varepsilon)$.*

Theorem 1 is an immediate corollary of the above theorem, by noting that

$$|F'| \leq k^{\mathrm{depth}(F')} = \left(2^{O(\log(1/\varepsilon))}\right)^{O((\log|F|)/\varepsilon)} = \mathrm{poly}_\varepsilon(|F|).$$

Here $k \approx \varepsilon^{-2}$ is the fan-in of $|F'|$ given by the alphabet size of the resilient interactive protocol π' constructed earlier.

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. Elwyn R. Berlekamp. *Block coding with noiseless feedback*. PhD thesis, Massachusetts Institute of Technology, 1964.
3. Zvika Brakerski, Yael T. Kalai, and Moni Naor. Fast Interactive Coding Against Adversarial Noise. *J. ACM*, 61(6):35:1–35:30, December 2014. doi:10.1145/2661628.
4. M. Braverman, R. Gelles, J. Mao, and R. Ostrovsky. Coding for Interactive Communication Correcting Insertions and Deletions. *IEEE Transactions on Information Theory*, 63(10):6256–6270, October 2017. doi:10.1109/TIT.2017.2734881.
5. M. Braverman and A. Rao. Toward Coding for Maximum Errors in Interactive Communication. *Information Theory, IEEE Transactions on*, 60(11):7248–7255, November 2014. doi:10.1109/TIT.2014.2353994.
6. Mark Braverman and Klim Efremenko. List and Unique Coding for Interactive Communication in the Presence of Adversarial Noise. *SIAM Journal on Computing*, 46(1):388–428, 2017. doi:10.1137/141002001.

7 Mark Braverman, Klim Efremenko, Ran Gelles, and Michael A. Yitayew. Optimal Short-Circuit Resilient Formulas. *CoRR*, arXiv:1807.05014, 2018. URL: http://arxiv.org/abs/1807.05014.

8 K. Efremenko, R. Gelles, and B. Haeupler. Maximal Noise in Interactive Communication Over Erasure Channels and Channels With Feedback. *IEEE Transactions on Information Theory*, 62(8):4575–4588, August 2016. doi:10.1109/TIT.2016.2582176.

9 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, January 2015. doi:10.1109/TIT.2014.2367094.

10 R. Gelles, B. Haeupler, G. Kol, N. Ron-Zewi, and A. Wigderson. Explicit Capacity Approaching Coding for Interactive Communication. *IEEE Transactions on Information Theory*, 64(10):6546–6560, October 2018. doi:10.1109/TIT.2018.2829764.

11 Ran Gelles. Coding for Interactive Communication: A Survey. *Foundations and Trends® in Theoretical Computer Science*, 13(1–2):1–157, 2017. doi:10.1561/0400000079.

12 Ran Gelles and Bernhard Haeupler. Capacity of Interactive Communication over Erasure Channels and Channels with Feedback. *SIAM Journal on Computing*, 46(4):1449–1472, 2017. doi:10.1137/15M1052202.

13 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.

14 Mohsen Ghaffari and Bernhard Haeupler. Optimal Error Rates for Interactive Coding II: Efficiency and List Decoding. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, FOCS '14, pages 394–403, 2014. doi:10.1109/FOCS.2014.49.

15 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.

16 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.

17 Bernhard Haeupler, Pritish Kamath, and Ameya Velingker. Communication with Partial Noiseless Feedback. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 881–897. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.881.

18 M. Horstein. Sequential transmission using noiseless feedback. *IEEE Transactions on Information Theory*, 9(3):136–143, July 1963. doi:10.1109/TIT.1963.1057832.

19 Y. T. Kalai, A. Lewko, and A. Rao. Formulas Resilient to Short-Circuit Errors. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 490–499, October 2012. doi:10.1109/FOCS.2012.69.

20 Mauricio Karchmer and Avi Wigderson. Monotone Circuits for Connectivity Require Super-Logarithmic Depth. *SIAM Journal on Discrete Mathematics*, 3(2):255–265, 1990. doi:10.1137/0403021.

21 Dan Kleitman, Tom Leighton, and Yuan Ma. On the Design of Reliable Boolean Circuits That Contain Partially Unreliable Gates. *J. Comput. Syst. Sci.*, 55(3):385–401, December 1997. doi:10.1006/jcss.1997.1531.

22 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.

23 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL: https://bitcoin.org/bitcoin.pdf.

24 Denis Pankratov. On the Power of Feedback in Interactive Channels. [Online:] http://people.cs.uchicago.edu/~pankratov/papers/feedback.pdf, 2013.

25 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.

26 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.

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

28 C. Shannon. The zero error capacity of a noisy channel. *IRE Transactions on Information Theory*, 2(3):8–19, September 1956. doi:10.1109/TIT.1956.1056798.

29 John von Neumann. Probabilistic logics and the synthesis of reliable organisms from unreliable components. In C. E. Shannon and J. McCarthy, editors, *Automata studies*, volume 34 of *Annals of Mathematics Studies*, pages 43–98. Princeton University Press, Princeton, 1956.