Electronic Colloquium on Computational Complexity, Report No. 166 (2016)

Optimal Resilience for Short-Circuit Noise in Formulas

Mark Braverman* Ran Gelles† Michael A. Yitayew*

*Department of Computer Science, Princeton University
mbraverm@cs.princeton.edu, myitayew@princeton.edu

†Faculty of Engineering, Bar-Ilan University
ran.gelles@biu.ac.il

Abstract

We show an efficient method for converting a logic circuit of gates with fan-out 1 into an equivalent circuit that works even if some fraction of its gates are short-circuited, i.e., their output is short-circuited to one of their inputs. Our conversion can be applied to any circuit with fan-in $k \geq 2$, yielding a resilient circuit whose size is polynomial in the size of the (non-resilient) input circuit.

The resilient circuit gives the correct output as long as less than $1/3$ of the gates in any of its input-to-output paths are corrupted. Furthermore, we prove that a resilience level of $1/3$ is optimal (maximal) for this type of faulty gates. This fully answers an open question by Kalai et al. (FOCS 2012).

ISSN 1433-8092

1 Introduction

Kleitman, Leighton and Ma [KLM97] 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 completely determined by the value of one of its input-legs. The specific input that is being connected to the output is assumed to be determined by an all powerful adversary, possibly as a function of the input to the circuit. This model is equivalent to the 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 [vN56], in which the noise flips the value of each wire in the circuit independently with probability p. See [KLM97, KLR12] 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. [KLM97]. 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 [KLR12] showing, for any constant $\varepsilon > 0$, how to convert any formula[1] F of size $|F| = s$ into a resilient formula F' of size $|F'| = \text{poly}(s)$ such that F' computes the same function that F computes, as long as at most $(\frac{1}{10} - \varepsilon)$-fraction of the gates in *any input-to-output path* in F' suffer from short-circuit noise. In their result, the gates in F' have fan-in 2. If the formula F' is allowed to have gates with fan-in 3, the fraction of noise improves to $(\frac{1}{6} - \varepsilon)$. Kalai et al. explicitly leave open the question of finding the *optimal* fraction of faulty gates for a resilient formula F'.[2]

We solve the above open question, and show that 1/3 is a tight bound on the tolerable fraction of faulty gates per input-to-output path. Namely, we show how to convert any formula to a resilient version that tolerates up to a fraction 1/3 of short-circuit gates per path, for any fan-in $k \geq 2$.

Theorem 1.1 (Main, informal)**.** *For any $\varepsilon > 0$, any formula F of size s can be efficiently converted into a formula F' of size $\text{poly}_\varepsilon(s)$ that computes the same function as F even when up to $1/3 - \varepsilon$ of the gates in any of its input-to-output path are short-circuited.*

We also show that our bound is tight, namely, that for an arbitrary formula, it is impossible to make a resilient version that tolerates a fraction 1/3 (or more) of short-circuit gates per path.

Theorem 1.2 (Converse)**.** *There exists a function f such that no formula F for computing f is resilient to a fraction 1/3 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 fan-in 2 formulas lies within the range $[\frac{1}{10}, 1]$.

Similar to [KLR12], a major ingredient for obtaining our result is a transformation, known as the Karchmer-Wigderson transformation (hereinafter KW-transformation) [KW90], 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 and Section 2.2 for details. The work of Kalai et al. adapts the KW-transformation to a noisy setting in which the formula may suffers from short-circuit noise, and the protocol may suffer from channel noise. Therefore, the "attack plan" in [KLR12] for making a given formula F resilient to short-circuit noise is (i) apply the KW-transformation to obtain an interactive protocol P; (ii) Convert P to a noise-resilient protocol P' that tolerates up to δ-fraction of noise; (iii) apply the (reverse) KW-transformation on P' to obtain a formula F'. The analysis of [KLR12] shows that the obtained F' is resilient to $\delta/2$ fraction of noise in any of its input-to-output paths. Furthermore, building upon recent progress in the field of coding for interactive protocols (see, e.g., [Gel15]), Kalai et al. construct a coding scheme for interactive protocols with resilience $\delta = 1/3 - \varepsilon$ for any $\varepsilon > 0$ (using alphabet of size 3; for binary alphabet the obtained resilience is $\delta = 1/5 - \varepsilon$), which gives their result.

We follow the above line of attack, yet improve it in two critical points that allow us to obtain our optimal resilience. First, we use an optimal coding scheme for interactive protocols by Efremenko, Gelles and Haeupler [EGH15, EGH16]. They observe that the interactive protocol obtained through the analysis of [KLR12] is in fact defined in a setting that contains *noiseless feedback*—that is, a setting in which the sender (and only the sender!) is aware of noise corrupting its transmission (this assumption was implied in [KLR12]). Additionally, Efremenko et al. show an optimal coding scheme for interactive protocols over binary channels (with noiseless feedback) with resilience is $\delta = 1/3 - \varepsilon$, for any $\varepsilon > 0$. This contribution on its own is straightforward, and immediately leads to improving the error resilience in [KLR12] to $\frac{1}{6} - \varepsilon$ (for any fan-in).

The second, and more important ingredient is what allows us to get rid of the 2 factor, and get circuit error resilience of δ instead of $\delta/2$: We show a certain variant of the KW-transformation, from formulas to protocols and vice-versa, which is *resilience-preserving*. This ingredient is our main technical as well as conceptual contribution.

One of the most striking features of the KW conversion between formulas and protocols is that it is lossless in both directions: formula depth converts into deterministic protocol length precisely (without even an additive difference). The challenge is to extend this relationship to the noisy regime — this appears necessary if we want the resulting error resilience to be optimal.

To explain the ideas behind our resilience-preserving transformation, 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 get 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, $\neg z_i$). Suppose F is of depth d. The conversion of F to a protocol P of length d 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 being 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's Alice 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 to 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.

The duality described above is simple, elegant, and, importantly, bi-directional: it can be reversed to convert a protocol P for the KW game into a formula for f. As noted earlier, short-circuit errors on the formula side correspond to transmission errors with feedback on the protocol side. It would be very elegant (and simple) if the plan we outlined above work for the noisy case as well, namely if converting F into a protocol P; converting P into a resilient protocol P'; and finally, converting P' back into a formula F', would lead to a resilient formula F'. Alas, this sequence of conversions does not work by itself.

We will use a very simple example to illustrate what goes wrong. Consider the (highly redundant) formula for computing $F(z) = z_1$, which is given by k layers of $\wedge$ gates, all leading to z_1 inputs. For example, for $k = 2$, the formula is of the form $F(z_1) = (z_1 \wedge z_1) \wedge (z_1 \wedge z_1)$. This leads to a protocol P where Alice can send any string of length k, and its output will be z_1. Note that there is a lot of freedom in specifying P. When the protocol is converted into the error-proof protocol P', the conversion process may specify how some of the communication happens. For example, for $k = 2$ the following protocol P' of length 2 protects against one error with feedback: Alice always tries to send '0'. On the leafs 00, 01, 10 output z_1, on the leaf 11 output $\neg z_1$. Note that z_1 is always the right answer, and $\neg z_1$ is always the wrong answer, but since Alice always tries to send a '0', the only way to reach the 11 leaf is through having two errors. Thus P' is error-proof against one error.

What happens when we convert P' into the formula F'? We obtain the formula $F'(z_1) = (z_1 \wedge z_1) \wedge (z_1 \wedge \neg z_1)$. Not only is this formula not error-proof, it is actually identically 0, and is wrong even with no errors present! What went wrong is that we have implemented the node 11 (corresponding to $\neg z_1$) in hardware, even though it is never even supposed to be reached in P'.

The solution for this problem, already identified in [KLR12], is to remove part of the protocol tree of P' before converting it into hardware. Specifically, [KLR12] remove all nodes that are the result of more than $\delta/2$ error fraction *per party* (thus, all reachable nodes suffer from a noise level of at most δ). Alas, the resilience of the formula F' in their result decreases by a factor of 2 with respect to the resilience of the protocol P'.

Our main contribution is to find the "right" way to cut down P', one which extends the precise correspondence of the KW construction into the noisy domain. Specifically, we remove from the protocol tree of P' any nodes that are not accessible by a *noiseless* execution of P' on some valid input. Once the right definition of the "accessible" part of P' is in place, the correctness proof is quite simple (every faulty computation in F' can be reduced to a faulty computation in P'), and the resilience is shown to be preserved.

In the example above, only the 00 leaf of P' is accessible without noise (since Alice always tries to send a '0'), and the resulting formula is $F'(z) = ((z_1 \wedge) \wedge)$. Note that the nodes in this formula have fan-in 1, yet we could always complete F' to full fan-in by replicating children; this replication does not affect the functionality of F'. In this specific case we would get the short-circuit resistant formula $(z_1 \wedge z_1) \wedge (z_1 \wedge z_1)$.

We note that in our case the construction can be made effective. In addition, we note that we can employ the (resilience-preserving) KW transformation again on the resilient F', obtaining a noise-resilient protocol ${P'}_{acc}$ for the KW game. The protocol ${P'}_{acc}$ would be a restriction of P' obtained by not allowing some branches of P' (specifically, ones that are never accessible by a noiseless execution). Converting ${P'}_{acc}$ into a formula again would result in the formula F' — leading to a stable cycle ${P'}_{acc} \rightleftarrows F'$ in this conversion process.

Paper Outline We begin in Section 2 by defining the notion of formulas, interactive protocols and noise. Additionally, we recall the (noiseless) KW-transformation. In Section 3 we present our noise-preserving KW-transformation. Specifically, in Section 3.1 we show how to convert a resilient formula into a resilient protocol, and in Section 3.2 we provide the other direction, from protocols to formulas. In Section 4 we prove our main theorem, showing how to efficiently compile any formula into a resilient version. In Section 5 we prove the converse theorem, showing that the resilience we obtain for formulas is optimal. Finally, in Section 6 we conclude with several open questions.

2 Preliminaries

For an integer $n \geq 1$, we let $[n] = \{1, 2, 3, \ldots, n\}$. Logarithms are assumed to be taken to base 2.

2.1 Formulas, Protocols, and Noise

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 for any k, 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 $\mathrm{depth}(F)$, 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.

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 behaves as defined above. Otherwise, the value of the node is the value of its E_v 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 hold that $F(z) = F_E(z)$. 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 denote the set of all such noise patterns by S_δ.

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. Unless otherwise mentioned, we assume $\Sigma = \{0, 1\}$. 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 P 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.

The length of a protocol, denoted $|P|$, 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 $|P|$. 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).

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.

Similar to the case of circuits, we define a noise pattern for protocols. 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 P_E the protocol P when the noise is dictated by E; we sometimes write P 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 P_E outputs the same value as P. We say that a protocol is resilient to a δ fraction of noise, if it is resilient to all noise patterns that corrupt at most a fraction δ of the transmissions in any instance of the protocol. We denote the set of such noise patterns by Φ_δ.

We will sometimes abuse notation and identify a short-circuit noise pattern with a transmission noise pattern for a formula F and a protocol P that share the same underlying tree structure. Furthermore, we will denote the two different objects with the same identifier E.

Remark 1. *For protocols and formulas that share the same tree-structure, a formula noise with at most δ-fraction on any input-to-output noise is equivalent to protocol noise that corrupts at most δ-fraction of the transmissions in any instance. In this case it holds that $S_\delta = \Phi_\delta$.*

2.2 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 [KW90] proved the following relation between formulas and protocols.

Theorem 2.1 ([KW90])**.** *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.*

The above theorem is proven by showing a conversion between a formula for f and a protocol for KW_f, which we term the *KW-transformation*. In this conversion, 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 $a_v : \{0,1\}^n \to \{0,1\}$ is set as follows. For a given input z, consider the evaluation of the formula F on z. The node v is an $\wedge$ gate and we can write $v(z) = v_0(z) \wedge v_1(z)$ where v_0 and v_1 are v's left and right descendants, respectively. If $v_0(z) = 0$ we set $a_v(z) = 0$; otherwise we set $a_v(z) = 1$. For an $\vee$ gate denote $v(z) = v_0(z) \vee v_1(z)$, and $b_v(z) = 0$ if $v_0(z) = 1$; otherwise $b_v(z) = 1$. If the protocol reaches a leaf which is marked with the literal z_i or $\neg z_i$, it outputs i. For technical reasons we will assume that the protocol outputs either z_i or $\neg z_i$ rather than just giving the index i. Note that the literal always evaluates to the value of f; In this work a KW_f protocol must satisfy this additional requirement.

It is easy to verify that the following invariant holds: for every node v reached by the protocol on some input $(x, y) \in f^{-1}(0) \times f^{-1}(1)$, it holds that $v(x) = 0$ while $v(y) = 1$. This holds for the root node by definition, and our selection of mappings a_v, b_v maintains this property. Specifically, for an $\wedge$ gate v for which $v(x) = 0$ it must hold that at least one of the gate inputs is zero, and indeed the way we chose a_v advances the protocol to a child node which evaluates to 0. Since $v(y) = 1$ then both children of v evaluate to 1 on y, thus both descendants satisfy the invariant. The analysis for an $\vee$ gate is symmetric. Thus, once the protocol reaches a leaf (the literal z_i or $\neg z_i$), we get that this literal evaluates differently for x and for y, so $x_i \neq y_i$ as required. In particular, the literal evaluates to 0 on x and to 1 on y.

The same reasoning allows us to convert a protocol for KW_f into a formula for f: consider the protocol tree and convert each node where Alice speaks to an $\wedge$ gate and each node where Bob speaks to an $\vee$ gate. If the protocol outputs z_i or $\neg z_i$ at some leaf, that literal is assigned to that leaf.

Proving that this conversion yields a formula for f is shown by induction on the length of the protocol. If $|KW_f| = 0$, then the protocol outputs (say) z_i without communicating. It is clear that all inputs in the domain satisfy $x_i \neq y_i$, and that $x_i = 0$ while $y_i = 1$ (negate these values if the output of the protocol is $\neg z_i$). For the induction step, assume without loss of generality that Alice is to speak first. For some partition $X^0 \cup X^1 = f^{-1}(0)$, Alice sends 0 when $x \in X^0$ and otherwise she sends 1. By induction, the continuation of the protocol can be converted into formulas F_0 and F_1 (corresponding the case Alice sends 0 or 1, respectively), for which $F_0(x) = 0$ when $x \in X^0$, $F_1(x) = 0$ when $x \in X^1$ and $F_0(y) = F_1(y) = 1$ when $y \in f^{-1}(1)$. Taking $F = F_0 \wedge F_1$ completes the proof. The other case, where Bob is to speak first is symmetric. See [KW90] for further details about the KW-transformation from formulas to protocols and vice versa, and for the formal proofs.

Remark 2. *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 $|\Sigma| = k$.*

3 The Noisy KW-Transformation

In this section we show a KW-transformation, from a short-circuited formula to a noisy protocol (with a similar noise pattern), that works similarly to the noiseless case, and preserves the noise-resilience of the formula/protocol. In the following subsection we show that if we start with a resilient formula, then the obtained protocol is resilient to the same noise fraction (Proposition 3.3). Next, we show that if we start with a resilient protocol, the transformation yields a formula which is resilient to the same fraction of noise (Proposition 3.5).

3.1 From resilient formulas to resilient protocols

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

Definition 3.1 (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 P^{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 3. *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 also valid (and gives an alternative protocol which still satisfies Proposition 3.1 and Corollary 3.2 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 $Z_0 \cup Z_1$).

Remark 4. *Note that extending the above to formulas with a larger fan-in (equivalently, protocols with a larger alphabet-size) is trivial. Similarly, the above applies to formulas with fan-in bounded by k.*

Proposition 3.1. *For any formula $F(z)$ and any noise pattern E for F, consider the noisy formula $F_E(z)$ and the noisy protocol P^{F_E} obtained by performing the noisy KW-transformation of Definition 3.1.*

Given any instance of P^{F_E} on inputs $(x, y) \in F_E^{-1}(0) \times F_E^{-1}(1)$ along with the noise induced on the protocol by E, it holds that any node v in the protocol tree reached by that instance maintains that $v(x) = 0$ and $v(y) = 1$ in F_E.

Proof. Denote the noisy instance of the protocol by $P_E^{F_E}$. The proof goes by induction on the depth d of $P_E^{F_E}$. For the base case $d = 0$, no noise/short-circuit is possible, the formula is just a leaf (either the literal z_i or the literal $\neg z_i$) and the protocol outputs that same literal. The claim trivially holds by the assumption that $(x, y) \in F_E^{-1}(0) \times F_E^{-1}(1)$.

For $d \geq 1$, consider the case where the top gate in F_E is an $\wedge$ gate (the case of an $\vee$ gate is shown in a similar manner). Denote the top gate by v and its left and right descendants as v_0 and v_1, respectively.

There are two cases according to the noise associated with the top gate. If there is no noise at v, $E_v = *$, then for any input z it holds that $v(z) = F_{E0}(z) \wedge F_{E1}(z)$, where F_{E0}, F_{E1} are the *noisy* subformulas of F_E rooted at v_0 and v_1 respectively.[3] Since $F_E(y) = v(y) = 1$ it must hold that $F_{E0}(y) = F_{E1}(y) = 1$. Additionally, $F_E(x) = v(x) = 0$ therefore at least one of $F_{E0}(x)$ and $F_{E1}(x)$ must be 0. The protocol $P_E^{F_E}$ proceeds to the left child if $F_{E0}(x) = 0$, or to the right child otherwise. By the induction hypothesis, the claim holds for the depth $d - 1$ subprotocol that corresponds to the (noisy) sub-formula F_{E0} or F_{E1} accordingly.

If there is noise at v, without loss of generality, $E_v = 0$, then $v(z) = F_{E0}(z)$. It follows that $F_E(z) = F_{E0}(z)$, and specifically, $F_{E0}(x) = 0$ while $F_{E0}(y) = 1$. Note that in the protocol $P_E^{F_E}$, the noise at node v dictates that the parties continue to node v_0 regardless to Alice's input and transmission[4]. By the induction hypothesis the claim holds for the depth $d - 1$ subprotocol that corresponds to the noisy sub-formula F_{E0} rooted at v_0. □

Corollary 3.2. *Assume that $F_E(z)$ computes the function $f(z)$. Then, $P_E^{F_E}$ computes KW_f.*

Proof. Say that on inputs $(x, y) \in F_E^{-1}(0) \times F_E^{-1}(1)$ the protocol terminates at a leaf v marked with either z_i or $\neg z_i$. By the above Proposition 3.1 it holds that $v(x) = 0$ while $v(y) = 1$, which implies that $x_i \neq y_i$. Note that the literal evaluates to the output of the function as we additionally require from KW_f protocols. □

With the above we can show our main proposition for converting formulas to protocols in a noise-preserving way.

Proposition 3.3. *Let F be a (complete) formula that computes the function f and is resilient to the set S_δ of up to δ-fraction of short circuit errors in every input-to-output path. Then, a noisy KW-transformation yields a protocol P that solves KW_f and is resilient to any noise from $\Phi_\delta = S_\delta$.*

Proof. We convert F into the protocol P defined for inputs $(x, y) \in F^{-1}(0) \times F^{-1}(1)$ in the following manner. The conversion is performed similar to the noisy KW-transformation (Definition 3.1), however the mappings a_v, b_v are set in a specific way we now describe.

[3] If v has only one child, the claim holds trivially.

[4] Alice is aware of the noise through the feedback, thus she can follow the progress of protocol as defined above. This shows that a noiseless feedback setting is critical when transforming formulas that suffer from short-circuit noise.

Order the nodes in the protocol tree in a BFS order starting from the root, and determine the mappings associated with each node in that order (i.e., before setting the mapping of some node, set the mapping of all its ancestors). Assume we wish to set the mapping of a node v. Let $S_{(v,x,y)}$ be the set of noise patterns in S_δ whose induced noise on P given the input (x, y) causes the protocol to reach the node v (note that this process is well defined due to the BFS order).

if v is an $\wedge$ node, for any x, the mapping $a_v(x)$ maps to the child w for which the subformula of F rooted at w evaluates to 0 on x for *all* noise patterns $E \in \bigcup_{y' \in F^{-1}(1)} S_{(v,x,y')}$. If v is an $\vee$ node, then for any y, the map $b_v(y)$ maps to the child w for which the subformula of F rooted at w evaluates to 1 on y for all noise patterns $E \in \bigcup_{x' \in F^{-1}(0)} S_{(v,x',y)}$. Note that the mappings may be partial functions, specifically, if an $\wedge$ node v is not reachable for the input x with any y's and any valid noise, then there is no meaning to define a_v on the input x. Claim 3.4 below guarantees that for any reachable node v we can always find a child w that satisfies the above condition.

We now prove that the protocol P is resilient to any noise pattern induced by some noise $E \in S_\delta$. Let $E \in S_\delta$ be given and let P_E be the protocol defined above for F, assuming the transmission noise induced by E. We claim that the protocol P_E solves $KW_{F_E} = KW_f$, which means that P is resilient to the noise E.

Consider the noisy formula F_E and its corresponding interactive protocol P^{F_E} given by the noisy KW-transformation (Definition 3.1). As mentioned in Remark 3 we can make it so the mapping at any reachable node in P^{F_E} where there is a choice whether to go to the left child or the right child, takes the same choice that P does.[5] Then, all the reachable nodes in $P_E^{F_E}$ behave exactly the same as in P_E: say that on input $x \in F_E^{-1}(0)$ we reach a node v in $P_E^{F_E}$ and there is no choice for the next node (e.g., $v_0(x) = 0$ while $v_1(x) = 1$, so we must continue with v_0) then also there is no choice in P, since v_1 does not evaluate to 0 with all errors (specifically, it does not evaluate to 0 on x with the noise E!); however if there is a choice in $P_E^{F_E}$, the protocol continues exactly as P. For any input $(x, y) \in F_E^{-1}(0) \times F_E^{-1}(1)$ the protocols P_E and $P_E^{F_E}$ advance exactly the same: they begin at the root; if they are at node v where $E_v \neq *$ they both advance to the node dictated by E_v; Otherwise, they both advance to the same node since their mapping (either $a_v(x)$ or $b_v(y)$) is exactly the same at that node. Hence, both protocols reach the same leaf and compute the same function for any input $(x, y) \in F_E^{-1}(0) \times F_E^{-1}(1) = F^{-1}(0) \times F^{-1}(1)$ (recall that F is resilient to the noise E). Then, Corollary 3.2 implies that P_E computes $KW_{F_E} = KW_f$, so P is resilient to the noise E. Note that this claim holds for any $E \in S_\delta$ which completes the proof. □

We are left to prove the following technical claim used in the above proof.

Claim 3.4. *Let F be a resilient formula and P the corresponding interactive protocol described in Proposition 3.3. Assume that for some input $(x, y) \in F^{-1}(0) \times F^{-1}(1)$ and noise $E' \in S_\delta$ it holds that $P_{E'}(x, y)$ reaches the node v, and let F_0 and F_1 be the subformulas (of F) rooted at the left and right child of v, respectively.*

Then there is at least one subformula $G \in \{F_0, F_1\}$ that satisfies $G_E(x) = 0$ for all noise patterns $E \in \bigcup_{y' \in F^{-1}(1)} S_{(v,x,y')}$ (when v is an $\wedge$ node), or $G_E(y) = 1$ for all noise patterns $E \in \bigcup_{x' \in F^{-1}(0)} S_{(v,x',y)}$ (when v is an $\vee$ node).

[5] P and P^{F_E} share the same underlying protocol tree structure, so for each node in one protocol there exists a corresponding node in the other protocol, which is owned by the same party. Hence, we can set the mappings a_v and b_v of one protocol via the corresponding mappings of the other one.

Proof. Assume the case where v is an $\wedge$ node (the other case is similar). First note that for any noise $E \in S_\delta$ and input (x, y) for which $P_E(x, y)$ reaches the node v it holds that $v(x) = 0$ and $v(y) = 1$ in F_E, as given by Proposition 3.1.

Assume towards contradiction that the claim does not hold, that is, there are two noise patterns $E_0, E_1 \in \bigcup_{y' \in F^{-1}(1)} S_{(v,x,y')}$ such that $(F_0)_{E_0}(x) = 1$ and $(F_1)_{E_1}(x) = 1$.

Define the noise pattern E^* (over the nodes of F) in the following way. For any ancestor of v, E^* is defined exactly as the *minimal* between E_0 and E_1 (i.e., the one that induces the least noise on the root-to-v path). For the nodes that belong to the subformula F_0, the noise E^* is identical to E_0, and for nodes that belong to the subformula F_1, E^* is identical to E_1. In all other nodes there is no noise in E^*.

Clearly by this construction, $E^* \in S_\delta$, and there exists some y' such that $P_{E^*}(x, y')$ reaches v. Therefore, we get that $E^* \in \bigcup_{y' \in F^{-1}(1)} S_{(v,x,y')}$. However we also get that in F_{E^*} the node v evaluates to 1 on x, because $(F_0)_{E^*}(x) = (F_0)_{E_0}(x) = 1$ and $(F_1)_{E^*}(x) = (F_1)_{E_1}(x) = 1$. But this contradicts the property, asserted at the beginning of this proof, that for any noise $E \in S_\delta$ (and specifically for E^*), any node v that is reachable by $P_{E^*}(x, y')$ must evaluate to 0. Therefore, at least one of $F_0(x)$ and $F_1(x)$ evaluates to 0 on all noise patterns in scope. □

3.2 From resilient protocols to resilient formulas

We now show the other direction, namely, that the KW-transformation converts a resilient protocol to a resilient formula. We perform the KW-transformation only on a certain subgraph of the protocol tree, namely, the all the nodes that are reachable for some input (x, y) assuming there is no noise. We call this subprotocol, the noiseless-reachable protocol tree.

An interesting effect of this reachability condition is pruning branches that can be reached only if there is a noise immediately leading to that branch—such branches are redundant when converting back to a formula. To illustrate this idea consider the following simple example where the protocol is defined over an alphabet of size $|\Sigma| = 3$. Assume that the root is owned by Alice, and assume that Alice either sends a 0 or a 1 according to her input, but she never sends a 2. Of course, the first transmission may be corrupted so that Bob receives 2, however this branch can happen only due to noise. When converting back to formula, there is no reason to convert the sub-tree rooted at the third child of the root. By eliminating this subprotocol we also eliminate the possibility that the root gate is short-circuited to its third input leg. [A nice feature of this single pruning is that it practically eliminates up to a third(!) of the formula size without affecting its functionality.]

Taking the above idea to its extreme leaves us exactly with the noiseless-reachable protocol tree. Quite surprisingly, although the noiseless-reachable protocol tree is defined in a way that assumes no errors, it is "resilient" enough so that when we convert it back to a formula, the obtained formula is resilient to δ-fraction of noise.

Definition 3.2 (noiseless-reachability)**.** *Given a protocol P we denote the noiseless-reachable protocol tree of P as all the nodes v that are reachable by P for some (x, y) assuming there is no noise.*

Remark 5. *If a node v is noiseless-reachable then at least one of its children is noiseless-reachable. Therefore, it cannot happen that an inner node in P becomes a leaf in the noiseless-reachable protocol tree of P.*

Proposition 3.5. *Let P be a protocol that solves KW_f for some function f and is resilient to δ-fraction of noise. The KW-transformation on the noiseless-reachable protocol tree of P yields a formula F that computes f and is resilient to up to δ-fraction of short-circuit noise in any of its input-to-output paths.*

Proof. Assume towards contradiction that there exists $(x, y) \in f^{-1}(0) \times f^{-1}(1)$ and a noise pattern $E \in S_\delta$ for F such that

$$f(x) = 0 \quad \text{while} \quad F_E(x) = 1$$

(the other case where $f(x) = 1$ but $F_E(x) = 0$ is similarly proven).

First we observe that short-circuiting an $\wedge$ gate can only turn the output from 0 to 1 (but not vice-versa); short-circuiting an $\vee$ gate can turn the output from 1 to 0 (but not vice-versa). We can therefore assume that the above noise pattern E corrupts only $\wedge$ nodes $V_\wedge$, and assigns $*$ to all the nodes in $V_\vee$.

Claim 3.6. *There exists a root-to-leaf path $\vec{p}$ in the underlying tree of P_E and F_E that satisfies the following properties:*

1. *Each node v along the path $\vec{p}$ satisfies $v(x) = 1$ in F_E.*
2. *Assume $v \to u$ is an edge in $\vec{p}$, then there exists some y' for which $P_E(x, y')$ reaches v and continues to u. then $a_v(x)$ maps to u.*

Proof. We construct the path $\vec{p}$ in an inductive manner. Add the the root of the underlying protocol/formula tree to $\vec{p}$. Suppose v is in $\vec{p}$. If v is an $\vee$ node (owned by Bob), then add a child for which $F_E(x)$ evaluates to 1. At least one child of v must evaluate to 1 on x exists since v is an $\vee$ gate and $v(x) = 1$. Note that property 1 holds by our selection, and property 2 holds vacuously.

If v is an $\wedge$ node (owned by Alice), let us assume for now that v is reachable by $P_E(x, y')$ for some y'. If $E_v \neq *$ then add the E_v-th child of v to $\vec{p}$. Both properties 1 and 2 hold by definition in this case. Otherwise, note that all v's children must evaluate to 1 (so property 1 trivially holds); add to $\vec{p}$ the child that $P_E(x, y')$ goes to when reaching v, so property (2) holds as well.

We are left to show that each node v on $\vec{p}$ that is owned by Alice is reachable by $P_E(x, y')$ for some y'. In fact, we show that there exists a y' for which $P_E(x, y')$ takes the path $\vec{p}$. Let ℓ be last node in $\vec{p}$. Due to the noiseless-reachability, there exist $(\tilde{x}, \tilde{y}) \in f^{-1}(0) \times f^{-1}(1)$, so that $P(\tilde{x}, \tilde{y})$ reaches ℓ. Note that $P_E(x, \tilde{y})$ reaches ℓ as well: for any node owned by Alice we walk along $\vec{p}$ by property 2, and for any node owned by Bob (and no noise) we follow $\vec{p}$ as implied by $P(\tilde{x}, \tilde{y})$. □

However, the existence of a path $\vec{p}$ from the above claim contradicts our assumption. To see that, consider again the instance $P_E(x, \tilde{y})$ discussed in the claim's proof. Since P computes KW_f and is resilient to the noise E, when it reaches the leaf ℓ it outputs a literal z_i (or $\neg z_i$) that evaluates to 0 on x (and to 1 on $\tilde{y}$).

On the other hand, property 1 suggests that $\ell(x) = 1$ in F_E, therefore the literal written at the leaf ℓ evaluates to 1 on x, which is a contradiction. □

4 Optimal Short-Circuit–Resilient Formulas

We can now derive our main result: formulas that are resilient to at most $(1/3 - \varepsilon)$ short-circuited gates in any of their input-to-output paths, for any $\varepsilon > 0$.

Theorem 4.1. *For any $\varepsilon > 0$, any formula F of depth n and fan-in 2 that computes a function f can be efficiently converted into formulas F_2 and F_3, where each computes f even up to $1/3 - \varepsilon$ of the gates in any of its input-to-output path are short-circuited. F_2 has fan-in 2 and depth $O(n/\varepsilon^2)$, and F_3 has fan-in 3 and depth $O(n/\varepsilon)$.*

Proof. The conversion is done in the following manner. Given F (that computes some function f) we first balance it, i.e., convert it to an equivalent formula $\tilde{F}$ of depth $\log |F|$ with no redundant branches. It is well known that such a formula always exists. Next, we convert $\tilde{F}$ into a protocol P for KW_f via the KW-transformation (Section 2.2); note that the length of P is at most the depth of $\tilde{F}$, that is, $O(\log |F|)$. Then, we convert P into a protocol P' that solves the same function KW_f and is resilient to a fraction $1/3 - \varepsilon$ of noise, assuming noiseless feedback. This step is possible due to the following theorem by Efremenko, Gelles, and Haeupler [EGH16].

Theorem 4.2 ([EGH16])**.** *For any $\varepsilon > 0$, any binary interactive protocol P of depth n can be efficiently converted into an interactive protocol P' (with feedback) over alphabet Σ which is resilient to $1/3 - \varepsilon$ corruptions. If $|\Sigma| = 2$ then $|P'| = O(n/\varepsilon^2)$ and if $|\Sigma| > 2$ then $|P'| = O(n/\varepsilon)$.*

The resilient P' is then transformed back into resilient formulas F_2 (if $|\Sigma| = 2$) or into F_3 (for $|\Sigma| = 3$) that satisfy the theorem assertions, using Proposition 3.5. Recall that the depth of the obtained formula is exactly the length of the resilient protocol.

To complete the proof we only need to argue that the conversion can be done efficiently. It is easy to verify that converting $\tilde{F}$ to P is efficient, and also converting P to P' is efficient by the above Theorem 4.2. The only part which is possibly inefficient is the reverse KW-transformation from P' back to a formula, which requires finding the noiseless-reachable protocol tree of P'. We complete the details in Appendix A □

Theorem 1.1 is an immediate corollary of the above theorem, by noting that $|F_2| \leq 2^{\text{depth}(F_2)} = 2^{O(\log |F|/\varepsilon^2)} = \text{poly}_\varepsilon(|F|)$.

5 Impossibility Bound

In this section we show that a noise level of $1/3 - \varepsilon$ is optimal for short-circuit noise in formulas. Intuitively, this follows from the fact that a noise resilience of $1/3$ is maximal for interactive protocols over channels with noiseless feedback [EGH16], and from the one-to-one correspondence between formulas and protocols.

Let us first recall that it is impossible for interactive protocols with noiseless feedback to have noise resilience $1/3$ or higher.

Theorem 5.1 ([EGH16])**.** *Let $\mathcal{F}$ be the set of functions $f(\cdot, \cdot)$ for which there are inputs x_1, x_2 such that $f(x_1, y) \neq f(x_2, y)$ for some y, as well as inputs y_1, y_2 such that $f(x, y_1) \neq f(x, y_2)$ for some x. For any function $f \in \mathcal{F}$, any (deterministic) interactive protocol that computes $f(x, y)$ over a channel with feedback, with any alphabet size, fails if a fraction $1/3$ of the transmissions may be corrupted.*

Specifically, there exists a party which outputs the same value for two different inputs; e.g. on (x, y_1) and (x, y_2) (if Alice) or on (x_1, y) and (x_2, y) (if Bob).

We repeat the proof of [EGH16] in Appendix B.

For $z \in \{0,1\}^n$, define $p(z) = z_1 \oplus \cdots \oplus z_n$ to be the parity function. It is easy to verify that $KW_p \in \mathcal{F}$. From Theorem 5.1 it follows that no interactive protocol for KW_p can tolerate

a fraction 1/3 of noise (or above): set for instance $x = 0^n$, $y_1 = 10^{n-1}$, $y_2 = 010^{n-1}$ (Assuming that Alice is the party that gets confused), then there exists an attack that corrupts 1/3 of the transmission and makes Alice output the same value on (x, y_1) and (x, y_2). Note that at least one of these outputs is incorrect for the KW_p task.

This implies that no formula for the parity function $p(z)$ can tolerate a fraction 1/3 of short-circuited gates.

Theorem 5.2. *There exists a function f such that no formula F for computing f is resilient to a fraction 1/3 of short-circuit noise.*

Proof. Assume F is formula for the parity function $p(z)$ that is resilient to a fraction 1/3 of noise, and assume the formula's underlying graph is a complete tree.[6] Then, using Proposition 3.3 we obtain an interactive protocol P for KW_p. Since F is a complete tree, each $E \in \Phi_{1/3}$ corrupting at most 1/3 of the transmissions in P in any instance, is also a valid noise for F that corrupts at most 1/3 of the gates in any input-to-output path, $E \in S_\delta$. Proposition 3.3 then suggests that P is resilient to $\Phi_{1/3}$, but this contradicts Theorem 5.1, so our assumption of the existence of F is invalid. □

6 Open Questions

While our work settles the optimal short-circuit *resilience* of formulas, it is still open to find the optimal *size* of formulas that are resilient to some given (small) amount of noise. Kalai et al. [KLR12] show that given any formula of depth d, one can obtain an equivalent formula that is resilient to e faulty gates with depth $4d + 10e$ assuming fan-in 2 gates (respectively, $2d + 6e$ for fan-in 3 gates). For small values of δ, binary formulas that are resilient to a noise fraction of δ have depth $D = 4d + 10\delta D$, or $D = \frac{4}{1-10\delta}d$. That is, the "rate" of the conversion is $d/D = 1/4 - 2.5\delta$. Since the KW-transformation preserves the depth of the formula (with respect to the length of the interactive protocol), this value should be contrasted with the optimal rate one obtains when converting interactive protocol into noise-resilient protocols.

Gelles and Haeupler [GH15] show a coding method for alternating binary interactive protocol (assuming noiseless feedback), that obtains a rate of $1 - O(H(\delta))$, which is tight up to the hidden constant in the second term.[7] On the other hand, the coding process of [GH15] is inherently random, and it is unclear if similar techniques can be used when translating the protocol back to a formula. Given an alternating $\wedge/\vee$ formula of depth d, finding a δ-resilient formula with depth $d(1+O(H(\delta)))$ and fan-in 2, or equivalently, finding a *deterministic* and efficient binary interactive coding schemes over channels with noiseless feedback that is resilient to δ-fraction of noise with rate $1 - O(H(\delta))$, remains an interesting open question.

Acknowledgements

Part of this work was done while R.G. was at Princeton University.

Research 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.

[6] 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.

[7] $H(x) = -x \log x - (1-x)\log(1-x)$ is the binary entropy function.

References

[EGH15] K. Efremenko, R. Gelles, and B. Haeupler. Maximal noise in interactive communication over erasure channels and channels with feedback. *Proceedings of the 6th Conference on Innovations in Theoretical Computer Science*, ITCS '15, pp. 11–20, 2015.

[EGH16] 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, 2016.

[Gel15] R. Gelles. Coding for interactive communication: A survey, 2015.

[GH15] R. Gelles and B. Haeupler. Capacity of interactive communication over erasure channels and channels with feedback. *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pp. 1296–1311, 2015.

[KLR12] Y. T. Kalai, A. Lewko, and A. Rao. Formulas resilient to short-circuit errors. *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pp. 490–499, 2012.

[KW90] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM Journal on Discrete Mathematics*, 3(2):255–265, 1990.

[KLM97] D. Kleitman, T. Leighton, and Y. Ma. On the design of reliable boolean circuits that contain partially unreliable gates. *J. Comput. Syst. Sci.*, 55(3):385–401, 1997.

[vN56] J. von Neumann. Probabilistic logics and the synthesis of reliable organisms from unreliable components. C. E. Shannon and J. McCarthy, eds., *Automata studies*, *Annals of Mathematics Studies*, vol. 34, pp. 43–98. Princeton University Press, Princeton, 1956.

A Theorem 4.1: Efficiency

We now argue that the conversion from P' to F_2 (or F_3) in Theorem 4.1 can be done efficiently in the size of the formula F. Note that in general, the conversion of Proposition 3.5 may not be efficient, but it is efficient for protocols obtained by the conversion described in Theorem 4.1. The ideas in this part resemble the analysis of [KLR12] (yet in a somewhat more intuitive manner), and we sketch here the details for self completeness.

The main part is is showing that given the (algorithmic) description of P' we can efficiently obtain its noiseless-reachable subtree in time poly($|F|$). The noiseless-reachable subtree is required for Proposition 3.5 as the KW-transformation is well-defined only over this protocol tree.

While in general, there no reason to believe that reachability of a given protocol is efficient, it is efficient for the restricted family of protocols we consider here. Specifically, the analysis greatly depends of the fact that P is a KW-transformation of a given formula, as well as on the specific coding protocol used for converting P to P'. Since our main theorem is stated for fan-in 2 formulas, we will assume the binary coding scheme stated as Algorithm 2 in [EGH16]; a similar analysis can be shown for the ternary coding scheme (which is somewhat more simple), see also [KLR12, Lemma 13].

A.1 The binary coding scheme with resilience $(1/3 - \varepsilon)$: A sketch

Let us first sketch the principle of the coding scheme converting P to P'. This description is not fully self-contained and the reader is referred to [EGH16] for the full details.

Given a binary interactive protocol Π of length n, its simulation Π' performs the following. The parties transfer messages of varying length. Each message is composed of two parts: (a) 2 *information* bits and (b) $t \geq 1$ *confirmation* bits. The information section is used to simulate the ternary-coding protocol of [EGH16] (or the equivalent of the coding in [KLR12]): the parties send a symbol in $\{0, 1, 2\}$ where 0 and 1 correspond to Π sending 0 or 1, and the symbol 2 means "backup" and rewind Π by two rounds. As long as there are no errors (as can be verified by the feedback), the simulation just runs Π. If an error is discovered, then the party that learns about this error (via the feedback) sends a "2", after which both parties erase the previous 2 simulated rounds of Π (hopefully, eliminating the corrupted transmission) and proceed from that point.

The confirmation bits allow the sender to communicate whether the information bits were correctly received at the other side (as learned by the feedback). Specifically, if the 2 information bits were received intact, the sender would send many '1' bits to confirm that event. On the other hand, if either of the 2 information bits were flipped by the noise, the sender would send many '0' bits. Note that the confirmation bits themselves may be flipped by the channel.

The parties stop communicating confirmation bits when one of the following two conditions occur. Either (1) the number of 0-confirmation bits is more than a third of the length of the current message (that is, the most recent two information bits and all the confirmation bits sent afterwards), or (2) the number of *received* 1-confirmation bits is at least n/ε more than the number of received 0-confirmation bits. If condition (1) occurs, the last message (i.e., the information bits) is ignored, and the sender re-transmit that message. If condition (2) occurs, the messages is processed, and the 2 information bits dictate the next move (i.e., backing up in Π or continuing to the 0- or 1-child of the current node in P).

It is shown [EGH16] that sending n/ε^2 bits with the above simulation is enough to simulate Π and be resilient to $1/3 - \varepsilon$ bit flips.

The following claims are immediate from the definition of the simulation.

Claim A.1. *Every node v in Π' can be associated with a node w_v in Π that corresponds to the node in Π currently being simulated by Π'.*

Claim A.2. *Additionally, for any node v in Π' the path $\gamma = (v_0, v_1, \ldots, v)$ from root to v in Π' depicts a walk $w_\gamma = (w_{v_0}, \ldots, w_v)$ in the protocol tree of Π which starts from the root and at each step either descends to a child, stays at same place, or goes to a grandfather of the current node.*

A.2 An efficient algorithm for finding the reachable protocol trees

Theorem 4.1 requires us to find the noiseless-reachable protocol tree of P' in an efficient way. In this section we show an even stronger claim: for any (efficiently computable[8]) set of noises Φ, we can obtain the Φ-reachable protocol tree of P' in an efficient way. The noiseless-reachable protocol tree can be obtained as a special case where $\Phi = \{*^{|V_a| \cup |V_b|}\}$.

Recall that $|P| = \log |F|$ and that $|P'| = O(|P|)$ and set $d = |P'|$. In the following, "efficiently" means time complexity of $\text{poly}(2^d) = \text{poly}(|F|)$. Given a node v in depth $h \leq d = |P'|$ (i.e., given

[8]Specifically, given a noise pattern E, determining whether or not $E \in \Phi$ should be done efficiently.

the path $(e_1, e_2, \ldots, e_h)$ where every edge $e_i \in \{0,1\}$ descends downwards starting from the root node), we wish to check if the node v is contained in the Φ-reachable protocol tree of P'.

The idea behind the algorithm is as follows. Given v let $(v_0, v_1, ..., v_h = v)$ be the nodes on the unique path from the root to v. We examine each one of the possible noise pattern on that path, that is, for each node v_i we decide whether it is corrupted or not; there are at most 2^d different such noise patterns. Once the noise is fixed, we verify that all the other edges (v_i, v_{i+1}) are consistent with the behavior of the simulation, and reject the noise pattern if they are not. If no inconsistency is found, we show that a valid run of P' on some input with that noise pattern leads to v.

First, we recall that we assume that the complete protocol tree of P is reachable, that is we prune all the redundant branches.[9]

Claim A.3. *For any node v in the protocol tree of P there exists an input (x, y) such that an instance of P on (x, y) reaches v.*

The reachability test is given by the following Algorithm 1.

Algorithm 1: Φ-reachability check for P'
Input: a node v in a complete binary tree of depth d.

1. Given v let $\gamma = (v_0, v_1, ..., v_h = v)$ be the path from root to v.
2. Let ϕ_v be the set of all path-noise patterns $(E_{v_0}, \ldots, E_{v_{h-1}}) \in \{0, 1, *\}^{h-1}$ for which
   (i) if $E_{v_i} \neq *$, for some $0 \leq i < h$ then the E_{v_i}-th child of v_i is v_{i+1}, and
   (ii) $E \in \Phi$.
3. For all $E \in \phi_v$ repeat:

   Verify that the path γ and the noise E are consistent:

   (a) Parse the path γ and distinguish nodes that are information or confirmation bits.
   (b) Verify the confirmation bits mechanism: all confirmation bits within a single "message" should be the same (unless corrupted) and their value should agree with the event of noise in the corresponding information bits.
   (c) Let w_γ be the induced walk in P. Verify that the walk is consistent for some leaf l in the protocol tree of P: loop over all leafs l in P, and call the path from root to l the "correct path". Check if w_γ is a valid walk that can be implied by P' given that the correct path (in P) ends at l: as long as there is no noise the information bits sent in P' for nodes v_i such that w_{v_i} is on the correct path lead to the correct child of w_{v_i}; The information bits for nodes such that w_{v_i} is not on the correct path are set for backing up.
   (d) If all verifications pass for a certain leaf l, output `Reachable`.
4. output `Non-Reachable`.

[9] This assumption means that the formula F we start with is optimal; note that obtaining the optimal formula F for a given function may not be efficient. Yet, this assumption is not crucial. Alternatively (as performed in [KLR12]), we can assume each leaf in F is an independent variable—surely a resilient coding for such a formula would also be a resilient version of F, i.e., when only considering inputs that are consistent with F. This, however, causes the (reachable) protocol tree of P to be larger, and respectively increases the size of the output resilient formula, yet keeping its size polynomial in $|F|$.

Claim A.4. *Algorithm 1 takes time* $\text{poly}(2^d)$.

Proof. It is easy to see there are at most 2^d valid noise pattens for the path γ that should be considered. For each, we need to go over at most 2^d possible leafs in P and perform $O(d)$ checks per leaf. The total time is clearly $\text{poly}(2^d)$. □

Theorem A.5. *For any input node v, Algorithm 1 outputs* `Reachable` *if and only if v is reachable by running P' over some input $(x, y) \in F^{-1}(0) \times F^{-1}(1)$ and some noise $E \in \Phi$.*

Proof. It is easy to see that if an inconsistency is found for some noise E at step (2b) then the obtained γ cannot describe a valid instance of P' with the noise E (regardless of the input). If the inconsistency is found at (2c) it means that γ cannot describe a valid instance of P' with noise E and any input that leads to the leaf l, thus if (2c) fails for all leafs l, there is no input that leads to v assuming that specific noise pattern E.

It remains to show that if no inconsistency is found in steps (2b)–(2c), then the node v is Φ-reachable. This follows the same reasoning. Let l and E be the leaf and error noise for at the time where the protocol ends and output that v is reachable. Since l is reachable in P (Claim A.3) let $(x, y) \in F^{-1}(0) \times F^{-1}(1)$ be the input that leads P to the leaf l; note that (x, y) is a valid input for P'. It is easy to verify that running P' on (x, y) with the noise E yields exactly the path γ. Therefore, v is reachable. □

B Proof of Theorem 5.1

For self-containment, we bring here the proof of Theorem 5.1. The proof is taken from [EGH16] almost verbatim.

Proof. Consider a protocol of length N, and suppose that on inputs (x, y_1), Bob is the party that speaks less during the first $2N/3$ rounds of the protocol (if there is no noise). Due to the feedback we can assume the parties are always in consensus regarding the party to speak in the next round, so that at every round only a single party talks; thus, Bob talks at most $N/3$ times during the first $2N/3$ rounds.

Consider the following experiment $\mathsf{EXP1}$ in on inputs (x, y_2), and we corrupt Bob's messages during the first $2N/3$ rounds so that they are the same as Bob's messages if he holds y_1. From Alice's point of view, the case where Bob holds y_1 and the case where he holds y_2 but all his messages are corrupted to look as if he had y_1, are equivalent. Therefore, with the consensus assumption, in both cases Bob's talking slots are exactly the same, and this strategy corrupts at most $N/3$ messages altogether.

Now consider the following experiment $\mathsf{EXP0}$ on input (x, y_1). Here, during the last $N/3$ rounds of the protocol we corrupt all Bob's messages to be the same as what he sends in $\mathsf{EXP1}$ during the same rounds. Note that due to the adaptiveness of the order of speaking in the protocol, it may be that Bob talks in *all* these $N/3$ rounds, but corrupting all of them is still within the corruption budget.

Finally, in both $\mathsf{EXP0}$ and $\mathsf{EXP1}$ Alice's view (messages sent, received and feedback) is the same, implying that she gives the same output for (x, y_1) and (x, y_2). The other claim works symmetrically. □

ECCC ISSN 1433-8092

http://eccc.hpi-web.de