

The Adversarial Noise Threshold for Distributed Protocols

William M. Hoza*

Leonard J. Schulman†

Abstract

We consider the problem of implementing distributed protocols, despite adversarial channel errors, on synchronous-messaging networks with arbitrary topology.

In our first result we show that any n -party T -round protocol on an undirected communication network G can be compiled into a robust simulation protocol on a sparse ($\mathcal{O}(n)$ edges) subnetwork so that the simulation tolerates an adversarial error rate of $\Omega(\frac{1}{n})$; the simulation has a round complexity of $\mathcal{O}(\frac{m \log n}{n} T)$, where m is the number of edges in G . (So the simulation is work-preserving up to a log factor.) The adversary's error rate is within a constant factor of optimal. Given the error rate, the round complexity blowup is within a factor of $\mathcal{O}(k \log n)$ of optimal, where k is the edge connectivity of G . We also determine that the maximum tolerable error rate on *directed* communication networks is $\Theta(1/s)$ where s is the number of edges in a minimum equivalent digraph.

Next we investigate adversarial *per-edge error rates*, where the adversary is given an error budget on each edge of the network. We determine the limit for tolerable per-edge error rates on an arbitrary directed graph to within a factor of 2. However, the construction that approaches this limit has exponential round complexity, so we give another compiler, which transforms T -round protocols into $\mathcal{O}(mT)$ -round simulations, and prove that for polynomial-query black box compilers, the per-edge error rate tolerated by this last compiler is within a constant factor of optimal.

1 Introduction

We consider the problem of protecting distributed protocols from channel noise. The two-party case has received extensive attention, while the multiparty case is less explored (see Gelles' survey [Gel15].) Two

prior works are especially relevant to this paper. Rajagopalan and Schulman [RS94] showed how to protect synchronous distributed protocols on digraphs with m edges and n vertices against stochastic noise (at a constant noise rate per bit transmission), slowing down by a factor of $\log(\max \text{degree})$. The first study of adversarial noise on multiparty ($n > 2$) networks is by Jain, Kalai, and Lewko [JKL15]. They focused on a "sequential" communication model, in which there is at most one message in-flight in the network at any time. Their networks are undirected (which throughout this work we equate with a symmetric or bidirected digraph), and they show that if the graph contains one party who is connected to every other (a star subnetwork), then every "semi-adaptive" T -round protocol can be compiled into an $\mathcal{O}(T)$ -round simulation protocol which tolerates an adversarial bit error rate of $\Omega(\frac{1}{n})$. They point out that this error rate is within a constant factor of optimal, because with an error budget of this order, the adversary can effectively cut off one party from the rest of the graph. They also prove another negative result, showing that in a certain black-box model, even if the adversary is restricted to a separate budget of errors for each party's outgoing messages, no constant error rate can be tolerated.

We return in this paper to the model of synchronous distributed protocols—in each unit of time, each party transmits one bit to each of its out-neighbors, as in [RS94]—but, unlike [RS94] and like [JKL15], we treat adversarial error. Specifically, the adversary is assumed to know the inputs to all the parties, and the entire history of communications up to the present. Only the private randomness of the parties is unknown to the adversary. Our primary objective is to determine (up to a constant) the noise threshold at which reliable communication becomes possible. On undirected networks, we provide simulation protocols which achieve this threshold and which are within a factor of $\mathcal{O}(k \log n)$ of optimal in round complexity (for protocols achieving the threshold), where k is the edge connectivity of the network.

1.1 Outline of our results The starting point for our main result is a slight variant of the compiler constructed by Rajagopalan and Schulman [RS94], which

*Caltech, Pasadena, CA 91125. Supported by a Nellie Bergen and Adrian Foster Tillotson Summer Undergraduate Research Fellowship from the California Institute of Technology, as well as by the ARCS Los Angeles Founder Chapter. whoza@caltech.edu

†Caltech, Pasadena, CA 91125. Supported in part by NSF Award 1319745. schulman@caltech.edu

we refer to as the *RS compiler* (see Appendix A for the modifications). Previously this compiler was analyzed for stochastic errors. We show (Proposition 1) that the RS compiler tolerates an adversarial error rate of $\Omega(\frac{1}{m})$. On networks with bounded edge connectivity, if we only consider simulation protocols which run on the same networks as the original protocols, this is within a constant factor of the best possible error rate: with an error budget of this order, the adversary can effectively disconnect the network. Thus, to tolerate a higher error rate, we are forced to consider simulations running on subnetworks. Note that it would not suffice for the parties to simply send dummy messages on those edges which they do not want to use; rather, our model explicitly allows simulations to run on subnetworks. It may seem strange that turning off edges can help with noise resiliency but the key is that we are able to redesign the protocol so that, informally, it “relies on all remaining edges evenly”; consequently, the adversary’s most effective attacks, which apply the entire error budget to a small region, have an advantage factor of only n rather than m . In outline, we achieve this as follows, given an arbitrary protocol on an undirected communication network:

- (a) We use multicommodity flow methods to route the messages of the original protocol through a cut sparsifier.
- (b) We modify the sparse network by adding back in some of the edges which were removed, so that the routes can be short in addition to having low congestion.
- (c) We apply the RS compiler to this new protocol on the second sparse subnetwork, so that the final simulation tolerates an error rate of $\Omega(\frac{1}{n})$.

This error rate is within a constant factor of optimal, as noted above. Furthermore, the round complexity blowup is within a factor of $\mathcal{O}(k \log n)$ of optimal (for protocols tolerant to this error rate), where k is the edge connectivity of the graph on which the original protocol ran (Theorem 4.) (If one permits shared randomness, there are cases in which this gap can be narrowed, as we describe in Theorem 5.)

The same basic strategy allows us to determine the optimal error rate on *directed* graphs. We say that two digraphs on the same vertex set are *reachability-equivalent* if they have the same reachability relation. A *minimum equivalent digraph* of G is a reachability-equivalent subgraph with the fewest possible edges. (See [MT69, Hsu75].) We show (Theorem 2) that any protocol on an arbitrary digraph can be simulated to tolerate an error rate of $\Omega(\frac{1}{s})$, where s is the number of edges in each minimum equivalent digraph. We

also show (Theorem 3) that this error rate is within a constant factor of optimal.

We also investigate a more restricted adversary, who has a separate budget of errors for each edge. We prove (Theorems 6 and 7) that the cutoff for tolerable per-edge error rates is $\Theta(\frac{1}{D})$, where D is the maximum finite directed distance between any two parties in the digraph. However, the positive side of that argument involves a simulation with exponential round complexity. We prove (Theorem 8) that there is a compiler which tolerates a per-edge error rate of $\Omega(\frac{1}{R})$, where R is the maximum number of distinct vertices visited in any walk through the graph; the simulations output by that compiler have round complexity $\mathcal{O}(mT)$. The proof of Theorem 8 mostly consists of extending the arguments in [RS94] to establish a tighter analysis of the RS compiler. By a similar argument to that used in [JKL15], we prove that this per-edge error rate is within a constant factor of optimal for polynomial-query black-box simulations (Theorem 9).

1.2 Prior work Classical coding theory methods designed for data transmission cannot be efficiently applied on a per-round basis to interactive protocols: either the slow-down or the error probability will be large. This problem was first addressed by Schulman for the case of two-party interactions. [Sch92] treated stochastic (positive capacity) channels and constructed a randomized compiler that transforms any T -round two-party protocol into a computationally efficient $\mathcal{O}(T)$ -round simulation protocol. Later [Sch93, Sch96] treated adversarial noise and constructed a deterministic compiler that transforms any T -round two-party protocol into an $\mathcal{O}(T)$ -round simulation protocol which tolerates adversarial error at the constant bit error rate $\frac{1}{240}$. This simulation, however, was not computationally efficient against adversarial error; it also relies on tree codes, which were shown to exist but have not yet been constructed (but see [Bra12, MS14]). Since then, the original two-party results have been improved in many respects. Braverman and Rao [BR11] improved the adversarial error rate to $\frac{1}{8}$. Gelles, Moitra, and Sahai [GMS11, GMS14] provided a computationally efficient simulation against stochastic errors which avoids the per-instance pre-sharing of random bits in [Sch92]. Brakerski and Kalai [BK12] and Brakerski and Naor [BN13] constructed computationally efficient simulations at constant adversarial error rates. Several papers focused on noise thresholds for various channels [GHS14, GH14, EGH15, BE14], while [CPT13, GSW14] investigated what is possible while preserving the privacy of information not re-

leased by the noiseless protocol. Haeupler [Hae14] showed how to extend the non-tree-code-based randomized protocol in [Sch92] to cope with adversarial error and at high rate. Kol and Raz [KR13] showed a strict separation between the communication rates in one-way and interactive two-party communication.

As mentioned previously, [RS94] treated the multiparty case for stochastic errors; since this solution depended upon tree codes, subsequent work provided effective simulations for a restricted class of communication protocols [ORS05, ORS09]. The computationally efficient simulation against stochastic errors in [GMS11, GMS14] extended to the multiparty setting, and Alon et al. [ABE⁺15] improved on that simulation by decreasing the round complexity in the case of highly connected networks. The paper closest to our work is [JKL15], which initiated the study of adversarial noise in protocols among $n > 2$ parties. The main points of comparison are: (a) We provide simulation protocols for general networks, not only those containing a spanning star subgraph—in this respect our work is more general. (b) We consider the edges of the network to be capable of carrying simultaneously one bit per edge per unit time, rather than there being only a single edge of the network on which active communication is occurring at any time—in this sense the two works are incomparable, the model in [JKL15] favoring communication complexity and ours favoring round complexity. Finally, [LV15] improved on [JKL15] in the case of a complete network by keeping the communication “balanced” across parties.

2 The Noise Threshold for Adversaries with a Global Budget

2.1 Asymptotically optimal error tolerance, and fast simulation, on undirected networks It was already shown by [JKL15] that reliable communication in an undirected n -vertex network is impossible against an adversary who can modify $\mathcal{O}(1/n)$ of the bit transmissions. (The model in [JKL15] is different but their argument applies mutatis mutandis to ours.) Our contribution is the converse to this statement:

THEOREM 1. *There exists a compiler C such that if π is a T -round protocol on a connected, undirected graph, then $C(\pi)$ tolerates a bit error rate of $\Omega(\frac{1}{n})$ and has a round complexity of $\mathcal{O}\left(\frac{m \log n}{n} T\right)$.*

(All of our graphs will be simple, i.e., without loops or multiple edges.)

2.1.1 The RS compiler The main coding-theoretic ingredient in the proof of Theorem 1 is (a

slight variant of) the *RS compiler*. The RS compiler was designed for stochastic errors, but it turns out to have good properties in the adversarial setting as well:

PROPOSITION 1. *There exists a compiler C (the RS compiler) such that if π is a T -round protocol on a digraph G , then $C(\pi)$ tolerates a bit error rate of $\Omega(\frac{1}{m})$ and has a round complexity of $\mathcal{O}(T)$.*

Proposition 1 follows easily from the analysis in [RS94]. We defer proof to Appendix A, where we prove a much stronger claim about the RS compiler, that is needed for adversaries with per-edge budgets (Theorem 8).

Since the simulations output by the RS compiler tolerate an error rate of $\Omega(1/m)$, we can increase error tolerance to $\Omega(1/\tilde{m})$ by first rerouting messages through a subgraph with $|\tilde{E}| = \tilde{m}$ edges. (See Equation 2.12.) This immediately allows us to tolerate an error rate of $\Omega(1/n)$ if G is undirected, by rerouting messages through a spanning tree. Naturally, we incur some round complexity overhead when we reroute through a sparse subgraph; most of the effort in this section will go toward minimizing this overhead.

2.1.2 Sparsification For a weighted, undirected graph (G, w) , let $\mathcal{L}_G(w)$ denote its Laplacian matrix. We will use the following theorem by de Carli Silva, Harvey, and Sato, which builds on [BSS09] (improving in turn on the earlier [BK96]).

LEMMA 1. ([DCSHS11, COROLLARY 5]) *Suppose $G = (V, E)$ is an undirected graph, $w : E \rightarrow \mathbb{R}_+$ is a weight function, and $E = E_1 \cup \dots \cup E_k$ is a partition of the edge set. For any real $\varepsilon \in (0, 1)$, there is a deterministic polynomial-time algorithm to find a subgraph $\tilde{G} = (V, \tilde{E})$ of G and a weight function $\tilde{w} : \tilde{E} \rightarrow \mathbb{R}_+$ such that for all $x \in \mathbb{R}^n$,*

$$(2.1) \quad x^T \mathcal{L}_G(w)x \leq x^T \mathcal{L}_{\tilde{G}}(\tilde{w})x \leq (1 + \varepsilon)x^T \mathcal{L}_G(w)x;$$

for all $1 \leq i \leq k$,

$$(2.2) \quad \sum_{e \in E_i} w_e \leq \sum_{e \in \tilde{E} \cap E_i} \tilde{w}_e \leq (1 + \varepsilon) \sum_{e \in E_i} w_e;$$

and $|\tilde{E}| \in \mathcal{O}\left(\frac{n+k}{\varepsilon^2}\right)$.

The following is a straightforward consequence of Lemma 1.

LEMMA 2. *Suppose $G = (V, E)$ is a connected, undirected graph. There exists a subgraph $\tilde{G} = (V, \tilde{E})$ with $|\tilde{E}| \in \mathcal{O}(n)$ such that for every cut $U \subseteq V$,*

$$(2.3) \quad \frac{5m}{n} \left| \tilde{\delta}(U) \right| \geq |\delta(U)|,$$

where $\delta(U)$ is the set of edges in G crossing U , and $\tilde{\delta}(U)$ is the set of edges in \tilde{G} crossing U .

Proof. Define $w(e) = 1$ for every $e \in E$. Partition the edge set E into n sets $E = E_1 \cup \dots \cup E_n$, where each E_i has at most $\lceil \frac{m}{n} \rceil$ edges in it. Pick $\varepsilon = \frac{1}{2}$, and let \tilde{G} be as in Lemma 1. Consider an arbitrary $e \in \tilde{E}$, say with $e \in E_i$. By Equation 2.2,

$$(2.4) \quad \sum_{e \in \tilde{E} \cap E_i} \tilde{w}_e \leq \frac{3}{2} \sum_{e \in E_i} w_e.$$

Since $w_e = 1$ for all e , the right-hand side is just $\frac{3}{2}|E_i|$, which is $\leq \frac{3}{2}\lceil m/n \rceil$. Thus, in particular, $\tilde{w}_e \leq \frac{3}{2}\lceil m/n \rceil$. Now, consider an arbitrary cut $U \subseteq V$. Let $x \in \mathbb{R}^n$ be the indicator function for U . By Equation 2.1,

$$(2.5) \quad \sum_{e \in \delta(U)} w_e \leq \sum_{e \in \delta(U)} \tilde{w}_e.$$

Since $w_e = 1$, the left sum is just $|\delta(U)|$. Since every $\tilde{w}_e \leq \frac{3}{2}\lceil m/n \rceil$, the right sum is $\leq \frac{3}{2}|\tilde{\delta}(U)| \cdot \lceil m/n \rceil$. Thus,

$$(2.6) \quad |\delta(U)| \leq \frac{3}{2} \left\lceil \frac{m}{n} \right\rceil |\tilde{\delta}(U)|$$

$$(2.7) \quad \leq \frac{3}{2} \left(\frac{m}{n} + 1 \right) |\tilde{\delta}(U)|$$

$$(2.8) \quad \leq \frac{5m}{n} |\tilde{\delta}(U)|. \quad \square$$

2.1.3 Routing Suppose \mathcal{N} is a multicommodity flow network on $G = (V, E)$; let d_i denote the demand of commodity i . We say that the *value* of a flow F is the largest number $\lambda \in [0, 1]$ such that for every i , λd_i units of commodity i flow from the source of i to the sink of i in F . The *maximum concurrent flow* of \mathcal{N} is the largest value of any flow. For any cut $U \subseteq V$, we let $\text{Cap}(U)$ denote the sum of the capacities of edges crossing U , and we let $\text{Dem}(U)$ denote the sum of the demands of commodities whose sources and sinks are on opposite sides of U . We rely on the following approximate max-flow min-cut theorem for multicommodity flow in undirected networks, due to Linial, London, and Rabinovich.

LEMMA 3. ([LLR95, THEOREM 4.1]) *Let \mathcal{N} be a k -commodity undirected flow network on $G = (V, E)$ and λ its maximum concurrent flow. There is a deterministic polynomial-time algorithm which, given \mathcal{N} , finds a cut $U \subseteq V$ such that*

$$(2.9) \quad \frac{\text{Cap}(U)}{\text{Dem}(U)} \leq \mathcal{O}(\log k) \cdot \lambda.$$

Given just a digraph G , we can naturally define an m -commodity flow network \mathcal{N}_G on G : the commodity associated with edge (P_i, P_j) has source P_i , sink P_j , and demand 1; every edge has capacity 1. (This idea also was useful in [LR99, sections 3.16, 3.17].) Combining Lemma 2 with Lemma 3, we can prove the following lemma.

LEMMA 4. *Suppose G is an undirected graph. There exists a flow for \mathcal{N}_G with value $\Omega\left(\frac{n}{m \log m}\right)$ which uses only $\mathcal{O}(n)$ edges.*

Proof. Let \tilde{G} be as in Lemma 2, and let \mathcal{N} denote the m -commodity flow network on \tilde{G} with all the same commodities as \mathcal{N}_G (and with every edge in \tilde{G} still having capacity 1.) For any cut U , the capacity $\text{Cap}(U)$ is just the number of edges in \tilde{G} which cross U , i.e. $|\tilde{\delta}(U)|$; the demand $\text{Dem}(U)$ is just the number of edges in G which cross U , i.e. $|\delta(U)|$. Thus, if we let U be that guaranteed by Lemma 3 for \mathcal{N} , we have

$$(2.10) \quad \frac{n}{5m} \leq \frac{|\tilde{\delta}(U)|}{|\delta(U)|} = \frac{\text{Cap}(U)}{\text{Dem}(U)} \leq \mathcal{O}(\log m) \cdot \lambda,$$

and hence $\lambda \in \Omega\left(\frac{n}{m \log m}\right)$. Of course, the same flow which achieves this λ in \mathcal{N} can be used in \mathcal{N}_G , completing the proof. \square

Flows are allowed to be fractional, but ultimately, we are interested in integer flows (i.e. collections of paths.) The following lemma quantifies the sense in which fractional flows do not cause too much trouble.

LEMMA 5. *Suppose $G = (V, E)$ is a digraph, and there is a flow F for \mathcal{N}_G with value λ which only uses s edges. Then there exists a set \mathcal{P} of m paths through G , containing one path from P_i to P_j for each $(P_i, P_j) \in E$, which uses at most s distinct edges in total and which has congestion at most $9\left(\frac{1}{\lambda} + \ln m\right)$.*

The proof of Lemma 5 is a straightforward probabilistic argument, which we defer to Appendix B.

LEMMA 6. *Suppose $G = (V, E)$ is a connected, undirected graph. There exists a subgraph $\tilde{G} = (V, \tilde{E})$ with $\mathcal{O}(n)$ edges and a set \mathcal{P} of m simple paths through \tilde{G} , such that*

(i) \mathcal{P} contains one path from P_i to P_j for each $(P_i, P_j) \in E$, and

(ii) \mathcal{P} has dilation $\mathcal{O}\left(\frac{m \log n}{n}\right)$ and congestion $\mathcal{O}\left(\frac{m \log n}{n}\right)$.

Proof. From Lemmas 4 and 5, there exists a set \mathcal{P}_0 of m paths, containing one path from P_i to P_j for each $(P_i, P_j) \in E$, which uses $\mathcal{O}(n)$ distinct edges in total and which has congestion $\mathcal{O}(\frac{m \log m}{n})$. Let p_{ij} denote the path from P_i to P_j in \mathcal{P}_0 . Define a path p'_{ij} from P_i to P_j by

$$(2.11) \quad p'_{ij} = \begin{cases} p_{ij} & \text{if } p_{ij} \text{ has length } \leq \frac{m \log m}{n} \\ (P_i, P_j) & \text{otherwise.} \end{cases}$$

Let $\mathcal{P} = \{p'_{ij} : (P_i, P_j) \in E\}$, and let \tilde{E} be the set of edges used by \mathcal{P} . Because of the bounds of \mathcal{P}_0 , the sum of the lengths of the paths in \mathcal{P}_0 must be $\mathcal{O}(m \log m)$. Therefore, in particular, the number of paths in \mathcal{P}_0 of length at least $\frac{m \log m}{n}$ is $\mathcal{O}(n)$. Therefore, \mathcal{P} still uses only $\mathcal{O}(n)$ distinct edges in total. Furthermore, by construction, the dilation of \mathcal{P} is no more than $\frac{m \log m}{n}$. Finally, the congestion on an edge e in \mathcal{P} is no more than the congestion of that edge in \mathcal{P}_0 , plus 1 for the length-1 path across e which may be in $\mathcal{P} \setminus \mathcal{P}_0$. Thus, in particular, the congestion of \mathcal{P} is still $\mathcal{O}(\frac{m \log m}{n})$. Of course, $\mathcal{O}(\log m) = \mathcal{O}(\log n)$, so we are done. \square

2.1.4 Scheduling We use the following fundamental theorem of Leighton, Maggs and Rao.

LEMMA 7. ([LMR94, THEOREM 3.4]) *Suppose G is a digraph, and \mathcal{P} is a set of simple paths through G with dilation ℓ and congestion c . There exists a schedule for routing packets along the paths in \mathcal{P} , with at most one packet traversing each edge in each time step, in a total of $\mathcal{O}(c + \ell)$ time steps.*

For positive results like Theorem 1, it suffices to show how to simulate the *universal protocol* $\pi^*[G, T]$, which is a T -round deterministic protocol running on the digraph G defined as follows. If P_i is a party with indegree d_i^- and outdegree d_i^+ , a T -round *transmission function* for P_i is a function

$$x_i : \left(\bigcup_{t=0}^{T-1} \{0, 1\}^{td_i^-} \right) \rightarrow \{0, 1\}^{d_i^+}.$$

In $\pi^*[G, T]$, each party receives a transmission function as input and “does as it instructs.” Formally, fix a party P_j and input x_j , and suppose t rounds have transpired. Let $y_i \in \{0, 1\}^t$ be the sequence of bits that P_j has received from P_i so far. Let $i_1, \dots, i_{d_j^-}$ be the indices of the in-neighbors of P_j , and let $k_1, \dots, k_{d_j^+}$ be the indices of the out-neighbors of P_j . Let $(z_1, \dots, z_{d_j^+}) = x_j(y_{i_1}, \dots, y_{i_{d_j^-}})$. Then in round $t + 1$, P_j sends P_{k_r} the bit z_r . Finally, after the T rounds of communication are complete, P_j 's output

is the sequence of all Td_j^- bits she received. We will just write π^* if G and T are clear.

Proof. [of Theorem 1] It suffices to describe $\tilde{\pi}^* = C(\pi^*)$. The compiler C is formed by composing a “sparsifying compiler” with the RS compiler, as depicted in Equation 2.12.

$$(2.12) \quad C : \pi^* \xrightarrow{\text{Sparsifying compiler}} \pi' \xrightarrow{\text{RS compiler}} \tilde{\pi}^*$$

Let \tilde{G} and \mathcal{P} be as in Lemma 6; the intermediate protocol π' runs on \tilde{G} . On input $x = (x_1, \dots, x_n)$, each round of π^* is simulated by $\mathcal{O}(\frac{m \log n}{n})$ rounds in π' as follows. Assume inductively that we have already simulated τ rounds. Based on these simulations, for each $(P_i, P_j) \in E$, there is some bit b_{ij} which x_i instructs P_i to send to P_j during round $\tau + 1$ of π^* . By Lemma 7, there is a schedule by which the parties can coordinate so that every b_{ij} reaches its destination after $\mathcal{O}(\frac{m \log n}{n})$ rounds; the parties follow this schedule. Thus, π' successfully simulates π^* on a noiseless network, and runs in $\mathcal{O}(\frac{m \log n}{n} T)$ rounds. Therefore, by Proposition 1, $\tilde{\pi}^*$ tolerates an error rate of $\Omega(\frac{1}{n})$ as a simulation of π^* , and still runs in $\mathcal{O}(\frac{m \log n}{n} T)$ rounds. \square

The sparse subgraph can be efficiently constructed, as stated in Lemma 1. There are efficient algorithms for constructing multicommodity flows that are within a factor of $1 + \epsilon$ of optimal; see e.g. [Mad10]. The proofs of Lemmas 5 and 6 can be implemented as efficient randomized algorithms in a straightforward way. Efficient randomized algorithms are also known which construct schedulers with the parameters of Lemma 7 [LMR99]. The RS compiler is not computationally efficient in the presence of adversarial errors.

2.2 The noise threshold in arbitrary digraphs

2.2.1 Positive result (lower bound on tolerable error rates)

We can now also easily obtain a lower bound on the maximum tolerable error rate on arbitrary *directed* graphs; in this setting, results on undirected sparsification do not help us to reduce the round complexity of the simulation. What is most interesting here is identification of the graph parameter that governs the adversarial noise threshold.

THEOREM 2. *Suppose $G = (V, E)$ is a digraph without isolated vertices, and suppose each minimum equivalent digraph of G has s edges. There exists a compiler C such that if π is a T -round protocol on G , then $C(\pi)$*

tolerates a bit error rate of $\Omega(\frac{1}{s})$ and has round complexity $\mathcal{O}(mT)$.

Proof. Pick some minimum equivalent digraph $\tilde{G} = (V, \tilde{E})$. Define \mathcal{P} to include, for each $(P_i, P_j) \in E$, some simple path from P_i to P_j through \tilde{G} . Clearly, \mathcal{P} has dilation no more than n and congestion no more than m , and there are at most s distinct edges used by \mathcal{P} . The same construction as in the proof of Theorem 1 works here. \square

We remark that finding a minimum equivalent digraph is NP-hard, but there is a polynomial-time approximation algorithm with a performance guarantee of about 1.64 [KRY02].

2.2.2 Negative result (upper bound on tolerable error rates) We now show that the error rate of Theorem 2 is within a constant factor of optimal. We begin with the following result by Moyses and Thompson.

LEMMA 8. ([MT69, THEOREM 1]) *Suppose $G = (V, E)$ is a directed acyclic graph, and $\tilde{G} = (V, \tilde{E})$ is a minimum equivalent digraph of G . Then \tilde{E} is exactly the set of edges $(P_i, P_j) \in E$ such that there is no path from P_i to P_j through G which avoids the edge (P_i, P_j) .*

Suppose $G = (V, E)$ is a digraph. We define the *relative edge connectivity (REC)* of G to be the least k such that there are k edges whose removal from G changes the reachability relation. For example, if G is strongly connected, then its REC is simply its edge connectivity.

LEMMA 9. *Suppose $G = (V, E)$ is a digraph with no isolated vertices, with $\text{REC}(G) = k$. Then it has a reachability-equivalent subgraph $\tilde{G} = (V, \tilde{E})$ with no more than $5m/k$ edges.*

Proof. Let G_1, \dots, G_q be the strongly connected components of G . Let $V^* = \{G_i\}_i$, and let $G^* = (V^*, E^*)$ be the condensation of G . Define a weight function $w : E^* \rightarrow \mathbb{N}$ by saying that the weight of (G_i, G_j) is the number of edges in E going from G_i to G_j . By Lemma 8, there is a reachability-equivalent subgraph $\tilde{G}^* = (V^*, \tilde{E}^*)$ of G^* , such that for each $(G_i, G_j) \in \tilde{E}^*$, every path from G_i to G_j through G^* uses the edge $(G_i, G_j) \in E^*$. Therefore, each edge $(G_i, G_j) \in \tilde{E}^*$ must have weight at least k , since removing the edges from G_i to G_j in G would make the vertices in G_j unreachable from the vertices in G_i .

We form the subgraph $\tilde{G} = (V, \tilde{E})$ as follows. For each G_i , we define \tilde{E}_i to be the set of edges in a minimum equivalent digraph of G_i . We define \tilde{E}^D to contain one edge from G_i to G_j for each $(G_i, G_j) \in \tilde{E}^*$. We define $\tilde{E} = \tilde{E}_1 \cup \dots \cup \tilde{E}_q \cup \tilde{E}^D$. By construction, clearly, \tilde{G} is a reachability-equivalent subgraph of G .

Say each G_i has n_i vertices. Then \tilde{E}_i has no more than $2(n_i - 1)$ edges, because we can form a reachability-equivalent subgraph of G_i with $2(n_i - 1)$ edges by picking a root vertex P_i in G_i and including all edges in an in-branching of G_i rooted at P_i , as well as all edges in an out-branching of G_i rooted at P_i . Therefore, the \tilde{E}_i s have, in total, no more than $2n$ edges. Furthermore, \tilde{E}^D has no more than m/k edges, since each edge in \tilde{E}^* has weight k . Now, $n \leq 2m/k$, because k is no more than the minimum number of edges adjacent to any vertex. Therefore, in total, \tilde{G} has no more than $4m/k + m/k = 5m/k$ edges. \square

LEMMA 10. *Suppose C is a compiler and $G = (V, E)$ is a digraph. Suppose that for some $T > 0$, $C(\pi^*[G, T])$ runs on a graph $\tilde{G} = (V, \tilde{E})$ with $\tilde{m} = |\tilde{E}|$ edges. Define λ to be $\text{REC}(\tilde{G})$ if \tilde{G} is reachability-equivalent to G , and $\lambda = 0$ otherwise. Then the failure probability of $C(\pi^*[G, T])$ in the presence of the bit error rate λ/\tilde{m} is at least $1 - 2^{-T}$.*

Proof. Say $(P_i, P_j) \in E$ and S is a set of λ edges in \tilde{E} such that after removing all the edges in S from \tilde{G} , there is no path from P_i to P_j . Consider the adversary \mathcal{A} who zeroes out all messages sent across every edge $e \in S$. Consider choosing an input x uniformly at random. For any transcript at P_j , the probability of that transcript conditioned on any input that P_i might receive is equally likely. Thus, P_j has only a 2^{-T} chance of correctly guessing the T bits that P_i would have sent P_j if they had followed $\pi^*[G, T]$. \square

Observe that Lemma 10 shows that on undirected networks with bounded edge connectivity, the error rate $\Omega(\frac{1}{m})$ is optimal, among simulations which run on that same network.

THEOREM 3. *Suppose C is a compiler and G is a digraph without isolated vertices, for which each minimum equivalent digraph has s edges. Then for all $T > 0$, the failure probability of $C(\pi^*[G, T])$ in the presence of the bit error rate $5/s$ is at least $1 - 2^{-T}$.*

Proof. Let $\tilde{G} = (V, \tilde{E})$ be the graph on which $C(\pi^*[G, T])$ runs. Say $\tilde{k} = \text{REC}(\tilde{G})$, and $\tilde{m} = |\tilde{E}|$. If \tilde{G} is not reachability-equivalent to G , we are done by Lemma 10. Otherwise, $s \leq \tilde{s}$, where \tilde{s} is the

number of edges in each minimum equivalent digraph of \tilde{G} . By Lemma 9, $\tilde{s} \leq 5\tilde{m}/k$. Thus, $5/s \geq \tilde{k}/\tilde{m}$; an application of Lemma 10 completes the proof. \square

2.3 Lower bound on the round complexity of robust simulations If G has a small relative edge connectivity, then simulations of protocols on G must run on subgraphs to achieve optimal error tolerance. Naturally, there is a round complexity cost associated with moving to a sparse subgraph. These two ideas prove the following theorem.

THEOREM 4. *Suppose C is a compiler, $\rho \in [0, 1]$, and G is a digraph with $\text{REC}(G) = k$. Suppose the round complexity of $C(\pi^*[G, T])$ is less than $\frac{m\rho}{k}T$. Then the failure probability of $C(\pi^*[G, T])$ in the presence of the bit error rate ρ is at least $\frac{1}{2}$.*

Proof. Suppose $C(\pi^*[G, T])$ runs on a subgraph \tilde{G} of G , with \tilde{m} edges and with $\text{REC}(\tilde{G}) = \tilde{k}$. By Lemma 10, if $\rho \geq \tilde{k}/\tilde{m}$, we are done, so assume $\rho < \tilde{k}/\tilde{m}$. We are also done if \tilde{G} is not reachability-equivalent to G , so assume that it is, which implies that $\tilde{k} \leq k$, and hence $\tilde{m} < k/\rho$.

Say $\tilde{\pi}^*[G, T]$ runs in \tilde{T} rounds, with $\tilde{T} < Tm/\tilde{m}$. Observe that the average indegree in \tilde{G} is no more than \tilde{m}/m times the average indegree in G , so there is some party P_i whose indegree \tilde{d}_i^- in \tilde{G} is no more than \tilde{m}/m times her indegree d_i^- in G . Fix some input x_i for P_i , and choose every other party's input uniformly at random. At the end of the execution of the simulation protocol, P_i must guess \tilde{d}_i^-T bits based on $\tilde{d}_i^- \tilde{T}$ bits that she receives. Since $\tilde{d}_i^- \tilde{T} < d_i^-T$, the probability of success is no more than $\frac{1}{2}$. \square

2.4 Magi coding When G is connected and undirected, taking $\rho \in \Omega(\frac{1}{n})$ in Theorem 4 shows that the round complexity blowup of Theorem 1 is within a factor of $\mathcal{O}(k \log n)$ of optimal, where k is now just the edge connectivity of G . On highly connected graphs, this leaves a sizable gap. It is quite possible that there are compilers with optimal error tolerance and with round complexity lower than that achieved in Theorem 1. The following theorem establishes that this is at least true if the parties share access to a common random string (unavailable to the adversary), and if we make a strong assumption on connectivity.

THEOREM 5. *Suppose $G = (V, E)$ is an undirected graph such that for every $(P_i, P_j) \in E$, the endpoints P_i and P_j have $\Omega(n)$ common neighbors. There exists a shared-randomness compiler C such that if π is a T -round protocol on G , then $C(\pi)$ tolerates a bit error*

rate of $\Omega(\frac{1}{n})$ with failure probability $e^{-\Omega(T)}$, and $C(\pi)$ has round complexity $\mathcal{O}(T \log n)$.

The compiler C used to prove Theorem 5 is formed by composing the RS compiler with a *magi coding*¹ compiler, as in Equation 2.13. Note that the RS compiler comes first in this composition, in contrast to the compilers used to prove Theorems 1 and 2.

$$(2.13) \quad C : \pi^* \xrightarrow{\text{RS compiler}} \pi' \xrightarrow{\text{Magi coding compiler}} \tilde{\pi}^*$$

We will need the following fact about the RS compiler, which is stronger than Proposition 1. (The proof is in Appendix A.)

PROPOSITION 2. *There exists a compiler C (the RS compiler) and $\eta > 0$ such that if an execution of $C(\pi^*)$ fails, then in that execution, the fraction of rounds in which bit errors occurred was at least η .*

The idea of the magi coding compiler is straightforward: parties send bits to randomly chosen third parties, who deliver them to their recipients.

2.4.1 Description of magi coding Since π' is deterministic, immediately upon receiving her input, every party P_i can compute the transmission function x'_i which describes her behavior in π' . Magi coding works by simulating each round of π' individually; the simulation of a single round is given by Algorithm 1. That is, at the beginning of an execution of Algorithm 1, for each edge (P_i, P_j) , P_i has in mind a bit b_{ij} that she would like to send to P_j ; at the end of the execution, P_j has a guess \hat{b}_{ij} about the value of b_{ij} . If we have inductively simulated τ rounds of π' in this way, then in round $\tau + 1$, each bit b_{ij} is determined by x'_i under the assumption that the previous estimates $\{\hat{b}_{j'i}\}$ were all correct.

Say that every adjacent pair of vertices have at least εn common neighbors. Algorithm 1 makes reference to a number ℓ . We define

$$(2.14) \quad \ell = \frac{24}{\varepsilon} \log \left(\frac{2m}{\alpha} \right),$$

where α is a parameter to be chosen later. For our purposes, it will suffice to take $\alpha = \frac{1}{4}\eta$, where η is that given in Proposition 2.

We refer to the 2ℓ rounds of communication needed to execute the preceding algorithm as one *segment*. The simulation runs for T' segments, where T' is the round complexity of π' .

¹To avoid King Herod, the biblical Magi went home along a different route than they had planned. (Matthew 2:12)

Algorithm 1 A single segment of magi coding.

- 1: **repeat** ℓ times:
 - 2: Using shared randomness, pick a random number $1 \leq r \leq n$, as well as two random bits w_{ij}, w'_{ij} for each $(P_i, P_j) \in E$.
 - 3: **for all** edges $(P_i, P_k) \in E$, all simultaneously:
 - 4: Define j so that $i + j + k \equiv r \pmod{n}$.
 - 5: P_i sends bit $b_{ij} \oplus w_{ij}$ to P_k , who receives $y_{ij} = b_{ij} \oplus w_{ij} \oplus \text{noise}$.
 - 6: **for all** edges $(P_k, P_j) \in E$, all simultaneously:
 - 7: Define i so that $i + j + k \equiv r \pmod{n}$.
 - 8: P_k sends bit $y_{ij} \oplus w'_{ij}$ to P_j , who receives $z_{ij} = b_{ij} \oplus w_{ij} \oplus \text{noise} \oplus w'_{ij} \oplus \text{more noise}$.
 - 9: P_j casts a vote for $z_{ij} \oplus w_{ij} \oplus w'_{ij}$, in an election for the office of \hat{b}_{ij} with candidates $\{0, 1\}$.
 - 10: By majority vote, for each $(P_i, P_j) \in E$, P_j decides on an estimate \hat{b}_{ij} .
-

2.4.2 Analysis of magi coding We say that a *simulated bit error* occurs in segment t if, at the end of segment t , for some edge (P_i, P_j) , $b_{ij} \neq \hat{b}_{ij}$.

LEMMA 11. *Fix a segment. Suppose that in that segment, the adversary introduces at most $\frac{1}{32}\varepsilon n\ell$ bit errors. Then the probability of a simulated bit error in that segment is no more than α .*

Proof. Fix $(P_i, P_j) \in E$. In each iteration of the loop, the probability that r is chosen such that $i + j + r \pmod{n}$ is the index of a common neighbor of P_i and P_j is at least ε . Thus, the expected number of votes that P_j casts regarding \hat{b}_{ij} is $\varepsilon\ell$. These are independent events, so by the Chernoff bound, the probability that fewer than $\frac{1}{2}\varepsilon\ell$ such votes are cast is no more than $\exp(-\frac{1}{8}\varepsilon\ell) \leq \frac{\alpha}{2m}$.

Say the number of bit errors that the adversary introduces during the τ th iteration of the main loop of the magi coding algorithm is a_τ , for $1 \leq \tau \leq \ell$. Fix some iteration τ of that loop. For any edge e , the probability that b_{ij} is sent across e during that iteration is no more than $\frac{2}{n}$, because of the choice of r . Furthermore, in both rounds in that iteration, it remains true conditioned on all bits that have been transmitted (i.e. on all the information that the adversary has) that the probability that b_{ij} is sent across e in that round is no more than $\frac{2}{n}$. (This was the purpose of the random bits w_{ij}, w'_{ij} .) Therefore, by the union bound, the probability that the adversary corrupts b_{ij} in this iteration is at most $\frac{4a_\tau}{n}$.

The expected number of iterations in which the adversary corrupts b_{ij} is no more than $\frac{1}{8}\varepsilon\ell$. These events are independent, so by the Chernoff bound,

the probability that more than $\frac{1}{4}\varepsilon\ell$ incorrect votes are cast is no more than $\exp(-\frac{1}{24}\varepsilon\ell) = \frac{\alpha}{2m}$.

If, at the end of the segment, $b_{ij} \neq \hat{b}_{ij}$, then either P_j cast fewer than $\frac{1}{2}\varepsilon\ell$ votes regarding \hat{b}_{ij} , or else P_j cast at least $\frac{1}{4}\varepsilon\ell$ incorrect votes regarding \hat{b}_{ij} . Therefore, by the union bound, $\Pr(\hat{b}_{ij} \neq b_{ij}) \leq \frac{\alpha}{m}$. Taking another union bound over the m edges (P_i, P_j) completes the proof. \square

Observe that magi coding does not simply make it difficult for the adversary to create a high simulated bit error rate. Indeed, if she wants to introduce a simulated bit error rate of ρ , she can simply choose a ρ fraction of the segments and corrupt every single edge in the even-numbered rounds of the chosen segments, costing her an actual bit error rate of $\rho/2$. Rather, the gain from magi coding is that it makes it difficult for the adversary to introduce a positive number of simulated bit errors in a large number of segments:

LEMMA 12. *Suppose that during the execution of $\tilde{\pi}^*$, the adversary introduces at most $\frac{1}{16}\alpha\varepsilon nT'$ bit errors. Then the probability that at least one simulated bit error occurs in each of $4\alpha T'$ different segments is $e^{-\Omega((1-2\alpha)T')}$.*

Proof. Say that a segment is *targeted* if the adversary introduces at least $\frac{1}{32}\varepsilon\ell n$ bit errors in that segment. By hypothesis, at most $2\alpha T'$ segments are targeted. On the other hand, from Lemma 11, we know that in each non-targeted segment, the probability that at least one simulated bit error occurs is at most α . There are T' segments total, so the expected number of non-targeted segments in which at least one simulated bit error occurs is at most $\alpha T'$. There are at least $(1 - 2\alpha)T'$ non-targeted segments, and these events are all independent. Therefore, by the Chernoff bound, the probability that at least one simulated bit error occurs in $2\alpha T'$ different non-targeted segments is $e^{-\Omega((1-2\alpha)T')}$. \square

Proof. [of Theorem 5] Note that $\tilde{\pi}^*$ runs in $2\ell T'$ rounds, which is $\mathcal{O}(T \log n)$ rounds as claimed. If $\tilde{\pi}^*$ fails, then by Proposition 2, there were at least $\eta T'$ distinct segments in which a simulated bit error occurred. By Lemma 12 with $\alpha = \frac{1}{4}\eta$, as long as the adversary is restricted to introducing at most $\frac{1}{16}\alpha\varepsilon nT'$ bit errors, then the probability of simulated bit errors occurring in $\eta T'$ distinct segments is $e^{-\Omega(T)}$. This tolerable amount of error corresponds to the bit error rate ρ given by

$$(2.15) \quad \rho = \frac{\frac{1}{16}\frac{1}{4}\eta\varepsilon\ell nT'}{m\tilde{T}} = \frac{\eta\varepsilon n}{128m} \geq \frac{\eta\varepsilon}{128n}. \quad \square$$

3 The Noise Threshold for Adversaries with a Per-Edge Budget

In this section, we are interested in *per-edge error rates*, where we restrict the distribution of errors as well as the total number. Specifically, we say that an adversary stays within the per-edge error rate ρ budget if on each edge, the fraction of bits transmitted on that edge which are flipped is no more than ρ . Note that when we are considering per-edge error rates, we can assume without loss of generality that simulations run on the same graphs as the original protocols.

For a digraph G , we define the *signal diameter* D of G to be the maximum finite distance between any two vertices in G . That is, the signal diameter of G is the maximum, over all P_i, P_j for which P_j is reachable from P_i , of the length of the shortest path from P_i to P_j . For example, if G is strongly connected, then the signal diameter of G is just the ordinary diameter of G .

3.1 Optimal per-edge error rates, ignoring round complexity

3.1.1 Positive result

THEOREM 6. *Suppose G is a digraph with signal diameter D . For every $\varepsilon > 0$, there exists a compiler C such that if π is a protocol on G , then $C(\pi)$ tolerates the per-edge error rate $\frac{1}{4D} - \varepsilon$, and $C(\pi)$ has round complexity $\mathcal{O}(D^2 n T 2^{nT})$.*

Proof. We describe $C(\pi^*[G, T])$. By the Gilbert-Varshamov bound, there is some family of error correcting codes χ with positive asymptotic rate and minimum relative distance at least $\frac{1}{2} - D\varepsilon$. Let ℓ be sufficiently long so that for any length- D list F of T -round transmission functions on G , $\chi(F)$ has length no more than ℓ . Set $\tilde{T} = D\ell$. Divide the \tilde{T} rounds into D segments of length ℓ . In the j th segment, P_i transmits (to all of her out-neighbors) the encoding under χ of the list $(x_{k_1}, \dots, x_{k_m})$ of transmission functions of parties P_{k_s} such that there is a path of length $< j$ from P_{k_s} to P_i .

First, suppose some decoding operation failed. Because of the minimum relative distance property, the adversary must have introduced at least $\frac{1}{4}\ell(1 - D\varepsilon)$ bit errors on some edge, which is a per-edge error rate of $\frac{1}{4D} - \frac{1}{4}\varepsilon$, which exceeds the specified budget. Suppose instead that all decoding operations succeed. Then every party P_i knows x_j for every party P_j from which P_i is reachable. Using this information, P_i can infer all of the bits that she would have received if the parties had followed π^* on a noiseless network. Finally, for the round complexity estimate, note that trivially every party has degree at most n . Hence, to

specify a T -round transmission function, it suffices to specify the nT bits that a party would send, given any arbitrary length- n list of T -bit incoming strings. Hence, a list of D such transmission functions can be specified with $DnT2^{nT}$ bits, so ℓ is $\mathcal{O}(DnT2^{nT})$. \square

3.1.2 Negative result We give a matching (up to a factor of 2) negative result, showing that the error rate $\frac{1}{2D}$ cannot be tolerated.

THEOREM 7. *Suppose C is a compiler and G is a digraph with signal diameter D . Then for all sufficiently large T , there exists a T -round protocol π such that the failure probability of $C(\pi)$ in the presence of the per-edge error rate $\frac{1}{2D}$ is at least $\frac{1}{4}$.*

Proof. Select vertices P_0, \dots, P_D such that (P_0, \dots, P_D) is a shortest path from P_0 to P_D . Pick any integer $L > 2mD$. In the protocol π_L , P_0 receives an L -bit string x as input, and transmits it to P_D along a shortest path, so that π_L runs in $T = L + D - 1$ rounds. Say $C(\pi_L)$ runs in \tilde{T} rounds.

The strategy of the adversary is to sample two possible inputs x, x' to P_0 in such a way that (a) $x \neq x'$ with probability at least $1/2$; (b) The probability distribution on the transcripts of all channel communications leading into P_D , is the same whether x or x' were given to P_0 as input. The theorem will follow.

However, the adversary cannot commit to the pair x, x' at the very outset so her strategy is slightly more complicated. Let ℓ be the largest even integer s.t. $\ell \leq \tilde{T}/D$, and let $B = \tilde{T} - D\ell$. Note then that $B < 2D$.

First the adversary selects an input x u.a.r. in $\{0, 1\}^L$. Then she allows the protocol to proceed without interference for B rounds. Let $\chi \in \{0, 1\}^{mB}$ be the random variable denoting the transcript generated by all parties in the network during these rounds. The adversary knows the (possibly randomized) simulation protocol and therefore knows the conditional probabilities of transcripts given inputs. She now samples $x' \in \{0, 1\}^L$ from the posterior distribution (given χ and the uniform prior on $\{0, 1\}^L$). For $a, b \in \{0, 1\}^L$ and $c \in \{0, 1\}^{mB}$, let $[a, b, c]$ denote the event that a was chosen as the input x , c was the transcript χ , and b was chosen as the “alternate” input x' . The key property of this construction is that for any a, b, c , $\Pr([a, b, c]) = \Pr([b, a, c])$.

Before continuing to describe the adversary’s strategy, let us argue already why (a) holds. Consider using the following alternate sampling rule for x' : for each χ , instead of selecting x' from the posteriori distribution, select x' to be the max-likelihood decoding of χ . This can only increase $\Pr(x = x')$.

This creates a deterministic decoding map from transcripts χ to inputs, which means that there is a set of at most 2^{mB} inputs x on which it can ever occur that $x = x'$. The probability that x is selected from this set is $2^{mB-L} < 2^{2mD-L} \leq 1/2$.

The adversary now breaks the remaining $D\ell$ rounds of the protocol into D segments, each of ℓ rounds. Each segment is further broken into two half-segments, each of $\ell/2$ rounds. Let $d(P, P')$ be the length of a shortest directed path (possibly infinite) from vertex P to vertex P' . Let $V_k = \{P : d(P_0, P) = k\}$, and let $W_k = \bigcup_{k' \geq k} V_{k'}$. At the beginning of segment k ($1 \leq k \leq D$) she flips a fair coin to decide whether to attack the first or second half of the segment. During the half-segment that she attacks, she substitutes messages of her choice for all the messages from V_{k-1} to V_k . The manner in which she generates these messages is as follows.

The adversary's strategy is to simulate an imaginary, "alternative reality" portion of the network, that gradually grows. See Figures 1, 2. For the duration of the first segment, the simulated network portion consists of a single vertex \bar{P}_0 , mirroring the actual network vertex P_0 . During the second segment the simulated region grows to mirror the induced network on $\{P_0\} \cup V_1$ (or what is the same, $V_0 \cup V_1$). In general during the k th segment the simulated network region is a copy of the induced graph on $V_0 \cup \dots \cup V_k$. Throughout the entire protocol, the adversary continues simulating communications on this gradually growing region; during attacking half-segments, the adversary replaces the $V_{k-1} \rightarrow V_k$ messages by $\bar{V}_{k-1} \rightarrow \bar{V}_k$ messages, that is, she substitutes the outgoing messages of the simulated reality on $V_0 \cup \dots \cup V_k$ for the outgoing messages of the real vertices in that region.

It remains to describe how the states of these imaginary vertices are initialized and updated.

Updates during a segment are as follows: during segment k , the state of each imaginary vertex in $\bar{V}_0 \cup \dots \cup \bar{V}_{k-1}$ is updated in each round just as it would in the protocol, using its prior state and, as inputs, the communications from the other imaginary vertices together with any communications coming from real vertices in \bar{V}_k .

Initialization at the beginnings of segments are as follows: at time B (the beginning of the first segment), \bar{P}_0 is initialized with a random state s chosen from the posteriori distribution conditional on her input being x' and on all messages that her genuine counterpart P_0 sent and received through time B . For $k \geq 2$, at the beginning of segment k (i.e., at time $B + (k-1)\ell$), we have to enlarge the simulation to include new vertices \bar{V}_{k-1} . The

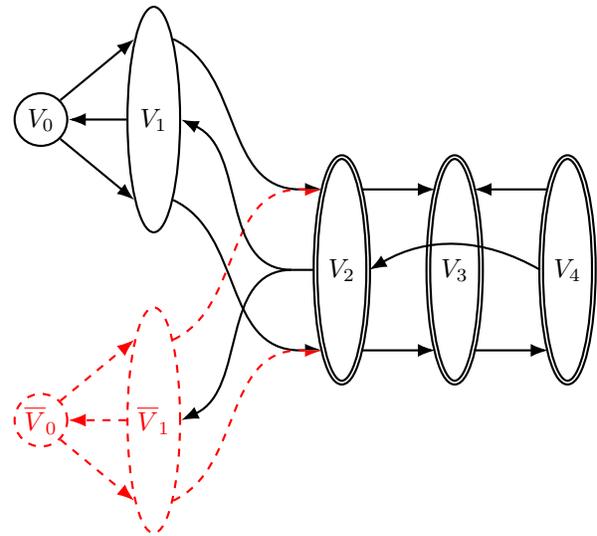


Figure 1: The adversarial strategy used to prove Theorem 7 on a graph with $D = 4$, during segment 2. Regions with solid black boundaries represent sets of actual parties, with double boundaries indicating sets of parties who do not know whether x or x' is the true input. Regions with dashed red boundaries represent sets of imaginary parties. Solid black arrows indicate channels controlled by actual parties; dashed red arrows indicate channels controlled by imaginary parties.

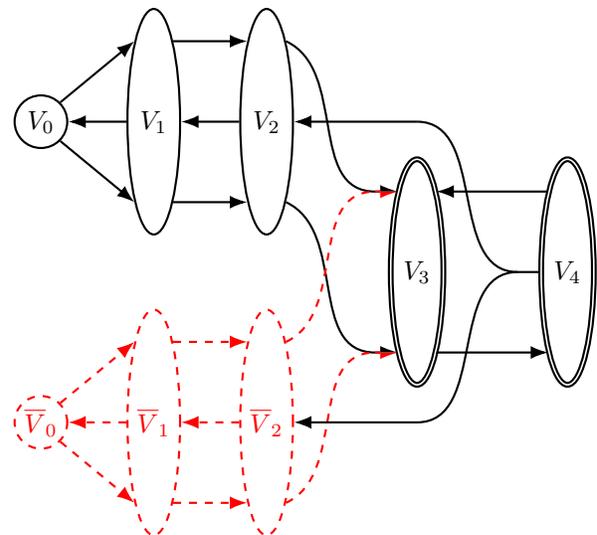


Figure 2: Segment 3 of the situation depicted in Figure 1.

existing vertices (those in $\bar{V}_0 \cup \dots \cup \bar{V}_{k-2}$) continue from their current state. Each vertex $\bar{P} \in \bar{V}_{k-1}$ is initialized with a random state s chosen from the posteriori distribution conditional on all messages that its genuine counterpart P sent and received up through time $B + (k - 1)\ell$.

Notice that the simulation is evolved forward in each round whether or not this is an attacking round. The imaginary vertices are always responding to messages coming from amongst themselves and from the real vertices. All that changes is whether V_k is hearing messages from V_{k-1} or from \bar{V}_{k-1} .

The key claim is this. Let \vec{s} denote the transcript at time $B + (k - 1)\ell$ of *all* messages ever received at vertices in W_k . Then:

LEMMA 13. For all \vec{s} , $\Pr(\vec{s} \mid [x, x', \chi]) = \Pr(\vec{s} \mid [x', x, \chi])$.

That is, to the vertices in W_k , the probability distribution over what they have (collectively) heard up until this time is the same whether the input is x or x' .

Proof. The proof is by induction on k . The base case is $k = 1$ and is simply our initial condition that $\Pr(\chi \mid x) = \Pr(\chi \mid x')$. Now for $k \geq 2$, let us denote by $h = 1$ ($h = 2$) the event that the adversary attacks during the first (resp. second) half of the $(k - 1)$ 'st segment. We claim:

(1) For all \vec{s} , $\Pr(\vec{s} \mid [x, x', \chi, h = 1]) = \Pr(\vec{s} \mid [x', x, \chi, h = 2])$.

(2) For all \vec{s} , $\Pr(\vec{s} \mid [x, x', \chi, h = 2]) = \Pr(\vec{s} \mid [x', x, \chi, h = 1])$.

We argue (1) (and (2) follows analogously). At the beginning of the $(k - 1)$ 'st segment the claim was true by induction; we need to argue that it remains so at the end of the $(k - 1)$ 'st segment, and this could break down only due to a difference in the statistics on messages from $V_{k-1} \rightarrow V_k$. This does not occur because for both events $[x, x', \chi, h = 1]$ and $[x', x, \chi, h = 2]$, what W_k hears during the first half of the $(k - 1)$ 'st segment, is messages from vertices “in the x' world”—more formally, in the event $[x, x', \chi, h = 1]$ it is vertices in \bar{V}_{k-1} acting as if the input is x' , while in the event $[x', x, \chi, h = 2]$, it is vertices in V_{k-1} , with the true input being x' ; while what W_k hears during the second half of the $(k - 1)$ 'st segment, is messages from vertices “in the x world”—more formally, in the event $[x, x', \chi, h = 1]$ it is vertices in V_{k-1} , with the true input being x , while in the event $[x', x, \chi, h = 2]$, it is vertices in \bar{V}_{k-1} acting as if the input is x .

Finally, $\Pr(\vec{s} \mid [x, x', \chi])$ is given by

$$\begin{aligned} & \frac{1}{2} \Pr(\vec{s} \mid [x, x', \chi, h = 1]) + \frac{1}{2} \Pr(\vec{s} \mid [x, x', \chi, h = 2]) \\ &= \frac{1}{2} \Pr(\vec{s} \mid [x', x, \chi, h = 2]) + \frac{1}{2} \Pr(\vec{s} \mid [x', x, \chi, h = 1]) \\ &= \Pr(\vec{s} \mid [x', x, \chi]). \end{aligned}$$

This completes the proofs of Lemma 13 and Theorem 7. \square

We remark that Theorem 7 extends to the model where parties send symbols from a large but constant-size alphabet Σ . In particular, with only trivial changes to the proof, one can show that the per-edge error rate $\frac{1}{2D}$ is not tolerable with any fixed-size alphabet. Conversely, any per-edge error rate $\frac{1}{2D} - \varepsilon$ can be tolerated using an alphabet whose size depends only on ε . This is established by using the large-alphabet version of the GV bound in the proof of Theorem 6.

3.2 Optimal per-edge error rates for black-box simulations with polynomial query complexity

The proof of Theorem 6 does not provide a practical compiler, since the round complexity \tilde{T} blows up exponentially. In this section, we give a compiler with polynomial round complexity, but with somewhat worse per-edge error tolerance. We don't show that this lower per-edge error rate is optimal for polynomial round complexity simulations, but we do show that it is optimal for polynomial-query compilers in a certain black-box model (described below).

For a digraph G , we define the *chain-length* R of G to be the maximum, over all directed walks W through G , of the number of distinct vertices visited in W . Observe that in any graph with at least one edge, the chain-length is strictly larger than the signal diameter, and that for a strongly connected graph, $R = n$.

3.2.1 Positive result

THEOREM 8. Suppose G is a digraph with chain-length R . There exists a compiler C such that if π is a T -round protocol on G , then $C(\pi)$ tolerates a per-edge error rate of $\Omega(\frac{1}{R})$ and has round complexity $\mathcal{O}(mT)$.

Most of the effort required to prove Theorem 8 consists of a new analysis of the RS compiler. The key fact (whose proof we defer to Appendix A) is the following.

LEMMA 14. There exists a compiler C (the RS compiler) such that if π is a T -round deterministic protocol

on a digraph G and an execution of $C(\pi)$ fails, then there is some walk through G on the edges of which were at least $\frac{T}{48}$ bit errors. The round complexity of $C(\pi)$ is $\mathcal{O}(T)$.

Proof. [of Theorem 8] We let C be as in the proof of Theorem 2, i.e. we reroute messages through a minimum equivalent digraph before using the RS compiler. Let $T' \in \mathcal{O}(mT)$ denote the round complexity of the intermediate protocol π' , so that the round complexity of $C(\pi^*)$ is $\mathcal{O}(T')$. If $C(\pi^*)$ fails, then by Lemma 14, there is some walk W through the minimum equivalent digraph \tilde{G} , on the edges of which were $\frac{T'}{48}$ bit errors. Since each strongly connected component H of \tilde{G} with n' vertices has no more than $2(n' - 1)$ edges, the number of edges in W is no more than $3R$. Thus, on some edge in W , there were $\frac{T'}{3R \cdot 48}$ bit errors, which is a per-edge error rate of $\Omega\left(\frac{1}{R}\right)$. \square

3.2.2 Black-box negative result We now give a result which shows that the per-edge error rate in Theorem 8 is within a constant factor of optimal, among polynomial-query compilers in a certain black-box model. Recall that in π^* (or any simulation thereof), each party P_i receives as input a transmission function x_i . We will call a simulation $\tilde{\pi}^*$ of π^* a *black-box simulation* if in $\tilde{\pi}^*$, the parties only ever access their inputs by making queries, wherein they specify an input to x_i and are given the corresponding output. Naturally, the *query complexity* of a black-box simulation is the largest number of total queries that the parties ever collectively make. A *polynomial-query black-box compiler* is a compiler C which takes as input a universal protocol $\pi^*[G, T]$ and gives as output a black-box simulation $C(\pi^*[G, T])$, such that for every graph G , there is a polynomial $\mathcal{Q}(T)$, so that the simulation $C(\pi^*[G, T])$ has a query complexity bounded by $\mathcal{Q}(T)$. Observe that (for a fixed graph G) the compiler which proved Theorem 8 makes $\mathcal{O}(T)$ queries per round, for a total query complexity of $\mathcal{O}(T^2)$. In contrast, the simulation in the proof of Theorem 6 has exponential query complexity.

THEOREM 9. *Suppose C is a polynomial-query black-box compiler. Then for any digraph G with no isolated vertices and with chain-length R , the failure probability of $C(\pi^*[G, T])$ in the presence of the per-edge error rate $\frac{4}{R}$ goes to 1 as $T \rightarrow \infty$.*

Both the statement and the proof of Theorem 9 are inspired by the black-box negative result in [JKL15].

Description of the adversary's strategy Fix some digraph G with no isolated vertices and with chain-length R . We begin with the following lemma.

LEMMA 15. *There is a walk through G of length $< n^2$ which visits R distinct vertices, with each vertex visited at most R times.*

Proof. Let v_1, \dots, v_R be the R distinct vertices visited in some chain-length walk through G , in the order in which they are visited. There is a path from v_i to v_{i+1} of length no more than n for each $1 \leq i < R$ which visits each vertex at most once. Chaining these paths together yields a walk with the desired properties. \square

Fix some positive integer T . Say \tilde{T} is the round complexity of the black-box simulation $\tilde{\pi}^* = C(\pi^*[G, T])$. Let $(P_{k_1}, P_{k_2}, \dots, P_{k_\ell})$ be the sequence of parties visited (with repetition) in the walk guaranteed by Lemma 15, so that $(P_{k_i}, P_{k_{i+1}}) \in E$ for all i and $\ell \leq n^2$. Let f_i denote the number of times that P_{k_i} is visited in this walk, so that $1 \leq f_i \leq R$. The adversary's strategy is given by Algorithm 2. Note

Algorithm 2 The strategy of the adversary \mathcal{A} used to prove Theorem 9.

- 1: **if** $\tilde{T} \geq R^2$:
 - 2: Divide the \tilde{T} rounds into ℓ segments, with segment i containing no more than $\left\lceil \frac{\tilde{T}}{Rf_i} \right\rceil$ rounds.
 - 3: **for** $i = 1$ **to** ℓ :
 - 4: Spend segment i zeroing out all messages going into or out of P_{k_i} .
 - 5: **else**:
 - 6: Do nothing.
-

that the step on line 2 is well defined, because

$$(3.16) \quad \sum_{i=1}^{\ell} \left\lceil \frac{\tilde{T}}{Rf_i} \right\rceil \geq \frac{\tilde{T}}{R} \sum_{i=1}^{\ell} \frac{1}{f_i} = \tilde{T},$$

where the last equation holds because each of R distinct parties contributes a total of 1 to the sum.

Analysis of the adversary's strategy

LEMMA 16. *The adversary \mathcal{A} described by Algorithm 2 introduces a per-edge error rate of no more than $4/R$.*

Proof. In the case $\tilde{T} < R^2$, the statement is trivial, so assume $\tilde{T} \geq R^2$. The number of rounds in which \mathcal{A} attacks an edge of the form (P_{k_i}, P_{k_j}) is no more than

$$(3.17) \quad f_i \left\lceil \frac{\tilde{T}}{Rf_i} \right\rceil + f_j \left\lceil \frac{\tilde{T}}{Rf_j} \right\rceil \leq f_i + \frac{\tilde{T}}{R} + f_j + \frac{\tilde{T}}{R}$$

$$(3.18) \quad \leq 2R + \frac{2\tilde{T}}{R}$$

$$(3.19) \quad \leq \frac{4\tilde{T}}{R}.$$

For an edge e with an endpoint which is not a P_{k_i} , \mathcal{A} attacks e in even fewer rounds than this. \square

Suppose P_i is a party with indegree d_i^- and outdegree d_i^+ . Observe that we can identify a T -round transmission function x_i for P_i with a $(2^{d_i^-})$ -ary tree of depth T whose vertices are labeled with strings in $\{0, 1\}^{d_i^+}$. The edges in a path from the root to a vertex v in this tree specify a sequence of bits received from each in-neighbor, and the label of v specifies the bits to send in the scenario described by that path. In these terms, when a party P_i in a black-box simulation $\tilde{\pi}^*$ makes a query, she effectively specifies a node in x_i and asks what its label is. For an input $x = (x_1, \dots, x_n)$, we can identify in each x_i the “true path” t_i^x from the root to a leaf of x_i , consisting of all the edges that would be taken if the parties followed π^* in a noiseless network. In these terms, the goal of the protocol is for each P_i to learn t_i^x .

LEMMA 17. *Suppose $\tilde{T} < R^2$. Then if we pick an input x uniformly at random, the probability that $\tilde{\pi}^*$ fails in the presence of \mathcal{A} on x is at least $1 - 2^{-d_1^- (R^2 - \tilde{T})}$, where d_1^- is the indegree of P_1 .*

Proof. Consider fixing an arbitrary transmission function x_1 and choosing the rest of x uniformly at random. Then P_1 needs to choose between $(2^{d_1^-})^T$ possible true paths (each of which is a priori equally likely), based on $< (2^{d_1^-})^{R^2}$ bits. Thus, the probability of success is no more than $2^{d_1^- (R^2 - T)}$. \square

DEFINITION 3.1. In the case that $\tilde{T} \geq R^2$, for $1 \leq i \leq \ell$, say that E_i is the event that at the end of segment $(i - 1)$, the following condition holds. Let u denote the node at depth $(i - 1) \lceil \sqrt{\tilde{T}} \rceil$ along $t_{k_i}^x$. Then the subtree hanging from u is completely unexplored, i.e. P_{k_i} has not made any queries about the labels of any vertices in that subtree.

LEMMA 18. *Suppose $\tilde{T} \geq R^2$ and $C(\pi^*)$ and has query complexity no more than $2^{\sqrt{\tilde{T}}}(1 - \delta)$, where $0 \leq \delta \leq 1$. Suppose we pick an input x uniformly at random, and execute $\tilde{\pi}^*$ on x in the presence of \mathcal{A} . Then for each $1 \leq i \leq \ell$, conditioned on E_1, \dots, E_{i-1} , the probability that E_i occurs is at least δ .*

Proof. For the analysis, it suffices to fix arbitrary values for any random bits that $\tilde{\pi}^*$ uses (still choosing x randomly.) Vacuously, E_1 occurs with probability 1, so assume $i > 1$. Consider an arbitrary input x such that E_1, \dots, E_{i-1} occur. Let w be the node at depth $(i - 2) \lceil \sqrt{\tilde{T}} \rceil$ in $t_{k_{i-1}}^x$, and let τ be the subtree hanging from w . Since E_{i-1} occurred, at the beginning of

segment $(i - 1)$, $P_{k_{i-1}}$ had not explored any of τ . Therefore, if x' is the same as x except for the labels of the nodes in τ , then the execution of $\tilde{\pi}^*(x')$ prior to segment $(i - 1)$ is the same as that of $\tilde{\pi}^*(x)$. Let U denote the set of locations of labels of nodes in τ which correspond to bits transmitted from $P_{k_{i-1}}$ to P_{k_i} . Consider altering x by assigning values to the bits in U uniformly at random.

Starting at depth $(i - 2) \lceil \sqrt{\tilde{T}} \rceil$ in $t_{k_i}^x$, at each level, the true path could go one of two ways, depending on the value of a bit in U which is at the same level in x_{k_i} . Thus, the node at depth $i \lceil \sqrt{\tilde{T}} \rceil$ in $t_{k_i}^x$ could be any of $2^{\lceil \sqrt{\tilde{T}} \rceil}$ different nodes, each with equal probability. Say this set of $2^{\lceil \sqrt{\tilde{T}} \rceil}$ nodes is S . At the end of segment $(i - 1)$, P_{k_i} has made fewer than $2^{\sqrt{\tilde{T}}}(1 - \delta)$ queries total, and thus the fraction of nodes in S whose subtrees she has not explored at all is at least δ . The queries she chooses to make during segment $(i - 1)$ cannot depend on the labels assigned to nodes in U , because $P_{k_{i-1}}$ is attacked by \mathcal{A} during segment $(i - 1)$. Therefore, when we assign values to U uniformly at random, the probability that E_i occurs is at least δ . Therefore, if we choose x uniformly at random, then conditioned on E_1, \dots, E_{i-1} , the probability that E_i occurs is at least δ . \square

LEMMA 19. *Suppose $\mathcal{Q}(T)$ is a polynomial. Then for all sufficiently large T ,*

$$(3.20) \quad \mathcal{Q}(T) < 2^{\sqrt{T}} (1 - 2^{-1/T}).$$

Proof. Note that $\frac{1}{2}e^{1/2} < 1$. Therefore, from the limit definition of the exponential function, we see that for sufficiently large T ,

$$(3.21) \quad \left(1 - \frac{1/2}{T}\right)^T > \frac{1}{2}e^{1/2} \cdot e^{-1/2} = \frac{1}{2}.$$

Taking a T th root of both sides gives $1 - \frac{1}{T} > 2^{-1/T}$, and therefore

$$(3.22) \quad 2^{\sqrt{T}} (1 - 2^{-1/T}) > \frac{2^{\sqrt{T}}}{T}.$$

Obviously, for sufficiently large T , the right-hand side is larger than $\mathcal{Q}(T)$. \square

Proof. [of Theorem 9] Say that the simulations output by C make a number of queries which is bounded by the polynomial $\mathcal{Q}(T)$. Since we only care about the limit as $T \rightarrow \infty$, we may assume that $T > 16n^4$, and by Lemma 19, we may also assume that

$$(3.23) \quad \mathcal{Q}(T) < 2^{\sqrt{T}} (1 - 2^{-1/T}).$$

We will show that if we pick an input x for $\pi^*[G, T]$ uniformly at random, the failure probability δ of $\tilde{\pi}^*$ in the presence of \mathcal{A} satisfies

$$(3.24) \quad \delta \geq \min\{2^{-(n^2/T)} \cdot (1 - 2^{-\frac{1}{2}T}), (1 - 2^{d_1(R^2-T)})\}$$

which in particular means that $\delta \rightarrow 1$ as $T \rightarrow \infty$. (Recall that G is fixed.) If $\tilde{T} < R^2$, then we are done by Lemma 17. Assume, therefore, that $\tilde{T} \geq R^2$.

The probability that E_i happens for all $1 \leq i \leq \ell$ is $\prod_{j=1}^{\ell} \Pr(E_i | E_1, \dots, E_{i-1})$, which by Lemma 18 is at least $2^{-\ell/T}$. Since $\ell \leq n^2$, the probability that E_j happens for all $1 \leq i \leq \ell$ is at least $2^{-n^2/T}$. Suppose $E_{\ell-1}$ happens. Then at the end of segment $(\ell - 2)$, $P_{k_{\ell-1}}$ has not made any queries about the labels of any of the vertices in the subtree τ hanging from the node at depth $(\ell - 2) \lceil \sqrt{T} \rceil$ along $t_{k_{\ell-1}}^x$. The height h of τ satisfies

$$(3.25) \quad h = T - (\ell - 2) \lceil \sqrt{T} \rceil$$

$$(3.26) \quad \geq T - n^2 \lceil \sqrt{T} \rceil$$

$$(3.27) \quad \geq T - 2n^2\sqrt{T}.$$

Since $T > 16n^4$, we have $\frac{1}{2}\sqrt{T} > 2n^2$, and hence $h \geq \frac{1}{2}T$.

During segment ℓ , all messages going into P_{k_ℓ} are zeroed out, so the output of P_{k_ℓ} at the end of the protocol does not depend on queries that $P_{k_{\ell-1}}$ makes during segment ℓ . After fixing the labels of all nodes other than those in τ , each step in $t_{k_\ell}^x$ at the level of τ could go one of two ways, based on a label of a node in τ . Therefore, conditioned in $E_{\ell-1}$, the probability that P_{k_ℓ} correctly guesses $t_{k_\ell}^x$ is no more than $2^{-\frac{1}{2}T}$, since τ has height at least $\frac{1}{2}T$. Thus, conditioned on $E_{\ell-1}$, the probability that $P_{k_{\ell+1}}$ fails to guess her transcript is at least $(1 - 2^{-\frac{1}{2}T})$. Multiplying, we see that (unconditionally) the probability that P_j fails to guess her transcript is at least $2^{-n^2/T}(1 - 2^{-\frac{1}{2}T})$, and thus Equation 3.24 is satisfied. \square

4 Acknowledgement

We thank anonymous reviewers for numerous suggestions, including the observation that Theorems 6 and 7 extend to large alphabets.

References

[ABE⁺15] N. Alon, M. Braverman, K. Efremenko, R. Gelles, and B. Haeupler. Reliable communication over highly connected noisy networks. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

[BE14] M. Braverman and K. Efremenko. List and unique coding for interactive communication in the presence of adversarial noise. In *Proc. FOCS*, 2014.

[BK96] A.A. Benczúr and D.R. Karger. Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC, pages 47–55, New York, NY, USA, 1996. ACM.

[BK12] Z. Brakerski and Y.T. Kalai. Efficient interactive coding against adversarial noise. In *Proc. 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 160–166. IEEE, 2012.

[BN13] Z. Brakerski and M. Naor. Fast algorithms for interactive coding. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 443–456, 2013.

[BR11] M. Braverman and A. Rao. Towards coding for maximum errors in interactive communication. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 159–166, New York, NY, USA, 2011. ACM.

[Bra12] M. Braverman. Towards deterministic tree code constructions. In *Proc. ITCS*, 2012.

[BSS09] J.D. Batson, D.A. Spielman, and N. Srivastava. Twice-Ramanujan sparsifiers. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC, pages 255–262, New York, NY, USA, 2009. ACM.

[CPT13] K. Chung, R. Pass, and S. Telang. Knowledge-preserving interactive coding. In *Proc. 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 449–458, Oct 2013.

[dCSHS11] M.K. de Carli Silva, N.J.A. Harvey, and C.M. Sato. Sparse sums of positive semidefinite matrices. *CoRR*, abs/1107.0088, 2011.

[EGH15] K. Efremenko, R. Gelles, and B. Haeupler. Maximal noise in interactive communication over erasure channels and channels with feedback. In *Proc. ITCS*, 2015.

[Gel15] R. Gelles. Coding for interactive communication: A survey, 2015. <http://www.cs.princeton.edu/~rgelles/papers/survey.pdf>.

[GH14] M. Ghaffari and B. Haeupler. Optimal error rates for interactive coding II: efficiency and list decoding. In *Proc. IEEE Symposium on Foundations of Computer Science (FOCS)*, 2014.

[GHS14] M. Ghaffari, B. Haeupler, and M. 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, New York, NY, USA, 2014. ACM.

[GMS11] R. Gelles, A. Moitra, and A. Sahai. Efficient and explicit coding for interactive communication. In *Proc. 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 768–777, Oct 2011.

[GMS14] R. Gelles, A. Moitra, and A. Sahai. Efficient coding for interactive communication. *IEEE Trans. on Information Theory*, 60(3):1899–191, 2014.

- [GSW14] R. Gelles, A. Sahai, and A. Wadia. Private interactive communication across an adversarial channel. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 135–144. ACM, 2014.
- [Hae14] B. Haeupler. Interactive channel capacity revisited. In *Proc. FOCS*, 2014.
- [Hsu75] H.T. Hsu. An algorithm for finding a minimal equivalent graph of a digraph. *J. ACM*, 22(1):11–16, January 1975.
- [JKL15] A. Jain, Y.T. Kalai, and A. Lewko. Interactive coding for multiparty protocols. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 1–10. ACM, 2015.
- [KR13] G. Kol and R. Raz. Interactive channel capacity. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC, pages 715–724, New York, NY, USA, 2013. ACM.
- [KRY02] S. Khuller, B. Raghavachari, and N.E. Young. Approximating the minimum equivalent digraph. *SIAM J. Comput.*, 24(cs. DS/0205040):859–872, 2002.
- [LLR95] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.
- [LMR94] F.T. Leighton, B.M. Maggs, and S.B. Rao. Packet routing and job-shop scheduling in $O(\text{congestion} + \text{dilation})$ steps. *Combinatorica*, 14(2):167–186, 1994.
- [LMR99] T. Leighton, B. Maggs, and A.W. Richa. Fast algorithms for finding $O(\text{congestion} + \text{dilation})$ packet routing schedules. *Combinatorica*, 19(3):375–401, 1999.
- [LR99] T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM (JACM)*, 46(6):787–832, 1999.
- [LV15] A. Lewko and E. Vitercik. Balancing communication for multi-party interactive coding. *arXiv preprint arXiv:1503.06381*, 2015.
- [Mad10] A. Madry. Faster approximation schemes for fractional multicommodity flow problems via dynamic graph algorithms. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 121–130. ACM, 2010.
- [MS14] C. Moore and L. J. Schulman. Tree codes and a conjecture on exponential sums. In *Proc. ITCS*, pages 145–153, 2014.
- [MT69] D.M. Moyles and G.L. Thompson. An algorithm for finding a minimum equivalent graph of a digraph. *J. ACM*, 16(3):455–460, July 1969.
- [ORS05] R. Ostrovsky, Y. Rabani, and L.J. Schulman. Error-correcting codes for automatic control. In *Proc. 46th FOCS*, pages 309–316, 2005.
- [ORS09] R. Ostrovsky, Y. Rabani, and L.J. Schulman. Error-correcting codes for automatic control. *IEEE Transactions on Information Theory*, 55(7):2931–2941, 2009.
- [RS94] S. Rajagopalan and L.J. Schulman. A coding theorem for distributed computation. In *STOC*, pages 790–799, 1994.
- [Sch92] L.J. Schulman. Communication on noisy channels: a coding theorem for computation. In *Proc. 33rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 724–733, Oct 1992.
- [Sch93] L.J. Schulman. Deterministic coding for interactive communication. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Theory of Computing*, STOC, pages 747–756, New York, NY, USA, 1993. ACM.
- [Sch96] L.J. Schulman. Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6):1745–1756, Nov 1996.

A The RS compiler

A.1 Description of compiler RS_0 The compiler described in [RS94] is essentially the compiler that we used and referred to as the RS compiler, but there are a couple of technicalities that require us to modify the compiler. We begin by briefly describing a compiler RS_0 for deterministic protocols. This compiler RS_0 is a slight variant of the compiler described in [RS94], which we will modify still further in Section A.3 to define the final RS compiler. Fix an arbitrary digraph $G = (V, E)$ and some deterministic T -round protocol π on G ; we will describe $RS_0(\pi)$. Our description of the simulation is not self-contained, and depends upon [RS94] (specifically the proof of Lemma 5.1.1). In terms of the description of Σ in [RS94], the only change we are making is to set $k = \log |S|$, a constant, instead of having k increase with the maximum indegree of G . This effectively eliminates the transmission code χ .

By [Sch96, Lemma 1], there exists some ternary tree code² \mathcal{T} of infinite depth, distance parameter $\frac{1}{2}$, and alphabet size 287. This tree code will be used to encode strings over the alphabet $\{0, 1, \mathbf{bkp}\}$, and the tree code characters, which can be represented by bitstrings of length 9, will be sent over the channels. We will refer collectively to the 9 rounds needed to send a single tree code character as one *step*.

In each step, a party P_i begins by tree-decoding all the characters she’s received so far from all of her in-neighbors, yielding, for each in-neighbor P_j , an *estimated unparsed incoming transcript* $\hat{y}_{ji} \in \{0, 1, \mathbf{bkp}\}^*$. Each estimate \hat{y}_{ji} is parsed into a *estimated parsed incoming transcript* $\hat{w}_{ji} \in \{0, 1\}^*$ by processing from left to right, interpreting each \mathbf{bkp} symbol as an instruction to delete the previous symbol. Similarly, for each out-neighbor P_j , P_i recalls the string $y_{ij} \in \{0, 1, \mathbf{bkp}\}^*$ that she has transmitted to P_j , and parses this into a *parsed outgoing transcript*

²See [Sch96] for the definition of a tree code.

$w_{ij} \in \{0, 1\}^*$. The *parsed transcript* at P_i at this moment is the collection of all these estimated parsed incoming transcripts and parsed outgoing transcripts.

We say that the parsed transcript at P_i is *consistent* if for every time τ and every out-neighbor P_j of P_i , the τ th bit in the parsed outgoing transcript w_{ij} is the bit specified by π to be sent to P_j , given the length- $(\tau - 1)$ prefixes of all the incoming transcripts. If the parsed transcript at P_i is consistent, she transmits whatever bits are specified by π (encoded using \mathcal{T} .) Otherwise, she transmits `bkp` to all of her out-neighbors. We run this simulation for T_2 steps, where T_2 is a parameter which will be chosen later.

A.2 Analysis of $RS_0(\pi)$ We recall some terminology from [RS94]. We say that an *edge character error* occurs on the edge (P_i, P_j) in step $\tau + 1$ if, in step τ , P_i sends some tree symbol, but P_j receives a different tree symbol. We say that an *edge tree error* occurs on (P_i, P_j) in step $\tau + 1$ if, in step $\tau + 1$, P_j 's estimated, unparsed transcript \hat{y}_{ij} differs from the true unparsed transcript y_{ij} . We simply say that a *tree error* occurs at P_j in step $\tau + 1$ if, for some in-neighbor P_i , an edge tree error occurs on (P_i, P_j) in step $\tau + 1$. We say that (P_i, τ) and (P_j, τ') are *time-like* if there is a path from P_i to P_j of length no more than $\tau' - \tau$. For edges e, e' , we say that (e, τ) and (e', τ') are *time-like* if there is some walk which begins with e , ends with e' , and has length no more than $\tau' - \tau + 1$. For example, (e, τ) and (e, τ') are time-like for any $\tau' \geq \tau$; if $e = (P_i, P_j)$ and $e' = (P_j, P_k)$, then (e, τ) and $(e', \tau + 1)$ are time-like. Finally, we say that (e, τ) and (P_i, τ') are time-like if there is a walk which begins with e , ends at P_i , and has length no more than $\tau' - \tau + 1$. For example, if $e = (P_i, P_j)$, then (e, τ) and (P_j, τ) are time-like. A *time-like sequence* is a sequence where each pair of successive elements is a time-like pair. The *time history cone* of (P_i, τ) is the set of all (P_j, τ') such that (P_j, τ') and (P_i, τ) are time-like, unioned with the set of all (e, τ') such that (e, τ') and (P_i, τ) are time-like.

For a party P_i and a time τ , $RP(P_i, \tau)$ is the number of rounds t such that the first t rounds in all of P_i 's parsed outgoing transcripts (at the end of step τ) match what she would send if all the parties followed π on a noiseless network. As in [RS94], our goal is to show that if the number of errors is sufficiently small, then $RP(P_i, \tau)$ will be close to τ for every P_i and every τ .³

³This will not immediately mean that the simulation is successful, since the criterion for success is that the parties give the correct outputs, which depends on their estimated incoming transcripts, not their outgoing transcripts. We deal with this small technicality in Section A.3.

Intuition. A key part of the intuition in [RS94] is that if two errors have a space-like separation, then they should cause no more delay than if only one of them occurred. The analysis in [RS94] establishes a quantitative version of this idea with regard to tree errors. Specifically, it is shown [RS94, Lemma 5.1.1] that if $RP(P_i, \tau) = \tau - \ell$, then there is a time-like sequence of at least $\ell/2$ tree errors in the time history cone of P_i at τ . But the intuition still applies if we look at the underlying edge tree errors, instead of just tree errors.

For example, suppose G is a star graph, with all edges but one directed inward; say there are q edges pointing inward. Consider the adversarial strategy of dividing the rounds of the simulation into q equal segments, and spending the i th segment zeroing out the bits sent across the i th inward-facing edge. As tree errors (rather than edge tree errors), this is a time-like sequence of errors, simply because they all have the same recipient. Thus, the analysis in [RS94] does not show that the simulation will succeed in the face of this adversary. However, if we pay attention to the underlying edge tree errors, we see that these errors do not have a time-like separation; signals sent along one incoming edge can never affect signals sent along another incoming edge. This suggests that for sufficiently large q , the simulation will succeed (for all round complexities T), since the longest sequence of time-like edge tree errors is only of length about T/q .

And indeed, this suggestion is easily seen to be true. After the central party recovers from the edge tree errors on the first couple of incoming edges, she is sufficiently far behind the other incoming parties in the simulation that further edge errors do not affect her; by the time a symbol actually affects the central party's transmissions, it has been "cleaned up" by the tree code mechanism, so she makes no further mistakes.

For a party P_i and a time τ_0 , we define $Y(P_i, \tau_0)$ to be the maximum length of any time-like sequence of edge character errors in the time history cone of P_i at τ_0 . (Compare $Y(P_i, \tau)$ to the quantity $X(P_i, \tau)$ analyzed in [RS94].) As in [RS94], we let $B(P_i, \tau)$ denote the number of times that P_i has transmitted `bkp` up to step τ , and we let $AT(P_i, \tau) = \tau - 2B(P_i, \tau)$, so that $AT(P_i, \tau)$ is the length of every outgoing transcript of P_i 's at time τ . Most of our effort will go toward proving the following proposition, analogous to [RS94, Proposition 5.2.1].

PROPOSITION 3. *For any party P_i and any time τ ,*

$$(A.1) \quad \tau \leq RP(P_i, \tau) + 24Y(P_i, \tau) + B(P_i, \tau).$$

Toward proving Proposition 3, we make the following definitions.

DEFINITION A.1. For an edge $(P_i, P_j) \in E$ and an alleged (parsed) transcript $z_{ij} \in \{0, 1\}^*$, we define the *accuracy* of z_{ij} to be the length of the longest prefix of z_{ij} which is a prefix of the sequence of bits that P_i would send P_j on a noiseless network.

DEFINITION A.2. The *action* of P_j in step $\tau + 1$ is defined as follows.⁴

- Suppose $RP(P_j, \tau + 1) > RP(P_j, \tau)$. Then the action of P_j in step $\tau + 1$ is **progress**.
- Suppose $RP(P_j, \tau + 1) = RP(P_j, \tau)$, and P_j backs up in step $\tau + 1$. Then the action of P_j in step $\tau + 1$ is **justified backup**.
- Suppose $RP(P_j, \tau + 1) < RP(P_j, \tau)$, or $RP(P_j, \tau + 1) = RP(P_j, \tau)$ and P_j transmits data in step $\tau + 1$. Say that an in-neighbor P_i of P_j is *accuracy minimizing* if the accuracy of the estimated, parsed transcript \hat{w}_{ij} is minimized at P_i among in-neighbors of P_j .
 - If, for every accuracy minimizing P_i , the accuracy of the true transcript w_{ij} is strictly more than the accuracy of the estimated transcript \hat{w}_{ij} , we say that the action of P_j in step $\tau + 1$ is **harmful tree error**.
 - Otherwise, we say that the action of P_j in step $\tau + 1$ is **harmful propagated error**.

Lemmas 20, 21, and 22 are fairly technical, and are motivated only by the fact that they will be useful for proving Proposition 3.

LEMMA 20. *Suppose that in step $\tau_0 + 1$, the action of P_j is either harmful tree error or harmful propagated error. Let P_i be accuracy minimizing, and say that the accuracy of \hat{w}_{ij} is $k - 1$. Then if P_j transmitted data in step $\tau_0 + 1$, then $RP(P_j, \tau_0) \geq k$, while if P_j backed up in step $\tau_0 + 1$, then $RP(P_j, \tau_0) \geq k + 1$.*

Proof. First, suppose P_j transmitted data in step $\tau_0 + 1$. Then P_j did not “regret” any of her transmissions, i.e. her parsed transcript was consistent. Thus, the length- k prefixes of her outgoing parsed transcripts must be correct, since they match the accurate length- $(k - 1)$ prefixes of her estimated incoming parsed transcripts. Next, suppose P_j backed up in step $\tau_0 + 1$. From the action, we must have $RP(P_j, \tau_0) = AT(P_j, \tau_0)$. Furthermore, since P_j “regrets” a transmission which is, in fact, correct, we must have $k \leq AT(P_j, \tau_0) - 1$. Therefore, $RP(P_j, \tau_0) \geq k + 1$ as claimed. \square

⁴This notion is analogous to, but not the same as, the function ACTION in [RS94].

Recall that the *magnitude* of an edge tree error on (P_i, P_j) is the length of the suffix of the affected estimated unparsed transcript \hat{y}_{ij} which begins with the first symbol which differs from the corresponding symbol of the true unparsed transcript y_{ij} .

LEMMA 21. *Suppose Equation A.1 holds for every party P_i and every $\tau \leq \tau_0$. Suppose that in step $\tau_0 + 1$, there is an edge tree error of magnitude $M > 0$ on (P_i, P_j) . Suppose that $RP(P_i, \tau_0) \leq RP(P_j, \tau_0) + 2M$, and $B(P_i, \tau_0) \leq B(P_j, \tau_0) + M$. Then Equation A.1 holds for P_j and $\tau = \tau_0 + 1$.*

Proof. Since RP and B change at most one each round, we have

$$\begin{aligned} \text{(A.2)} \quad RP(P_i, \tau_0 + 1 - M) &\leq RP(P_i, \tau_0) + M - 1 \\ \text{(A.3)} \quad &\leq RP(P_j, \tau_0) + 3M - 1 \\ \text{(A.4)} \quad &\leq RP(P_j, \tau_0 + 1) + 3M, \end{aligned}$$

and similarly

$$\text{(A.5)} \quad B(P_i, \tau_0 + 1 - M) \leq B(P_j, \tau_0 + 1) + 2M.$$

From the tree code condition, we can be sure that in the M steps preceding and including step $\tau_0 + 1$, the character error rate on (P_i, P_j) was at least $\frac{1}{4}$. Therefore, we have

$$\text{(A.6)} \quad Y(P_i, \tau_0 + 1 - M) \leq Y(P_j, \tau_0 + 1) - \frac{1}{4}M.$$

Applying Equation A.1 to P_i at $\tau = \tau_0 + 1 - M$ and using Equations A.4, A.5, and A.6, we have

$$\begin{aligned} \text{(A.7)} \quad \tau_0 + 1 - M &\leq RP(P_i, \tau_0 + 1 - M) \\ \text{(A.8)} \quad &+ 24Y(P_i, \tau_0 + 1 - M) \\ \text{(A.9)} \quad &+ B(P_i, \tau_0 + 1 - M) \\ \text{(A.10)} \quad &\leq RP(P_j, \tau_0 + 1) + 3M \\ \text{(A.11)} \quad &+ 24Y(P_j, \tau_0 + 1) - 6M \\ \text{(A.12)} \quad &+ B(P_j, \tau_0 + 1) + 2M \\ \text{(A.13)} \quad &= RP(P_j, \tau_0 + 1) \\ \text{(A.14)} \quad &+ 24Y(P_j, \tau_0 + 1) \\ \text{(A.15)} \quad &+ B(P_j, \tau_0 + 1) - M, \end{aligned}$$

which completes the proof. \square

The following lemma is proven in exactly the same way as an analogous statement in [RS94].

LEMMA 22. *Suppose Equation A.1 holds for all P_i and all $\tau \leq \tau_0$. Suppose that $RP(P_j, \tau_0) = AT(P_j, \tau_0)$, and for some in-neighbor P_i , $B(P_i, \tau_0) > B(P_j, \tau_0)$. Then Equation A.1 holds for P_j and $\tau = \tau_0 + 1$.*

Proof. By hypothesis, $RP(P_j, \tau_0) = \tau_0 - 2B(P_j, \tau_0)$. We also have $RP(P_i, \tau_0) \leq \tau_0 - 2B(P_i, \tau_0)$. Subtracting gives $RP(P_j, \tau_0) + 2B(P_j, \tau_0) \geq RP(P_i, \tau_0) + 2B(P_i, \tau_0)$. Using our assumption $B(P_i, \tau_0) \geq B(P_j, \tau_0) + 1$, this implies

$$(A.16) \quad RP(P_j, \tau_0) + B(P_j, \tau_0) \geq RP(P_i, \tau_0)$$

$$(A.17) \quad + B(P_i, \tau_0) + 1.$$

Now, observe that $RP(P_j, \tau_0 + 1) + B(P_j, \tau_0 + 1) \geq RP(P_j, \tau_0) + B(P_j, \tau_0)$, because the B term can only increase in step $\tau_0 + 1$, while the RP term can only decrease by at most 1 and in that case the B term increased. Therefore, we have

$$(A.18) \quad RP(P_i, \tau_0) + B(P_i, \tau_0) < RP(P_j, \tau_0 + 1)$$

$$(A.19) \quad + B(P_j, \tau_0 + 1).$$

Of course, $Y(P_i, \tau_0) \leq Y(P_j, \tau_0 + 1)$, so an application of Equation A.1 at P_i at $\tau = \tau_0$ completes the proof. \square

Proof. [of Proposition 3] We proceed by induction on τ . At $\tau = 0$, all terms of Equation A.1 are zero. For the inductive step, assume that Equation A.1 holds for all $\tau < \tau_0$; we will prove that it holds for $\tau = \tau_0 + 1$. Fix some party P_i . If the action of P_i in step $\tau + 1$ is **progress**, then $RP(P_i, \tau_0 + 1) = RP(P_i, \tau_0) + 1$, and the B and Y terms do not decrease from τ_0 to $\tau_0 + 1$, so we are done. Similarly, if the action of P_i is **justified backup**, then $B(P_i, \tau_0 + 1) = B(P_i, \tau_0) + 1$, and the RP and Y terms do not decrease. The final two cases, **harmful tree error** and **harmful propagated error**, are treated in Lemmas 23 and 24 below.

LEMMA 23. *Suppose Equation A.1 holds for all P_i and all $\tau \leq \tau_0$. Suppose that in step $\tau_0 + 1$, the action of P_j is harmful tree error. Then Equation A.1 holds for P_j and $\tau = \tau_0 + 1$.*

Proof. Let P_i be an accuracy minimizing in-neighbor of P_j , and say that the accuracy of \hat{w}_{ij} is $k - 1$. As the name of the action indicates, because the k th symbol of the true transcript w_{ij} is correct and present while the k th symbol of the estimated transcript \hat{w}_{ij} is not, there must have been an edge tree error on (P_i, P_j) . Say that the tree error was of magnitude M . Then w_{ij} and \hat{w}_{ij} agree in their first $(|w_{ij}| - 2M)$ positions. In particular, $|w_{ij}| \leq k + 2M - 1 < k + 2M$.

By Lemma 20, $RP(P_j, \tau_0) \geq k$. Therefore, $|w_{ij}| \leq RP(P_j, \tau_0) + 2M$. Of course, $|w_{ij}| = AT(P_i, \tau_0) \geq RP(P_i, \tau_0)$, so

$$(A.20) \quad RP(P_i, \tau_0) \leq RP(P_j, \tau_0) + 2M.$$

Again because w_{ij} and \hat{w}_{ij} agree in their first $(|w_{ij}| - 2M)$ positions, $|w_{ij}| \geq |\hat{w}_{ij}| - 2M$.

First, suppose that P_j transmitted data in step $\tau_0 + 1$. Then $|\hat{w}_{ij}| \geq AT(P_j, \tau_0)$, so $AT(P_i, \tau_0) \geq AT(P_j, \tau_0) - 2M$, which implies that

$$(A.21) \quad B(P_i, \tau_0) \leq B(P_j, \tau_0) + M.$$

An application of Lemma 21 completes the proof in this case. Next, suppose that P_j backed up in step $\tau_0 + 1$. If $B(P_i, \tau_0) \leq B(P_j, \tau_0)$, then once again, Lemma 21 completes the proof. But if $B(P_i, \tau_0) > B(P_j, \tau_0)$, then Lemma 22 completes the proof, since the action implies that $AT(P_j, \tau_0) = RP(P_j, \tau_0)$. \square

LEMMA 24. *Suppose Equation A.1 holds for every party P_i and every $\tau \leq \tau_0$. Suppose that in step $\tau_0 + 1$, the action of P_j is harmful propagated error. Then Equation A.1 holds for P_j and $\tau = \tau_0 + 1$.*

Proof. Let P_i be an accuracy minimizing in-neighbor such that the accuracy of w_{ij} is no more than the accuracy of \hat{w}_{ij} , and say that the accuracy of \hat{w}_{ij} is $k - 1$. Then $RP(P_i, \tau_0) < k$. We claim that

$$(A.22) \quad RP(P_i, \tau_0) \leq RP(P_j, \tau_0 + 1) - 1.$$

To see why, first suppose P_j transmitted data in step $\tau_0 + 1$; then $RP(P_j, \tau_0 + 1) = RP(P_j, \tau_0)$, and by Lemma 20, $RP(P_j, \tau_0) \geq k$, which completes the proof of Equation A.22. Next, suppose P_j backed up in step $\tau_0 + 1$; then $RP(P_j, \tau_0 + 1) = RP(P_j, \tau_0) - 1$, and by Lemma 20, $RP(P_j, \tau_0) \geq k + 1$, so $RP(P_j, \tau_0 + 1) \geq k$, again completing the proof of Equation A.22.

Now, for a first case, suppose $B(P_i, \tau_0) \leq B(P_j, \tau_0 + 1)$. Then applying Equation A.1 to P_i at $\tau = \tau_0$ gives

$$(A.23) \quad \tau_0 \leq RP(P_i, \tau_0) + 24Y(P_i, \tau_0) + B(P_i, \tau_0)$$

$$(A.24) \quad \leq RP(P_j, \tau_0 + 1) - 1 + 24Y(P_j, \tau_0 + 1)$$

$$(A.25) \quad + B(P_j, \tau_0 + 1),$$

completing the proof. For the second case, suppose $B(P_i, \tau_0) = B(P_j, \tau_0 + 1) + \ell$, with $\ell > 0$.

- For the first subcase, suppose that P_j transmitted data in step $\tau_0 + 1$; then $B(P_i, \tau_0) = B(P_j, \tau_0) + \ell$, so there was an edge tree error of magnitude $M \geq \ell$ on edge (P_i, P_j) in step $\tau_0 + 1$. Therefore, we can apply Lemma 21 to complete the proof, since $RP(P_i, \tau_0) \leq RP(P_j, \tau_0 + 1) - 1$.
- Finally, for the second subcase, suppose that P_j backed up in step $\tau_0 + 1$. Then $RP(P_j, \tau_0) = AT(P_j, \tau_0)$ and $B(P_i, \tau_0) > B(P_j, \tau_0 + 1) > B(P_j, \tau_0)$, so we can apply Lemma 22.

This completes the proofs of Lemma 24 and Proposition 3. \square

The following lemma follows easily from Proposition 3; it is analogous to [RS94, Lemma 5.1.1].

LEMMA 25. *Suppose $RP(P_i, \tau) = \tau - \ell$. Then there is some time-like sequence of $\frac{\ell}{48}$ edge character errors in the time history cone of P_i at τ .*

Proof. Doubling Equation A.1 and rearranging gives

$$\begin{aligned} RP(P_i, \tau) - \tau + 48Y(P_i, \tau) &\geq \tau - RP(P_i, \tau) \\ &\quad - 2B(P_i, \tau) \\ &= AT(P_i, \tau) \\ &\quad - RP(P_i, \tau). \end{aligned}$$

The right-hand side is nonnegative, so $Y(P_i, \tau) \geq \frac{\ell}{48}$. \square

A.3 The final RS compiler All the work we have done to analyze RS_0 has focused on RP as a measure of progress, but RP is not directly related to our success criterion, which is that the parties give the correct outputs at the end of the simulation, or equivalently that the parties are able to correctly guess the bits they would have *received* if the parties had followed the original protocol on a noiseless network. To address this technicality, before applying RS_0 , we will modify the protocol so that any bits that the parties receive are immediately retransmitted over dummy channels to dummy parties. The resulting compiler is the RS compiler, which we denote RS.

We describe $RS(\pi)$ for arbitrary deterministic protocols π . Let $\bar{V} = V \cup \{P_{n+i} : 1 \leq i \leq n\}$ be two disjoint copies of V , where P_{n+i} is a copy of P_i . Let $\bar{E} = E \cup \{(P_i, P_{n+j}) : (P_j, P_i) \in E\}$. Let $\bar{G} = (\bar{V}, \bar{E})$. From the T -round protocol π on G , we define a $(T+1)$ -round protocol $\bar{\pi}$ on \bar{G} as follows. The inputs for $\bar{\pi}$ are exactly the same as the inputs for π . When $P_i \in V$ receives an input x_i , she uses the “ordinary” edges in G to do what π instructs her. (In the extra round at the end, she just sends a zero on every ordinary edge.) On each “dummy” edge (P_i, P_{n+j}) , in round τ , she transmits the bit that she received in round $\tau - 1$ on the corresponding ordinary edge (P_j, P_i) . (In the first round, she just sends a zero on every dummy edge.)

The protocol $RS(\pi)$, which runs on the original graph G , is essentially $RS_0(\bar{\pi})$, with $T_2 = 2T + 1$. Technically, $RS_0(\bar{\pi})$ runs on \bar{G} ; naturally, the parties in $RS(\pi)$ do not literally send bits across the dummy edges, but they keep track of which bits they would have sent across the dummy edges, and on the ordinary edges, they behave exactly as in $RS_0(\bar{\pi})$. At the end of the simulation, P_i gives as output whatever π instructs her to give as output, under the assumption that her parsed outgoing transcripts accurately reflect the (incoming and outgoing) transcripts that would have

occurred if the parties had followed π on a noiseless network.

We can now prove Lemma 14 and Proposition 2, from which Proposition 1 immediately follows.

Proof. [of Lemma 14] The round complexity of $RS(\pi^*[G, T])$ is $9T_2 \leq 27T$. If the execution fails, then there must be some party P_i such that $RP(P_i, 2T + 1) < T + 1$. Therefore, by Lemma 25, there was some time-like sequence of $\frac{T}{48}$ edge character errors. Each character error is associated with at least one bit error. \square

Proof. [of Proposition 2] Again, if the execution of $RS(\pi^*)$ fails, there was some time-like sequence of $\frac{T}{48}$ edge character errors. Time-like character errors must occur in distinct steps. So we can take $\eta = \frac{1}{48} \cdot \frac{1}{27} = \frac{1}{1296}$. \square

B Proof of Lemma 5

Without loss of generality, assume that precisely λ units of each commodity flow in F . For each (P_i, P_j) , randomly form a path p_{ij} from P_i to P_j as follows. Initially, p_{ij} is just P_i ; in each step, extend p_{ij} by randomly selecting an outgoing edge, with the probability of selecting an edge being proportional to the amount of flow of commodity (P_i, P_j) which goes across that edge. Repeat until the path reaches P_j . Let \mathcal{P} be the set of paths p_{ij} formed in this way.

Fix some edge $e \in E$. For each $(P_i, P_j) \in E$, the probability that $e \in p_{ij}$ is equal to the flow of commodity (P_i, P_j) across e in F divided by λ , and these are independent events. Therefore, the expected value of the congestion c_e of e is exactly equal to f_e/λ , where f_e is the total flow across e in F , and by the Chernoff bound, for any $\varepsilon > 0$,

$$\begin{aligned} \Pr \left[c_e \geq (1 + \varepsilon) \frac{f_e}{\lambda} \right] &\leq \exp \left(-\frac{\varepsilon^2}{2 + \varepsilon} \frac{f_e}{\lambda} \right) \\ &\leq \exp \left(-\frac{\varepsilon^2}{(2 + \varepsilon)^2} \cdot (1 + \varepsilon) \frac{f_e}{\lambda} \right). \end{aligned}$$

If we define ε so that $(1 + \varepsilon) \frac{f_e}{\lambda} = 9 \left(\frac{1}{\lambda} + \ln m \right)$, then certainly $\varepsilon > 1$ simply because $f_e \leq 1$, and hence $\frac{\varepsilon^2}{(2 + \varepsilon)^2} > \frac{1}{9}$. Therefore,

$$\begin{aligned} \Pr \left[c_e \geq 9 \left(\frac{1}{\lambda} + \ln m \right) \right] &< \exp \left[-\frac{1}{9} \cdot 9 \left(\frac{1}{\lambda} + \ln m \right) \right] \\ &= \frac{1}{m} e^{-1/\lambda}. \end{aligned}$$

Taking a union bound over all m edges e , we see that with positive probability, the total congestion of \mathcal{P} is no more than $9 \left(\frac{1}{\lambda} + \ln m \right)$. \square