Noisy Radio Network Lower Bounds
Via Noiseless Beeping Lower Bounds

Klim Efremenko[*]   Gillat Kol[†]   Dmitry Paramonov[‡]

Ben-Gurion University   Princeton University   Princeton University

Raghuvansh R. Saxena[§]

Microsoft

Abstract

Much of today's communication is carried out over large wireless systems with different input-output behaviors. In this work, we compare the power of central abstractions of wireless communication through the general notion of *boolean symmetric f-channels*: In every round of the f-channel, each of its n parties decides to either broadcast or not, and the channel outputs $f(m)$, where m is the number of broadcasting parties.

Our first result is that the well studied *beeping channel*, where f is the threshold-1 function, is not stronger than *any* other f-channel. To this end, we design a protocol over the f-channel and prove that any protocol that simulates it over the beeping channel blows up the round complexity by a factor of $\Omega(\log n)$. Our lower bound technique may be of independent interest, as it essentially generalizes the popular fooling set technique by exploiting a "local" relaxation of combinatorial rectangles.

Curiously, while this result shows the limitations of a *noiseless* channel, namely, the beeping channel, we are able to use it to show the limitations of the *noisy* version of many other channels. This includes the extensively studied *single-hop radio network model with collisions-as-silence* (CAS), which is equivalent to the f-channel with $f(m) = 1$ iff $m = 1$.

In particular, our second and main result, obtained from the first, shows that converting CAS protocols to *noise resilient* ones may incur a large performance overhead, *i.e.*, no constant rate *interactive code* exists. To this end, we design a CAS protocol and prove that any protocol that simulates it over the noisy CAS model with correlated stochastic noise, blows up the round complexity by a factor of $\Omega(\log n)$. We mention that the $\Omega(\log n)$ overhead in both our results is tight.

[*]klimefrem@gmail.com
[†]gillat.kol@gmail.com
[‡]dp20@cs.princeton.edu
[§]raghuvansh.saxena@gmail.com

1   Introduction

As wireless systems have become massively popular, theoretical models for such systems have become the topic of numerous works. In this paper we study the relative power of central models abstracting broadcast communication. To this end, we use the notion of *symmetric boolean-valued f-channels*.

For a function $f : \mathbb{N} \cup \{0\} \to \{0, 1\}$, the f-channel is a synchronous channel that, in every round, allows any number of parties to broadcast a bit. The round ends with all participants receiving the bit $f(\mathsf{wt}(x))$, where x_i is the bit broadcast by party i, and $\mathsf{wt}(x)$ is the Hamming weight of $x \in \{0,1\}^*$. It is useful to think of parties with $x_i = 1$ as broadcasting, and of parties with $x_i = 0$ as silent. The output of the channel is then a function of the number of broadcasting parties.

Given two such channels f and g, we say that the f-channel is *at least as strong as* (or *stronger than*) the g-channel if any protocol over the g-channel can be simulated by a protocol over the f-channel with only a constant multiplicative blowup in the number of rounds. The f-channel is *strictly stronger* than the g-channel if it is stronger, but not vice-versa. The two channels are *equivalent* if each is at least as strong as the other. We note that two channels may be incomparable, meaning that neither is stronger than the other.

Of special interest to us are binary versions of two well-studied channels, the *single-hop radio networks* model and the *beeping* model:

The BEEPING model.   In an n-party protocol over the binary beeping model [CK10], denoted here BEEPING, the parties interact in synchronous rounds. In every round, each party can decide to either beep (emit a signal) or not to beep (stay silent). If at least one party beeps, all parties hear a beep, otherwise, all parties hear silence[1]. The BEEPING model received a lot of attention in recent years, largely as it captures the simplest possible communication primitive, a detectable burst of 'energy', making it very well suited for describing certain wireless networks as well as signal-driven biological systems[2]. It is easy to see that the BEEPING model is essentially the f-channel for $f(m) = \mathbb{1}[m \geq 1]$ (the THRESHOLD[1] function).

The CAS model.   In a round of an n-party protocol over the binary *collision-as-silence* single-hop radio networks model [CK85], each party can choose to either broadcasts a bit or stay silent. In rounds where exactly one party broadcasts, his bit is received by all parties. Otherwise (if all parties were silent or if at least two parties broadcast), a special $\perp$ symbol

---

[1] In the original definition of BEEPING by [CK10], the parties are woken up by an adversary. In this paper we use a relaxed definition where all parties are woken up at the beginning of the protocol. Since we show impossibility results, this relaxation of the model only strengthens our results.

[2] For example, cells communicating by secreting proteins and other chemical markers that are diffused and sensed by neighboring cells, or fireflies reacting to flashes of light from nearby fireflies. Also see, *e.g.*, [AAB+11, NB15].

is received by all[3]. While the above description of the channel does not fit our definition of an f-channel, as the channel's inputs and output are non-binary, a simple time-sharing argument shows that it is equivalent to the f-channel for $f(m) = \mathbb{1}[m = 1]$, and we denote this channel CAS for short[4].

The CAS vs. BEEPING problem. It is easy to see that CAS is at least as strong as BEEPING, as a BEEPING round can be simulated by a CAS round: Parties broadcast in the CAS round if they broadcast in the BEEPING round, but there is an additional "dummy" party that broadcasts with them.[5] Observe that the output of the CAS round is 1 iff the dummy is the only one to broadcast, which happens iff the value of the BEEPING round is 0. What about the other direction?

Problem 1.1. *Is* CAS *strictly* stronger than *BEEPING?*

The noisy CAS problem. So far in our discussion of the CAS model, we made the strong assumption that the parties utilizing the channel receive the correct output bit in every round. We now switch gears and consider the noisy version of this channel, where this assumption is relaxed. For a function f and $\epsilon > 0$, the parties utilizing the (correlated) ϵ-*noisy* f-*channel*, received the correct output bit $f(\mathsf{wt}(x))$ with probability $1 - \epsilon$, and with probability ϵ they receive the opposite bit. It is easy to see that CAS is at least as strong as noisy CAS. A natural question is again about the reverse direction:

Problem 1.2. *Is* CAS *strictly* stronger than *noisy* CAS*?*

1.1 Our Results

We settle both of the above problems in the affirmative. While seemingly very different, we show that Problem 1.1 and Problem 1.2 are actually tightly connected, and exploit this connection.

---

[3]The collision-as-silence model is, perhaps, the most common single-hop radio networks model in the literature. Another very popular model is the *collision detection* model, where collision and silence are perceived as different symbols. Our results are stated for the collision-as-silence model, but apply to the collision detection model as well.

[4]To see the equivalence, simulate a round of the original non-binary channel with three CAS rounds: In the first CAS round, parties broadcast (*i.e.*, their x_i is 1) if they broadcast a 0 or a 1 in the original round. In the second round, parties broadcast only if they broadcast 0 in the original round. In the third round, parties broadcast only if they broadcast 1 in the original round. A simulation in the other direction is also possible.

[5]Even without an extra dummy party, a BEEPING round can be simulated by constantly many CAS rounds.

1.1.1   No f-channel is Weaker than BEEPING

Theorem 1.3 (see formal statement in Theorem 4.1) shows that no non-trivial f-channel is weaker than BEEPING.[6] A positive answer for Problem 1.1 is obtained from Theorem 1.3 as a special case.

Theorem 1.3 (Informal). *Let $f : \mathbb{N} \cup \{0\} \to \{0, 1\}$ be a function such that f is not constant on $\mathbb{N}$. Then, for all $n > 0$, there exists an n-party protocol Π over the f-channel, such that any protocol Π' that computes the same function as Π over the BEEPING model has $\Omega(|\Pi| \log n)$ communication rounds.*

Here, $|\Pi|$ denotes the worst-case number of rounds of Π. We mention that the $\Omega(\log n)$ blowup in the number of rounds suggested by Theorem 1.3 is *tight*, as for many f-channels (*e.g.*, CAS), $\mathcal{O}(\log n)$ BEEPING rounds suffice in order to compute f (see Section 1.1.3)[7]. We also mention that the condition that f is not constant on $\mathbb{N}$ cannot be replaced by the condition that f is a non-constant function. The reason is that, as remarked before, the BEEPING channel is the THRESHOLD[1]-channel, but THRESHOLD[1] is not a constant function (note that it is constant on $\mathbb{N}$).

Technique.   Proving Theorem 1.3 requires proving a lower bound in the BEEPING model. Our lower bound proof is inspired by the classical *fooling set* technique for proving *deterministic, two-party* communication complexity lower bounds. We mention though that the fooling set technique completely breaks when applied to the beeping model, as this technique crucially assumes that, at any point in the execution of the protocol, the set of possible inputs constitutes a *combinatorial rectangle*. While this is not the case for the BEEPING model, we show that a "local" version of this method can be made to work for this model (and even handle randomized protocols[8]), in a sense made precise in Section 2. We believe that our new technique will find other applications. A detailed overview of our technique can be found in Section 2.

1.1.2   Noisy CAS is Strictly Weaker than CAS

While Theorem 1.3 implies that CAS is strictly stronger than BEEPING, we observe that BEEPING is at least as strong as noisy CAS (we sketch this argument in Section 1.1.3 and prove it in Theorem 5.2). A direct corollary of the above is the following Theorem 1.4 (see formal statement in Theorem 5.1) that shows that CAS is strictly stronger than noisy CAS, answering Problem 1.2 (which was the original motivation for this paper) in the positive.

---

[6]We note that, as the f-channel's output is only a function of the number of broadcasting parties (and is oblivious to the number of silent parties), some channels, like the AND channel, cannot be represented as f-channels.

[7]There are, however, specific f-channels that require a substantially greater overhead when simulated by BEEPING. For example, in [BKOT19], it is shown that the parity function, PARITY$(m) = m$ mod 2, requires $\Omega(n^{1/6})$ BEEPING rounds. Clearly, PARITY requires only 1 round over the PARITY-channel.

[8]See [MSY16] for another (unrelated) randomized lower bound using the fooling set technique.

Theorem 1.4 (Main, Informal). *Let $\epsilon \geq 0$. For all $n > 0$, there exists an n-party protocol Π over the CAS channel, such that any n-party protocol Π' that computes the same function as Π over the ϵ-noisy CAS channel has $\Omega(|\Pi| \log n)$ rounds.*

We mention that understanding whether CAS protocols can be converted to noise resilient ones with small overhead, was the original motivation for this work. As is typically the case for (classical or interactive) error correcting codes, a CAS protocol Π can be made resilient by simply repeating every round $\mathcal{O}(\log n)$ times and taking majority (at least as long as Π is not too long, say, $|\Pi| \leq \text{poly}(n)$). This shows that the $\Omega(\log n)$ round blowup in Theorem 1.4 is *tight*.

We mention that our argument showing that BEEPING is at least as strong as noisy CAS[9], generalizes to other noisy f-channels (*e.g.*, to all noisy THRESHOLD$[k]$-channels with $k \geq 2$). For such fs, a separation of the f-channel and the noisy f-channel, analogue to the one in Theorem 1.4, is also implied by Theorem 1.3. We remark however that there may be functions f that satisfy the condition of Theorem 1.3 for which the noisy f-channel is strictly stronger than BEEPING and the analogue of Theorem 1.4 may not hold[10]. Characterizing the set of functions f for which the analogue of Theorem 1.4 holds is an intriguing problem.

1.1.3   BEEPING is At Least As Strong as Noisy CAS

We next sketch an easy argument showing that a noisy CAS round can be simulated by constantly many BEEPING rounds, showing that Theorem 1.4 can be obtained as a corollary of Theorem 1.3: Randomly partition the parties into two roughly equal sets $[n] = S_1 \cup S_2$. For $i \in \{1, 2\}$, in the ith BEEPING round, the only beeping parties are parties in S_i that broadcast in the noisy CAS round. Now, if no party broadcasts in the noisy CAS round, then no beep is heard in either of the BEEPING rounds. If exactly one party broadcasts in the noisy CAS round, then a beep is heard in exactly one of the two BEEPING rounds. Else, if two or more parties broadcast in the noisy CAS round, then with probability at least $1/2$ a beep is heard in both BEEPING rounds (as the probability that all broadcasting parties beep in the same round is at most $1/2$).

By simply repeating the above simulation, we can simulate an ϵ-noisy CAS round using $\mathcal{O}(\log(1/\epsilon))$ BEEPING rounds. Additionally, by repeating the above simulation in a "binary search"-like manner, we can also simulate one round of noiseless CAS ($\epsilon = 0$) using $\mathcal{O}(\log n)$ BEEPING rounds[11]. This implies that the $\Omega(\log n)$ blowup in the number of rounds in Theorem 1.3 is tight for the CAS channel. We mention that similar simulation protocols can be obtained for channels other than CAS (*e.g.*, THRESHOLD channels).

---

[9]We mention that CAS is equivalent to the THRESHOLD$[2]$-channel (see Theorem 5.3).

[10]A good candidate for such a function is the PARITY function. Owing to the existence of good linear error correcting codes, the noisy PARITY-channel may be as strong as the PARITY-channel.

[11]In more detail, if exactly one of the first two BEEPING rounds beeps, say, round $i \in \{1, 2\}$, we wish to know if at least two of the parties in S_i broadcast in the CAS round. To do that, we repeat the simulation with only the parties in S_i participating.

1.2 Related Work

1.2.1 Interactive Coding

Interactive error correcting codes encode interactive communication protocols designed to work over noiseless channels to protocols that also work over noisy channels. The study of interactive codes was initiated by a seminal paper of Schulman [Sch92] that considered two-party protocols, and these were also studied by many follow-up works. Interactive codes for multi-party distributed channels received quite a bit of attention over the last few years. These include codes for *peer-to-peer* channels [RS94, JKL15, HS16, ABE+16, BEGH16, GK19, GKR19] and codes for various *wireless* channels [CHHZ17, EKS18, CHHHZ19, EKS19, EKS20, AGL20, EKPS21a, EKPS21b].

We next compare Theorem 1.4 to the most relevant prior work on interactive coding over wireless channels.

Relation to [EKS18] (interactive coding over CAS). [EKS18] shows that any "simple" protocol Π over CAS can be compiled to a protocol Π' that works over ϵ-noisy CAS, while only incurring a constant multiplicative overhead in the number of rounds. Here, "simple" means that the protocol is *non-adaptive* (*a.k.a, oblivious* or *static*). That is, in every round of Π exactly one party broadcasts and, moreover, the identity of this party is fixed in advance (thus, it is independent of the received transcript, channel noise, and inputs)[12].

As adaptivity can hugely boost the power of wireless channels and is widely used, in this paper we revisit the [EKS18] result for general (possibly adaptive) protocols. Theorem 1.4 proves that the simulation scheme of [EKS18] cannot be extended to work for general protocols.

We point out that the [EKS18] protocol works for both the correlated noise model, like the one assumed by this paper, and the *independent noise model*, where each party gets a noisy bit with probability ϵ, independently from the others. The correlated and independent noise models abstract different phenomena – the correlated noise model describes the situation where there is global interference (*e.g.*, global network problems due to weather, contaminated environment, *etc.*), while independent noise is better suited to describe situations where the noise source is local. We also mention that the interactive coding problems over these two noise models are incomparable: On the one hand, with correlated noise all parties have the same received transcript, but on the other hand, if a symbol is corrupted by noise, none of the parties receive the correct symbol and can warn the others. Whether or not an impossibility result similar to our Theorem 1.4 can be proved for independent noise is a great open problem.

---

[12]Note that while Π is assumed to be non-adaptive, the simulation protocol Π' constructed by [EKS18] is adaptive. However, Π' also works in the model where if it is not the case that exactly one party broadcasts, an adversary is controlling the received symbols.

Relation to [EKS20] (interactive coding over BEEPING). [EKS20] studies the beeping channel under correlated noise and shows that an $\Omega(\log n)$ overhead is required to convert a protocol to a noise resilient one. In other words, it shows that Theorem 1.4 also holds for BEEPING vs. ϵ-noisy BEEPING. To this end, [EKS20] shows a lower bound in the *noisy* BEEPING model. In contrast, in this paper we prove a lower bound for the *noiseless* BEEPING model, which is a harder task.

We note however, that our work does not reproduce the main result of [EKS20], as the protocol that we show is costly over the noiseless BEEPING channel is not known to be more expensive over the noisy BEEPING channel. Rather, we view the result of [EKS20] as complementing ours: Recall that BEEPING is the THRESHOLD[1]-channel, and that, as mention in Section 1.1.2, our proof shows that Theorem 1.4 holds for every THRESHOLD[k] channel for $k \geq 2$. Thus, putting these results together, we get an interactive coding impossibility result for all THRESHOLD[k]-channels.

We remark that while CAS is at least as strong as BEEPING, the interactive coding questions for these models are incomparable. The reason is that the additional power of CAS can be used by both the noisy and noiseless CAS channels. More generally, given two communication models, even if the first is at least as strong as the second, the interactive coding questions for these models are incomparable.

Finally, we mention the result of [HMP20], that also studies the beeping model under correlated noise and shows that the Consensus problem can be made noise resilient. Note however, that in their model, the noise may only corrupt beeps to be received as silence, but not vice-versa.

Relation to [EKS19] and [AGL20] (interactive coding over general networks). [EKS19] and [AGL20] study noisy wireless *multi-hop networks* (general graph topologies as opposed to the single-hop clique topology). [EKS19] gives an impossibility result for interactive coding over CAS with independent noise, and [AGL20] (see also [CMT17]) has an impossibility result for interactive coding over BEEPING with independent noise (although their main result is an upper bound). While lower bounds are harder to prove for the single-hop networks than for arbitrary topology of our choice, our result is incomparable to [EKS19], as the noise model is different. Characterizing the set of topologies for which Theorem 1.4 holds is an interesting problem.

Theorem 1.3 gives a lower bound for the *noiseless* BEEPING channel. We next compare it to the related result of [BKOT19]:

Relation to [BKOT19] (noiseless BEEPING lower bounds). [BKOT19] uses circuit lower bound techniques to show that protocols over the BEEPING model (and other related models) must have at least $\Omega(n^{\alpha})$ rounds, for a constant $\alpha > 0$, to compute certain boolean functions (*e.g.*, the PARITY function). This result is incomparable to Theorem 1.3. As CAS is the $\mathbb{1}[m=1]$-channel and since [BKOT19] does not give lower bounds for the $\mathbb{1}[m=1]$

function, it cannot be used to derive Theorem 1.4. Furthermore, the technique of [BKOT19] is not "fine enough" to prove Theorem 1.3 for CAS (recall from Section 1.1.3 that the $\Omega(\log n)$ overhead in Theorem 1.3 is tight for CAS).

1.2.2 Broadcast Lower Bounds

Our work is also related to prior work proving lower bounds for (noiseless) broadcast channels.

Lower bounds for noiseless network channels. In this work we consider single-hop topologies (in particular, the topology and the number of parties is known to everyone). We also allow the parties to have unique ids, and assume that the parties are always available to carry out the protocol (*e.g.*, no wake-ups or Byzantine failures). Since this setting is easier from an algorithm design perspective, our (noiseless) lower bounds carry over to the harder settings for which prior work in distributed computing showed lower bounds (see, *e.g.*, [CMS01, New14]).

Multiple access channels. We mention that our definition of f-channels is a special case of *multiple access channels* in information theory (for a great survey see [GGLR05]). A general multiple access channel allows each party $i \in [n]$ to communicate a symbol from the input alphabet $\mathcal{I}_i$. The output of the channel is a symbol from a, possibly different, output alphabet $\mathcal{O}$, that is computed from the n communicated symbols via a probabilistic function. Our (noiseless) f-channels have $\mathcal{I}_1 = \cdots = \mathcal{I}_n = \mathcal{O} = \{0, 1\}$ and the function for computing the output is deterministic and only depends on the number of 1 bits sent by the parties.

We also mention that there are known connections between coding over multiple access channels and *counterfeit coin problems* (*a.k.a.*, *weighing problems*) [Can64, SS63, ER63, Lin64, Lin65, Lin71, Lin75, Bsh12, MK13, KKP22, to cite a few]. In the basic counterfeit coin problem, we are given n coins, some of which are heavy (defective), while the remaining are light (legal). We know the weight of both the legal and the defective coins and have access to a spring scale that can be used to weigh any subset of the coins. It can be shown that any weighing scheme for the counterfeit coin problem can be converted to a signature coding scheme for the multiple access adder channel, see, *e.g.*, Section 7.1 in [GGLR05].

2 Overview

We now provide a detailed overview of the proof of Theorem 1.3. For this, we have to show that the BEEPING channel is strictly weaker than the f-channel, for any function $f : \mathbb{N} \cup \{0\} \to \{0, 1\}$ that is not constant on $\mathbb{N}$. As f is not constant on $\mathbb{N}$, there exist $a < b \in \mathbb{N}$ such that $f(a) \neq f(b)$. Using these, we define the following problem that can be solved easily over the f-channel and show later that solving it over the BEEPING channel takes many more rounds.

The hard function. We divide the n players into k sets of n/k players each. For concreteness, the reader may assume $n = k^2$ although we shall not heavily rely on this relation and only use the fact that k is some polynomial in n. Our hard function is the (a, b)-weight problem where each of the n players has an input bit and the goal is to find out for each group j, whether group j has exactly a players with input 1 (in which case the output for this group should be 0) or exactly b players with input 1 (in which case the output for this group should be 1). The parties can output anything for a group where the number of players with input 1 is different from a and b.

Note that there exists a k round protocol for the (a, b)-weight problem over the f-channel: Simply go over all $j \in [k]$ and have people in group j broadcast in round j. To finish the proof, we are going to argue an $\Omega(k \cdot \log(n/k))$ lower bound for any protocol solving this problem over the BEEPING channel. For the purposes of this overview, we restrict attention to the case $a = 1$ and $b = 2$ as a similar proof works for other values of a and b.

2.1 Fooling Sets

Consider a deterministic two-party communication protocol. A transcript π of the protocol is the concatenation of the messages sent by the parties in all rounds of the protocol. We say that a transcript π is realized on the input (x, y), if, when Alice has the input x and Bob has the input y, the transcript of the protocol is π. It is well known that the set of inputs on which π is realized is a *combinatorial rectangle*: If π is realized on (x, y) and on (x', y'), then it is also realized on (x', y) and (x, y'). If the protocol has a short output, it can be assumed that its output is a function of its transcript. It follows that if the protocol correctly computes a function f with a short output, and $f(x, y) = f(x', y')$, then either the transcript realized on input (x, y) is different from that realized on input (x', y'), or we also have $f(x, y) = f(x', y) = f(x, y')$.

This simple fact can be used to get lower bounds on the length of deterministic communication protocols computing a function as follows: Consider for example a protocol for computing the equality function $f(x, y) = \mathbb{1}[x = y]$. By definition, we have $f(x, x) = f(x', x')$ for all x, x' but $f(x, x) \neq f(x, x')$ for all $x \neq x'$. Thus, from our conclusion in the previous paragraph, it must be the case that the transcript realized when the inputs for the parties are (x, x) is different from that realized when the inputs are (x', x') for all $x \neq x'$. This means that there must be many different transcripts, implying the protocol must be long.

The above technique for proving lower bounds on the length of deterministic protocols computing a function f is called the *fooling set technique*. It works by proving that f has a large *fooling set*, where a fooling set for f is a subset of the inputs with the same f value, such that for every two different inputs (x, y) and (x', y') in the set, at least one of (x', y) and (x, y') has a different f value. In the above example for the equality function, the set of all inputs of the form (x, x) forms a fooling set.

Even though we presented the technique for two party communication protocols, the same arguments can also be extended to multi-party protocols with a fixed order of speaking. In

this context, if there are n parties, we get that if a protocol correctly computes a function f, and $f(x_1, x_2, \cdots, x_n) = f(x_1', x_2', \cdots, x_n')$, then either the transcript realized when the inputs to the parties are $(x_1, \cdots, x_n)$ is different from that realized when the inputs are $(x_1', \cdots, x_n')$, or we have $f(x_1, \cdots, x_n) = f(x_1'', \cdots, x_n'')$ for all $i \in [n]$ and $x_i'' \in \{x_i, x_i'\}$. We shall call such inputs x'' *hybrids*.

The reason these arguments work for the above channels is that for all these channels, the set of inputs that lead to any given transcript is a combinatorial rectangle (like in the case of the two-party channel). However, for the BEEPING channel this is not the case. For example, if we consider a one round protocol where the parties simply beep the first bit of their input, then the set of inputs that lead to the transcript 0 is a combinatorial rectangle, but the set of inputs that lead to the transcript 1 is the complement of this rectangle, and may not itself be a rectangle. Thus, fooling set arguments in general do not work for the BEEPING channel.

Despite the fact that these fooling set arguments do not hold for the BEEPING channel, we shall assume for now that they do (that is, that it suffices to demonstrate a large fooling set), and present our lower bound under this assumption. We additionally mention that these arguments, and also the lower bound for the equality function above, hold only for deterministic protocols. Later, we shall extend them to randomized protocols.

2.2 Lower Bounds Assuming Fooling Sets

In this subsection, we present a proof of our lower bound using *fooling sets*. Even though arguments based on fooling sets do not work for the BEEPING channel, the ideas present will be instructive and will form an important part of the proof.

2.2.1 Lower Bound Against Deterministic Protocols

We now present a fooling set for the $(1, 2)$-weight problem. Assuming the arguments above, this implies a lower bound on the length of a protocol over the BEEPING channel that solve the $(1, 2)$-weight problem. Our fooling set has all the inputs $(x_1, \cdots, x_n)$ that have exactly 1 player with the input 1 in each of the k groups. To see why any two such inputs must have a different transcript (equivalently, why these inputs form a fooling set), note that $f(x_1, \cdots, x_n)$ is the same for all such inputs and also, for any two distinct inputs $(x_1, \cdots, x_n)$ and $(x_1', \cdots, x_n')$, there must exist one group (out of the k groups) such that these two inputs, when restricted to this group, are different.

Without loss of generality, let this group be the first group (the group of players $[k]$), and, as the other cases are similar, assume that:

$$(x_1, \cdots, x_k) = (1, \underbrace{0, 0, \cdots, 0}_{k-1 \text{ times}}) \qquad (x_1', \cdots, x_k') = (0, 1, \underbrace{0, \cdots, 0}_{k-2 \text{ times}}). \qquad (1)$$

Defining:
$$(x_1'', \cdots, x_k'') = (1, 1, \underbrace{0, \cdots, 0}_{k-2 \text{ times}}), \tag{2}$$

we see that it is a hybrid as described in the previous section but the correct output on the hybrid input $(x_1'', \cdots, x_n'')$ is different from that on $(x_1, \cdots, x_n)$. This finishes the proof that any two such inputs must have different transcripts. Thus, the number $\left(\frac{n}{k}\right)^k$ of such inputs is also a lower bound on the number of distinct transcripts. As these many transcripts need a protocol of length at least $k \cdot \log \frac{n}{k}$, we are done.

2.2.2 Lower Bound Against Randomized Protocols

We now describe how to extend the lower bound to work against general randomized protocols while continuing our assumption that fooling sets based arguments work for the BEEPING channel. For this, we first use Yao's minimax theorem to get that it is enough to define a "hard" distribution $\mathcal{D}$ over the inputs for the $(1, 2)$-weight problem such that any *deterministic* protocol for this problem fails on a large fraction of inputs from $\mathcal{D}$.

The hard distribution $\mathcal{D}$.   We work with the following hard distribution: First, for each group $j \in [k]$, we sample a uniformly and independently random bit to decide whether this group will have one or two players with input 1. Then, based on the output of this bit, we uniformly and independently sample a subset of players in this group of size 1 or 2 and give them the input 1. All the remaining players in the group get the input 0.

The proof.   Recall that we need to show that any deterministic protocol is incorrect on a set of inputs that has a high probability according to $\mathcal{D}$. Also recall that, if any two inputs in a fooling set give rise to the same transcript, then the protocol is incorrect on at least one of the possible hybrids formed by those inputs. Thus, if we can show that, unless a protocol has $\Omega(k \cdot \log(n/k))$ rounds, the number of inputs in a fooling set (to be specified later) that give rise to the same transcript is so large that the probability of incorrect hybrids generated by these inputs is a constant, we will have our randomized lower bound. We show this next in the following two steps:

1. **Showing that the number of incorrect hybrids is large:** Fix a short, correct protocol and let π be a typical transcript. Define the set X^{all} to be the set of all inputs for which the output induced by π is correct, regardless of whether or not they realize π. Also, define $X^{\mathsf{true}} \subseteq X^{\mathsf{all}}$ to be the set of all inputs in X^{all} that also realize the transcript π. Observe that all inputs in X^{all} are equally likely under our hard distribution $\mathcal{D}$.

   Next, observe that as π is a typical transcript of a correct protocol that has length $o(k \cdot \log(n/k))$, X^{true} has at least a $\left(\frac{n}{k}\right)^{-o(k)}$ fraction of the inputs in X^{all} and the output of π says that at least $\Omega(k)$ of the groups have only one player with the input 1. Let

11

$\mathsf{Ones} \subseteq [k]$ be this set of groups. As for all $j \in \mathsf{Ones}$, there are at most $\frac{n}{k}$ choices for the inputs of players in group j in X^{all}, the above conditions imply that for at least $\Omega(k)$ of the groups in Ones, the number of possible inputs for the players in this group amongst all the inputs in X^{true} is at least $M = \left(\frac{n}{k}\right)^{1-o(1)}$.

For any such input $x \in X^{\mathsf{true}}$ and any such group, one can take any of the at least $M-1$ possible choices of the inputs of the players in this group that are different from x on this group, and define a hybrid as in Eqs. (1) and (2),[13] that has the same transcript but for which the output is incorrect. Moreover, as Eq. (2) has exactly two players with input 1, it can be formed in exactly two ways. Conclude that the number of incorrect hybrids is at least $\Omega(k \cdot M \cdot |X^{\mathsf{true}}|)$.

2. **Showing that the probability of incorrect hybrids is large:** We now use the lower bound on the number of incorrect hybrids to get a lower bound on the probability of incorrect hybrids. For this, note from the definition of the distribution $\mathcal{D}$ that the probability of an input x is proportional to $\left(\mathcal{O}\left(\frac{n}{k}\right)\right)^{|\mathsf{Ones}(x)|}$, where $\mathsf{Ones}(x)$ is the set of groups that have exactly one player with the input 1 in x. As the number of such groups is one smaller in each hybrid, we get that the probability of incorrect hybrids relative to the probability of correct inputs is at least $\Omega\left(\frac{kM}{n}\right) \gg 1$ by our choice of k and M. Thus, the protocol errs with high probability.

2.3   Removing the Fooling Set Assumption

The only problem with the argument above is that the fooling set argument does not work for the BEEPING channel, *i.e.*, it is not true that if $(x_1, \cdots, x_n)$ and $(x'_1, \cdots, x'_n)$ give rise to the same transcript, then any hybrid input $(x''_1, \cdots, x''_n)$, where $x''_i \in \{x_i, x'_i\}$ for all $i \in [n]$ will also give rise to the same transcript. However, note from the foregoing subsection that we never actually use the full power of the fooling set argument, restricting attention only to hybrids that differ in the inputs of exactly one of the n players from one of the original inputs (Eqs. (1) and (2)). Specifically, each hybrid only differs on one coordinate from the input $x \in X^{\mathsf{true}}$ it was obtained from.

Such a strong condition on the types of hybrids that we use in our argument allow us to use the following trick to remove the fooling set assumption: Recall that the BEEPING channel, in every round, computes and sends the logical OR of the bits sent by the players in this round. As the hybrids we work with differ in exactly one player, say player i, from one of the original inputs, the only way this hybrid can lead to a different transcript is if there exists a round such that player i is the only player beeping (broadcasting a 1) in that round.

---

[13]In the hybrid, the inputs of the players that are not in the chosen group are the same as their inputs in x. As for players in the chosen group, the player that has a 1 input in x has a 1 input in the hybrid, but there is an additional player with a 1 input.

This means that, unless the protocol has more than $\Omega(k \cdot \log(n/k))$ rounds, there are at most $\mathcal{O}(k \cdot \log(n/k))$ players i such that changing the input of player i while taking a hybrid can affect the transcript. Thus, we can simply remove these "bad" players from the $\mathcal{O}(k \cdot M)$ players we take hybrids over and make the argument work.

Geometrical interpretation. We finish by also explaining this weaker form of the fooling set arguments in terms of combinatorial rectangles. Recall that the general fooling set argument works when the set of inputs X that lead to any given transcript is a combinatorial rectangle. An equivalent way to express the fact that X is a combinatorial rectangle is that for any $x = (x_1, \cdots, x_n) \in X$ and $i \in [n]$, the single-dimensional projection of X on to coordinate i is equal to the intersection of X with the line $\forall j \neq i : X_j = x_j$.[14] What our weaker fooling set argument shows is that for any $x \in X$, this holds for most of the coordinates i. Precisely, we show that it holds for all but $\mathcal{O}(k \cdot \log(n/k))$ values of i.

Put another way, even if we do not get combinatorial rectangles exactly, we do get a "local" version of them, that allows us to take hybrids as long as we do not change a lot of coordinates, and do not change coordinates corresponding to "bad" players. Our fooling set argument is strong enough to work with this weaker guarantee.

3 Models

Notation. We use $\mathbb{1}[E]$ to denote the indicator random variable for the event E. For a string s, we use $|s|$ to denote the length of s. For $i \in [|s|]$, let s_i denote the ith coordinate of s and $s_{<i}, s_{\leq i}$ denote the prefix of the first $i-1$ and i coordinates of s, respectively. For $x \in \{0,1\}^*$, we denote the Hamming weight of x by $\mathsf{wt}(x) = |\{i \in [|x|] : x_i = 1\}|$.

f-channels. The *symmetric boolean-valued channel f with noise rate ϵ* (ϵ-*noisy f-channel*, for short) is specified by a constant $\epsilon \geq 0$ and a function $f : \mathbb{N} \cup \{0\} \to \{0,1\}$. Let $n \in \mathbb{N}$. A (deterministic) *n-party protocol* over the ϵ-noisy f-channel is given by a tuple

$$\Pi = \left(T, \{\mathcal{X}^i\}_{i \in [n]}, \mathcal{Y}, \{g_t^i\}_{i \in [n], t \in [T]}, \mathsf{out} \right).$$

Here, $T = |\Pi|$ is the number of rounds (or the *length*) of the protocol, $\mathcal{X}^i$ is the input space for player i, $\mathcal{Y}$ is the output space of the protocol, $g_t^i : \mathcal{X}^i \times \{0,1\}^{t-1} \to \{0,1\}$ is the function player i uses to determine what message to broadcast in round t, and $\mathsf{out} : \{0,1\}^T \to \mathcal{Y}$ is the function used to determine the output from the received transcript. As usual, we define a *randomized protocol* to be a distribution over (deterministic) protocols.

---

[14]More precisely, if $y = \left(\underbrace{0, \cdots, 0}_{i-1 \text{ times}}, y_i, \underbrace{0, \cdots, 0}_{n-i \text{ times}} \right)$ is in the former set, then $y' = (x_1, \cdots, x_{i-1}, y_i, x_{i+1}, \cdots, x_n)$ is in the latter set.

Execution of a protocol. The protocol Π starts with all players $i \in [n]$ having an input $x^i \in \mathcal{X}^i$ and proceeds in T rounds, maintaining the invariant that before round t, for all $t \in [T]$, all players know a *transcript* $\pi_{<t} \in \{0,1\}^{t-1}$. In round t, player i broadcasts the bit $b_t^i = g_t^i(x^i, \pi_{<t})$. Then, in round t, if b_t denotes the string $b_t = b_t^1 \cdots b_t^n$, *all* the players receive the bit π_t which equals $f(\mathsf{wt}(b_t))$ with probability $1 - \epsilon_t$, and equals $1 - f(\mathsf{wt}(b_t))$ with probability ϵ_t, where $\epsilon_t \leq \epsilon$ is adversarially chosen. All players append π_t to $\pi_{<t}$ to get a transcript $\pi_{\leq t}$ and continue the execution of the protocol. After T rounds, all players output $\Pi(x) = \mathsf{out}(\pi_{\leq T}) \in \mathcal{Y}$.

Noiseless channels. Let $f : \mathbb{N} \cup \{0\} \to \{0,1\}$. The f-channel is the ϵ-noisy f-channel for $\epsilon = 0$. Let Π be a deterministic n-party protocol over the f-channel as above. Observe that the transcript π of the protocol Π is a deterministic function of the input x, and denote it by $\mathsf{trans}_\Pi(x)$. Denote $\mathsf{trans}_{\Pi,t}(x) = (\mathsf{trans}_\Pi(x))_t$, $\mathsf{trans}_{\Pi,\leq t}(x) = (\mathsf{trans}_\Pi(x))_{\leq t}$, and $\mathsf{trans}_{\Pi,<t}(x) = (\mathsf{trans}_\Pi(x))_{<t}$.

Let $F : \mathcal{X}^1 \times \cdots \times \mathcal{X}^n \to \mathcal{Y}$ be a (partial) function and $x \in \mathcal{X}^1 \times \cdots \times \mathcal{X}^n$. We say that Π is *correct* on x with respect to F if $\Pi(x) = F(x)$. Let $\pi \in \{0,1\}^T$ be a candidate transcript for Π. We say that π is *correct* on x with respect to F if $\mathsf{out}(\pi) = F(x)$. When F is clear from the context, we sometimes simply say that Π (or π) is correct on x. Finally, for $p > 0$ we say that Π *solves/computes* F with success probability p, if for all x such that $F(x)$ is well defined, it holds that $\Pi(x) = F(x)$ with probability at least p regardless of the choices of ϵ_t made by the adversary (in case Π is over an ϵ-noisy channel).

Special channels. We define the following channels: For $\epsilon \geq 0$ and $k \geq 0$,

1. ϵ-noisy $\mathrm{THRESHOLD}[k]$ is the ϵ-noisy f-channel for $f(m) = \mathbb{1}[m \geq k]$. In addition, $\mathrm{THRESHOLD}[k]$ is ϵ-noisy $\mathrm{THRESHOLD}[k]$ with $\epsilon = 0$.

2. $\mathrm{BEEPING} = \mathrm{THRESHOLD}[1]$.

3. ϵ-noisy CAS is the ϵ-noisy f-channel for $f(m) = \mathbb{1}[m = 1]$. In addition, CAS is ϵ-noisy CAS with $\epsilon = 0$.

For the $\mathrm{BEEPING}$ channel, we will sometimes use special terminology. Let Π be an n-party protocol over $\mathrm{BEEPING}$. For $t \in [|\Pi|]$, denote by $\mathsf{beep}_{\Pi,t}(x) = \{i \in [n] : b_t^i = 1\}$ the set of all players who *beep* (*i.e.*, broadcast a 1) in round t. Note that $\mathsf{trans}_{\Pi,t}(x) = 1$ if and only if $\mathsf{beep}_{\Pi,t}(x) \neq \emptyset$.

Channel equivalence. Let $f, g : \mathbb{N} \cup \{0\} \to \{0,1\}$. We say that the f-channel can *simulate* the g-channel if for every (deterministic) n-party protocol Π over the g-channel, there is a (deterministic) n-party protocol Π' over the f-channel with $|\Pi'| = \mathcal{O}(|\Pi|)$ and a function h, such that for every input x for Π, $h(\mathsf{trans}_{\Pi'}(x)) = \mathsf{trans}_\Pi(x)$. We say that the f-channel and the g-channel are *equivalent* if the f-channel can simulate the g-channel and vice-versa.

4   BEEPING Lower Bound

In this section we prove Theorem 1.3. Fix a function f satisfying the condition of Theorem 1.3, and let $0 < a < b \in \mathbb{N}$ be such $f(a) \neq f(b)$. We prove that the following problem is hard for BEEPING but easy for the f-channel.

In the (a, b)-*weight problem with n players and k groups*, n parties, each holding a private input bit, are partitioned into k sets of n/k parties each. The goal is for the first party to output a string $v \in \{0, 1\}^k$ such that if exactly a parties in group $j \in [k]$ have input 1 then $v_j = 0$, and if exactly b parties in group i have input 1 then $v_j = 1$.

Observe that the (a, b)-weight problem can be computed by a k-round protocol over the f-channel, by having parties from group j holding a 1 input broadcasting in round j. Thus, the following Theorem 4.1 will suffice in order to prove Theorem 1.3. The rest of this section is devoted to proving Theorem 4.1.

Theorem 4.1. *Let $b > a \geq 1$ and $k > (100b)^{(100b)^{100}}$ be integers. Let $n = k(k+a-1)$. Then, every (randomized) protocol over BEEPING that solves the (a, b)-weight problem with n players and k groups with success probability greater than $1/n^{1/300}$, requires at least $\Omega_{a,b}(k \log(n/k))$ rounds.*

We will next prove a restricted version of Theorem 4.1, where $a = 1$:

Theorem 4.2. *Let $b > 1$ and $k > (100b)^{(100b)^{100}}$ be integers. Let $n = k^2$. Then, every (randomized) protocol over BEEPING that solves the $(1, b)$-weight problem with n players and k groups with success probability greater than $1/n^{1/250}$, requires at least $\Omega_b(k \log(n/k))$ rounds.*

Equipped with Theorem 4.2, we are able to prove Theorem 4.1:

Proof of Theorem 4.1. Suppose for contradiction that there existed some protocol Π over the BEEPING model which solves the (a, b)-weight problem with k groups and $k + a - 1$ parties in each group, with success probability greater than $1/n^{1/300}$ running in $o_{a,b}(k \log(n/k))$.

Let $c = b - a + 1$. We construct a protocol Π' for the $(1, c)$-weight problem where there are k groups with k players in each group (k^2 parties in total). We will first add $k \cdot (a-1)$ "dummy" parties, $a - 1$ dummy parties in each group, such that all dummy parties are assumed to have the input 1. To run Π', the k^2 parties will run Π with player 1 also simulating all the extra $k \cdot (a - 1)$ dummy parties. Note that player 1 can indeed simulate all these extra parties: Player 1 can compute the bit that each such dummy party would have sent (as each player hears the same transcript and player 1 knows the inputs of the dummy parties), and will compute and send the logical OR of all these bits and the bit that he himself wishes to send.

If a group has only 1 player with a 1 input before the dummy players were added, the dummy players ensure that this group now has a players with a 1 input. Likewise, if there were c players with a 1 input in a group, the dummy players ensure that the group now has b

such players. Thus, if Π outputs the correct answer for the inputs with the dummy players, then Π' will output the correct answer.

However, this leads to a contradiction: The protocol Π' only runs Π, which, by assumption takes $o_{a,b}(k \log(n/k))$ rounds of communication, and it gets the correct answer for the $(1,c)$-weight problem on any input with probability greater than $1/n^{1/300}$. This success probability is greater than $1/(n')^{1/250}$, where $n' = n - k(a-1)$ is the number of parties participating in Π', contradicting Theorem 4.2. □

4.1 Proof of Hardness of the $(1,b)$-weight problem

The rest of this section is devoted to proving Theorem 4.2. Fix b, k, n that satisfy the conditions of Theorem 4.2.

We view an input for the $(1,b)$-weight problem with n players and k groups as a string $x \in \{0,1\}^n$, such that party i gets the input bit x_i. Let $x \in \{0,1\}^n$ and $j \in [k]$. We denote by $x_{|j} = x_{(j-1) \cdot n/k+1}, \ldots, x_{j \cdot n/k} \in \{0,1\}^{n/k}$ the inputs of the parties in group j. Let $X' \subseteq \{0,1\}^n$ and $j \in [k]$. We denote by $X'_{|j}$ the projection of X' to group j. Formally, $X'_{|j} = \{x_{|j} \mid x \in X'\}$.

The hard distribution $\mathcal{D}$. By Yao's minimax theorem, to prove Theorem 4.2 it suffices to define a hard distribution $\mathcal{D}$ such that any deterministic protocol is correct with probability at most $1/n^{1/250}$ when the inputs x to the parties are drawn from $\mathcal{D}$.

We next define the hard distribution $\mathcal{D}$ over the inputs $x \in \{0,1\}^n$ for the $(1,b)$-weight problem with n players and k groups. For every $j \in [k]$ independently: with probability $1/2$ sample $x_{|j}$ uniformly under the constraint that $\mathsf{wt}(x_{|j}) = 1$, and with probability $1/2$ sample $x_{|j}$ uniformly under the constraint that $\mathsf{wt}(x_{|j}) = b$.

Fixing the protocol Π. Define $\epsilon = \frac{1}{100(b-1)}$. For the rest of the text, fix a deterministic protocol Π with at most $T = \epsilon^{10} k \log(n/k) = \mathcal{O}_b(k \log(n/k))$ rounds. We assume without loss of generality that the protocol Π has exactly T rounds. Recall that we require the first party to give the output. We will also assume that the output of Π can be determined from the transcript and does not require knowing the input of the first party. This is without loss of generality, as the first party can beep its input using one round of communication. We will often omit the subscript Π, $e.g.$, we will write $\mathsf{beep}(x,t)$ to mean $\mathsf{beep}_\Pi(x,t)$.

Protocol notation. Let $x \in \{0,1\}^n$ and let π be a transcript for Π. Define

$$\mathsf{Ones}(x) = \{j \in [k] \mid \mathsf{wt}(x_{|j}) = 1\},$$
$$\mathsf{Ones}(\pi) = \{j \in [k] \mid (\mathsf{out}(\pi))_j = 0\}.$$

Additionally, let $X(\pi) = \{x \in \mathsf{supp}(\mathcal{D}) \mid \mathsf{trans}(x) = \pi\}$ be the set of inputs in $\mathsf{supp}(\mathcal{D})$ that generate π. Let $X^{\mathsf{true}}(\pi)$ be the set of all $x \in X(\pi)$ such that the output induced by π

is correct for input x, *i.e.*, $\mathsf{Ones}(x) = \mathsf{Ones}(\pi)$.

Bad sets. Let $x \in \{0,1\}^n$. Define $B(x)$ be the set of $i \in [n]$ such that there exists $t \in [T]$ satisfying $i \in \mathsf{beep}(x,t)$ and $|\mathsf{beep}(x,t)| < b$. Also define, for $j \in [k]$, the set $B_j(x)$ be the set of players in $B(x)$ that are also in group j and:

$$X_j^{\mathsf{Bad}}(x) = \{x' \in \{0,1\}^n \mid (x'_{|j} = x_{|j}) \vee (\exists i \in B_j(x) : x'_i = 1)\}.$$

Finally, define:

$$J^{\mathsf{Bad}}(x) = \{j \in [k] \mid |B_j(x)| > \epsilon^5 \log(n/k)\}.$$

Hybrids. Let $x, x^1, \ldots, x^{b-1} \in \{0,1\}^n$ and $j \in [k]$. Define $\mathsf{hyb}_j(x, x^1, \ldots, x^{b-1})$ to be a vector $x^* \in \{0,1\}^n$ such that for every $j' \in [k] \setminus \{j\}$ it holds that $x^*_{|j'} = x_{|j'}$ and $x^*_{|j} = \mathsf{OR}(x_{|j}, x^1_{|j}, \ldots, x^{b-1}_{|j})$, where OR stands for the bit-wise OR (*i.e.*, coordinate i of $x^*_{|j}$ is the logical OR of the ith coordinates of $x_{|j}, x^1_{|j}, \ldots, x^{b-1}_{|j}$). Observe that it is always the case that $\mathsf{hyb}_j(x, x^1, \ldots, x^{b-1}) \geq x$, where for $x, y \in \{0,1\}^n$, we write $y \geq x$ if for all $i \in [n]$ it holds that $y_i \geq x_i$. .

4.2 Key Lemma

The following lemma will be key to Theorem 4.2.

Lemma 4.3. *Let π be a transcript such that $\mathcal{D}(X^{\mathsf{true}}(\pi)) \geq 2^{-10T}$ and $|\mathsf{Ones}(\pi)| \geq k/5$. We have:*

$$\mathcal{D}(X^{\mathsf{true}}(\pi)) \leq \frac{\mathcal{D}(X(\pi))}{n^{1/200}}.$$

We now present our proof of Theorem 4.2 assuming the key lemma, which shall be proved in Section 4.3.

Proof of Theorem 4.2. Let $\mathsf{Trans} = \{0,1\}^T$ be the set of all transcripts π for Π. Define:

$$\mathsf{Trans}_1 = \{\pi \in \mathsf{Trans} \mid \mathcal{D}(X^{\mathsf{true}}(\pi)) < 2^{-10T}\},$$
$$\mathsf{Trans}_2 = \{\pi \in \mathsf{Trans} \mid |\mathsf{Ones}(\pi)| < k/5\},$$
$$\mathsf{Trans}_3 = \mathsf{Trans} \setminus (\mathsf{Trans}_1 \cup \mathsf{Trans}_2).$$

The success probability of Π is

$$\sum_{\pi \in \mathsf{Trans}} \mathcal{D}(X^{\mathsf{true}}(\pi)) \leq \sum_{j \in [3]} \sum_{\pi \in \mathsf{Trans}_j} \mathcal{D}(X^{\mathsf{true}}(\pi)). \tag{3}$$

We bound each of the three terms on the right hand side one by one. For $j = 1$, we have by

17

definition of Trans_1 that:

$$\sum_{\pi \in \mathsf{Trans}_j} \mathcal{D}(X^{\mathsf{true}}(\pi)) \leq \sum_{\pi \in \mathsf{Trans}_j} 2^{-10T} \leq |\mathsf{Trans}| \cdot 2^{-10T} \leq 2^{-9T}. \tag{4}$$

For $j = 2$, note that any $\pi \in \mathsf{Trans}_2$ satisfies $|\mathsf{Ones}(\pi)| < k/5$. We group the π according to the value of $\mathsf{Ones}(\pi)$ and use the fact that the number of possible values for $\mathsf{Ones}(\pi)$ is at most $\sum_{0 \leq \ell < k/5} \binom{k}{\ell} \leq (5e)^{k/5} \leq 1.7^k$ to get:

$$\begin{aligned}
\sum_{\pi \in \mathsf{Trans}_2} \mathcal{D}(X^{\mathsf{true}}(\pi)) &= \sum_{O \subseteq [k]} \sum_{\substack{\pi \in \mathsf{Trans}_2 \\ \mathsf{Ones}(\pi) = O}} \mathcal{D}(X^{\mathsf{true}}(\pi)) \\
&\leq 1.7^k \cdot \max_{O \subseteq [k]} \sum_{\substack{\pi \in \mathsf{Trans}_2 \\ \mathsf{Ones}(\pi) = O}} \mathcal{D}(X^{\mathsf{true}}(\pi)) \\
&\leq 1.7^k \cdot \max_{O \subseteq [k]} \sum_{\substack{\pi \in \mathsf{Trans}_2 \\ \mathsf{Ones}(\pi) = O}} \sum_{\substack{x : \mathsf{trans}(x) = \pi \\ \mathsf{Ones}(x) = \mathsf{Ones}(\pi)}} \mathcal{D}(x) \qquad \text{(Definition of } X^{\mathsf{true}}(\pi)) \\
&\leq 1.7^k \cdot \max_{O \subseteq [k]} \sum_{x : \mathsf{Ones}(x) = O} \mathcal{D}(x) \\
&\leq 1.7^k \cdot 2^{-k} \qquad \text{(Definition of } X^{\mathsf{true}}(\pi)) \\
&\leq 0.9^k.
\end{aligned}$$

Finally, for $j = 3$, we use Lemma 4.3, we have:

$$\sum_{\pi \in \mathsf{Trans}_3} \mathcal{D}(X^{\mathsf{true}}(\pi)) \leq \frac{1}{n^{1/200}} \cdot \sum_{\pi \in \mathsf{Trans}_3} \mathcal{D}(X(\pi)) \leq \frac{1}{n^{1/200}}.$$

Summing up, we can continue Eq. (3) as;

$$\sum_{\pi \in \mathsf{Trans}} \mathcal{D}(X^{\mathsf{true}}(\pi)) \leq 2^{-9T} + 0.9^k + \frac{1}{n^{1/200}} \leq \frac{1}{n^{1/250}}.$$

$\square$

4.3 Proof of Key Lemma

This section is devoted to proving Lemma 4.3. Fix a transcript π that satisfies the conditions of Lemma 4.3. We will often omit π from the notation, $e.g.$, we'll write X to mean $X(\pi)$ and Ones to mean $\mathsf{Ones}(\pi)$. Let $x \in X^{\mathsf{true}}$ and $j \in \mathsf{Ones}$. Define:

$$H_{x,j} = \left\{ \mathsf{hyb}_j(x, x^1, \ldots, x^{b-1}) \mid x^1, \ldots, x^{b-1} \in X^{\mathsf{true}} \setminus X_j^{\mathsf{Bad}}(x) \text{ and } x_{|j}^1, \ldots, x_{|j}^{b-1} \text{ all distinct} \right\}. \tag{5}$$

Lemma 4.4. *For all $x \in X^{\text{true}}$ and $j \in \text{Ones}$, we have $H_{x,j} \subseteq X$. Additionally, for all $x^* \in H_{x,j}$, we have $\text{Ones}(x^*) = \text{Ones} \setminus \{j\}$. As a corollary, we get $H_{x,j} \subseteq X \setminus X^{\text{true}}$.*

Proof. We fix an arbitrary $x^* \in H_{x,j}$ and show that $x^* \in X$ and $\text{Ones}(x^*) = \text{Ones} \setminus \{j\}$. Let $x^1, \ldots, x^{b-1} \in X^{\text{true}} \setminus X_j^{\text{Bad}}(x)$ be such that $x_{|j}^1, \ldots, x_{|j}^{b-1}$ are all distinct and $x^* = \text{hyb}_j(x, x^1, \ldots, x^{b-1})$. We first claim that $\text{wt}(x_{|j}^*) = b$: Let $\ell \in [b-1]$. Since $x^\ell \notin X_j^{\text{Bad}}(x)$, it holds that $x_{|j}^\ell \neq x_{|j}$. Since $j \in \text{Ones}$ and $x, x^\ell \in X^{\text{true}}$, it holds that $j \in \text{Ones}(x^\ell) = \text{Ones}(x)$. Since it is also the case that $x_{|j}^1, \ldots, x_{|j}^{b-1}$ are all distinct, we conclude that each of $x_{|j}, x_{|j}^1, \ldots, x_{|j}^{b-1}$ have exactly one 1 coordinate in group j and that all these coordinates are distinct. We get that $\text{wt}(x_{|j}^*) = b$ as $x_{|j}^* = \text{OR}(x_{|j}, x_{|j}^1, \ldots, x_{|j}^{b-1})$.

Since the definition of hyb implies $x_{|j'}^* = x_{|j'}$ for all $j' \neq j \in [k]$, and since $\text{Ones}(x) = \text{Ones}$ (as $x \in X^{\text{true}}$) and $\text{wt}(x_{|j}^*) = b$, we get that $x^* \in \text{supp}(\mathcal{D})$ and that $\text{Ones}(x^*) = \text{Ones} \setminus \{j\}$. This also means that x and x^* differ on the inputs of at most $b - 1$ parties.

To finish the proof, we need to show that $\text{trans}(x^*) = \pi$. For this, we show by induction that for every $0 \leq t \leq T$, it holds that $\text{trans}_{\leq t}(x^*) = \pi_{\leq t}$. The base case $t = 0$ is trivial. We next assume that the claim holds for some $0 \leq t \leq T - 1$ and prove that it holds for $t + 1$. Consider the following two cases:

Case ($\pi_{t+1} = 1$): Note that as $\pi_{t+1} = 1$, it holds that $\text{beep}(x, t+1) \neq \emptyset$. We will show that there exists a player $i \in \text{beep}(x, t+1)$ such that $x_i^* = x_i$. This is enough as the fact that player i beeped in round $t + 1$ given the input x to the players implies that player i beeps if he sees the transcript $\pi_{\leq t}$ and has the input x_i. Since, by the induction hypothesis, $\text{trans}_{\leq t}(x^*) = \pi_{\leq t}$, this would mean that he also beeps in round $t + 1$ given the input x^*. We conclude that $\text{beep}(x^*, t+1) \neq \emptyset$, and, in particular, $\text{trans}_{\leq t+1}(x^*) = \pi_{\leq t+1}$.

We next show that there exists a player $i \in \text{beep}(x, t+1)$ such that $x_i^* = x_i$. Since x and x^* differ on the inputs of at most $b - 1$ parties, this is clearly true in the case where $|\text{beep}(x, t+1)| \geq b$. Therefore, we assume that $|\text{beep}(x, t+1)| < b$ and show the stronger statement that $x_i^* = x_i$ for all $i \in \text{beep}(x, t+1)$. Suppose for the sake of contradiction that there exists $i \in \text{beep}(x, t+1)$ such that $x_i^* \neq x_i$. By definition of hyb, such an i must be group j. Also, from the definition of $B(x)$ and the fact that $|\text{beep}(x, t+1)| < b$, we can further conclude that $i \in B_j(x)$. Now, use the fact that $x^\ell \notin X_j^{\text{Bad}}(x)$ for all $\ell \in [b-1]$ to conclude that $\forall \ell \in [b-1] : x_i^\ell = 0$. By the definition of hyb, we also get that $x_i^* = x_i$, and we are done.

Case ($\pi_{t+1} = 0$): Assume for contradiction that there exists $i \in \text{beep}(x^*, t+1)$ (that is, $\text{beep}(x^*, t+1) \neq \emptyset$). By the definition of hyb, it holds that $x_i^* = x_i$ or there exists $\ell \in [b-1]$ such that $x_i^* = x_i^\ell$. Assume the latter and let $\ell \in [b-1]$ be such that $x_i^* = x_i^\ell$ (the former case where $x_i^* = x_i$ is similar). Then, since $i \in \text{beep}(x^*, t+1)$ and $x_i^* = x_i^\ell$, it also holds that $i \in \text{beep}(x^\ell, t+1)$. Since $x^\ell \in X$, we get a contradiction to the assumption $\pi_{t+1} = 0$. $\square$

Lemma 4.5. *For all $x \in X^{\text{true}}$ and $j \in \text{Ones}$, it holds that:*

$$|H_{x,j}| \geq \binom{|X_{|j}^{\text{true}}| - |B_j(x)| - 1}{b - 1}.$$

Proof. Throughout the proof, we use G to denote $G = X^{\text{true}} \setminus X_j^{\text{Bad}}(x)$. We first show that $|H_{x,j}| = \binom{|G_{|j}|}{b-1}$ by showing a bijection between the set $H_{x,j}$ and the set of all subsets of $G_{|j}$ of size $b-1$. The bijection $f_{x,j}$ takes an element $x^* \in H_{x,j}$ and acts as follows: By definition of $H_{x,j}$, there exists $x^1, \ldots, x^{b-1} \in G$ with $x_{|j}^1, \ldots, x_{|j}^{b-1}$ all distinct such that $x^* = \text{hyb}_j(x, x^1, \ldots, x^{b-1})$. Pick the lexicographically smallest such $x^1, \ldots, x^{b-1}$ and observe that $\left\{ x_{|j}^1, \ldots, x_{|j}^{b-1} \right\}$ is a subset of $G_{|j}$ of size $b-1$. This is the set output by $f_{x,j}$.

We now show that $f_{x,j}$ is indeed a bijection by showing that $f_{x,j}$ is both one-to-one and onto. For the former, note from the definition of hyb that, together with x and j, the set $\left\{ x_{|j}^1, \ldots, x_{|j}^{b-1} \right\}$ completely determines x^*. For the latter, note that for any subset $\left\{ t^1, \ldots, t^{b-1} \right\}$ of $G_{|j}$ of size $b-1$, there exists $x^1, \ldots, x^{b-1} \in G$ such that $\forall \ell \in [b-1]$, we have $x_{|j}^\ell = t^\ell$. Observe that for the lexicographically smallest such $x^1, \ldots, x^{b-1} \in G$, we have

$$f_{x,j}\big(\text{hyb}_j(x, x^1, \ldots, x^{b-1}) \big) = \left\{ t^1, \ldots, t^{b-1} \right\},$$

implying that the function $f_{x,j}$ is onto.

Next, we finish the proof of the lemma by showing that $|G_{|j}| = \left| \big(X^{\text{true}} \setminus X_j^{\text{Bad}}(x) \big)_{|j} \right| \geq \left| X_{|j}^{\text{true}} \right| - |B_j(x)| - 1$. For this, we use the fact that for any two sets S, T, we have $\left| (S \setminus T)_{|j} \right| \geq |S_{|j}| - \left| (S \cap T)_{|j} \right|$ and that $\left| \big(X^{\text{true}} \cap X_j^{\text{Bad}}(x) \big)_{|j} \right| \leq |B_j(x)| + 1$. The latter is because for any $x' \in X^{\text{true}} \cap X_j^{\text{Bad}}(x)$, the definition of X^{true} implies that there is exactly one player in group j with input 1, *i.e.*, we have $\text{wt}(x'_{|j}) = 1$ and the definition of $X_j^{\text{Bad}}(x)$ and the fact that $x \in X^{\text{true}}$ implies that there are at most $|B_j(x)| + 1$ choices for this coordinate. $\qquad \square$

Observation 4.6. *For $x \in \text{supp}(\mathcal{D})$ it holds that $\mathcal{D}(x) = 2^{-k} \cdot \left(\binom{n/k}{b} \right)^{|\text{Ones}(x)| - k} \cdot (n/k)^{-|\text{Ones}(x)|}$.*

Lemma 4.7. *For all $x \in X^{\text{true}}$, there exists $J \subseteq \text{Ones}$ with $|J| \geq 0.75|\text{Ones}|$ such that for all $j \in J$, we have:*

$$|H_{x,j}| \geq \left(\frac{n}{k} \right)^{(1-0.75\epsilon)(b-1)}.$$

Proof. Let $J = J^{\text{Large}} \setminus J^{\text{Bad}}(x)$, where

$$J^{\text{Large}} = \left\{ j \in \text{Ones} \mid |X_{|j}^{\text{true}}| \geq (n/k)^{1-\epsilon/2} \right\},$$
$$J^{\text{Bad}}(x) = \{ j \in [k] \mid |B_j(x)| > \epsilon^5 \log(n/k) \}.$$

We claim that $|J| \geq 0.75|\text{Ones}|$, by showing that $|J^{\text{Large}}| > 0.8|\text{Ones}|$ and that $|J^{\text{Bad}}(x)| \leq 0.05|\text{Ones}|$. We first show that $|J^{\text{Bad}}(x)| \leq 0.05|\text{Ones}|$: It holds that $|B(x)| \leq (b-1)T = \frac{T}{100\epsilon}$, as i can only be in $B(x)$ if there exists a round $t \in [T]$ such that $i \in \text{beep}(x, t)$ and $|\text{beep}(x, t)| < b$. But, if $|J^{\text{Bad}}(x)| > 0.01k$ then $|B(x)| = \sum_{j \in [k]} |B_j(x)| \geq \epsilon^5 \log(n/k) \cdot 0.01k = \frac{T}{100\epsilon^5}$, a contradiction. We get that $|J^{\text{Bad}}(x)| \leq 0.01k \leq 0.05|\text{Ones}|$, where the last inequality is by our selection of π.

We next show that $|J^{\mathsf{Large}}| > 0.8|\mathsf{Ones}|$: Assume for contradiction that $|J^{\mathsf{Large}}| \leq 0.8|\mathsf{Ones}|$. Fix $j \in [k]$. We upper bound $|X_{|j}^{\mathsf{true}}|$ by considering the following three cases:

1. If $j \in J^{\mathsf{Large}}$, then since $j \in \mathsf{Ones}$, it holds that $|X_{|j}^{\mathsf{true}}| \leq n/k$.

2. If $j \in \mathsf{Ones} \setminus J^{\mathsf{Large}}$, then by the definition of J^{Large}, it holds that $|X_{|j}^{\mathsf{true}}| < (n/k)^{1-\epsilon/2}$.

3. If $j \in [k] \setminus \mathsf{Ones}$, then, for all $x \in X^{\mathsf{true}}$, exactly b parties in group j have a 1 input. We get $|X_{|j}^{\mathsf{true}}| \leq \binom{n/k}{b}$.

Now, using these upper bounds and the fact that $X^{\mathsf{true}} \subseteq X_{|1}^{\mathsf{true}} \times \ldots \times X_{|k}^{\mathsf{true}}$, we get

$$|X^{\mathsf{true}}| \leq (n/k)^{|J^{\mathsf{Large}}|} \cdot \left((n/k)^{1-\epsilon/2}\right)^{|\mathsf{Ones}|-|J^{\mathsf{Large}}|} \cdot \left(\binom{n/k}{b}\right)^{k-|\mathsf{Ones}|}$$

$$= (n/k)^{|J^{\mathsf{Large}}| \cdot \epsilon/2} \cdot \left((n/k)^{1-\epsilon/2}\right)^{|\mathsf{Ones}|} \cdot \left(\binom{n/k}{b}\right)^{k-|\mathsf{Ones}|}$$

$$\leq (n/k)^{(1-\epsilon/10)|\mathsf{Ones}|} \cdot \left(\binom{n/k}{b}\right)^{k-|\mathsf{Ones}|}. \qquad (\text{As } |J^{\mathsf{Large}}| \leq 0.8|\mathsf{Ones}|)$$

Observe that for all $x \in X^{\mathsf{true}}$ it holds that $\mathsf{Ones}(x) = \mathsf{Ones}$, as the output induced by π is correct for input x. Using Observation 4.6, $\mathcal{D}(x) = 2^{-k} \cdot \left(\binom{n/k}{b}\right)^{|\mathsf{Ones}|-k} \cdot (n/k)^{-|\mathsf{Ones}|}$. This implies,

$$\mathcal{D}(X^{\mathsf{true}}) \leq 2^{-k} \cdot \left(\binom{n/k}{b}\right)^{|\mathsf{Ones}|-k} \cdot (n/k)^{-|\mathsf{Ones}|} \cdot (n/k)^{(1-\epsilon/10)|\mathsf{Ones}|} \cdot \left(\binom{n/k}{b}\right)^{k-|\mathsf{Ones}|}$$

$$\leq (n/k)^{-|\mathsf{Ones}| \cdot \epsilon/10}.$$

Because $|\mathsf{Ones}| \geq k/5$, we get $\mathcal{D}(X^{\mathsf{true}}) \leq (n/k)^{-k\epsilon/50}$. However, we choose π such that $\mathcal{D}(X^{\mathsf{true}}) \geq 2^{-10T}$, and thus, $(n/k)^{-k\epsilon/50} \geq 2^{-10T}$. This implies $T \geq (k\epsilon/500) \log(n/k)$, which is a contradiction to the assumed length of Π.

To finish the proof of the lemma, we lower bound $|H_{x,j}|$ as follows:

$$|H_{x,j}| \geq \binom{|X_{|j}^{\mathsf{true}}| - |B_j(x)| - 1}{b-1} \qquad (\text{Lemma 4.5})$$

$$\geq \binom{(n/k)^{1-\epsilon/2} - \epsilon^5 \log(n/k) - 1}{b-1} \qquad (\text{As } j \in J)$$

$$\geq \left(\frac{\max\left(0, (n/k)^{1-\epsilon/2} - \epsilon^5 \log(n/k) - 1\right)}{b-1}\right)^{b-1} \qquad \left(\text{As } \binom{s}{t} \geq \left(\frac{s}{t}\right)^t\right)$$

$$\geq n^{-0.01\epsilon} \cdot \left(\max\left(0, (n/k)^{1-\epsilon/2} - \epsilon^5 \log(n/k) - 1\right)\right)^{b-1}.$$

$$\left(\text{As } \epsilon = \frac{1}{100(b-1)} \text{ and } n > (100b)^{(100b)^{100}}\right)$$

To finish the proof, note that $f(x) = \left(x^{1-\epsilon/2} - \epsilon^5 \log(x) - 1\right) - x^{1-0.7\epsilon}$ is monotone for $x \geq 5$. Moreover, $f\left((100b)^{(100b)^{100}}\right) \geq 0$. Therefore, $f(n/k) \geq 0$ and we get

$$|H_{x,j}| \geq n^{-0.01\epsilon} \cdot (n/k)^{(1-0.7\epsilon)(b-1)} \geq \left(\frac{n}{k}\right)^{(1-0.75\epsilon)(b-1)}.$$

$\square$

Define the set:
$$H = \bigcup_{x \in X^{\text{true}}, j \in \text{Ones}} H_{x,j}.$$

Lemma 4.8. *It holds that* $|H| \geq |X^{\text{true}}| \cdot \left(\frac{n}{k}\right)^{(1+\epsilon)(b-1)}$.

Proof. For $x^* \in H$, define $S(x^*)$ to be the set of all pairs $(x, j) \in X^{\text{true}} \times \text{Ones}$ such that $x^* \in H_{x,j}$. We claim that $|S(x^*)| \leq b$ for all $x^* \in H$. Before we prove this claim, we show that it implies the lemma. Indeed, we have:

$$|H| = \sum_{x^* : |S(x^*)| > 0} 1$$

$$= \sum_{x^* : |S(x^*)| > 0} \frac{1}{|S(x^*)|} \cdot \sum_{x \in X^{\text{true}}, j \in \text{Ones}} \mathbb{1}[(x,j) \in S(x^*)]$$

$$\geq \frac{1}{b} \cdot \sum_{x^* : |S(x^*)| > 0} \sum_{x \in X^{\text{true}}, j \in \text{Ones}} \mathbb{1}[(x,j) \in S(x^*)] \qquad (\text{As } \forall x^* \in H : |S(x^*)| \leq b)$$

$$= \frac{1}{b} \cdot \sum_{x^*} \sum_{x \in X^{\text{true}}, j \in \text{Ones}} \mathbb{1}[(x,j) \in S(x^*)]$$

$$= \frac{1}{b} \cdot \sum_{x \in X^{\text{true}}, j \in \text{Ones}} |H_{x,j}|$$

$$\geq \frac{1}{b} \cdot |X^{\text{true}}| \cdot 0.75 \cdot |\text{Ones}| \cdot \left(\frac{n}{k}\right)^{(1-0.75\epsilon)(b-1)} \qquad (\text{Lemma 4.7})$$

$$\geq \frac{k}{10b} \cdot |X^{\text{true}}| \cdot \left(\frac{n}{k}\right)^{(1-0.75\epsilon)(b-1)} \qquad (\text{As } |\text{Ones}| \geq k/5)$$

$$\geq |X^{\text{true}}| \cdot \left(\frac{n}{k}\right)^{(1+\epsilon)(b-1)}. \qquad (\text{As } \epsilon = \frac{1}{100(b-1)} \text{ and } n = k^2 > (100b)^{(100b)^{100}})$$

It remains to show that $|S(x^*)| \leq b$ for all $x^* \in H$. For this, use Lemma 4.4 to conclude that for all $(x, j) \in S(x^*)$, we have $\{j\} = \text{Ones} \setminus \text{Ones}(x^*)$ implying that j is unique. This also implies that for all $(x, j) \in S(x^*)$, we have $\text{wt}(x_{|j}^*) = b$ and (using the definition of hyb) that $x_{|j'}^* = x_{|j'}$ and $x^* \geq x$ for all $j' \neq j \in [k]$. Together, we get that there are at most b choices of x which when combined with the fact that j is unique implies that $|S(x^*)| \leq b$. $\square$

Proof of Lemma 4.3. Note first that for all $x \in X^{\text{true}}$, it holds that $\text{Ones}(x) = \text{Ones}$, as the output induced by π is correct for input x. Additionally, for all $x^* \in H$, by Lemma 4.4,

$|\mathsf{Ones}(x^*)| = |\mathsf{Ones}| - 1$. By Observation 4.6 and Lemma 4.8, this means that:

$$\frac{\mathcal{D}(H)}{\mathcal{D}(X^{\mathsf{true}})} = \frac{|H|}{|X^{\mathsf{true}}|} \cdot \frac{\mathcal{D}(x^*)}{\mathcal{D}(x)} = \frac{|H|}{|X^{\mathsf{true}}|} \cdot \frac{n/k}{\binom{n/k}{b}} \geq \frac{|H|}{|X^{\mathsf{true}}|} \cdot \left(\frac{k}{n}\right)^{b-1} \geq \left(\frac{n}{k}\right)^{\epsilon \cdot (b-1)}.$$

Now, use the definition of H and Lemma 4.4 to get:

$$0 \leq \mathcal{D}(X^{\mathsf{true}}) \leq \mathcal{D}(X) - \mathcal{D}(H) \leq \mathcal{D}(X) - \mathcal{D}(X^{\mathsf{true}}) \cdot \left(\frac{n}{k}\right)^{\epsilon \cdot (b-1)}.$$

Rearranging and using $\epsilon = \frac{1}{100(b-1)}$ and $n = k^2$ finishes the proof. $\qquad\square$

5   Hardness of Interactive Coding

In this section, we prove the following Theorem 5.1, which implies Theorem 1.4.

Theorem 5.1. *Let $\epsilon \geq 0$ and $k_1, k_2 \geq 2$. For all $n > 0$, there exists a (deterministic) n-party protocol Π over the $\mathrm{THRESHOLD}[k_1]$ channel, such that any (possibly randomized) n-party protocol Π' that computes the same function as Π with success probability greater than $1/n^{1/300}$ over the ϵ-noisy $\mathrm{THRESHOLD}[k_2]$ channel requires $\Omega(|\Pi| \log n)$ rounds.*

The same holds true if the $\mathrm{THRESHOLD}[k_1]$ channel is replaced by the CAS channel, or if the ϵ-noisy $\mathrm{THRESHOLD}[k_2]$ channel is replaced by the ϵ-noisy CAS channel, or both.

To prove Theorem 5.1, we use the following Theorem 5.2 that shows that for all $k \geq 2$ every noisy $\mathrm{THRESHOLD}[k]$ round can be simulated by constantly many rounds of the $\mathrm{BEEPING}$ channel.

Theorem 5.2. *Let $\epsilon \geq 0$ and $k \geq 2, n > 0$ be integers. There exists an $\mathcal{O}_{k,\epsilon}(1)$ length n-party protocol over $\mathrm{BEEPING}$ such that if patry $i \in [n]$ has the input bit $x_i \in \{0, 1\}$, then at the end of the protocol all the parties output a bit b that equals $\mathbb{1}[\mathsf{wt}(x) \geq k]$ with probability least $1 - \epsilon$.*

Proof of Theorem 5.1. For the case of $\mathrm{THRESHOLD}[k_1]$ channel, define $b = k_1$. Otherwise, for the CAS channel, define $b = 2$. Assume without loss of generality that that there exists a $k > (100b)^{(100b)^{100}}$ such that $n = k^2$, and observe that there exists a deterministic k round protocol over the channel concerned that solves the $(1, b)$-weight problem with n players and k groups. Additionally, we have from Theorem 4.2 that every (randomized) protocol over the $\mathrm{BEEPING}$ model that solves this problem with success probability greater than $1/n^{1/300}$ has at least $\Omega_b(k \log(n/k))$ rounds.

To finish the proof, we argue that if there is such a protocol over the ϵ-noisy $\mathrm{THRESHOLD}[k_2]$ channel, or over the ϵ-noisy CAS channel, then there also exists such a protocol over the $\mathrm{BEEPING}$ channel, a contradiction. In the case of the ϵ-noisy $\mathrm{THRESHOLD}[k_2]$, this follows directly by simulating the protocol round by round and using Theorem 5.2 with $k = k_2$,

while in the case of the ϵ-noisy CAS channel, this follows similarly from Theorem 5.2 using $k = 2$ and the identity $\mathbb{1}[\mathsf{wt}(x) = 1] = \mathbb{1}[\mathsf{wt}(x) \geq 1] - \mathbb{1}[\mathsf{wt}(x) \geq 2]$ for all $x \in \{0, 1\}^n$. Note that the first term can be computed exactly in one round over the BEEPING channel. $\square$

Proof of Theorem 5.2. We claim that Algorithm 1 below outputs 0 if $\mathsf{wt}(x) < k$ and outputs 1, except with probability at most ϵ, if $\mathsf{wt}(x) \geq k$.

Algorithm 1 An algorithm computing $\mathbb{1}[\mathsf{wt}(x) \geq k]$ over BEEPING.

Input: Each player $i \in [n]$ has an input $x_i \in \{0, 1\}$.
Output: Each player $i \in [n]$ outputs a bit b (supposedly, $b = 1$ if $\mathsf{wt}(x) \geq k$ and $b = 0$ otherwise).
1: **for** $3 \log(1/\epsilon)$ times **do**
2:      Each player $i \in [n]$ randomly chooses a number $n_i \in [k^2]$.
3:      **for** $j \in [k^2]$ **do**
4:          Each player sets $b_j^i = x_i \wedge \mathbb{1}[n_i = j]$.
5:          In one beeping round: Each player i beeps b_j^i. All parties receive the bit z_j from the channel.
6:      **end for**
7:      **if** $\sum_{j \in [k^2]} z_j \geq k$ **then**
8:          **return** 1
9:      **end if**
10: **end for**
11: **return** 0

Let us consider just one iteration of the loop at Line 1. To begin with, note that the only players who ever broadcast during the algorithm are those with $x_i = 1$, as can be seen at Line 4. As such, we can restrict our analysis to just those players, as the remaining players do not influence the execution of the algorithm in any way. Let this group of players be S.

We can now partition S into k^2 sets $S_1, \ldots, S_{k^2}$. In particular, S_j consists of the players $i \in [n]$ such that $n_i = j$. It is clear that this partitions S. We also note that at Line 5, exactly the players in S_j broadcast. Thus, z_j is 1 if and only if S_j is non-empty. Furthermore, we thus get that $\sum_{j \in [k^2]} z_j$ is exactly the number of non-empty S_j sets.

Now, we can consider two cases, depending on if $\mathsf{wt}(x) \geq k$ or if $\mathsf{wt}(x) < k$. Let us analyze the latter case. It is evident that $\mathsf{wt}(x) = |S|$. Furthermore, as $S = S_1 \cup \cdots \cup S_{k^2}$ and these sets are disjoint, we get that the number of non-empty S_j is at most the size of $|S|$. Thus, the sum at Line 7 will be strictly less than k, so the algorithm will never return 1. As such, the algorithm returns 0 with probability 1, as desired.

Now, suppose that $\mathsf{wt}(x) \geq k$. Thus, $|S| \geq k$. Fix k elements of S. We can now analyze the probability that these k elements each fall into different S_j. Let this event be A.

$$\Pr[A] = \left(\frac{k^2}{k^2}\right)\left(\frac{k^2 - 1}{k^2}\right) \cdots \left(\frac{k^2 - k + 1}{k^2}\right) \geq \left(\frac{k^2 - k}{k^2}\right)^k = \left(1 - \frac{1}{k}\right)^k \geq \frac{1}{4}.$$

Thus, with probability at least $\frac{1}{4}$, each of the k elements are in a different S_j, so there are at least k non-empty S_j. Thus, the sum at Line 7 will be at least k, so the algorithm will return 1. In other words, the probability that the algorithm doesn't return 1 during this iteration of the loop at Line 1 is at most $\frac{3}{4}$. By independence of the iterations, the probability that the algorithm doesn't return 1 during any of the iterations of the loop is thus at most

$$\left(\frac{3}{4}\right)^{3\log(1/\epsilon)} \leq \left(\frac{1}{2}\right)^{\log(1/\epsilon)} = \epsilon.$$

Thus, the algorithm will return 1, except with probability at most ϵ, as desired. □

5.1 CAS is THRESHOLD[2]

In this subsection, we formally show the claim made in Section 1 that the CAS channel is equivalent to the THRESHOLD[2] channel.

Theorem 5.3. *The* CAS *channel and the* THRESHOLD[2]*-channel are equivalent.*

To show this, we first show a simpler claim about threshold channels, which shall be used throughout this paper.

Theorem 5.4. *For all integers $r_1 \leq r_2 \in \mathbb{N}$, for all $n > 0$, there exists an n-party protocol over* THRESHOLD[r_2] *which solves the function $f(x) = \mathbb{1}[\mathsf{wt}(x) \geq r_1]$ with success probability 1 in $\mathcal{O}_{r_1,r_2}(1)$ rounds.*

Proof. We can assume $1 \leq r_1 < r_2$ without loss of generality. We begin by analysing the algorithm $\texttt{find-zeros}^{r_2}_{\ell_0,\ell_1}$, shown in Algorithm 2. We claim that for all $\ell_0, \ell_1 \geq 1$, when $\texttt{find-zeros}^{r_2}_{\ell_0,\ell_1}$ is run over THRESHOLD[r_2], it has each player output a tuple (b, S) such that $b \in \{0, 1\}$, $S \subseteq [n]$, $x_i = b$ for all $i \in S$, and such that $|S| = \ell_b$.

Note that during an execution of Algorithm 2, each player receives the same z_j at Line 4. As the values of S_0 and S_1 that each player maintains depend only on the values of z_j that they receive, this means that each player outputs the same result once the algorithm outputs. As such, we can look at the algorithm from an outside perspective.

We claim that during an execution of Algorithm 2 $z_j = x_j$ for all j considered during the loop. For this, note that for exactly $r_2 - 1$ other players, $b^i_j = 1$. Thus, $\sum_{i \in [n]} b^i_j = r_2 - 1 + x_j$. As such, the THRESHOLD[r_2] channel sets

$$z_j = \mathbb{1}\left[\sum_{i \in [n]} b^i_j \geq r_2\right] = \mathbb{1}[r_2 - 1 + x_j \geq r_2] = \mathbb{1}[x_j \geq 1] = x_j.$$

Then, at Line 5, each player adds j to the set $S_{z_j} = S_{x_j}$. As such, in each iteration, one of S_0 or S_1 increases by 1, and ensures that for each $j \in S_0$, $x_j = 0$ and for each $j \in S_1$, $x_j = 1$.

Algorithm 2 Algorithm $\texttt{find-zeros}_{\ell_0,\ell_1}^{r_2}$. This algorithm is used to find a set of players with the same input.

Input: Each player $i \in [n]$ has an input $x_i \in \{0, 1\}$.
Output: Each player $i \in [n]$ outputs a pair (b, S).
 1: Each player initializes two sets $S_0 = \emptyset$ and $S_1 = \emptyset$.
 2: **for** $j \in [\ell_0 + \ell_1 - 1]$ **do**
 3:     Each player $i \in [n]$ sets b_j^i according to

$$
b_j^i = \begin{cases} x_i, & i - j = 0 \\ 1, & i - j \in [r_2 - 1] \\ 0, & \text{otherwise} \end{cases}
$$

 4:     Players broadcast with b_j^i to get z_j.
 5:     $S_{z_j} \leftarrow S_{z_j} \cup \{j\}$
 6:     **if** $S_0 \geq \ell_0$ **then**
 7:         **return** $(0, S_0)$
 8:     **end if**
 9:     **if** $S_1 \geq \ell_1$ **then**
10:         **return** $(1, S_1)$
11:     **end if**
12: **end for**

Furthermore, after $\ell_0 + \ell_1 - 1$ repetitions of Line 2, either $|S_0| \geq \ell_0$ or $|S_1| \geq \ell_1$. Thus, at that point, each player outputs either $(0, S_0)$ or $(1, S_1)$, which ensures that the output of $\texttt{find-zeros}_{\ell_0,\ell_1}^{r_2}$ is as desired.

Using this result, we now demonstrate Algorithm 3, which we claim computes $\mathbb{1}[\mathsf{wt}(x) \geq r_1]$ when run over THRESHOLD$[r_2]$.

The first thing Algorithm 3 does is run $\texttt{find-zeros}_{r_2-r_1,r_1}^{r_2}$ at Line 13. This takes $\mathcal{O}(r_2)$ broadcasts, and returns some pair (b, S). If $b = 1$, then S is a subset of $[n]$ satisfying $|S| = r_1$ such that $x_{i'} = 1$ for all $i' \in S$, which implies that

$$
\mathbb{1}[\mathsf{wt}(x) \geq r_1] \geq \mathbb{1}\left[\sum_{i \in S} x_i \geq r_1\right] = \mathbb{1}[|S| \geq r_1] = 1,
$$

which means that returning 1 at Line 15 actually gives the correct result.

In the alternative case that $b = 0$, then we know that S is a subset of $[n]$ satisfying $|S| = r_2 - r_1$ such that $x_{i'} = 0$ for all $i' \in S$. Thus, due to Line 17, we know that the b^i values satisfy

$$
\mathbb{1}\left[\sum_{i \in [n]} b^i \geq r_2\right] = \mathbb{1}\left[\sum_{i \in [n], x_i = 1} 1 + \sum_{i \in S} 1 \geq r_2\right] = \mathbb{1}[\mathsf{wt}(x) + |S| \geq r_2]
$$

Algorithm 3 An algorithm for calculating $\mathbb{1}[\mathsf{wt}(x) \geq r_1]$ over THRESHOLD$[r_2]$.

Input: Each player $i \in [n]$ has an input $x_i \in \{0,1\}$.
Output: Each player $i \in [n]$ outputs whether or not $\mathsf{wt}(x) \geq r_1$.
13: The players run $\mathtt{find\text{-}zeros}_{r_2-r_1,r_1}^{r_2}$ on their inputs x_i to obtain (b, S).
14: **if** $b = 1$ **then**
15:     **return** 1
16: **end if**
17: Set b^i for players $i \in [n]$ according to

$$b^i = \begin{cases} 1, & x_i = 1 \\ 1, & i \in S \\ 0, & \text{otherwise} \end{cases}$$

18: Players broadcast with b^i to get z.
19: **return** z

$$= \mathbb{1}[\mathsf{wt}(x) + r_2 - r_1 \geq r_2] = \mathbb{1}[\mathsf{wt}(x) \geq r_1].$$

Thus, the value of z that is returned at the end of Algorithm 3 is exactly the desired value, concluding the proof. $\qquad\square$

We also get a direct consequence from this.

Corollary 5.5. *For all integers $r_1 \leq r_2 \in \mathbb{N}$, if there exists a protocol computing some function F over* THRESHOLD$[r_1]$ *in T rounds, then there exists a protocol computing F over* THRESHOLD$[r_2]$ *in $\mathcal{O}_{r_1,r_2}(T)$ rounds.*

This follows via simulating each round of the original protocol using the above result. Anyway, we now return to the proof of Theorem 5.3.

Proof of Theorem 5.3. To show the desired result, it suffices to show that there exist protocols solving $\mathbb{1}[\mathsf{wt}(x) \geq 2]$ over CAS and solving $\mathbb{1}[\mathsf{wt}(x) = 1]$ over THRESHOLD$[2]$ in $\mathcal{O}(1)$ rounds of communication (with success probability 1).

First, note that Theorem 5.4 shows that there exists a $\mathcal{O}(1)$ round protocol over THRESHOLD$[2]$ computing $\mathbb{1}[\mathsf{wt}(x) \geq 1]$. Furthermore, there exists a trivial one-round protocol over THRESHOLD$[2]$ computing $\mathbb{1}[\mathsf{wt}(x) \geq 2]$. Finally, as

$$\mathbb{1}[\mathsf{wt}(x) = 1] = \mathbb{1}[\mathsf{wt}(x) \geq 1] - \mathbb{1}[\mathsf{wt}(x) \geq 2],$$

we get that there exist an $\mathcal{O}(1)$ round protocol computing $\mathbb{1}[\mathsf{wt}(x) = 1]$ over THRESHOLD$[2]$. Thus, any protocol over CAS can be simulated over THRESHOLD$[2]$ with only $\mathcal{O}(1)$ blow-up in the number of rounds.

Algorithm 4 An algorithm computing $\mathbb{1}[\mathsf{wt}(x) \geq 2]$ over CAS.

Input: Each player $i \in [n]$ has an input $x_i \in \{0, 1\}$.
Output: Each player $i \in [n]$ outputs whether or not $\mathsf{wt}(x) \geq 2$.
20: Players broadcast with $x_i \vee \mathbb{1}[i = 1]$ to get z_1.
21: Players broadcast x_i to get z_2.
22: **return** $\neg(z_1 \vee z_2)$

Now, consider Algorithm 4. We claim that this computes $\mathbb{1}[\mathsf{wt}(x) \geq 2]$ over CAS in $\mathcal{O}(1)$ rounds, which would complete the proof. To show this, it suffices to consider three possible cases for $\mathsf{wt}(x)$:

- **If $\mathsf{wt}(x) = 0$:** In this case, $x_i = 0$ for all $i \in [n]$. Thus, $z_1 = 1$ and $z_2 = 0$, and the algorithm will return $0 = \mathbb{1}[\mathsf{wt}(x) \geq 2]$.

- **If $\mathsf{wt}(x) = 1$:** In this case, $z_2 = \mathbb{1}[\mathsf{wt}(x) = 1] = 1$, so the algorithm will return $0 = \mathbb{1}[\mathsf{wt}(x) \geq 2]$.

- **If $\mathsf{wt}(x) \geq 2$:** In this case, $z_1 = z_2 = 0$, and the algorithm returns $1 = \mathbb{1}[\mathsf{wt}(x) \geq 2]$.

Thus, this algorithm computes $\mathbb{1}[\mathsf{wt}(x) \geq 2]$, using only 2 broadcasts, which concludes the proof. $\qquad\square$

References

[AAB+11]   Yehuda Afek, Noga Alon, Omer Barad, Eran Hornstein, Naama Barkai, and Ziv Bar-Joseph. A biological solution to a fundamental distributed computing problem. *Science*, 331(6014):183–185, 2011. 2

[ABE+16]   Noga Alon, Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Reliable communication over highly connected noisy networks. In *Symposium on Principles of Distributed Computing (DISC)*, pages 165–173, 2016. 6

[AGL20]   Yagel Ashkenazi, Ran Gelles, and Amir Leshem. Brief announcement: Noisy beeping networks. In *Symposium on Principles of Distributed Computing (PODC)*, pages 458–460, 2020. 6, 7

[BEGH16]   Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Constant-rate coding for multiparty interactive communication is impossible. In *Symposium on Theory of Computing (STOC)*, pages 999–1010, 2016. 6

[BKOT19]   Mark Braverman, Gillat Kol, Rotem Oshman, and Avishay Tal. On the computational power of radio channels. In *International Symposium on Distributed Computing (DISC)*, volume 146, pages 8:1–8:17, 2019. 4, 7, 8

[Bsh12]   Nader H. Bshouty. On the coin weighing problem with the presence of noise. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques (RANDOM/APPROX)*, volume 7408 of *Lecture Notes in Computer Science*, pages 471–482, 2012. 8

[Can64]   David G Cantor. Determining a set from the cardinalities of its intersections with other sets. *Canadian Journal of Mathematics*, 16:94–97, 1964. 8

[CHHHZ19]   Keren Censor-Hillel, Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Erasure correction for noisy radio networks. In *International Symposium on Distributed Computing (DISC)*, 2019. 6

[CHHZ17]   Keren Censor-Hillel, Bernhard Haeupler, D. Ellis Hershkowitz, and Goran Zuzic. Broadcasting in noisy radio networks. In *Symposium on Principles of Distributed Computing (PODC)*, pages 33–42, 2017. 6

[CK85]   Imrich Chlamtac and Shay Kutten. On broadcasting in radio networks-problem analysis and protocol design. *IEEE Trans. Communications*, 33(12):1240–1246, 1985. 2

[CK10]   Alejandro Cornejo and Fabian Kuhn. Deploying wireless networks with beeps. In *International Symposium on Distributed Computing (DISC)*, pages 148–162, 2010. 2

[CMS01]   Andrea E. F. Clementi, Angelo Monti, and Riccardo Silvestri. Selective families, superimposed codes, and broadcasting on unknown radio networks. In *Symposium on Discrete Algorithms (SODA)*, pages 709–718, 2001. 8

[CMT17]   Bogdan S. Chlebus, Gianluca De Marco, and Muhammed Talo. Naming a channel with beeps. *Fundamenta Informaticae*, 153(3)(199–219), 2017. 7

[EKPS21a]   Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Computation over the noisy broadcast channel with malicious parties. In *Innovations in Theoretical Computer Science Conference, (ITCS)*, volume 185, pages 82:1–82:19, 2021. 6

[EKPS21b]   Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Tight bounds for general computation in noisy broadcast networks. In *Symposium on Foundations of Computer Science (FOCS)*, 2021. 6

[EKS18]   Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive coding over the noisy broadcast channel. In *Symposium on Theory of Computing (STOC)*, pages 507–520, 2018. 6

[EKS19]   Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Radio network coding requires logarithmic overhead. In *Foundations of Computer Science (FOCS)*, pages 348–369, 2019. 6, 7

[EKS20]   Klim Efremenko, Gillat Kol, and Raghuvansh R. Saxena. Noisy beeps. In *Symposium on Principles of Distributed Computing (PODC)*, pages 418–427, 2020. 6, 7

[ER63]   Paul Erdos and Alfréd Rényi. On two problems of information theory. *Magyar Tud. Akad. Mat. Kutató Int. Közl*, 8:229–243, 1963. 8

[GGLR05] László Györfi, S Győri, Bálint Laczay, and M Ruszinkó. Lectures on multiple access channels, book draft, 2005. 8

[GK19]   Ran Gelles and Yael T Kalai. Constant-rate interactive coding is impossible, even in constant-degree networks. *IEEE Transactions on Information Theory*, 65:3812–3829, 2019. 6

[GKR19]   Ran Gelles, Yael Tauman Kalai, and Govind Ramnarayan. Efficient multiparty interactive coding for insertions, deletions, and substitutions. In *Symposium on Principles of Distributed Computing (PODC)*, pages 137–146, 2019. 6

[HMP20]   Kokouvi Hounkanli, Avery Miller, and Andrzej Pelc. Global synchronization and consensus using beeps in a fault-prone multiple access channel. *Theoretical Computer Science*, 806:567–576, 2020. 7

[HS16]   William M. Hoza and Leonard J. Schulman. The adversarial noise threshold for distributed protocols. In *Symposium on Discrete Algorithms (SODA)*, pages 240–258, 2016. 6

[JKL15]   Abhishek Jain, Yael Tauman Kalai, and Allison Bishop Lewko. Interactive coding for multiparty protocols. In *Innovations in Theoretical Computer Science (ITCS)*, pages 1–10, 2015. 6

[KKP22]   Marek Klonowski, Dariusz R. Kowalski, and Dominik Pajak. Generalized framework for group testing: Queries, feedbacks and adversaries. *Theoretical Computer Science*, 919:18–35, 2022. 8

[Lin64]   Bernt Lindström. On a combinatory detection problem i. *I. Magyar Tud. Akad. Mat. Kutató Int. Közl*, 9:195–207, 1964. 8

[Lin65]   Bernt Lindström. On a combinatorial problem in number theory. *Canadian Mathematical Bulletin*, 8(4):477–490, 1965. 8

[Lin71]   Bernt Lindström. On möbius functions and a problem in combinatorial number theory. *Canadian Mathematical Bulletin*, 14(4):513–516, 1971. 8

[Lin75]   B Lindström. Determining subsets by unramified experiments. in editor jn srivastava, editor, a survey of statistical designs and linear models, 1975. 8

[MK13]   Gianluca De Marco and Dariusz R. Kowalski. Searching for a subset of counterfeit coins: Randomization vs determinism and adaptiveness vs non-adaptiveness. *Random Struct. Algorithms*, 42(1):97–109, 2013. 8

[MSY16]   Shay Moran, Makrand Sinha, and Amir Yehudayoff. Fooling pairs in randomized communication complexity. In Jukka Suomela, editor, *Structural Information and Communication Complexity (SIROCCO)*, volume 9988, pages 49–59, 2016. 4

[NB15]   Saket Navlakha and Ziv Bar-Joseph. Distributed information processing in biological and computational systems. *Communications of the ACM*, 58(1):94–102, 2015. 2

[New14]   Calvin C. Newport. Radio network lower bounds made easy. In *International Symposium on Distributed Computing (DISC)*, volume 8784 of *Lecture Notes in Computer Science*, pages 258–272, 2014. 8

[RS94]   Sridhar Rajagopalan and Leonard J. Schulman. A coding theorem for distributed computation. In *Symposium on the Theory of Computing (STOC)*, pages 790–799, 1994. 6

[Sch92]   Leonard J Schulman. Communication on noisy channels: A coding theorem for computation. In *Foundations of Computer Science (FOCS)*, pages 724–733, 1992. 6

[SS63]   H Shapiro and S Soderberg. A combinatory detection problem. *Amer. Math. Monthly*, 70:1066–1070, 1963. 8