Some Lower Bounds in Dynamic Networks with Oblivious Adversaries[*]

Irvan Jahja
National University of Singapore
irvan@comp.nus.edu.sg

Haifeng Yu
National University of Singapore
haifeng@comp.nus.edu.sg

Yuda Zhao[†]
Grab, Singapore
yudazhao@gmail.com

Abstract

This paper considers several closely-related problems in synchronous dynamic networks with *oblivious adversaries*, and proves novel $\Omega(d + \text{poly}(m))$ lower bounds on their time complexity (in terms of rounds). Here d is the dynamic diameter of the dynamic network and m is the total number of nodes. Before this work, the only known lower bounds on these problems under oblivious adversaries were the trivial $\Omega(d)$ lower bounds. Our novel lower bounds are hence the first non-trivial lower bounds and also the first lower bounds with a poly(m) term. Our proof relies on a novel reduction from a certain two-party communication complexity problem. Our central proof technique is unique in the sense that we consider the communication complexity with a special *leaker*. The leaker helps Alice and Bob in the two-party problem, by disclosing to Alice and Bob certain "non-critical" information about the problem instance that they are solving.

1 Introduction

Dynamic networks [25] is a flourishing topic in recent years. We consider a synchronous setting where the m (fixed) nodes in the network proceed in synchronous rounds. Each node has a unique id of size $O(\log m)$, and the messages are of size $O(\log m)$ as well. The nodes never fail. The topology of the dynamic network can change from round to round, as determined by an *adversary*, subject to the only constraint that the topology in each round must be a connected and undirected graph. The *time complexity* of a protocol is the number of rounds needed for all nodes to generate the final output, over the worst-case adversary, worst-case initial values, and average coin flips of the protocol. We consider a number of fundamental distributed computing problems within such a context:

- CONSENSUS: Each node has a binary input. The nodes aim to achieve a consensus (with the standard agreement, validity, and termination requirements) and output the final decision.
- LEADERELECT: Each node should output the leader's id.
- CONFIRMEDFLOOD: A certain node ν aims to propagate a token of size $O(\log m)$ to all other nodes, and wants to further confirm that all nodes have received the token.[1] Formally, node ν's

[*]The authors of this paper are alphabetically ordered. This work is partly supported by the research grant MOE2014-T2-2-030 from Singapore Ministry of Education Academic Research Fund Tier-2.

[†]This work was done while this author was in National University of Singapore

[1]Such confirmation does not have to come from explicit acknowledgements, and can be via implicit means, such as counting the number of rounds.

output is correct only if by the time that ν outputs, the token has already been received by all the nodes. (The value of the output is not important.) The remaining nodes can output any time.

- AGGREGATION: Each node has a value of $O(\log m)$ bits, and the nodes aim to compute a certain aggregation function over all these values. We consider two specific aggregation functions, SUM and MAX.

Let d be the *(dynamic) diameter* (see definition later) of the dynamic network. (Note that since the topology is controlled by an adversary, the protocol never knows d beforehand.) Given an optimal protocol for solving any of the above problems, let $\text{tc}(d, m)$ denote the protocol's time complexity, when it runs over networks with d diameter and m nodes. It is easy to see that $\text{tc}(d, m)$ crucially depends on d, since we trivially have $\text{tc}(d, m) = \Omega(d)$. Given such, this paper focus on the following central question:

Ignoring polylog(m) terms, is tc(d, m) independent of the network size m?

Answering this fundamental question will reveal whether the complexity of all these basic problems is due to the diameter or due to both the diameter and the network size.

Existing results. If the network were *static*, then building a spanning tree would solve all these problems in either $O(d)$ or $O(d \log m)$ rounds, implying a **yes** answer to the above question. In dynamic networks, the picture is more complex. In a dynamic network model without congestion (i.e., message size unlimited), Kuhn et al. [23] have proposed elegant upper bound protocols with $O(d)$ complexity for all these problems. Hence the answer is **yes** as well. For dynamic networks with congestion (i.e., message size limited to $O(\log m)$), Yu et al. [32] recently have proved that $\text{tc}(d, m) = O(d \log m)$ for CONSENSUS and LEADERELECT, if the nodes know a *good* estimate on m.[2] Hence the answer is **yes** in such cases. One the other hand, if nodes' estimate on m is *poor*,[3] then Yu et al. [32] prove a lower bound of $\Omega(d+\text{poly}(m))$ for CONSENSUS and LEADERELECT, implying a **no** answer. For CONFIRMEDFLOOD and AGGREGATION, they have also proved $\text{tc}(d, m) = \Omega(d + \text{poly}(m))$, even if the nodes know m. This implies a **no** answer for those two problems.

All the lower bound proofs in [32], however, critically relies on a powerful *adaptive adversary*: In each round, the adaptive adversary sees all the coin flip outcomes so far of the protocol $\mathscr{P}$ and manipulates the topology based on those. In particular, in each round the adversary sees whether each node will be sending (and can then manipulate the topology accordingly), *before* the nodes actually send their messages. Their proof breaks under *oblivious adversaries*, which do not see $\mathscr{P}$'s coin flip outcomes and have to decide the topologies in all the rounds before $\mathscr{P}$ starts.[4]

In summary, our central question of whether $\text{tc}(d, m)$ is largely independent of the network size m has been answered in: i) static networks, ii) dynamic networks without congestion under both adaptive and oblivious adversaries, and iii) dynamic networks with congestion under adaptive adversaries.

Our results. This work gives the last piece of the puzzle for answering our central question. Specifically, we show that in dynamic networks with congestion and under oblivious adversaries, for CONSENSUS and LEADERELECT, the answer to the question is **no** when the nodes' estimate on m is poor. (If the nodes' estimate on m is good, results from [32] already implied a **yes** answer.) Specifically, we prove a novel $\Omega(d + \text{poly}(m))$ lower bound on CONSENSUS under oblivious adversaries, when the nodes' estimate on m is poor. This is the first non-trivial lower bound and also the first lower bound with a $\text{poly}(m)$ term, for CONSENSUS under oblivious adversaries. The best lower bound before this work was the trivial $\Omega(d)$ lower bound. Our CONSENSUS lower bound directly carries over to LEADERELECT since CONSENSUS reduces to LEADERELECT [32].

[2]More precisely, if the nodes know m' such that $|\frac{m'-m}{m}| \leq \frac{1}{3} - c$ for some positive constant c. Obviously, this covers the case where the nodes know m itself.

[3]More precisely, if the nodes only knows m' such that $|\frac{m'-m}{m}|$ reaches $\frac{1}{3}$ or above. Obviously, this covers the case where the nodes do not have any knowledge about m.

[4]Note however that all upper bounds, from [23] and [32], will directly carry over to oblivious adversaries.

Our approach will also lead to a $\Omega(d + \text{poly}(m))$ lower bound under oblivious adversaries for CONFIRMEDFLOOD, which in turn reduces to SUM and MAX [32]. Such a lower bound similarly gives a **no** answer for CONFIRMEDFLOOD and AGGREGATION. But since the lower bound proof for CONFIRMEDFLOOD is similar to and in fact easier than our CONSENSUS proof, for clarity, we will not separately discuss it in this paper.

Different adversaries. In dynamic networks, different kinds of adversaries often require different algorithmic techniques and also yield different results. Hence it is common for researchers to study them separately. For example, lower bounds for information dissemination were proved separately, under adaptive adversaries [16] and then later under oblivious adversaries [1]. Dynamic MIS was investigated separately under adaptive adversaries [20] and later under oblivious adversaries [11]. Broadcasting was first studied under adaptive adversaries [21], and later under oblivious adversaries [17].

Our approach. Our novel CONSENSUS lower bound under oblivious adversaries is obtained via a reduction from a two-party communication complexity (CC) problem called *Gap Disjointness with Cycle Promise* or GDC. Our reduction partly builds upon the reduction in [32] for adaptive adversaries, but has two major differences. In fact, these two novel aspects also make our central proof technique rather unique, when compared with other works that use reductions from CC problems [12, 15, 24].

The first novel aspect is that we reduce from GDC with a special *leaker* that we design. The leaker is an oracle in the GDC problem, and is separate from the two parties Alice and Bob . It helps Alice and Bob, by disclosing to them certain "non-critical" information in the following way. For a CC problem Π, let $\Pi_n(X, Y)$ be the answer to Π for length-n inputs X and Y. Let x_i and y_i denote the i-th character of X and Y, respectively. A pair (a, b) is defined to be a *leakable pair* if for all n, X, Y, and $i \in [0, n]$, $\Pi_n(x_1x_2 \ldots x_n, y_1y_2 \ldots y_n) = \Pi_{n+1}(x_1x_2...x_iax_{i+1}x_{i+2}...x_n, y_1y_2...y_iby_{i+1}y_{i+2}...y_n)$. Intuitively, inserting or removing a leakable pair does not impact the answer to Π. For each index i where (x_i, y_i) is leakable, independently with probability $\frac{1}{2}$, our leaker *leaks* the index i, by letting both Alice and Bob know for free the value of i and the value of the pair (x_i, y_i), before Alice and Bob start running their protocol.

Our reduction from GDC (with our leaker) to CONSENSUS still does not allow us to directly use an oblivious adversary. Instead, as the second novel aspect, we will use a special kind of adaptive adversaries which we call *sanitized adaptive adversaries*. These adversaries are still adaptive, but their adaptive decisions have been "sanitized" by taking XOR with independent coin flips. We then show that a sanitized adaptive adversary is no more powerful than an oblivious adversary, in terms of incurring the cost of a protocol.

2 Related Work

This section discusses related works beyond those already covered in the previous section.

Related work on CONSENSUS and LEADERELECT. Given the importance of CONSENSUS and LEADERELECT in dynamic networks, there is a large body of related efforts and we can only cover the most relevant ones. In dynamic networks without congestion, Kuhn et al. [23] show that the *simultaneous consensus* problem has a lower bound of $\Omega(d + \text{poly}(m))$ round. In this problem, the nodes need to output their consensus decisions simultaneously. Their knowledge-based proof exploits the need for simultaneous actions, and does not apply to our setting. Some other researchers (e.g., [3, 4, 5]) have studied CONSENSUS and LEADERELECT in a dynamic network model where the set of nodes can change and where the topology is an *expander*. Their techniques (e.g., using random walks) critically reply on the expander property of the topology, and hence do not apply to our setting. Augustine et al. [2] have proved an upper bound of $O(d \log m)$ for LEADERELECT in dynamic networks while assuming d is known to all nodes. This does not contradict with our lower bound, since we do not assume the knowledge of d. Certain CONSENSUS and LEADERELECT protocols (e.g., [19]) assume that the network's topology eventually stops changing, which is different from our setting where the change does not stop. CONSENSUS and

LEADERELECT have also been studied in *directed* dynamic networks (e.g., [8, 14, 28, 29]), which are quite different from our undirected version. In particular, lower bounds there are mostly obtained by exploiting the lack of guaranteed bidirectional communication in directed graphs. Our AGGREGATION problem considers the two aggregation functions SUM and MAX. Cornejo et al. [13] considers a different aggregation problem where the goal is to collect distributed tokens (without combining them) to a small number of nodes. Some other research (e.g., [9]) on AGGREGATION assumes that the topology is each round is a (perfect) matching, which is different from our setting where the topology must be connected.

Related work on reductions from CC. Reducing from two-party CC problems to obtain lower bounds for distributed computing problem has been a popular approach in recent years. For example, Kuhn et al. [24] and Das Sarma et al.[15] have obtained lower bounds on the *hear-from* problem and the *spanning tree verification* problem, respectively, by reducing from DISJOINTNESS. In particular, Kuhn et al.'s results suggest that the *hear-from* problem has a lower bound of $\Omega(d + \sqrt{m}/\log m)$ in *directed static networks*. Chen et al.'s work [12] on computing SUM in *static networks with node failures* has used a reduction from the $\text{GDC}_n^{1,q}$ problem. Our reduction in this paper is unique, in the sense that none of these previous reductions use the two key novel techniques in this work, namely CC with our leaker and sanitized adaptive adversaries.

Related work on CC. To the best of our knowledge, we are the first to exploit the CC with a leaker in reductions to distributed computing problems such as CONSENSUS. Our leaker serves to allow oblivious adversaries. Quite interestingly, for completely different purposes, the notions of leakable pairs and a leaker have been extensively (but implicitly) used in proofs for obtaining direct sum results on the information complexity (IC) (e.g., [6, 10, 31]) of various communication problems: First, leakable pairs have been used to construct a *collapsing input*, for the purpose of ensuring that the answer to the problem Π is entirely determined by (x_i, y_i) at some index i. Second, an (implicit) leaker has often been used (e.g., in [10, 31]) to enable Alice and Bob to draw $(\mathbf{X}, \mathbf{Y})$ from a non-product distribution.

Because of the fundamentally different purposes of leaking, our leaker differs from those (implicit) leakers used in works on IC, in various specific aspects. For example in our work, all leakable pairs are subject to leaking, while in the works on IC, there is some index i that is never subject to leaking. Also, when our leaker leaks index j, it discloses both $\mathbf{x}_j$ and $\mathbf{y}_j$ to both Alice and Bob. In comparison, in works on IC, the (implicit) leaking is usually done differently: For example, Alice and Bob may use public coins to draw $\mathbf{x}_j$ and Bob may use his private coins to draw $\mathbf{y}_j$. Doing so (implicitly) discloses $\mathbf{x}_j$ to both Alice and Bob and (implicitly) discloses $\mathbf{y}_j$ *only* to Bob.

A key technical step in our work is to prove a lower bound on the CC of $\text{GDC}_n^{g,q}$ with our leaker. For simpler problems such as DISJOINTNESS (which is effectively $\text{GDC}_n^{1,2}$), we believe that such a lower bound could alternatively be obtained by studying its IC with our leaker. But the gap promise and the cycle promise in $\text{GDC}_n^{g,q}$ make IC arguments rather tricky. Hence we will (in Section 8) obtain our intended lower bound by doing a direct reduction from the CC of $\text{GDC}_{n'}^{g,q}$ without the leaker to the CC of $\text{GDC}_n^{g,q}$ with the leaker.

3 Model and Definitions

Conventions. All protocols in this paper refer to Monte Carlo randomized algorithms. We always consider public coin protocols, which makes our lower bounds stronger. All $\log$ is base 2, while $\ln$ is base e. Upper case fonts (e.g., X) denote strings, vectors, sets, etc. Lower case fonts (e.g., x) denote scalar values. In particular, if X is a string, then x_i means the i-th element in X. Bold fonts (e.g., $\mathbf{X}$ and $\mathbf{x}$) refer to random variables. Blackboard bold fonts (e.g., $\mathbb{D}$) denote distributions. We write $\mathbf{x} \sim \mathbb{D}$ if $\mathbf{x}$ follows the distribution $\mathbb{D}$. Script fonts (e.g., $\mathscr{P}$ and $\mathscr{Q}$) denote either protocols or adversaries.

Dynamic networks. We consider a synchronous dynamic network with m fixed nodes, each with a unique id of $\Theta(\log m)$ bits. A protocol in such a network proceeds in synchronous rounds, and starts executing on all nodes in round 1. (Clearly such simultaneous start makes our lower bound stronger.)

In each round, each node v first does some local computation, and then chooses to either send a single message of $O(\log m)$ size or receive. All nodes who are v's neighbors in that round and are receiving in that round will receive v's message at the end of the round. A node with multiple neighbors may receive multiple messages.

The topology of the network may change arbitrarily from round to round, as determined by some *adversary*, except that the topology in each round must be a connected undirected graph. (This is the same as the 1-interval model [22].) A node does not know the topology in a round. It does not know its neighbors either unless it receives messages from them in that round. Section 1 already defined *oblivious adversaries* and *adaptive adversaries*. In particular in each round, an adaptive adversary sees all $\mathscr{P}$'s coin flip outcomes up to and including the current round, and manipulates the topology accordingly, before $\mathscr{P}$ uses the current round's coin flip outcomes.

We use the standard definition for the *(dynamic) diameter* [25] of a dynamic network: Intuitively, the diameter of a dynamic network is the minimum number of rounds needed for every node to influence all other nodes. Formally, we say that $(\omega, r) \to (v, r+1)$ if either ω is v's neighbor in round r or $\omega = v$. The *diameter* d of a dynamic network is the smallest d such that $(\omega, r) \rightsquigarrow (v, r+d)$ for all ω, v, and r, where "$\rightsquigarrow$" is the transitive closure of "$\to$". Since the topology is controlled by an adversary, a protocol never knows d beforehand.

Communication complexity. In a two-party communication complexity (CC) problem Π_n, Alice and Bob each hold input strings X and Y respectively, where each string has n *characters*. A character here is q-ary (i.e., an integer in $[0, q-1]$) for some given integer $q \geq 2$. For any given i, we sometimes call (x_i, y_i) as a *pair*. Alice and Bob aim to compute the value of the binary function $\Pi_n(X, Y)$. Given a protocol $\mathscr{P}$ for solving Π (without a leaker), we define cc$(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}})$ to be the communication incurred (in terms of number of bits) by $\mathscr{P}$, under the input (X, Y) and $\mathscr{P}$'s coin flip outcomes $\mathbf{C}_{\mathscr{P}}$. Note that $\mathbf{C}_{\mathscr{P}}$ is a random variable while cc() is a deterministic function. We similarly define err$(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}})$, which is 1 if $\mathscr{P}$'s output is wrong, and 0 otherwise. We define the *communication complexity* of $\mathscr{P}$ to be cc$(\mathscr{P}) = \max_X \max_Y E_{\mathbf{C}_{\mathscr{P}}}[\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}})]$, and the *error* of $\mathscr{P}$ to be err$(\mathscr{P}) = \max_X \max_Y E_{\mathbf{C}_{\mathscr{P}}}[\text{err}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}})]$. We define the δ-error $(0 < \delta < \frac{1}{2})$ *communication complexity* of Π_n to be $\mathfrak{R}_\delta(\Pi_n) = \min \text{cc}(\mathscr{P})$, with the minimum taken over all $\mathscr{P}$ where err$(\mathscr{P}) \leq \delta$. For convenience, we define $\mathfrak{R}_\delta(\Pi_0) = 0$ and $\mathfrak{R}_\delta(\Pi_a) = \mathfrak{R}_\delta(\Pi_{\lfloor a \rfloor})$ for non-integer a.

Communication complexity with our leaker. We define similar concepts for CC with our leaker. Section 1 already defined leakable pairs and how our leaker works. Given $\mathscr{P}$ for solving Π with our leaker, cc$(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{L}})$ is the communication incurred by $\mathscr{P}$, under the input (X, Y), $\mathscr{P}$'s coin flip outcomes $\mathbf{C}_{\mathscr{P}}$, and the leaker's coin flip outcomes $\mathbf{C}_{\mathscr{L}}$. Here (X, Y) and $\mathbf{C}_{\mathscr{L}}$ uniquely determine which indices get leaked. We define cc$(\mathscr{P}) = \max_X \max_Y E_{\mathbf{C}_{\mathscr{L}}} E_{\mathbf{C}_{\mathscr{P}}}[\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{L}})]$. We similarly define err$(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{L}})$ and err$(\mathscr{P})$. Finally, we define the δ-error $(0 < \delta < \frac{1}{2})$ *communication complexity* of Π_n with our leaker, denoted as $\mathfrak{L}_\delta(\Pi_n)$, to be $\mathfrak{L}_\delta(\Pi_n) = \min \text{cc}(\mathscr{P})$, with the minimum taken over all $\mathscr{P}$ such that $\mathscr{P}$ solves Π_n with our leaker and err$(\mathscr{P}) \leq \delta$. Note that we always have $\mathfrak{L}_\delta(\Pi_n) \leq \mathfrak{R}_\delta(\Pi_n)$.

4 Preliminaries on Gap Disjointness with Cycle Promise

The section defines the two-party GDC problem and describes some basic properties of GDC.

Definition 1 (Gap Disjointness with Cycle Promise). *In* Gap Disjointness with Cycle Promise*, denoted as* $\text{GDC}_n^{g,q}$*, Alice and Bob have input strings X and Y, respectively. X and Y each have n characters, and each character is an integer in* $[0, q-1]$. *Alice and Bob aim to compute* $\text{GDC}_n^{g,q}(X, Y)$*, defined to be* 1 *if* (X, Y) *contains no* $(0, 0)$ *pair, and* 0 *otherwise. The problem comes with the following two promises:*

- **Gap promise:** (X, Y) *contains either no* $(0, 0)$ *pair or at least g such pairs.*

- **Cycle promise [12]:** *For each index i, x_i and y_i satisfy exactly one of the following four conditions: i) $x_i = y_i = 0$, ii) $x_i = y_i = q - 1$, iii) $x_i = y_i + 1$, or iv) $x_i = y_i - 1$.*

One can easily verify that the cycle promise is trivially satisfied when $q = 2$. It is also easy to see $\text{GD}_n^{1,2}$ degenerates to the classic DISJOINTNESS problem. The gap promise and the cycle promise start to impose material restrictions when $g \geq 2$ and $q \geq 3$, respectively. For example for $g = 2$ and $q = 4$, $X = 02103$ and $Y = 03003$ satisfy both the two promises, where (X, Y) contains 2 pairs of $(0, 0)$, at indices 1 and 4. The following result on the CC of GDC is a simple adaption from the result in [12]:

Theorem 1. *For any δ where $0 < \delta < 0.5$, there exist constants $c_1 > 0$ and $c_2 > 0$ such that for all n, g, and q, $\mathfrak{R}_\delta(\text{GDC}_n^{g,q}) \geq \frac{c_1 n}{gq^2} - c_2 \log \frac{n}{g}$.*

Proof. First, we show $\mathfrak{R}_\delta(\text{GDC}_{n/g}^{1,q}) \leq \mathfrak{R}_\delta(\text{GDC}_n^{g,q})$, via a simple reduction: Given an oracle protocol $\mathscr{P}$ for solving $\text{GDC}_n^{g,q}$, we construct a protocol $\mathscr{Q}$ for solving $\text{GDC}_{n/g}^{1,q}$. In $\mathscr{Q}$, Alice replicates her length-(n/g) input g times to get a length-n input. Bob does the same. Alice and Bob then invoke $\mathscr{P}$ and output $\mathscr{P}$'s output. It is easy to verify the correctness of this trivial reduction. Next, the theorem directly follows from an existing result from Chen et al. [12] showing that $\mathfrak{R}_\delta(\text{GDC}_{n/g}^{1,q}) \geq \frac{c_1 n}{gq^2} - c_2 \log \frac{n}{g}$. □

For GDC, all $(0, 0)$ pairs are non-leakable, while all other pairs are leakable. For example for $X = 02103$ and $Y = 03003$, those 3 pairs at index 2, 3, and 5 are leakable. The proof of Theorem 1 leveraged $\mathfrak{R}_\delta(\text{GDC}_n^{g,q}) \geq \mathfrak{R}_\delta(\text{GDC}_{n/g}^{1,q})$. It is important to note that $\mathfrak{L}_\delta(\text{GDC}_n^{g,q}) \geq \mathfrak{L}_\delta(\text{GDC}_{n/g}^{1,q})$ does *not* hold in general. (We omit a counter-example here due to space limitations.) In particular, the previous reduction fails for $\mathfrak{L}_\delta$: After Alice replicates her length-(n/g) input g times, the leaker (over the length-n input) may leak different parts in each of the g segments, and Alice cannot simulate such behavior. Hence when later proving the lower bound on $\mathfrak{L}_\delta(\text{GDC}_n^{g,q})$, we will have to work with the gap promise directly, instead of obtaining the lower bound via $\mathfrak{L}_\delta(\text{GDC}_{n/g}^{1,q})$.

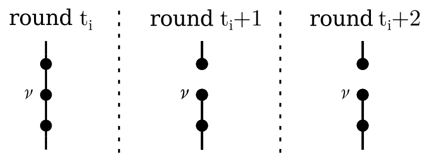
5 Review of Existing Proof under Adaptive Adversaries

This section gives an overview of the recent CONSENSUS lower bound proof [32] under adaptive adversaries. That proof is quite lengthy and involved, hence we will stay at the high-level, while focusing on aspects that are more relevant to this paper.

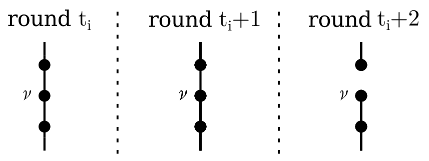
Overview. Consider any oracle CONSENSUS protocol $\mathscr{P}$ with $\frac{1}{10}$ error. Let $\text{tc}(d, m)$ be $\mathscr{P}$'s time complexity, when running over dynamic network controlled by adaptive adversaries and with d diameter and m nodes. The proof in [32] is mainly for proving $\text{tc}(8, m) = \Omega(\text{poly}(m))$. The proof trivially extends to $\text{tc}(d, m)$ for all $d \geq 8$. Combining with the trivial $\Omega(d)$ lower bound will lead to the final lower bound of $\Omega(d + \text{poly}(m))$.

To prove $\text{tc}(8, m) = \Omega(\text{poly}(m))$, [32] uses a reduction from $\text{GDC}_n^{g,q}$ to CONSENSUS. To solve $\text{GDC}_n^{g,q}(X, Y)$, Alice knowing X and Bob knowing Y simulate the CONSENSUS protocol $\mathscr{P}$ in the following way: In the simulation, the input (X, Y) is mapped to a dynamic network. Roughly speaking, if $\text{GDC}_n^{g,q}(X, Y) = 1$, the resulting dynamic network will have a diameter of 8. Hence $\mathscr{P}$ should decide within $r_1 = \text{tc}(8, m)$ rounds on expectation. If $\text{GDC}_n^{g,q}(X, Y) = 0$, then the resulting dynamic network will have a diameter of roughly $\frac{q}{2}$. It is then shown [32] that $\mathscr{P}$ must take $r_2 = \Omega(q)$ rounds to decide in dynamic networks with such a diameter. The value of q is chosen, as a function of $\text{tc}(8, m)$, such that $r_2 > 10r_1$. Alice and Bob determine the answer to GDC based on when $\mathscr{P}$ decides: If $\mathscr{P}$ decides within $10r_1$ rounds, they claim that $\text{GDC}_n^{g,q}(X, Y) = 1$. Otherwise they claim $\text{GDC}_n^{g,q}(X, Y) = 0$.

To solve GDC using the above simulation, Alice and Bob need to simulate $\mathscr{P}$ for $10r_1 = 10\text{tc}(8, m)$ rounds. In each round, to enable the simulation to continue, Alice and Bob will need to incur $O(\log m)$ bits of communication. Hence altogether, they incur $10\text{tc}(8, m) \cdot O(\log m)$ bits for solving $\text{GDC}_n^{g,q}$. The lower bound on the CC of $\text{GDC}_n^{g,q}$ then immediately translates to a lower bound on $\text{tc}(8, m)$.



(a) ν is sending in round $t_i + 1$



(b) ν is receiving in round $t_i + 1$

Figure 1: *The adaptive decisions of the adversary in* [32].

Crux of the proof. When solving GDC, Alice only knows X and not Y. This means that Alice does *not* actually have the full knowledge of the dynamic network, which is a function of (X, Y). Hence the proof's central difficulty is to design the dynamic network in such a way that Alice can nevertheless still properly simulate $\mathscr{P}$ over that dynamic network. The proof in [32] overcomes this key difficulty by i) leveraging the cycle promise in GDC, and ii) using an *adaptive* adversary — in particularly, using an adaptive adversary is highlighted [32] as a key technique. We give a concise review below.

Given (X, Y), the dynamic network constructed in [32] has one *chain* for each index $i \in [1, n]$. Each chain has 3 node in a line (Figure 1). Consider as an example the i-th chain where $x_i = 0$. Since $x_i = 0$, y_i must be either 0 or 1 (by the cycle promise). The set of edges on this chain will be different depending on whether y_i is 0 or 1 — this serves to make the diameter of the dynamic network different when GDC $= 1$ and when GDC $= 0$, as discussed earlier. The difficulty for Alice, is that she does not know y_i, and hence does not know the exact set of edges on this chain. This prevents her from properly simulating those nodes that she need to simulate for this chain. Similar difficulty applies to Bob.

To overcome this difficulty, if a pair (x_i, y_i) is not $(0, 0)$, the adversary in [32] will make an adaptive decision for manipulating the edges on the i-th chain,[5] to help enable Alice (and also Bob) to simulate. The cycle promise already tells us that for given x_i (e.g., 0), there are two possibilities for y_i (e.g., 0 and 1). The adaptive decisions of the adversary will have the following end effects: Under the topology resulted from such adaptive decisions, the behavior of those nodes that Alice needs to simulate will depend only on x_i and no longer depend on y_i. A similar property holds for Bob.

The details on why those adaptive decisions can achieve such end effects are complex, and are related to the fundamental fact that a node does not know its neighbors in a round until it receives messages from them. At the same time, those details are entirely orthogonal to this work. Hence due to space limitations, we refer interested readers to [32] for such details. Here we will only describe the specifics of all the adaptive decisions made by the adversary, which is needed for our later discussion: Consider any i where (x_i, y_i) is not $(0, 0)$. At the beginning of round $t_i + 1$ where t_i is some function of x_i and y_i, the adversary examines the coin flip outcomes of $\mathscr{P}$ and determines whether the middle node ν on the i-th chain is sending or receiving in round $t_i + 1$ (see Figure 1). If ν is sending, the adversary removes a certain edge e that is incidental to ν, immediately in round $t_i + 1$. Otherwise the adversary will remove the edge e in round $t_i + 2$. Except these adaptive decisions, the adversary does not make any other adaptive decisions. In particular, the adversary does not need to make adaptive decisions for chains corresponding to $(0, 0)$.

6 Roadmap for Lower Bound Proof under Oblivious Adversaries

This section provides the intuition behind, and the roadmap for, our novel proof of CONSENSUS lower bound under oblivious adversaries.

Some concepts. To facilitate discussion, we define a few simple concepts. Consider the i-th chain in the previous section where (x_i, y_i) is not $(0, 0)$, and the middle node ν on the chain. We define binary random variable $\mathbf{z} = 0$ if ν is sending in round $t_i + 1$, and define $\mathbf{z} = 1$ otherwise. We use $\mathscr{A}'$ to denote

[5]In the actual proof, the adversary only needs to make adaptive decisions for a subset (usually a constant fraction) of such chains. But it is much easier to understand if we simply let the adversary make an adaptive decision on all of them. Doing so has no impact on the asymptotic results.

the adaptive adversary described in the previous section. We define $\lambda_{\mathscr{A}'}$ to be the adaptive decision made by $\mathscr{A}'$, where $\mathscr{A}'$ removes the edge e in round $t_i + 1 + \lambda_{\mathscr{A}'}$. With these concepts, $\mathscr{A}'$ essentially sets its decision $\lambda_{\mathscr{A}'}$ to be $\lambda_{\mathscr{A}'} = \mathbf{z}$.

Making guesses. $\mathscr{A}'$ is adaptive since $\lambda_{\mathscr{A}'}$ depends on $\mathbf{z}$, and $\mathbf{z}$ in turn is a function of $\mathscr{P}$'s coin flips. An oblivious adversary $\mathscr{A}$ cannot have its decision $\lambda_{\mathscr{A}}$ depend on $\mathbf{z}$. At the highest level, our idea of allowing $\mathscr{A}$ in the reduction is simple: We let $\mathscr{A}$ make a blind guess on whether ν is sending. Specifically, imagine that $\mathscr{A}$ itself flips a fair coin $\mathbf{c}$, and then directly set its decision to be $\lambda_{\mathscr{A}} = \mathbf{c}$. Same as $\mathscr{A}'$, $\mathscr{A}$ still removes the edge e in round $t_i + 1 + \lambda_{\mathscr{A}}$, except that now $\lambda_{\mathscr{A}} = \mathbf{c}$. Some quick clarifications will help to avoid confusion here. First, such a guess $\mathbf{c}$ may be either correct (i.e., $\mathbf{c} = \mathbf{z}$) or wrong (i.e., $\mathbf{c} = \bar{\mathbf{z}}$). $\mathscr{A}$ itself cannot tell whether the guess is correct, since $\mathscr{A}$ (being oblivious) does not know $\mathbf{z}$. Alice and Bob, however, can tell if the guess is correct, because they are simulating both the protocol $\mathscr{P}$ and the adversary $\mathscr{A}$, and hence know both $\mathbf{z}$ and $\mathbf{c}$. But they cannot interfere with the guess even if they know it is wrong.

Now if the guess is correct, then the decision of $\mathscr{A}$ will be exactly the same as $\mathscr{A}'$, and everything will work out as before. But if the guess is wrong, then $\mathscr{A}$ can no longer enable Alice to simulate without knowing Y. More specifically, if the guess is wrong, then for the i-th chain, the behavior of those nodes that Alice needs to simulate will depend on the value of y_i, and Alice does not know y_i. To overcome this main obstacle, our key idea is to add a special *leaker* entity in the two-party CC problem, which should be viewed as an oracle that is separate from Alice and Bob. If the guess is wrong for the i-th chain, the leaker will disclose for free to Alice and Bob the pair (x_i, y_i). The knowledge of y_i then immediately enables Alice to infer the exact behavior of the nodes that she needs to simulate. Similar arguments apply to Bob.

Roadmap. There are two non-trivial technical issues remaining in the above approach: i) when to make guesses, and ii) how the leaker impacts the CC of GDC. Overcoming them will be the main tasks of Section 7 and 8, respectively. Section 9 will present our final CONSENSUS lower bound, whose lengthy and somewhat tedious proof is deferred to the appendix.

7 Sanitized Adaptive Adversaries

The difficulty. It turns out that it does not quite work for Alice and Bob to approach the leaker for help when they feel needed. Consider the following example $\text{GDC}_6^{2,4}$ instance with $X = 000000$ and $Y = 111100$. As explained in Section 5, the dynamic network corresponding to this instance has six chains. For all i, we say that the i-th chain is an "$|^a_b$ chain" if $x_i = a$ and $y_i = b$. The first four chains in the dynamic network are thus all $|^0_1$ chains, while the remaining two are $|^0_0$ chains. The adaptive adversary $\mathscr{A}'$ in [32] (see Section 5) will make adaptive decisions for all $|^0_1$ chains, but does not need to do so for $|^0_0$ chains. Applying the idea from Section 6, the oblivious adversary $\mathscr{A}$ should thus make guesses for those four $|^0_1$ chains. Note that $\mathscr{A}$ needs to be simulated by Alice and Bob. The difficulty is that Alice does not know for which chains a guess should be made, since she does not know which chains are $|^0_1$ chains. In fact if she knew, she would have already solved GDC in this instance. Similar arguments apply to Bob.

A naive fix is to simply make a guess for each of the six chains. Imagine now that the guess turns out to be wrong for the last chain, which is a $|^0_0$ chain. Alice and Bob will then ask the leaker to disclose (x_6, y_6). Such disclosure unfortunately directly reveals the answer to the GDC instance. This in turn, reduces the CC of GDC to 0, rendering the reduction meaningless. (Refusing to disclose (x_6, y_6) obviously does not work either, since the refusal itself reveals the answer.)

Our idea. To overcome this, we do not let Alice and Bob decide for which chains the adversary $\mathscr{A}$ should make a guess. Instead, we directly let our leaker decide which indices should be leaked: For every i where $(x_i, y_i) \neq (0, 0)$, the leaker leaks the pair (x_i, y_i) with half probability, to both Alice and Bob. In the earlier example, the leaker will leak each of the indices 1 through 4 independently with half probability.

For any given i, define binary random variable $\mathbf{s} = 1$ iff the leaker leaks index i. If $\mathbf{s} = 1$, then Alice and Bob will "fabricate" a wrong guess for the adversary $\mathscr{A}$ that they are simulating, so that the guess of $\mathscr{A}$ is wrong (and hence index i needs to be leaked). Specifically, Alice and Bob examine the coin flip outcomes of the protocol $\mathscr{P}$ to determine the value of $\mathbf{z}$, and then set the guess $\mathbf{c}$ of $\mathscr{A}$ to be $\mathbf{c} = \bar{\mathbf{z}}$. (Recall that $\mathbf{z}$ indicates whether the middle node is sending in round $t_i + 1$.) In such a case, the decision $\lambda_{\mathscr{A}}$ of $\mathscr{A}$ will be $\lambda_{\mathscr{A}} = \mathbf{c} = \bar{\mathbf{z}}$. On the other hand, if $\mathbf{s} = 0$ (meaning that index i is not leaked), then Alice and Bob let $\mathscr{A}$ behave exactly the same as the adaptive adversary $\mathscr{A}'$ in Section 5. In particular, if $\mathscr{A}'$ makes an adaptive decision $\lambda_{\mathscr{A}'} = \mathbf{z}$ for this chain, then the decision $\lambda_{\mathscr{A}}$ of $\mathscr{A}$ will also be $\lambda_{\mathscr{A}} = \mathbf{z}$ (i.e., as if $\mathscr{A}$ guessed correctly). Combining the two cases gives $\lambda_{\mathscr{A}} = \mathbf{z} \oplus \mathbf{s}$.

Obviously $\mathscr{A}$ here is no longer oblivious (since $\lambda_{\mathscr{A}}$ now depends on $\mathbf{z}$), which seems to defeat the whole purpose. Fortunately, this adaptive adversary $\mathscr{A}$ is special in the sense that all the adaptivity (i.e., $\mathbf{z}$) has been "sanitized" by taking XOR with the independent coin of $\mathbf{s}$. Intuitively, this prevents $\mathscr{A}$ from effectively adapting. The following discussion will formalize and prove that such an $\mathscr{A}$ is no more powerful than an oblivious adversary, in terms of incurring the cost of a protocol.

Formal framework and results. Without loss of generality, we assume that an adversary makes *binary decisions* that fully describe the behavior of the adversary. An adversary is *deterministic* if its decisions are fixed given the protocol's coin flip outcomes, otherwise it is *randomized*. Consider any deterministic adaptive adversary $\mathscr{A}'$. A decision $\lambda_{\mathscr{A}'}$ made by $\mathscr{A}'$ is called *adaptive* if $\lambda_{\mathscr{A}'}$ can be different under different coin flip outcomes of the protocol. A randomized adaptive adversary $\mathscr{A}$ is called a *sanitized version* of $\mathscr{A}'$, if $\mathscr{A}$ behaves the same as $\mathscr{A}'$ except that $\mathscr{A}$ *sanitizes* all adaptive decisions made by $\mathscr{A}'$ and also an arbitrary (possibly empty) subset of the non-adaptive decisions made by $\mathscr{A}'$. Here $\mathscr{A}$ *sanitizes* a decision $\lambda_{\mathscr{A}'}$ made by $\mathscr{A}'$ by setting its own decision $\lambda_{\mathscr{A}}$ to be $\lambda_{\mathscr{A}} = \lambda_{\mathscr{A}'} \oplus \mathbf{s}$, where $\mathbf{s}$ is a separate fair coin and is independent of all other coins. We also call the above $\mathscr{A}$ as a *sanitized adaptive adversary*. In our discussion above, $\lambda_{\mathscr{A}'} = \mathbf{z}$, while $\lambda_{\mathscr{A}} = \mathbf{z} \oplus \mathbf{s} = \lambda_{\mathscr{A}'} \oplus \mathbf{s}$. The following simple theorem confirms that $\mathscr{A}$ is no more powerful than an oblivious adversary (see proof in the appendix):

Theorem 2. *Let* $\mathrm{cost}(\mathscr{P}, \mathscr{A}, C_{\mathscr{P}}, C_{\mathscr{A}})$ *be any deterministic function (which the adversary aims to maximize) of the protocol $\mathscr{P}$, the adversary $\mathscr{A}$, the coin flip outcomes $C_{\mathscr{P}}$ of $\mathscr{P}$, and the coin flip outcomes $C_{\mathscr{A}}$ (if any) that may also influence the behavior of $\mathscr{A}$. For any protocol $\mathscr{P}$, any deterministic adaptive adversary $\mathscr{A}'$, and its sanitized version $\mathscr{A}$, there exists a deterministic oblivious adversary $\mathscr{B}$ such that* $E_{\mathbf{C}_{\mathscr{P}}}[\mathrm{cost}(\mathscr{P}, \mathscr{B}, \mathbf{C}_{\mathscr{P}}, -)] \geq E_{\mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{A}}}[\mathrm{cost}(\mathscr{P}, \mathscr{A}, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{A}})]$. *Furthermore, for every $C_{\mathscr{P}}$ in the support of $\mathbf{C}_{\mathscr{P}}$, there exists $C_{\mathscr{A}}$ in the support of $\mathbf{C}_{\mathscr{A}}$, such that $\mathscr{B}$'s decisions are exactly the same as the decisions made by $\mathscr{A}$ under $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$.*

Summary of this section. Recall that $\mathscr{A}'$ denotes the adaptive adversary used in [32] and reviewed in Section 5. Based on the discussion in this section, our reduction from GDC (with a leaker) to CONSENSUS will use a sanitized adaptive adversary $\mathscr{A}$ for the dynamic network. $\mathscr{A}$ behaves exactly the same as $\mathscr{A}'$ except: For each i-th chain where $\mathscr{A}'$ makes an adaptive decision $\lambda_{\mathscr{A}'}$ for that chain, $\mathscr{A}$ sets its own decision $\lambda_{\mathscr{A}}$ for that chain to be $\lambda_{\mathscr{A}} = \lambda_{\mathscr{A}'} \oplus \mathbf{s}$. Here $\mathbf{s}$ denotes whether index i is leaked by the leaker. Theorem 2 confirms that the consensus protocol $\mathscr{P}$'s end guarantees, even though $\mathscr{P}$ was designed to work against oblivious adversaries instead of adaptive adversaries, will continue to hold under $\mathscr{A}$.

8 Communication Complexity with The Leaker

To get our final CONSENSUS lower bound , the next key step is to prove a lower bound on the CC of GDC with the leaker. At first thought, one may think that having the leaker will not affect the CC of GDC much, since i) the leakable pairs have no impact on the answer to the problem and are hence "dummy" parts, and ii) the leaker only leaks about half of such "dummy" parts. As a perhaps surprising example, Lemma 1 in the appendix shows that having the leaker reduces the CC of $\mathrm{GDC}_n^{16\sqrt{n}\ln\frac{1}{\delta},2}$ from $\Omega(\sqrt{n})$ to

0. This implies that the impact of the leaker is more subtle than expected. In particular, without a careful investigation, it is not even clear whether the CC of GDC with our leaker is large enough to translate to our intended $\Omega(d + \text{poly}(m))$ lower bound on CONSENSUS.

This section will thus do a careful investigation and eventually establish a formal connection between the CC with the leaker ($\mathfrak{L}_\delta$) and the CC without the leaker ($\mathfrak{R}_\delta$):

Theorem 3. *For any constant $\delta \in (0, \frac{1}{2})$, there exist constants $c_1 > 0$ and $c_2 > 0$ such that for all n, g, q, and $n' = c_2\sqrt{n}/(q^{1.5}\log q)$, $\mathfrak{L}_\delta(\text{GDC}_n^{g,q}) \geq c_1\mathfrak{R}_\delta(\text{GDC}_{n'}^{g,q})$.*

Later we will see that the lower bound on GDC with our leaker as obtained in the above theorem (combined with Theorem 1) is sufficient for us to get a final $\Omega(d + \text{poly}(m))$ lower bound on CONSENSUS. The theorem actually also holds for many other problems beyond GDC, though we do not present the general form here due to space limitations.

8.1 Our Approach and Key Ideas

While we will only need to prove Theorem 3 for GDC, we will consider general two-party problem Π, since the specifics of GDC are not needed here. We will prove Theorem 3 via a reduction: We will construct a protocol $\mathscr{Q}$ for solving $\Pi_{n'}$ without the leaker, by using an oracle protocol $\mathscr{P}$ for solving Π_n with the leaker, where n' is some value that is smaller than n. Such a reduction will then lead to $\mathfrak{R}_{\delta'}(\Pi_{n'}) = O(\mathfrak{L}_\delta(\Pi_n))$.

We will call each kind of leakable pairs as a *leakable pattern*. For example, $\text{GDC}_n^{1,2}$ has leakable patterns of $(1,1)$, $(0,1)$, and $(1,0)$. Note that leakable patterns are determined by the problem Π and not by an instance of the problem. We use $k \in [0, q^2]$ to denote the total number of leakable patterns for Π whose inputs are q-ary strings. For $\text{GDC}_n^{g,q}$, $k = 2q - 1$.

Simulating the leaker via padded pairs. The central difficulty in the reduction is that Alice and Bob running $\mathscr{Q}$ need to simulate the leaker, in order to invoke the oracle protocol $\mathscr{P}$. (Note that $\mathscr{P}$ here is the two-party protocol, and has nothing to do with the CONSENSUS protocol.) This is difficult because each party only knows her/his own input. Our first step to overcome this difficulty is to pad known characters to the inputs and then leak *only* those padded characters, as explained next.

Let (X', Y') be the given input to $\mathscr{Q}$. Assume for simplicity that $(2, 1)$ is the only leakable pattern in Π, and consider the problem instance in Figure 2 where $X' = 02$ and $Y' = 01$. Alice and Bob will append/pad a certain number of occurrences of each leakable pattern to (X', Y'). Let (X, Y) denote the resulting strings after the padding. In the example in Figure 2, Alice and Bob append 1 occurrence of $(2, 1)$ to (X', Y') — or more specifically, Alice appends 2 to X' and Bob appends 1 to Y'. Doing so gives $X = 022$ and $Y = 011$. Note that doing so does not involve any communication, since the leakable patterns are publicly known. Imagine that Alice and Bob now invoke $\mathscr{P}$ using (X, Y), where $X = 022$ and $Y = 011$. Note that the two-party protocol $\mathscr{P}$ assumes the help from our leaker. Alice and Bob can easily simulate the leaking of (x_3, y_3), since (x_3, y_3) is the padded pair and they both know that the pair is exactly $(2, 1)$. However, (x_2, y_2) is also a leakable pair. Alice and Bob still cannot simulate the leaking of this pair, since this pair originated from (X', Y') and they do not know the value of this pair.

To overcome this, Alice and Bob use public coins to generate a random permutation, and then use the permutation to permute X and Y, respectively (Figure 2). This step does not involve communication. For certain problems Π (e.g., for GDC), one can easily verify that such permutation will not affect the answer to Π. Such permutation produces an interesting effect, as illustrated in Figure 2. The upper part of Figure 2 plots the 6 possible outcomes after the permutation, for our earlier example of $X = 022$ and $Y = 011$. Before the permutation, the last pair in (X, Y) is a padded pair. Imagine that Alice and Bob leak this pair. Now after the permutation, this leaked pair will occupy different indices in the 6 outcomes of the permutation.

The bottom part of Figure 2 illustrates the (real) leaker's behavior over certain inputs. To help understanding, assume here for simplicity that the leaker leaks *exactly* half of all the leakable pairs. Now

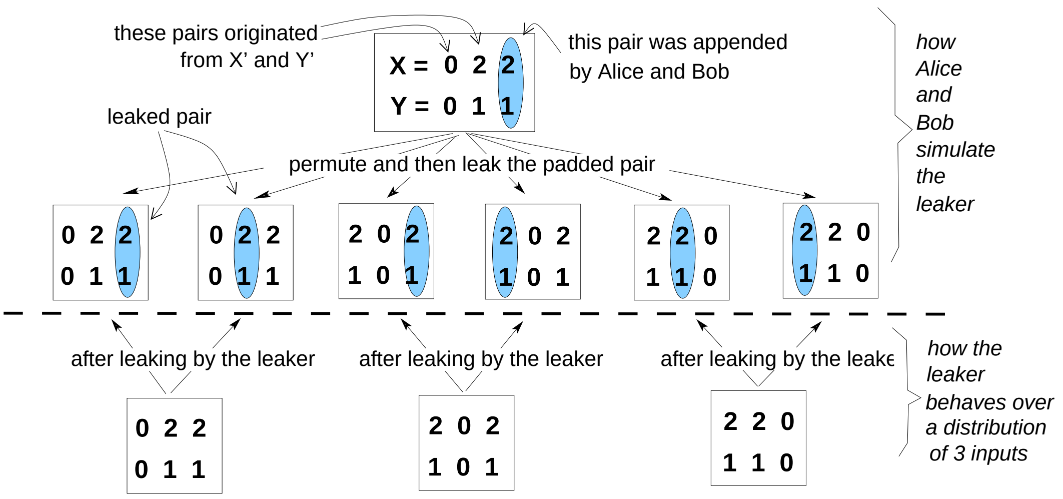

these pairs originated from X' and Y'
X = 0 2 2
Y = 0 1 1
this pair was appended by Alice and Bob
how Alice and Bob simulate the leaker
leaked pair
permute and then leak the padded pair
0 2 2 / 0 1 1
0 2 2 / 0 1 1
2 0 2 / 1 0 1
2 0 2 / 1 0 1
2 2 0 / 1 1 0
2 2 0 / 1 1 0
after leaking by the leaker
after leaking by the leaker
after leaking by the leake
how the leaker behaves over a distribution of 3 inputs
0 2 2 / 0 1 1
2 0 2 / 1 0 1
2 2 0 / 1 1 0

Figure 2: *How padding and permutation enable Alice and Bob to simulate the leaker. In this example $X' = 02$, $Y' = 01$, $X = 022$, and $Y = 011$. Here to help understanding, we assume that the leaker leaks* exactly *half of all the leakable pairs.*

consider 3 different inputs $(022, 011)$, $(202, 101)$, and $(220, 110)$. One can see that the behavior of the leaker over these 3 inputs (see Figure 2) exactly matches the result of permutation as done by Alice and Bob. Hence when Alice and Bob feed the result of the permutation into $\mathscr{P}$ while leaking the padded pair, it is as if $\mathscr{P}$ were invoked over the previous 3 inputs (each chosen with $1/3$ probability) together with the real leaker. This means that $\mathscr{P}$'s correctness and CC guarantees should continue to hold, when Alice and Bob invoke $\mathscr{P}$ while leaking only the padded pair.

How many pairs to leak. Imagine that (X', Y') contain o pairs of $(2, 1)$, and Alice and Bob pad p pairs of $(2, 1)$ to (X', Y'). The result of the padding, (X, Y), will contain $o + p$ pairs of $(2, 1)$. Let $\mathbf{f}$ be the number of $(2, 1)$ pairs in (X, Y) that should be leaked, which obviously follows a binomial distribution with a mean of $\frac{o+p}{2}$. Ideally, Alice and Bob should draw $\mathbf{f}$ from the above binomial distribution, and then simulate the leaking of $\mathbf{f}$ pairs of $(2, 1)$. (They can do so as long as $\mathbf{f} \le p$ — with proper p, we easily throw $\Pr[\mathbf{f} > p]$ into the error.) The difficulty, however, is that Alice and Bob do not know o, and hence cannot draw $\mathbf{f}$ with the correct mean of $\frac{o+p}{2}$.

To overcome this, Alice and Bob will estimate the value of o by sampling: For each sample, they use public coin to choose a uniformly random $i \in [1, n']$, and then send each other the values of x'_i and y'_i. They will spend total $\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2}$ bits for doing this, so that such sampling is effectively "free" and does not impact the asymptotic quality of the reduction. Alice and Bob will nevertheless still not obtain the exact value of o. This means that the distribution they use to draw $\mathbf{f}$ will be different from the distribution that the (real) leaker uses. Our formal proof will carefully take into account such discrepancy.

8.2 Formal Reduction and Final Guarantees

Pseudo-code. Protocol 1 presents the protocol $\mathscr{Q}$ for solving $\Pi_{n'}$ without our leaker, as run by Alice. $\mathscr{Q}$ internally invokes the oracle two-party protocol $\mathscr{P}$, where $\mathscr{P}$ solves Π_n with our leaker. At Line 1–6, Alice and Bob first exchange sampled indices to estimate the occurrences of each leakable pattern. Next Line 7–9 calculate the amount of padding needed. Line 10–15 do the actual padding, and then for each leakable pattern, flag a certain number of padded pairs as "to be leaked". At Line 16–20, Alice and Bob do a random permutation to obtain $(\mathbf{X}, \mathbf{Y})$, and then invoke $\mathscr{P}$ on $(\mathbf{X}, \mathbf{Y})$ while leaking all those flagged pairs.

Final properties of $\mathscr{Q}$. The appendix will prove that $\mathscr{Q}$ solves Π without our leaker, with an error of $\delta + \frac{11}{12}(\delta' - \delta)$, while incurring $\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + 5.5\text{cc}(\mathscr{P})$ bits of communication. This will eventually lead to

```
   Input: $X'$, $n$, $n'$, $\delta$, $\delta'$, where $\delta < \delta'$
   /*** Line 1–6 use $s$ samples to estimate the occurrences of each leakable pattern in $(X', Y')$. ***/
1  $s \leftarrow \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{4\log q}$;  foreach $j = 1, \ldots, k$ do  $\mathbf{v}_j \leftarrow 0$ ;
2  repeat $s$ times do
3      draw a uniformly random integer $i \in [1, n']$ using public coins;
4      send $x'_i$ to Bob and receive $y'_i$ from Bob ;
5      foreach $j = 1, \ldots, k$ do if $(x'_i, y'_i)$ equals the $j$-th leakable pattern then $\mathbf{v}_j \leftarrow \mathbf{v}_j + \frac{n'}{s}$;
6  end
   /*** Line 7–15 pad leakable pairs, and then flag some of them as "to be leaked". ***/
   /*** Here $h_j$ is the number of times that the $j$-th leakable pattern is padded to $(X', Y')$. ***/
7  $h \leftarrow 2n' + \frac{500}{(\delta'-\delta)^2}(k^2 + \frac{kn'^2}{2s} \ln \frac{24k}{\delta'-\delta})$;
8  foreach $j = 1, \ldots, k-1$ do  $h_j \leftarrow h$ ;
9  $h_k \leftarrow n - n' - (k-1)h$;  if $h_k < h$ then generate an arbitrary output and exit;
10 foreach $j = 1, \ldots, k$ do
11     draw an integer $\mathbf{b}_j$ from the binomial distribution $\mathbb{B}(\frac{h_j + \mathbf{v}_j}{2})$ using public coins ;
       // $\mathbb{B}(\mu)$ is the distribution for the number of heads obtained when flipping $2\mu$ fair coins.
12     if $\mathbf{b}_j > h_j$ then $\mathbf{b}_j \leftarrow h_j$;
13     let $(a, b)$ be the $j$-th leakable pattern ;
14     append $h_j$ copies of $a$ to $X'$, and flag the first $\mathbf{b}_j$ indices of these $h_j$ indices as "to be leaked";
15 end
   /*** Line 16–17 randomly permute the input. ***/
16 generate a uniformly random permutation $\mathbf{M}$ using public coins;
17 $\mathbf{X} \leftarrow \mathbf{M}(X')$ ;
   /* the flags in $X'$ will be treated as part of $X'$ and be permuted as well. */;
   /*** Line 18–20 invoke $\mathscr{P}$. ***/
18 invoke $\mathscr{P}$ (together with the other party) using $\mathbf{X}$ as input, while leaking all those indices that are flagged,
     until either $\mathscr{P}$ outputs or $\mathscr{P}$ has incurred $(\frac{6}{\delta'-\delta})\mathrm{cc}(\mathscr{P})$ bits of communication ;
   /* when leaking index $i$, both $x'_i$ and $y'_i$ will be given to $\mathscr{P}$ — this can be done since a leaked index here
     must correspond to a padded pair at Line 14 */;
19 if $\mathscr{P}$ has incurred $(\frac{6}{\delta'-\delta})cc(\mathscr{P})$ bits of communication then  exit with an arbitrary output ;
20 else  output $\mathscr{P}$'s output and exit ;
```

Protocol 1: *Our δ'-error protocol $\mathscr{Q}$ for solving $\Pi_{n'}$ without our leaker. $\mathscr{Q}$ invokes the δ-error oracle two-party protocol $\mathscr{P}$ that solves Π_n with our leaker. The above only shows Alice's part of $\mathscr{Q}$. Bob's part of $\mathscr{Q}$ can be obtained similarly.*

Theorem 3 (see proof in the appendix).

9 CONSENSUS Lower Bound under Oblivious Adversaries

Following is our final theorem on CONSENSUS under oblivious adversaries:

Theorem 4. *If the nodes only know a poor estimate m' for m such that $|\frac{m'-m}{m}|$ reaches $\frac{1}{3}$ or above, then a $\frac{1}{10}$-error* CONSENSUS *protocol for dynamic networks with oblivious adversaries must have a time complexity of $\Omega(d + m^{\frac{1}{12}})$ rounds.*

Our proof under oblivious adversaries partly builds upon the previous proof under adaptive adversaries [32], as reviewed in Section 5. The key difference is that we reduce from GDC *with our leaker* to CONSENSUS. The full proof is lengthy and tedious as it needs to build upon the lengthy proof in [32]. Since Section 7 and 8 already discussed the key differences between our proof and [32], we leave the full proof to the appendix, and only provide an overview here on how to put the pieces from Section 7 and 8 together.

Consider any oracle CONSENSUS protocol $\mathscr{P}$ with $\frac{1}{10}$ error. Let $\text{tc}(d, m)$ denote $\mathscr{P}$'s time complexity when running over dynamic networks controlled by oblivious adversaries and with d diameter and m nodes. As explained in Section 5, the crux will be to prove $\text{tc}(8, m) \geq m^{\frac{1}{12}}$. To do so, we consider $\text{GDC}_n^{g,q}$ with $n = \frac{m-4}{3}$, $q = 20\text{tc}(8, m) + 20$, and $g = 15q \ln q$. To solve $\text{GDC}_n^{g,q}(X, Y)$, Alice and Bob simulate $\mathscr{P}$ in the following way: In the simulation, the input (X, Y) is mapped to a *sanitized* adaptive adversary $\mathscr{A}$ that determines the topology of the dynamic network. Roughly speaking, if $\text{GDC}_n^{g,q}(X, Y) = 1$, the resulting dynamic network will have a diameter of 8. Even though $\mathscr{A}$ is an adaptive adversary, by Theorem 2 in Section 7, $\mathscr{P}$'s time complexity should remain $\text{tc}(d, m)$ under $\mathscr{A}$. Hence $\mathscr{P}$ should decide within $\text{tc}(8, m)$ rounds on expectation. If $\text{GDC}_n^{g,q}(X, Y) = 0$, then the resulting dynamic network will have a diameter of $\Theta(q)$. For $\mathscr{P}$ to decide in this dynamic network, we prove that it takes at least roughly $\frac{q}{2}$ rounds. Note that $\frac{q}{2} > 10\text{tc}(8, m)$ — in other words, it takes longer for $\mathscr{P}$ to decide if $\text{GDC}_n^{g,q}(X, Y) = 0$. Alice and Bob do not know the other party's input, and hence does not have full knowledge of the dynamic network. But techniques from [32], together with the help from our leaker, enable them to still properly simulate $\mathscr{P}$'s execution. Finally, if $\mathscr{P}$ decides within $10\text{tc}(8, m)$ rounds, Alice and Bob claim that $\text{GDC}_n^{g,q}(X, Y) = 1$. Otherwise they claim $\text{GDC}_n^{g,q}(X, Y) = 0$. Our proof will show that to solve $\text{GDC}_n^{g,q}$ with our leaker, using the above simulation, Alice and Bob incur $\Theta(\text{tc}(8, m) \cdot \log n)$ bits of communication. We thus have $\Theta(\text{tc}(8, m) \log n) \geq \mathfrak{L}_\delta(\text{GDC}_n^{g,q})$. Together with the lower bound on $\mathfrak{L}_\delta(\text{GDC}_n^{g,q})$ from Theorem 3 in Section 8 (and Theorem 1 in Section 4), this will lead to a lower bound on $\text{tc}(8, m)$.

References

[1] J. Augustine, C. Avin, M. Liaee, G. Pandurangan, and R. Rajaraman. Information spreading in dynamic networks under oblivious adversaries. In *DISC*, 2016.

[2] J. Augustine, T. Kulkarni, P. Nakhe, and P. Robinson. Robust leader election in a fast-changing world. In *Workshop on Foundations of Mobile Computing*, 2013.

[3] J. Augustine, G. Pandurangan, and P. Robinson. Fast byzantine leader election in dynamic networks. In *DISC*, October 2015.

[4] J. Augustine, G. Pandurangan, P. Robinson, and E. Upfal. Towards robust and efficient computation in dynamic peer-to-peer networks. In *SODA*, 2012.

[5] John Augustine, Gopal Pandurangan, and Peter Robinson. Fast byzantine agreement in dynamic networks. In *PODC*, July 2013.

[6] Z. Bar-Yossef, T. S. Jayram, R. Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, June 2004.

[7] N. Batir. Sharp inequalities for factorial n. *Proyecciones*, 27(1):97–102, 2008.

[8] M. Biely, P. Robinson, and U. Schmid. Agreement in directed dynamic networks. In *SIROCCO*, July 2012.

[9] Q. Bramas, T. Masuzawa, and S. Tixeuil. Distributed online data aggregation in dynamic graphs. In *ICDCS*, 2016.

[10] M. Braverman. Interactive information complexity. *SIAM Journal on Computing*, 44(6):1698–1739, 2015.

[11] K. Censor-Hillel, E. Haramaty, and Z. Karnin. Optimal dynamic distributed MIS. In *PODC*, 2016.

[12] Binbin Chen, Haifeng Yu, Yuda Zhao, and Phillip B. Gibbons. The cost of fault tolerance in multi-party communication complexity. *Journal of the ACM*, 61(3), May 2014.

[13] Alejandro Cornejo, Seth Gilbert, and Calvin Newport. Aggregation in dynamic networks. In *PODC*, July 2012.

[14] E. Coulouma, E. Godard, and J. Peters. A characterization of oblivious message adversaries for which consensus is solvable. *Theoretical Computer Science*, 584:80–90, June 2015.

[15] A. Das Sarma, S. Holzer, L. Kor, A. Korman, D. Nanongkai, G. Pandurangan, D. Peleg, and R. Wattenhofer. Distributed verification and hardness of distributed approximation. In *STOC*, 2011.

[16] C. Dutta, G. Pandurangan, R. Rajaraman, Z. Sun, and E. Viola. On the complexity of information spreading in dynamic networks. In *SODA*, January 2013.

[17] Mohsen Ghaffari, Nancy Lynch, and Calvin Newport. The cost of radio network broadcast for different models of unreliable links. In *PODC*, July 2013.

[18] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[19] R. Ingram, P. Shields, and J. Walter. An asynchronous leader election algorithm for dynamic networks. In *IPDPS*, 2009.

[20] M. Konig and R. Wattenhofer. On local fixing. In *OPODIS*, 2013.

[21] Fabian Kuhn, Nancy Lynch, Calvin Newport, Rotem Oshman, and Andrea Richa. Broadcasting in unreliable radio networks. In *PODC*, July 2010.

[22] Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *STOC*, June 2010.

[23] Fabian Kuhn, Yoram Moses, and Rotem Oshman. Coordinated consensus in dynamic networks. In *PODC*, June 2011.

[24] Fabian Kuhn and Rotem Oshman. The complexity of data aggregation in directed networks. In *DISC*, September 2011.

[25] Fabian Kuhn and Rotem Oshman. Dynamic networks: Models and algorithms. *SIGACT News*, 42(1):82–96, March 2011.

[26] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1996.

[27] F. Liese and I. Vajda. *Convex Statistical Distances*. Leipzig: Teubner, 1987.

[28] U. Schmid, B. Weiss, and I. Keidar. Impossibility results and lower bounds for consensus under link failures. *SIAM Journal on Computing*, 38(5):1912–1951, 2009.

[29] M. Schwarz, K. Winkler, and U. Schmid. Fast consensus under eventually stabilizing message adversaries. In *ICDCN*, January 2016.

[30] A. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

[31] O. Weinstein. Information complexity and the quest for interactive compression. *SIGACT News*, 46(2):41–64, June 2015.

[32] H. Yu, Y. Zhao, and I. Jahja. The cost of unknown diameter in dynamic networks. In *SPAA*, July 2016. (Journal version currently under review).

A Key Notations

Table 1 summarizes the key notations in this paper.

Table 1: Key notations

Π	A two-party communication complexity problem
X	Alice's input string for Π
Y	Bob's input string for Π
n	Number of characters in X and Y
q	Each character in X and Y is an integer between 0 and $q-1$, inclusively
k	The number of leakable patterns
g	The minimum number of $\vert_0^0$ patterns that will appear in (X, Y) if $\textsc{Gdc}_n^{g,q}(X, Y) = 0$
$\mathbf{C}_{\mathscr{L}}$	Coin flip outcomes of the leaker
$\mathscr{P}, \mathscr{Q}$	Protocols
$\mathscr{A}, \mathscr{B}$	Adversaries
$\mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{Q}}$	Coin flip outcomes of the protocol $\mathscr{P}$, $\mathscr{Q}$
$\mathbf{C}_{\mathscr{A}}, \mathbf{C}_{\mathscr{B}}$	Coin flip outcomes of the adversary $\mathscr{A}$, $\mathscr{B}$
$\text{cc}(\mathscr{P})$	The communication complexity of $\mathscr{P}$
$\text{err}(\mathscr{P})$	The error of $\mathscr{P}$
$\mathfrak{R}_\delta(\Pi)$	The δ-error communication complexity of problem Π
$\mathfrak{L}_\delta(\Pi)$	The δ-error communication complexity of problem Π with our leaker
$\vert_b^a(X, Y)$	The number of occurrences of the $\vert_b^a$ pattern in (X, Y)
m	Number of nodes in the dynamic network
d	(Dynamic) diameter of the dynamic network
$\alpha, \beta, \gamma, \lambda$	Special nodes in the dynamic network constructed by the reference adversary
$\tau, \upsilon, \nu, \omega$	Generic nodes in a dynamic network
$\mathbb{B}(\mu)$	Binomial distribution corresponding to the number of heads in 2μ fair coin flips
$\mathbb{N}(\mu)$	Normal distribution with mean μ and variance $\frac{\mu}{2}$
$\vec{\mathbb{N}}(\vec{\mu}, \Sigma)$	k-variate normal distribution with mean vector $\vec{\mu}$ and covariance matrix Σ
$\vert\vert\mathbb{D} - \mathbb{D}'\vert\vert$	L_1 distance between two distributions $\mathbb{D}$ and $\mathbb{D}'$
$f_{\mathbb{D}}$	Probability density function of distribution $\mathbb{D}$

B Lemma 1

Lemma 1. *For all* $0 < \delta < \frac{1}{2}$, *there exists constant constant* $c \geq 0$ *such that* $\mathfrak{R}_\delta(\textsc{Gdc}_n^{16\sqrt{n}\ln\frac{1}{\delta},2}) \geq c\sqrt{n}/(16\ln\frac{1}{\delta})$ *and* $\mathfrak{L}_\delta(\textsc{Gdc}_n^{16\sqrt{n}\ln\frac{1}{\delta},2}) = 0$ *for all* $n \geq 1$.

Proof. We first show that for all $0 < \delta < \frac{1}{2}$, there exists constant $c \geq 0$ such that $\mathfrak{R}_\delta(\textsc{Gdc}_n^{16\sqrt{n}\ln\frac{1}{\delta},2}) \geq c\sqrt{n}/(16\ln\frac{1}{\delta})$ for all $n \geq 1$. $\textsc{Gdc}_n^{1,2}$ is the same as the Disjointness problem. Hence from well-known results on Disjointness [26], for all $0 < \delta < \frac{1}{2}$, we can find positive constant c such that $\mathfrak{R}_\delta(\textsc{Gdc}_n^{1,2}) \geq cn$ for all $n \geq 1$. One can trivially reduce from $\textsc{Gdc}_n^{1,2}$ to

$\text{GDC}^{g,2}_{gn}$, by Alice and Bob replicating their respective input strings by g times. This immediately gives $\mathfrak{R}_\delta(\text{GDC}^{1,2}_n) \le \mathfrak{R}_\delta(\text{GDC}^{g,2}_{gn})$. Hence for all n where $n \ge 16\sqrt{n}\ln\frac{1}{\delta}$:

$$\mathfrak{R}_\delta(\text{GDC}_n^{16\sqrt{n}\ln\frac{1}{\delta},2}) \ge \mathfrak{R}_\delta(\text{GDC}^{1,2}_{n/(16\sqrt{n}\ln\frac{1}{\delta})}) \ge c\cdot n/(16\sqrt{n}\ln\frac{1}{\delta})) = (c/(16\ln\frac{1}{\delta}))\cdot\sqrt{n}$$

Next, we need to show that for all $0 < \delta < \frac{1}{2}$, $\mathfrak{L}_\delta(\text{GDC}_n^{16\sqrt{n}\ln\frac{1}{\delta},2}) = 0$ for all $n \ge 1$. Define $g = 16\sqrt{n}\ln\frac{1}{\delta}$, and we will prove $\mathfrak{L}_\delta(\text{GDC}^{g,2}_n) = 0$.

If $g < (16\ln\frac{1}{\delta})^2$ and $n = g^2/(16\ln\frac{1}{\delta})^2$, then $n < g$ and hence the only possible answer for $\text{GDC}^{g,2}_n$ is 1. This means that Alice and Bob can trivially solve the problem without any communication. Next if $g \ge (16\ln\frac{1}{\delta})^2$ and $n = g^2/(16\ln\frac{1}{\delta})^2$, we construct the following protocol for solving $\text{GDC}^{g,2}_n$ with our leaker: Alice and Bob output 0 if the total number of leaked indices is at least $\frac{n}{2} - 2\sqrt{n}\ln\frac{1}{\delta}$, and 1 otherwise. We next show that this protocol has at most δ error.

Let random variable $\mathbf{z}$ denote the number of leaked indices. If the answer to the $\text{GDC}^{g,2}_n$ problem is 1, then $\mathbf{z}$ is the number of heads obtained when flipping n independent fair coins. The protocol's error probability is:

$$\begin{aligned}
\Pr[\mathbf{z} < \frac{n}{2} - 2\sqrt{n}\ln\frac{1}{\delta}] &= \Pr[\mathbf{z} < (1 - \frac{4\ln\frac{1}{\delta}}{\sqrt{n}})\cdot\frac{n}{2}] \\
&\le exp(-\frac{1}{2}\cdot\frac{n}{2}\cdot\frac{16\ln^2\frac{1}{\delta}}{n}) < \delta \quad \text{(by Chernoff bound)}
\end{aligned}$$

If the answer to the $\text{GDC}^{g,2}_n$ problem is 0, then $\mathbf{z}$ is the number of heads obtained when flipping $n - b$ independent fair coins where $b \ge g = 16\sqrt{n}\ln\frac{1}{\delta}$. Let $\mathbf{z}'$ be the number of heads obtained when flipping $n - 14\sqrt{n}\ln\frac{1}{\delta}$ independent fair coins. Then the protocol's error probability is:

$$\begin{aligned}
\Pr[\mathbf{z} \ge \frac{n}{2} - 2\sqrt{n}\ln\frac{1}{\delta}] &< \Pr[\mathbf{z}' \ge \frac{n}{2} - 2\sqrt{n}\ln\frac{1}{\delta}] \\
&\le \Pr[\mathbf{z}' \ge \frac{n}{2} - 2\sqrt{n}\ln\frac{1}{\delta} - 70\ln^2\frac{1}{\delta}] \\
&= \Pr[\mathbf{z}' \ge (1 + \frac{10\ln\frac{1}{\delta}}{\sqrt{n}})\cdot(\frac{n}{2} - 7\sqrt{n}\ln\frac{1}{\delta})] \\
&\le exp(-\frac{1}{3}\cdot(\frac{n}{2} - 7\sqrt{n}\ln\frac{1}{\delta})\cdot\frac{100\ln^2\frac{1}{\delta}}{n}) \quad \text{(by Chernoff bound)} \\
&= exp(-\ln\frac{1}{\delta}\cdot(\frac{100\ln\frac{1}{\delta}}{6} - \frac{700\ln^2\frac{1}{\delta}}{3\sqrt{n}})) \\
&\le exp(-\ln\frac{1}{\delta}\cdot(\frac{100\ln\frac{1}{\delta}}{6} - \frac{700\ln\frac{1}{\delta}}{48})) \\
&< \delta
\end{aligned}$$

□

The second to the last inequality holds since $n = g^2/(16\ln\frac{1}{\delta})^2 \ge (16\ln\frac{1}{\delta})^2$.

C Proof for Theorem 2

Theorem 2. *Let* $\text{cost}(\mathscr{P}, \mathscr{A}, C_{\mathscr{P}}, C_{\mathscr{A}})$ *be any deterministic function (which the adversary aims to maximize) of the protocol* $\mathscr{P}$*, the adversary* $\mathscr{A}$*, the coin flip outcomes* $C_{\mathscr{P}}$ *of* $\mathscr{P}$*, and the coin flip outcomes* $C_{\mathscr{A}}$ *(if any) that may also influence the behavior of* $\mathscr{A}$*. For any protocol* $\mathscr{P}$*, any deterministic adaptive adversary* $\mathscr{A}'$*, and its sanitized version* $\mathscr{A}$*, there exists a deterministic oblivious adversary* $\mathscr{B}$ *such that*

$E_{\mathbf{C}_{\mathscr{P}}}[\text{cost}(\mathscr{P}, \mathscr{B}, \mathbf{C}_{\mathscr{P}}, -)] \geq E_{\mathbf{C}_{\mathscr{P}},\mathbf{C}_{\mathscr{A}}}[\text{cost}(\mathscr{P}, \mathscr{A}, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{A}})]$. *Furthermore, for every $C_{\mathscr{P}}$ in the support of $\mathbf{C}_{\mathscr{P}}$, there exists $C_{\mathscr{A}}$ in the support of $\mathbf{C}_{\mathscr{A}}$, such that $\mathscr{B}$'s decisions are exactly the same as the decisions made by $\mathscr{A}$ under $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$.*

Proof. We will construct a $\mathscr{B}$ to satisfy the properties in the theorem. Let random variable $\mathbf{Z}$ be the sequences of decisions made by $\mathscr{A}$ that have been sanitized by $\mathscr{A}$. $\mathscr{A}$ may make other decisions that are not part of $\mathbf{Z}$ — those decisions must be non-adaptive decisions made by $\mathscr{A}'$. Under given $C_{\mathscr{P}}$, there exists a one-to-one mapping from the sequence Z of decisions made by $\mathscr{A}$ to the coin flip outcomes $C_{\mathscr{A}}$ obtained by $\mathscr{A}$. Let the function $h(Z) = C_{\mathscr{A}}$ be this one-to-one mapping. Since all $C_{\mathscr{A}}$ occurs with the same probability, for all $C_{\mathscr{P}}$, Z, and $C_{\mathscr{A}}$, we have $\Pr[\mathbf{Z} = Z|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] = \Pr[\mathbf{C}_{\mathscr{A}} = h(Z)|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] = \Pr[\mathbf{C}_{\mathscr{A}} = C_{\mathscr{A}}|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}]$. Since $\mathbf{C}_{\mathscr{P}}$ and $\mathbf{C}_{\mathscr{A}}$ are independent, we further have:

$$\begin{aligned}
\Pr[\mathbf{Z} = Z] &= \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{Z} = Z|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \times \Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}]) \\
&= \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{A}} = C_{\mathscr{A}}|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \times \Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}]) \\
&= \Pr[\mathbf{C}_{\mathscr{A}} = C_{\mathscr{A}}] = \Pr[\mathbf{C}_{\mathscr{A}} = C_{\mathscr{A}}|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] = \Pr[\mathbf{Z} = Z|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \qquad (1)
\end{aligned}$$

Define $f(\mathscr{P}, C_{\mathscr{P}}, Z) = \text{cost}(\mathscr{P}, \mathscr{A}, C_{\mathscr{P}}, C_{\mathscr{A}})$ where $h(Z) = C_{\mathscr{A}}$ under the given $C_{\mathscr{P}}$ and $\mathscr{A}$. Note that $f(\mathscr{P}, C_{\mathscr{P}}, Z)$ no longer depends on $\mathscr{A}$ itself. We have:

$$\begin{aligned}
& E_{\mathbf{C}_{\mathscr{P}},\mathbf{C}_{\mathscr{A}}}[\text{cost}(\mathscr{P}, \mathscr{A}, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{A}})] \\
= & \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \sum_{C_{\mathscr{A}}}(\Pr[\mathbf{C}_{\mathscr{A}} = C_{\mathscr{A}}|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \times \text{cost}(\mathscr{P}, \mathscr{A}, C_{\mathscr{P}}, C_{\mathscr{A}}))) \\
= & \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \sum_{Z}(\Pr[\mathbf{Z} = Z|\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \times f(\mathscr{P}, C_{\mathscr{P}}, Z))) \\
& \text{(previous step was since given } C_{\mathscr{P}}\text{, there is a one-to-one mapping between } C_{\mathscr{A}} \text{ and } Z) \\
= & \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] \sum_{Z}(\Pr[\mathbf{Z} = Z] f(\mathscr{P}, C_{\mathscr{P}}, Z))) \quad \text{(from Equation 1)} \\
= & \sum_{Z}(\Pr[\mathbf{Z} = Z] \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] f(\mathscr{P}, C_{\mathscr{P}}, Z)))
\end{aligned}$$

Now pick Z_0 such that $\sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] f(\mathscr{P}, C_{\mathscr{P}}, Z_0))$ is maximized, and we have:

$$\begin{aligned}
\sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] f(\mathscr{P}, C_{\mathscr{P}}, Z_0)) &\geq \sum_{Z} \Pr[\mathbf{Z} = Z] \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] f(\mathscr{P}, C_{\mathscr{P}}, Z)) \\
&= E_{\mathbf{C}_{\mathscr{P}},\mathbf{C}_{\mathscr{A}}}[\text{cost}(\mathscr{P}, \mathscr{A}, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{A}})]
\end{aligned}$$

Since Z_0 is a fixed sequence of decisions, we construct the adversary $\mathscr{B}$ the same as $\mathscr{A}$ except that $\mathscr{B}$ always make those decisions Z_0 in place of the decisions $\mathbf{Z}$ made by $\mathscr{A}$. Obviously $\mathscr{B}$ is deterministic and oblivious, and furthermore:

$$\begin{aligned}
E_{\mathbf{C}_{\mathscr{P}}}[\text{cost}(\mathscr{P}, \mathscr{B}, \mathbf{C}_{\mathscr{P}}, -)] &= \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}]\text{cost}(\mathscr{P}, \mathscr{B}, \mathbf{C}_{\mathscr{P}}, -)) \\
&= \sum_{C_{\mathscr{P}}}(\Pr[\mathbf{C}_{\mathscr{P}} = C_{\mathscr{P}}] f(\mathscr{P}, C_{\mathscr{P}}, Z_0)) \\
&\geq E_{\mathbf{C}_{\mathscr{P}},\mathbf{C}_{\mathscr{A}}}[\text{cost}(\mathscr{P}, \mathscr{A}, \mathbf{C}_{\mathscr{P}}, \mathbf{C}_{\mathscr{A}})]
\end{aligned}$$

Finally, given $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$ where $C_{\mathscr{A}} = h(Z_0)$, $\mathscr{B}$'s decisions Z_0 are exactly the same as $\mathscr{A}$'s decisions under $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$. □

D Proof for Theorem 3

The proof for Theorem 3 is based on the reduction in Section 8. We will prove various properties of Protocol 1 (whose description can be found in Section 8.2), which will ultimately lead to the proof for Theorem 3. These properties include Lemma 2, Theorem 5, Lemma 3, Lemma 4, Theorem 6, Lemma 5, Theorem 7, Lemma 6, and Lemma 7. In particular, we aim to prove that Protocol 1 is correct for all two-party permutation-invariant problem Π. For length-n string X, define $M(X) = x_{m_1}x_{m_2}...x_{m_n}$, where M is any given permutation of 1 through n, and m_i is the i-th integer in M. A two-party problem Π is *permutation-invariant* iff for all X, Y, and M, $\Pi(X,Y) = \Pi(M(X), M(Y))$. Throughout this section, we assume that Π is permutation invariant, and when we mention a line number (e.g., Line 5), we refer to the corresponding line of Protocol 1.

We first quantify the estimation quality on the occurrence counts of each leakable pattern as done by the protocol. For $1 \le j \le k$, let w_j denote the occurrence count of the j-th leakable pattern in (X', Y'). The $\mathbf{v}_j$'s in Protocol 1 are essentially estimates for w_j. We say that *Protocol 1's estimates are good* if immediately after Line 6, $\max_{1\le j\le k}(\mathbf{v}_j - w_j)^2 \le \frac{n'^2}{2s}\ln\frac{24k}{\delta-\delta'}$.

Lemma 2. *Protocol 1's estimates are good with probability at least* $1 - \frac{\delta'-\delta}{12}$.

Proof. Let $\epsilon = \sqrt{\frac{n'^2}{2s}\ln\frac{24k}{\delta'-\delta}}$. By the definition of good, it suffices to prove

$$\Pr[\max_{1\le j\le k}|\mathbf{v}_j - w_j| \le \epsilon] \ge 1 - \frac{\delta'-\delta}{12}$$

For any $j \in [1,k]$, let $\mathbf{s}_j$ be the number of times $\mathbf{v}_j$ is incremented by $\frac{n'}{s}$ in Line 5 of Protocol 1. Each time Line 3 through 5 is executed, $\mathbf{v}_j$ is incremented only when (x'_i, y'_i) is the j-th leakable pattern. Since i is drawn uniformly at random from $[1, n']$ and since there exists exactly w_j indices $i \in [1, n']$ such that (x'_i, y'_i) is the j-th leakable pattern, each time Line 3 through 5 is executed $\mathbf{v}_j$ is incremented with probability exactly $\frac{w_j}{n'}$. Since Line 3 through 5 is executed s times, $\mathbf{s}_j$ is the sum of s independent and identical Bernoulli random variables, with each Bernoulli trial having a success probability of $\frac{w_j}{n'}$. We will apply the Chernoff-Hoeffding bound [18] for absolute error, which states for any $0 \le \frac{w_j}{n'} \le 1$ and $a \ge 0$,

$$\begin{aligned}\Pr\left(\frac{\mathbf{s}_j}{s} \ge \frac{w_j}{n'} + a\right) &\le e^{-2a^2s}\\ \Pr\left(\frac{\mathbf{s}_j}{s} \le \frac{w_j}{n'} - a\right) &\le e^{-2a^2s}\end{aligned}$$

$\mathbf{v}_j$ is modified only in Line 1, where it is initially set to zero, and then in Line 5. Thus, $\mathbf{v}_j = \frac{\mathbf{s}_j n'}{s}$. Hence we have:

$$\Pr[|\mathbf{v}_j - w_j| > \epsilon] = \Pr[|\frac{\mathbf{s}_j n'}{s} - w_j| > \epsilon] = \Pr[|\frac{\mathbf{s}_j}{s} - \frac{w_j}{n'}| > \frac{\epsilon}{n'}] \le 2exp(-2s(\frac{\epsilon}{n'})^2) = \frac{\delta'-\delta}{12k}$$

Finally, taking a union bound for j from 1 through k, we have:

$$\Pr[\max_{1\le j\le k}|\mathbf{v}_j - w_j| > \epsilon] \le k \times \frac{\delta'-\delta}{12k} \le \frac{\delta'-\delta}{12}$$

□

Let $\mathbb{B}(\mu)$ denote the binomial distribution corresponding to the number of heads in 2μ fair coin flips.

Theorem 5. *Consider any positive integer k, any μ_i and μ'_i where $2\mu_i$ and $2\mu'_i$ are all integers ($1 \le i \le k$). Let $\mu = \min_{1\le i\le k}(\min(\mu_i, \mu'_i))$. Let δ and δ' be any given constants where $0 < \delta' < \delta < 0.5$. Let product distribution $\mathbb{D} = \mathbb{B}(\mu_1) \times \mathbb{B}(\mu_2) \times ... \times \mathbb{B}(\mu_k)$ and $\mathbb{D}' = \mathbb{B}(\mu'_1) \times \mathbb{B}(\mu'_2) \times ... \times \mathbb{B}(\mu'_k)$. If $\mu \ge \frac{250}{(\delta-\delta')^2}(k^2 + k\max_{1\le i\le k}(\mu_i - \mu'_i)^2)$, then $||\mathbb{D} - \mathbb{D}'|| \le \frac{2(\delta-\delta')}{3}$.*

The proof of Theorem 5 is long and complex, but is entirely independent of all other proofs in this paper. Hence we will prove it separately later in Appendix F.

Protocol 1 has (X', Y') as its inputs to Alice and Bob, respectively. It internally converts (X', Y') to a randomized input $(\mathbf{X}, \mathbf{Y})$. For any given (X, Y), conditioned upon $(\mathbf{X}, \mathbf{Y}) = (X, Y)$, we define $\hat{\mathbb{T}}(X, Y)$ to be the distribution of the *leaked sets*, as induced by Protocol 1 at Line 18. Here a *leaked set* is the set $\{(i, x_i, y_i) \,|\, \text{index } i \text{ is leaked}\}$. Define $\mathbb{T}(X, Y)$ to be the distribution of the leaked sets that would have resulted, if (X, Y) were subjected to the (real) leaker. Alice is using $\hat{\mathbb{T}}(X, Y)$ to approximate $\mathbb{T}(X, Y)$. We thus want to quantify the distance between these two distributions. Consider any two distributions $\mathbb{D}$ and $\mathbb{D}'$ over the same sample space $\mathcal{D}$. We define their L_1 *distance* (denoted as $||\mathbb{D} - \mathbb{D}'||$ and also called *variation distance*) to be $\int_{x\in\mathcal{D}} |f_{\mathbb{D}}(x) - f_{\mathbb{D}'}(x)|dx$ if $\mathcal{D}$ is continuous, and $\sum_{x\in\mathcal{D}} |f_{\mathbb{D}}(x) - f_{\mathbb{D}'}(x)|$ if $\mathcal{D}$ is discrete. Here $f_{\mathbb{D}}$ and $f_{\mathbb{D}'}$ are the density functions of the two distributions, respectively.

Lemma 3. *If Protocol 1's estimates are good and if it does not exit at Line 9, then for all (X, Y), we have $||\hat{\mathbb{T}}(X, Y) - \mathbb{T}(X, Y)|| \le \frac{9(\delta'-\delta)}{12}$.*

Proof. Consider any (X, Y) in the support of $(\mathbf{X}, \mathbf{Y})$. For clarity, we write $\hat{\mathbb{T}}(X, Y)$ as $\hat{\mathbb{T}}$ and $\mathbb{T}(X, Y)$ as $\mathbb{T}$. Let random variables $\hat{\mathbf{T}} \sim \hat{\mathbb{T}}$ and $\mathbf{T} \sim \mathbb{T}$. Given a leaked set (e.g., $\mathbf{T}$), define its corresponding *leaked vector* as a vector of k integers, where the jth entry is the number of indices i such that (i, x_i, y_i) is in the leaked set and (x_i, y_i) is the jth leakable pattern. Define many-to-one mapping $\rho(\cdot)$, which maps each leaked set to its corresponding leaked vector. Define random variables $\hat{\mathbf{S}} = \rho(\hat{\mathbf{T}})$ and $\mathbf{S} = \rho(\mathbf{T})$. Let the distributions of $\hat{\mathbf{S}}$ and $\mathbf{S}$ be $\hat{\mathbb{S}}$ and $\mathbb{S}$, respectively. By the behavior of the leaker, it is obvious that $\mathbb{S} = \Pi_{j=1}^{k}\mathbb{B}(\frac{h_j+w_j}{2})$. On the other hand, due to Line 12 in Protocol 1, $\hat{\mathbb{S}}$ is different from $\Pi_{j=1}^{k}\mathbb{B}(\frac{h_j+\mathrm{v}_j}{2})$. Define $\rho^{-1}(S) = \{T|\rho(T) = S\}$. Let supp$(\cdot)$ denote the support of a distribution, and let $f_{\mathbb{D}}(\cdot)$ denote the probability density function of a distribution $\mathbb{D}$. It is easy to see that supp$(\hat{\mathbb{T}}) = \{T|T \in \rho^{-1}(S) \text{ and } S \in \text{supp}(\hat{\mathbb{S}})\}$ and supp$(\mathbb{T}) = \{T|T \in \rho^{-1}(S) \text{ and } S \in \text{supp}(\mathbb{S})\}$.

We will first prove that $||\hat{\mathbb{T}} - \mathbb{T}|| = ||\hat{\mathbb{S}} - \mathbb{S}||$. To do so, we observe that conditioned on $\hat{\mathbf{S}} = \mathbf{S}$, the distributions of $\hat{\mathbf{T}}$ and $\mathbf{T}$ are the same, which implies that for all $T \in \text{supp}(\hat{\mathbb{T}}) \cup \text{supp}(\mathbb{T})$ and all $S \in \text{supp}(\hat{\mathbb{S}}) \cap \text{supp}(\mathbb{S})$, $\Pr[\hat{\mathbf{T}} = T|\hat{\mathbf{S}} = S] = \Pr[\mathbf{T} = T|\mathbf{S} = S]$. To see why such an observation holds, let $\hat{\mathrm{s}}_j$ ($1 \le j \le k$) denote the jth entry in $\hat{\mathbf{S}}$. By the construction of Protocol 1, $\hat{\mathbf{T}}$ can be viewed as choosing $\hat{\mathrm{s}}_j$ uniformly random indices among all indices that corresponds to the jth leakable pattern, and then leak those indices. This holds because all the indices were permuted using a uniformly random permutation at Line 17 in Protocol 1. Similarly, by the behavior of the leaker, T can be viewed as being generated via such a process as well.

Thus we have:

$$\begin{aligned}
||\hat{\mathbb{T}} - \mathbb{T}|| &= \sum_{T\in\text{supp}(\hat{\mathbb{T}})\cup\text{supp}(\mathbb{T})} |f_{\hat{\mathbb{T}}}(T) - f_{\mathbb{T}}(T)| \\
&= \sum_{S\in\text{supp}(\hat{\mathbb{S}})\setminus\text{supp}(\mathbb{S})} \sum_{T\in\rho^{-1}(S)} f_{\hat{\mathbb{T}}}(T) + \sum_{S\in\text{supp}(\mathbb{S})\setminus\text{supp}(\hat{\mathbb{S}})} \sum_{T\in\rho^{-1}(S)} f_{\mathbb{T}}(T) + \\
&\quad \sum_{S\in\text{supp}(\hat{\mathbb{S}})\cap\text{supp}(\mathbb{S})} \sum_{T\in\rho^{-1}(S)} |f_{\hat{\mathbb{S}}}(S)\Pr[\hat{\mathbf{T}} = T|\hat{\mathbf{S}} = S] - f_{\mathbb{S}}(S)\Pr[\mathbf{T} = T|\mathbf{S} = S]|
\end{aligned}$$

Leveraging the earlier claim that $\Pr[\hat{\mathbf{T}} = T|\hat{\mathbf{S}} = S] = \Pr[\mathbf{T} = T|\mathbf{S} = S]$, we can simplify the third term:

$$\begin{aligned}
& \sum_{S\in \text{supp}(\hat{\mathbb{S}})\cap \text{supp}(\mathbb{S})} \sum_{T\in\rho^{-1}(S)} |f_{\hat{\mathbb{S}}}(S)\Pr[\hat{\mathbf{T}}=T|\hat{\mathbf{S}}=S] - f_{\mathbb{S}}(S)\Pr[\mathbf{T}=T|\mathbf{S}=S]| \\
= & \sum_{S\in \text{supp}(\hat{\mathbb{S}})\cap \text{supp}(\mathbb{S})} \sum_{T\in\rho^{-1}(S)} (|f_{\hat{\mathbb{S}}}(S) - f_{\mathbb{S}}(S)| \times \Pr[\mathbf{T}=T|\mathbf{S}=S]|) \\
= & \sum_{S\in \text{supp}(\hat{\mathbb{S}})\cap \text{supp}(\mathbb{S})} (|f_{\hat{\mathbb{S}}}(S) - f_{\mathbb{S}}(S)| \times \sum_{T\in\rho^{-1}(S)} \Pr[\mathbf{T}=T|\mathbf{S}=S]|) \\
= & \sum_{S\in \text{supp}(\hat{\mathbb{S}})\cap \text{supp}(\mathbb{S})} |f_{\hat{\mathbb{S}}}(S) - f_{\mathbb{S}}(S)|
\end{aligned}$$

Combining this result with the earlier equation, we have:

$$\begin{aligned}
& ||\hat{\mathbb{T}} - \mathbb{T}|| \\
= & \sum_{S\in \text{supp}(\hat{\mathbb{S}})\backslash \text{supp}(\mathbb{S})} \sum_{T\in\rho^{-1}(S)} f_{\hat{\mathbb{T}}}(T) + \sum_{S\in \text{supp}(\mathbb{S})\backslash \text{supp}(\hat{\mathbb{S}})} \sum_{T\in\rho^{-1}(S)} f_{\mathbb{T}}(T) + \\
& \sum_{S\in \text{supp}(\hat{\mathbb{S}})\cap \text{supp}(\mathbb{S})} |f_{\hat{\mathbb{S}}}(S) - f_{\mathbb{S}}(S)| \\
= & \sum_{S\in \text{supp}(\hat{\mathbb{S}})\backslash \text{supp}(\mathbb{S})} f_{\hat{\mathbb{S}}}(S) + \sum_{S\in \text{supp}(\mathbb{S})\backslash \text{supp}(\hat{\mathbb{S}})} f_{\mathbb{S}}(S) + \sum_{S\in \text{supp}(\hat{\mathbb{S}})\cap \text{supp}(\mathbb{S})} |f_{\hat{\mathbb{S}}}(S) - f_{\mathbb{S}}(S)| \\
= & ||\hat{\mathbb{S}} - \mathbb{S}||
\end{aligned}$$

We have proved that $||\hat{\mathbb{T}} - \mathbb{T}|| = ||\hat{\mathbb{S}} - \mathbb{S}||$. Let distribution $\mathbb{D} = \Pi_{j=1}^{k}\mathbb{B}(\frac{h_j+\mathbf{v}_j}{2})$. The next will prove that:

$$||\mathbb{S} - \mathbb{D}|| \leq \frac{8(\delta'-\delta)}{12} \tag{2}$$

$$||\hat{\mathbb{S}} - \mathbb{D}|| \leq \frac{\delta'-\delta}{12} \tag{3}$$

If these two inequalities do hold, then we will have $||\hat{\mathbb{T}} - \mathbb{T}|| = ||\hat{\mathbb{S}} - \mathbb{S}|| \leq ||\mathbb{S} - \mathbb{D}|| + ||\hat{\mathbb{S}} - \mathbb{D}|| \leq \frac{9(\delta'-\delta)}{12}$.

We first prove $||\mathbb{S} - \mathbb{D}|| \leq \frac{8(\delta'-\delta)}{12}$, by using Theorem 5. Recall that $\mathbb{S} = \Pi_{j=1}^{k}\mathbb{B}(\frac{h_j+w_j}{2})$. Let $\mu_j = \frac{h_j+w_j}{2}$ and $\hat{\mu}_j = \frac{h_j+\mathbf{v}_j}{2}$ for $1 \leq j \leq k$. Let $\mu = \min_{1\leq j\leq k}(\min(\mu_j, \hat{\mu}_j)) \geq \min_{1\leq j\leq k}(\frac{h_j}{2}) = \min(\frac{h}{2}, \frac{h_k}{2})$. Since Protocol 1 does not exit at Line 9, we have $h_k \geq h$. Hence $\mu \geq \frac{h}{2}$. Since Protocol 1's estimates are good, by the definition of estimates being good, we have:

$$\begin{aligned}
\mu \quad \geq \quad & \frac{h}{2} \geq n' + \frac{250}{(\delta'-\delta)^2}\left(k^2 + k \max_{1\leq j\leq k}(\mathbf{v}_j - w_j)^2\right) \\
> \quad & \frac{250}{(\delta'-\delta)^2}(k^2 + 4k \max_{1\leq j\leq k}(\mu_j - \hat{\mu}_j)^2) \\
> \quad & \frac{250}{(\delta'-\delta)^2}(k^2 + k \max_{1\leq j\leq k}(\mu_j - \hat{\mu}_j)^2)
\end{aligned}$$

Applying Theorem 5 immediately tell us that $||\mathbb{S} - \mathbb{D}|| \leq \frac{8(\delta'-\delta)}{12}$.

Next we prove $||\hat{\mathbb{S}} - \mathbb{D}|| \leq \frac{\delta'-\delta}{12}$. The difference between $\hat{\mathbb{S}}$ and $\mathbb{D}$ arises solely from Line 12 in Protocol 1. when $\mathbf{b}_j > h_j$ for some j. Hence:

$$||\hat{\mathbb{S}} - \mathbb{D}|| \leq \Pr[\exists j \text{ where } 1 \leq j \leq k, \text{such that } \mathbf{b}_j > h_j]$$

We trivially have $h \geq 2n' \geq 2\mathbf{v}_j$ for all j. Since Protocol 1 does not exit at Line 9, we have $h \leq h_j$ for all j. Hence $\mathbf{v}_j \leq \frac{h}{2} \leq \frac{h_j}{2}$. The random variable $\mathbf{b}_j$ is drawn from the binomial distribution $\mathbb{B}(\frac{h_j+\mathbf{v}_j}{2})$ at Line 11 of Protocol 1, and thus $\frac{1}{2}h_j \leq E[\mathbf{b}_j] \leq \frac{3}{4}h_j$. By Chernoff bound, for any given j, we have:

$$\begin{aligned}
\Pr[\mathbf{b}_j \geq h_j] &\leq \Pr[\mathbf{b}_j \geq \frac{4}{3}E[\mathbf{b}_j]] \leq exp(-\frac{1}{3}(\frac{1}{3})^2 E[\mathbf{b}_j]) \leq exp(-\frac{1}{3}(\frac{1}{3})^2\frac{1}{2}h_j) = exp(-\frac{1}{54}h_j) \\
&< exp(-\frac{500k^2}{54(\delta'-\delta)^2}) < \frac{54(\delta'-\delta)^2}{500k^2} \quad \text{(since } exp(-x) < 1/x \text{ for } x > 0) \\
&< \frac{\delta'-\delta}{12k} \quad \text{(since } 0 < \delta < \delta' < \tfrac{1}{2})
\end{aligned}$$

Finally, taking a union bound for j from 1 through k gives:

$$||\hat{\mathbb{S}} - \mathbb{D}|| \leq \Pr[\exists j \text{ where } 1 \leq j \leq k, \text{such that } \mathbf{b}_j > h] < k \cdot \frac{\delta'-\delta}{12k} = \frac{\delta'-\delta}{12}$$

□

Lemma 4. *If Protocol 1's estimates are good and if it does not exit at Line 9, then for all (X, Y) in the support of $(\mathbf{X}, \mathbf{Y})$, we have:*

$$E_{\mathbf{C}_\mathscr{Q}}[\text{err}(\mathscr{Q}, X', Y', \mathbf{C}_\mathscr{Q})|(\mathbf{X}, \mathbf{Y}) = (X, Y)] \leq \delta + \frac{11}{12}(\delta' - \delta) \tag{4}$$

$$E_{\mathbf{C}_\mathscr{Q}}[\text{cc}(\mathscr{Q}, X', Y', \mathbf{C}_\mathscr{Q})|(\mathbf{X}, \mathbf{Y}) = (X, Y)] \leq \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + 5.5cc(\mathscr{P}) \tag{5}$$

Proof. Recall that given input (X', Y'), Protocol 1 invokes $\mathscr{P}$ internally. When $(\mathbf{X}, \mathbf{Y}) = (X, Y)$, Protocol 1 invokes $\mathscr{P}$ with input (X, Y). The input (X, Y) is obtained by i) padding some extra leakable patterns to (X', Y') at Line 14, and ii) doing a permutation at Line 17.

- Proof for Equation 4. Consider any (X, Y) in the support of $(\mathbf{X}, \mathbf{Y})$. By definition of leakable patterns and permutation-invariant functions, we know that $\Pi((X', Y')) = \Pi((X, Y))$. Hence Protocol 1's result must be correct if i) Protocol 1 does not exit at Line 9, ii) Protocol 1 does not exit in Line 20 due to $\mathscr{P}$ incurring more than $\frac{6}{\delta'-\delta}\text{cc}(\mathscr{P})$ bits of communication, and iii) $\mathscr{P}$ gives the correct result for (X, Y). Recall that $\hat{\mathbb{T}}(X, Y)$ is defined to be the distribution of the leaked set fed into $\mathscr{P}$ by Protocol 1, while $\mathbb{T}(X, Y)$ is defined to be the distribution of the leaked set that would have been generated by the leaker for (X, Y). For clarity, we write them as $\hat{\mathbb{T}}$ and $\mathbb{T}$. Define $\mathbb{C}_Q$ and $\mathbb{C}_P$ to be the distribution of coin flips made by $\mathscr{Q}$ and $\mathscr{P}$, respectively. Define $\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_\mathscr{P}, \mathbf{T})$ to be communication incurred (in terms of number of bits) by $\mathscr{P}$, under the input (X, Y), protocol's coin flip outcomes $\mathbf{C}_\mathscr{P}$, and leaked set $\mathbf{T}$. Note that $\mathbf{T}$ captures all coins flipped by the leaker. We similarly define $\text{err}(\mathscr{P}, X, Y, \mathbf{C}_\mathscr{P}, \mathbf{T})$, which is 1 if the $\mathscr{P}$'s output is wrong, and 0 otherwise. From the condition of the lemma, we already know that Protocol 1 does not exit at Line 9. We have:

$$
\begin{aligned}
& \Pr_{\mathbf{C}_{\mathscr{Q}}\sim\mathbb{C}_Q}[\text{err}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}}) = 1|(\mathbf{X},\mathbf{Y}) = (X, Y)] \\
= & \Pr_{\mathbf{C}_{\mathscr{P}}\sim\mathbb{C}_P, \mathbf{T}\sim\hat{\mathbb{T}}}[(\text{err}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T}) = 1) \text{ or } (\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T}) > \frac{6}{\delta'-\delta}\text{cc}(\mathscr{P}))] \\
\leq & ||\hat{\mathbb{T}} - \mathbb{T}|| + \Pr_{\mathbf{C}_{\mathscr{P}}\sim\mathbb{C}_P, \mathbf{T}\sim\mathbb{T}}[(\text{err}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T}) = 1) \text{ or } (\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T}) > \frac{6}{\delta'-\delta}\text{cc}(\mathscr{P}))] \\
\leq & ||\hat{\mathbb{T}} - \mathbb{T}|| + \Pr_{\mathbf{C}_{\mathscr{P}}\sim\mathbb{C}_P, \mathbf{T}\sim\mathbb{T}}[\text{err}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T}) = 1] + \\
& \Pr_{\mathbf{C}_{\mathscr{P}}\sim\mathbb{C}_P, \mathbf{T}\sim\mathbb{T}}[\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T}) > \frac{6}{\delta'-\delta}\text{cc}(\mathscr{P})] \\
\leq & ||\hat{\mathbb{T}} - \mathbb{T}|| + \delta + \frac{\delta'-\delta}{6} \\
\leq & \delta + \frac{11}{12}(\delta' - \delta) \quad \text{(by Lemma 3)}
\end{aligned}
$$

- Proof for Equation 5. Protocol 1's communication only involves two parts. The first part is for taking $\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{4\log q}$ samples in Step 1, which incurs at most $\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{4\log q} \times 2\log q = \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2}$ bits. The second part is for invoking $\mathscr{P}$, incurring no more than $\frac{6}{\delta'-\delta}\text{cc}(\mathscr{P})$ bits. We have:

$$
\begin{aligned}
& E_{\mathbf{C}_{\mathscr{Q}}}[\text{cc}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}})|(\mathbf{X},\mathbf{Y}) = (X, Y)] \\
\leq & \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + E_{\mathbf{C}_{\mathscr{P}}\sim\mathbb{C}_P, \mathbf{T}\sim\hat{\mathbb{T}}}[\min(\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T}), \frac{6}{\delta'-\delta}\text{cc}(\mathscr{P}))] \\
\leq & \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + ||\hat{\mathbb{T}} - \mathbb{T}|| \times \frac{6}{\delta'-\delta}\text{cc}(\mathscr{P}) + E_{\mathbf{C}_{\mathscr{P}}\sim\mathbb{C}_P, \mathbf{T}\sim\mathbb{T}}[\text{cc}(\mathscr{P}, X, Y, \mathbf{C}_{\mathscr{P}}, \mathbf{T})] \\
\leq & \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + \frac{9(\delta'-\delta)}{12} \times \frac{6}{\delta'-\delta}\text{cc}(\mathscr{P}) + \text{cc}(\mathscr{P}) \quad \text{(by Lemma 3)} \\
= & \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + 5.5\text{cc}(\mathscr{P})
\end{aligned}
$$

□

Theorem 6. *For all permutation-invariant problem* Π, *all constants* δ *and* δ' *such that* $0 < \delta < \delta' < 1/2$, *all* n *and* n' *such that* $n \geq n' + 2kn' + \frac{500}{(\delta'-\delta)^2}\left(k^3 + \frac{2k^2n'^2}{\mathfrak{R}_{\delta'}(\Pi_{n'})}(\log q)(\ln\frac{24k}{\delta'-\delta})\right)$, *we have* $\mathfrak{L}_{\delta}(\Pi_n) \geq \frac{1}{14}\mathfrak{R}_{\delta'}(\Pi_{n'})$.

Proof. For any given protocol $\mathscr{P}$ for solving Π_n with the leaker and with error δ, we construct a protocol $\mathscr{Q}$ for solving $\Pi_{n'}$ without our leaker and with error δ' as in Protocol 1. It is easy to verify that n is large enough such that Protocol 1 does not exit at Line 9:

$$
\begin{aligned}
n & \geq n' + 2kn' + \frac{500}{(\delta'-\delta)^2}\left(k^3 + \frac{2k^2n'^2}{\mathfrak{R}_{\delta'}(\Pi_{n'})}(\log q)(\ln\frac{24k}{\delta'-\delta})\right) \\
& = n' + k\left(2n' + \frac{500}{(\delta'-\delta)^2}\left(k^2 + \frac{2kn'^2}{\mathfrak{R}_{\delta'}(\Pi_{n'})}(\log q)(\ln\frac{24k}{\delta'-\delta})\right)\right) = n' + kh
\end{aligned}
$$

Denote z as the event that Protocol 1's estimates are good. By Lemma 2, $\Pr[z] \geq 1 - \frac{\delta'-\delta}{12}$. Consider any given input (X', Y') to our reduction protocol in Protocol 1 and the corresponding random variables

$(\mathbf{X}, \mathbf{Y})$ obtained at Line 17 of Protocol 1. By Lemma 4, we have:

$$\begin{aligned}
& \Pr[\text{err}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}}) = 1|z] \\
= & \sum_{(X,Y)} \Pr[\text{err}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}}) = 1|(\mathbf{X}, \mathbf{Y}) = (X, Y), z] \times \Pr[(\mathbf{X}, \mathbf{Y}) = (X, Y)|z] \\
\le & \sum_{(X,Y)} (\delta + \frac{11}{12}(\delta' - \delta)) \times \Pr[(\mathbf{X}, \mathbf{Y}) = (X, Y)|z] = \delta + \frac{11}{12}(\delta' - \delta)
\end{aligned}$$

$$\begin{aligned}
& E_{\mathbf{C}_{\mathscr{Q}}}[\text{cc}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}})|z] \\
= & \sum_{(X,Y)} E_{\mathbf{C}_{\mathscr{Q}}}[\text{cc}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}})|(\mathbf{X}, \mathbf{Y}) = (X, Y), z] \times \Pr[(\mathbf{X}, \mathbf{Y}) = (X, Y)|z] \\
\le & \sum_{(X,Y)} \left(\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + 5.5\text{cc}(\mathscr{P}) \right) \times \Pr[(\mathbf{X}, \mathbf{Y}) = (X, Y)|z] \\
= & \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + 5.5\text{cc}(\mathscr{P})
\end{aligned}$$

Now we consider the case where z does not hold, i.e., the protocol's estimates are not good. Although most our previous technical lemmas no longer hold, we still know that the error probability is at most 1, and the communication cost, by our protocol design, is at most $\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{4\log q} \cdot 2\log q + \frac{6}{\delta' - \delta}\text{cc}(\mathscr{P})$ bits. Hence we have:

$$\begin{aligned}
\Pr[\text{err}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}}) = 1] & \le \Pr[\text{err}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}}) = 1|z] + (1 - \Pr[z]) \\
& \le \delta + \frac{11}{12}(\delta' - \delta) + \frac{1}{12}(\delta' - \delta) = \delta'
\end{aligned}$$

$$\begin{aligned}
& E_{\mathbf{C}_{\mathscr{Q}}}[\text{cc}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}})] \\
\le & E_{\mathbf{C}_{\mathscr{Q}}}[\text{cc}(\mathscr{Q}, X', Y', \mathbf{C}_{\mathscr{Q}})|z] + (1 - \Pr[z]) \left(\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{4\log q} \cdot 2\log q + \frac{6}{\delta' - \delta}\text{cc}(\mathscr{P}) \right) \\
\le & 5.5\text{cc}(\mathscr{P}) + \frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{2} + \frac{\delta' - \delta}{12} \left(\frac{\mathfrak{R}_{\delta'}(\Pi_{n'})}{4\log q} \cdot 2\log q + \frac{6}{\delta' - \delta}\text{cc}(\mathscr{P}) \right) \\
\le & 6\text{cc}(\mathscr{P}) + \frac{25}{48}\mathfrak{R}_{\delta'}(\Pi_{n'})
\end{aligned}$$

Since $\mathscr{Q}$ solves $\Pi_{n'}$ with at most δ' error, we have $6\text{cc}(\mathscr{P}) + \frac{25}{48}\mathfrak{R}_{\delta'}(\Pi_{n'}) \ge \text{cc}(\mathscr{Q}) \ge \mathfrak{R}_{\delta'}(\Pi_{n'})$. Let $\mathscr{P}$ be the optimal protocol for solving Π_n with the leaker and with error δ, we have $\mathfrak{L}_\delta(\Pi_n) = \text{cc}(\mathscr{P}) \ge \frac{1}{14}\mathfrak{R}_{\delta'}(\Pi_{n'})$. □

Lemma 5. *For any given two-party problem* Π*, any given constants* δ_1 *and* δ_2 *such that* $0 < \delta_1 < \delta_2 < 0.5$*, we have* $\mathfrak{R}_{\delta_1}(\Pi) \le \frac{\ln(1/\delta_1)}{2(0.5-\delta_2)^2}\mathfrak{R}_{\delta_2}(\Pi)$.

Proof. Given a protocol $\mathscr{P}$ for Π with error δ_2, we will construct a protocol with error at most δ_1 as follows: we invoke $\mathscr{P}$ for $\frac{\ln(1/\delta_1)}{2(0.5-\delta_2)^2}$ times, and take the majority of these outputs as the final output. Let random variable $\mathbf{z}$ denote the fraction of correct outputs. Since $E[\mathbf{z}] \ge 1 - \delta_2$, by the Chernoff-Hoeffding bound [18] for absolute error, we have:

$$\Pr[\mathbf{z} \le 0.5] \le \Pr[\mathbf{z} \le (E[\mathbf{z}] - (1 - \delta_2 - 0.5))] \le exp(-2\frac{\ln(1/\delta_1)}{2(0.5 - \delta_2)^2}(1 - \delta_2 - 0.5)^2) = \delta_1$$

□

Theorem 7. *For all permutation-invariant problem Π, all constants $\delta \in (0, \frac{1}{2})$, all n and n' such that $n \geq n' + 2kn' + \frac{500}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})\left(k^3 + \frac{k^2 n'^2}{\mathfrak{R}_\delta(\Pi_{n'})}(\log q)(\ln\frac{48k}{0.5-\delta})\right)$, we have $\mathfrak{L}_\delta(\Pi_n) \geq \frac{(0.25-0.5\delta)^2}{7\ln(1/\delta)}\mathfrak{R}_\delta(\Pi_{n'})$.*

Proof. Obviously, $0 < \delta < 0.5\delta + 0.25 < 0.5$. Apply Lemma 5 (with $\delta_1 = \delta$ and $\delta_2 = 0.5\delta + 0.25$) and we have:

$$\begin{aligned}
n &\geq n' + 2kn' + \frac{500}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})\left(k^3 + \frac{k^2 n'^2}{\mathfrak{R}_\delta(\Pi_{n'})}(\log q)(\ln\frac{48k}{0.5-\delta})\right)\\
&\geq n' + 2kn' + \frac{500}{(0.25-0.5\delta)^2}\left(k^3 + \frac{k^2 n'^2 \ln\frac{1}{\delta}}{(0.25-0.5\delta)^2\mathfrak{R}_\delta(\Pi_{n'})}(\log q)(\ln\frac{48k}{0.5-\delta})\right)\\
&> n' + 2kn' + \frac{500}{(0.25-0.5\delta)^2}\left(k^3 + \frac{2k^2 n'^2 \ln\frac{1}{\delta}}{2(0.5-(0.5\delta+0.25))^2\mathfrak{R}_\delta(\Pi_{n'})}(\log q)(\ln\frac{48k}{0.5-\delta})\right)\\
&\geq n' + 2kn' + \frac{500}{((0.5\delta+0.25)-\delta)^2}\left(k^3 + \frac{2k^2 n'^2}{\mathfrak{R}_{0.5\delta+0.25}(\Pi_{n'})}(\log q)(\ln\frac{24k}{(0.5\delta+0.25)-\delta})\right)
\end{aligned}$$

The above equation shows that n satisfies the condition needed for Theorem 6. Invoke Theorem 6 and we have $\mathfrak{L}_\delta(\Pi_n) \geq \frac{1}{14}\mathfrak{R}_{0.5\delta+0.25}(\Pi_{n'})$. Applying Lemma 5 a second time (with $\delta_1 = \delta$ and $\delta_2 = 0.5\delta + 0.25$) yields $\mathfrak{L}_\delta(\Pi_n) \geq \frac{1}{14}\mathfrak{R}_{0.5\delta+0.25}(\Pi_{n'}) \geq \frac{(0.25-0.5\delta)^2}{7\ln(1/\delta)}\mathfrak{R}_\delta(\Pi_{n'})$. □

Lemma 6. *For all $q \geq 2$ and all δ where $0 < \delta < 0.5$, we have $2(\ln\frac{140}{0.5-\delta})(\log q) > \ln(\frac{48q^2}{0.5-\delta})$.*

Proof.

$$\begin{aligned}
\ln(\frac{48q^2}{0.5-\delta}) &= \ln q^2 + \ln(\frac{48}{0.5-\delta}) = (\ln q^2)(1 + \frac{\ln(\frac{48}{0.5-\delta})}{\ln q^2})\\
&\leq 2(\ln q)(1 + \ln(\frac{48}{0.5-\delta})) < 2(\ln\frac{140}{0.5-\delta})(\ln q) < 2(\ln\frac{140}{0.5-\delta})(\log q)
\end{aligned}$$

□

Lemma 7. *For any constant $\delta \in (0, \frac{1}{2})$, there exist constants $c_1 > 0$ and $c_2 > 0$ such that for all permutation-invariant problem Π satisfying the cycle promise, we have $\mathfrak{L}_\delta(\Pi_n) \geq c_1\mathfrak{R}_\delta(\Pi_{c_2\sqrt{n}/(q^{1.5}\log q)})$.*

Proof. We will leverage the fact that under the cycle promised, $k \leq 2q$ for all problem Π. Let $c_1 = \frac{(0.25-0.5\delta)^2}{7\ln(1/\delta)}$, and let c_2 be the positive constant such that $\frac{1}{2c_2^2} = 3 + \frac{4000}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})(\ln\frac{140}{0.5-\delta})$. If $\frac{c_2\sqrt{n}}{q^{1.5}\log q} < 1$, the lemma trivially holds. Otherwise let $n' = \frac{c_2\sqrt{n}}{q^{1.5}\log q} \geq 1$. We have:

$$\begin{aligned}
n &= \frac{q^3 n'^2 \log^2 q}{c_2^2} > \frac{q^3 + q^2 n'^2 \log^2 q}{2c_2^2} \geq \frac{1}{2c_2^2}(q^3 + \frac{q^2 n'^2 \log^2 q}{\mathfrak{R}_\delta(\Pi_{n'})})\\
&> n' + 2q^2 n' + \frac{4000}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})(\ln\frac{140}{0.5-\delta})(q^3 + \frac{q^2 n'^2 \log^2 q}{\mathfrak{R}_\delta(\Pi_{n'})})\\
&> n' + 2q^2 n' + \frac{500}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})(\ln\frac{140}{0.5-\delta})(8q^3 + \frac{8q^2 n'^2 \log^2 q}{\mathfrak{R}_\delta(\Pi_{n'})})\\
&> n' + 2kn' + \frac{500}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})(k^3 + \frac{k^2 n'^2 \log^2 q}{\mathfrak{R}_\delta(\Pi_{n'})} \times (2\ln\frac{140}{0.5-\delta}))\\
&> n' + 2kn' + \frac{500}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})(k^3 + \frac{k^2 n'^2 \log q}{\mathfrak{R}_\delta(\Pi_{n'})} \times \ln(\frac{48q^2}{0.5-\delta})) \quad \text{(by Lemma 6)}\\
&> n' + 2kn' + \frac{500}{(0.25-0.5\delta)^4}(\ln\frac{1}{\delta})(k^3 + \frac{k^2 n'^2 \log q}{\mathfrak{R}_\delta(\Pi_{n'})}\ln(\frac{48k}{0.5-\delta}))
\end{aligned}$$

This means that n satisfies the condition needed by Theorem 7. Invoke Theorem 7, and we have $\mathfrak{L}_\delta(\Pi_n) \geq c_1\mathfrak{R}_\delta(\Pi_{n'}) = c_1\mathfrak{R}_\delta(\Pi_{c_2\sqrt{n}/(q^{1.5}\log q)})$. □

Theorem 3. *For any constant $\delta \in (0, \frac{1}{2})$, there exist constants $c_1 > 0$ and $c_2 > 0$ such that for all n, g, q, and $n' = c_2\sqrt{n}/(q^{1.5}\log q)$, $\mathfrak{L}_\delta(\text{GDC}_n^{g,q}) \geq c_1\mathfrak{R}_\delta(\text{GDC}_{n'}^{g,q})$.*

Proof. GDC is obviously permutation-invariant. The theorem directly follows from Lemma 7 with $\Pi_n = \text{GDC}_n^{g,q}$. □

E Proof for Theorem 4

This section aims to eventually prove Theorem 4. The proof essentially follows the framework in [32], with the key differences already explained in Section 7 and 8. In the discussion, we will sometimes refer to round 0, where the CONSENSUS protocol does nothing and where every node is in the receiving state.

E.1 Simulation Framework

Preprocessing. Given an input (X, Y) to the $\text{GDC}_n^{g,q}$ problem, Alice and Bob first process the input to obtain a *processed input* $(\mathbf{X}', \mathbf{Y}')$. To do so, they use public coins to generate a uniformly random permutation π, and set $\mathbf{X}' = \pi(X)$ and $\mathbf{Y}' = \pi(Y)$, respectively. Next for each i ($1 \leq i \leq n$), Alice and Bob use public coins to draw an independent random integer $\mathbf{o}_i$ as an *offset* such that $\Pr[\mathbf{o}_i = 0] = \frac{1}{2}$ and $\Pr[\mathbf{o}_i = 2j] = \frac{1}{q-1}$ for $1 \leq j \leq \frac{q-1}{2}$, and then set $\mathbf{x}'_i = \min(\mathbf{x}'_i + \mathbf{o}_i, q-1)$ and $\mathbf{y}'_i = \min(\mathbf{y}'_i + \mathbf{o}_i, q-1)$, respectively.

Define $|_b^a(X, Y)$ to be the number of occurrences of the $|_b^a$ pattern in (X, Y). We say that (X, Y) is of:

- *type*-0 iff $|_0^0(X, Y) \geq q$ and $|_{2j}^{2j}(X, Y) \geq 1$ for $1 \leq j \leq \frac{q-1}{2}$;
- *type*-1 iff $|_0^0(X, Y) = |_2^2(X, Y) = \ldots = |_{q-3}^{q-3}(X, Y) = 0$ and $|_{q-1}^{q-1}(X, Y) \geq 1$.

Note that it is possible for (X, Y) to be neither type-0 nor type-1.

Define left$(X, Y) = (x_1 \ldots x_{\frac{n}{2}}, y_1 \ldots y_{\frac{n}{2}})$ and right$(X, Y) = (x_{\frac{n}{2}+1} \ldots x_n, y_{\frac{n}{2}+1} \ldots y_n)$. For $z \in \{0, 1\}$, we say that (X, Y) is of *double-type-z* if both left(X, Y) and right(X, Y) are type-z. Otherwise (X, Y) is of *double-type-$\perp$*.

Lemma 8. *Consider any input (X, Y) of the $\text{GDC}_n^{g,q}$ problem and its corresponding processed input $(\mathbf{X}', \mathbf{Y}')$. For $z \in \{0, 1\}$, if $q \geq 20$, $g \geq 15q\ln q$, $n \geq 4g$, and $\text{GDC}(X, Y) = z$, then $\Pr[(\mathbf{X}', \mathbf{Y}')$ is of double-type-$z] > 1 - \frac{1}{q}$.*

Proof. We separately consider two cases:

- $\text{GDC}(X, Y) = 1$. By definition, $|_0^0(X, Y) = 0$. In turn, $|_0^0(\mathbf{X}', \mathbf{Y}') = |_2^2(\mathbf{X}', \mathbf{Y}') = \ldots = |_{q-3}^{q-3}(\mathbf{X}', \mathbf{Y}') = 0$. Next, since for each index $\Pr[\mathbf{o} = q-1] = \frac{1}{q-1}$, the probability that there does not exist any index from 1 to $\frac{n}{2}$ such that $\mathbf{o} = q-1$ is:

$$(1 - \frac{1}{q-1})^{\frac{n}{2}} < exp(-\frac{1}{q-1} \cdot \frac{n}{2}) < exp(-\frac{n}{2q}) \leq exp(-\frac{4g}{2q}) \leq exp(-30\ln q) = \frac{1}{q^{30}} < \frac{1}{2q}$$

  As long as there exists some $\mathbf{o} = q-1$, left$(\mathbf{X}', \mathbf{Y}')$ will be of type-1. The probability of right$(\mathbf{X}', \mathbf{Y}')$ being type-1 is the same. A simple union bound then shows that with probability at least $1 - \frac{1}{q}$, both left$(\mathbf{X}', \mathbf{Y}')$ and right$(\mathbf{X}', \mathbf{Y}')$ are type-1.

- $\textsc{Gdc}(X, Y) = 0$. By definition, $|^0_0(X, Y) \geq g$. We first show that immediately after the permutation and before adding the offsets in the preprocessing step, with probability at least $1 - \frac{1}{q^2}$, $|^0_0(\text{left}(\mathbf{X}', \mathbf{Y}')) \geq 4q \ln q$. To see why, we view the permutation as obtained by assigning each index between 1 and n in (X, Y), one by one, to a new position between 1 and n after the permutation. Each position can only accommodate one index. Hence when we assign an index, we will choose a uniformly random position among all remaining unoccupied positions. Furthermore, we can imagine that we assign those indices corresponding to the $|^0_0$ patterns in (X, Y) first, before assigning other indices. There are at least g indices corresponding to the $|^0_0$ pattern, and let us consider the first g of them. For each such index, regardless what happened prior to our assigning this index, the probability of this index being assigned to the first half of the positions (i.e., to some position between 1 and $\frac{n}{2}$) must be no smaller than $\frac{1}{3}$. The reason is that there will always be at least $\frac{n}{2} - g \geq \frac{n}{4}$ unoccupied positions in the first half of the positions, and at most $\frac{n}{2}$ unoccupied positions in the second half. Consider the sum $\mathbf{z}$ of $15q \ln q$ independent Bernoulli random variables each taking a value of 1 with probability $\frac{1}{3}$. We have, via a simply coupling argument and Chernoff bound:

$$\Pr[|^0_0(\text{left}(\mathbf{X}', \mathbf{Y}')) < 4q \ln q] \leq \Pr[\mathbf{z} < (1 - \frac{1}{5})5q \ln q] \leq exp(-\frac{1}{2} \times \frac{1}{25} \times 5q \ln q) \leq \frac{1}{q^2}$$

  Next, conditioned upon the event that $|^0_0(\text{left}(\mathbf{X}', \mathbf{Y}')) \geq 4q \ln q$ before adding the offsets in the preprocessing step, we will show that after adding the offsets, $\Pr[\text{left}(\mathbf{X}', \mathbf{Y}') \text{ is type-0}] > 1 - \frac{1}{q^3} - \frac{1}{q^5}$. Consider any given j where $1 \leq j \leq \frac{q-1}{2}$. For each index corresponding to the $|^0_0$ pattern in $\text{left}(\mathbf{X}', \mathbf{Y}')$ immediately after the permutation, Alice and Bob will choose an offset $\mathbf{o}$ where $\Pr[\mathbf{o} = 2j] = \frac{1}{q-1}$. Hence the probability that $\mathbf{o} = 2j$ for none of the $4q \ln q$ indices is:

$$(1 - \frac{1}{q-1})^{4q \ln q} < exp(-\frac{4q \ln q}{q-1}) < exp(-4 \ln q) = \frac{1}{q^4}$$

  There are total $\frac{q-1}{2}$ such j's. Hence by a union bound, with probability at least $1 - \frac{1}{q^3}$, after adding the offsets, $|^{2j}_{2j}(\text{left}(\mathbf{X}', \mathbf{Y}')) \geq 1$ for all j. Next, after adding the offsets, by a Chernoff bound, we also have:

$$Pr[|^0_0(\text{left}(\mathbf{X}', \mathbf{Y}')) < q] < \Pr[|^0_0(\text{left}(\mathbf{X}', \mathbf{Y}')) < (1-\frac{1}{2})2q \ln q] \leq exp(-\frac{1}{2} \times \frac{1}{4} \times 2q \ln q) \leq \frac{1}{q^5}$$

  Taking a union bound thus shows that conditioned upon the event that $|^0_0(\text{left}(\mathbf{X}', \mathbf{Y}')) \geq 4q \ln q$ before adding the offsets, after adding the offsets, $\Pr[\text{left}(\mathbf{X}', \mathbf{Y}') \text{ is type-0}] > 1 - \frac{1}{q^3} - \frac{1}{q^5}$.

  Putting everything together, we have $\Pr[\text{left}(\mathbf{X}', \mathbf{Y}') \text{ is type-0}] > (1 - \frac{1}{q^2})(1 - \frac{1}{q^3} - \frac{1}{q^5}) > 1 - \frac{3}{q^2}$. By same argument, we similarly have $\Pr[\text{right}(\mathbf{X}', \mathbf{Y}') \text{ is type-0}] > 1 - \frac{3}{q^2}$. Then $\Pr[(\mathbf{X}', \mathbf{Y}') \text{ is double-type-0}] > 1 - \frac{6}{q^2} > 1 - \frac{1}{q}$ holds by union bound.

□

The reference adversary. We now define the *reference adversary*. Given a processed input and an oracle CONSENSUS protocol $\mathscr{P}$, the reference adversary determines a dynamic network, over which we will prove our lower bound. Our reference adversary will be a sanitized adaptive adversary, and hence the dynamic network generated depends on the coin flip outcomes of the CONSENSUS protocol, as well as the internal coin flip outcomes of the reference adversary itself. Note that Alice and Bob, not knowing the other party's input, do not know the reference adversary for the processed input that they are holding.

In the following we define the details of the reference adversary. If the processed input $(\mathbf{X}', \mathbf{Y}')$ is of double-type-$\perp$, then the dynamic network will have $\frac{3n}{2} + 2$ nodes, which are all called *stable nodes*.

There are two special nodes α and β in the topology. For each index i ($1 \le i \le \frac{n}{2}$), there is a vertical chain consisting of three nodes and two edges. (Note that we only use the first half of $(\mathbf{X}', \mathbf{Y}')$.) One edge connects the top and middle nodes, and one edge connects the middle and bottom nodes. The top nodes of all chains are connected to α, and the bottom nodes of all chains are connected to β. The topology does not change from round to round. A chain for index i is called a $|_b^a$ chain if $\mathbf{x}'_i = a$ and $\mathbf{y}'_i = b$. If a is even, we call the top edge (i.e., the edge between the top node and the middle node on the chain) as an *even edge* on this chain. Similarly, if b is even, the bottom edge is an *even edge*. We say a chain is *leaked* if the corresponding index is leaked by the leaker in the two-party GDC problem.

If $(\mathbf{X}', \mathbf{Y}')$ is of double-type-1, then the reference adversary is the same as the double-type-$\perp$ reference adversary, except that for all $t \ge 1$:

- For every $|_{2t-1}^{2t}$ and $|_{2t}^{2t-1}$ chain, the adversary removes the even edge at the beginning of round $t+1$.
- For every $|_{2t+1}^{2t}$ and $|_{2t}^{2t+1}$ chain, the adversary removes the even edge at the beginning of round $t + 1 + (\mathbf{z} \oplus \mathbf{s})$. Here $\mathbf{z} = 1$ if the middle node on the chain is receiving in round $t+1$, and $\mathbf{z} = 0$ otherwise. The random variable $\mathbf{s} = 1$ if the chain is leaked, and $\mathbf{s} = 0$ otherwise.

If $(\mathbf{X}', \mathbf{Y}')$ is of double-type-0, then the reference adversary is the same as the double-type-1 reference adversary, except that:

- The dynamic network has total $3n + 4$ nodes. Out of these, $\frac{3n}{2} + 2$ nodes are the stable nodes as in the double-type-1 reference adversary, while the remaining nodes are *unstable* nodes.
- Topology in round 0: The topology among the stable nodes are exactly the same as before. The topology among the unstable nodes are constructed in the same way as the stable nodes except that we use the second half of $(\mathbf{X}', \mathbf{Y}')$ to construct the $\frac{n}{2}$ chains. Let the two special nodes (among the unstable nodes) be γ and λ, where γ is node corrected to all the top nodes, while λ connects to all the bottom nodes.
- At the beginning of round 1, the adversary connects the middle nodes of all $|_0^0$ chains (including both chains of stable nodes and chains of unstable nodes) into a line such that all stable nodes are before the unstable nodes on the line. It then connects the end with a stable node to the middle node of some $|_2^2$ chain of stable nodes, and the other end (ending with an unstable node) to the middle node of some $|_2^2$ chain of unstable nodes. Since $(\mathbf{X}', \mathbf{Y}')$ is of double-type-0, such $|_2^2$ chains must exist. Finally, the adversary removes all the top edges and bottom edges on all the $|_0^0$ chains.
- For every $|_{2t}^{2t}$ chain of stable nodes (having $1 \le t < \frac{q-1}{2}$), at the beginning of round $t+1$, the adversary connects the middle node of the chain to the middle node of some arbitrary $|_{2t+2}^{2t+2}$ chain of stable nodes. Same as earlier, such $|_{2t+2}^{2t+2}$ chain must exist. Similarly, the adversary connects the middle node of every $|_{2t}^{2t}$ chain of unstable nodes (having $1 \le t < \frac{q-1}{2}$) to the middle node of some arbitrary $|_{2t+2}^{2t+2}$ chain of unstable nodes. Finally, the adversary removes the top and bottom edges of all $|_{2t}^{2t}$ chains.

Spoiled nodes. We inherit the notion of spoiled nodes from [32]. Each node is either *spoiled* or *non-spoiled* for Alice. Roughly speaking, a node is non-spoiled for Alice in round r if, based solely on Alice's input X and all the messages sent by the node β in the dynamic network so far, Alice can simulate the execution of the CONSENSUS protocol on this node against the reference adversary in round r. Formally, we define all unstable nodes as always spoiled for Alice. Among the stable nodes, we define α as always non-spoiled for Alice, while β as always spoiled. The remaining stable nodes are all on the chains. Consider any given chain with stable nodes, and let υ, ν, and ω be the three nodes on the chain, from the top to the bottom:

- If the chain is not leaked and is in the form of $|_*^{2t}$, then ν and ω become spoiled since the beginning of round $t+1$.
- If the chain is not leaked and is in the form of $|_*^{2t+1}$, then ω becomes spoiled since the beginning of round $t+1$.
- A node is non-spoiled for Alice until it becomes spoiled. In particular, all nodes on leaked chains are always non-spoiled.

We similarly define these concepts for Bob: All unstable nodes and the stable node α are always spoiled for Bob. β is never spoiled for Bob. For any chain with stable nodes υ, ν, and ω, from the top to the bottom:

- If the chain is not leaked and is in the form of $|_{2t}^*$, then υ and ν become spoiled since the beginning of round $t+1$.
- If the chain is not leaked and is in the form of $|_{2t+1}^*$, then υ becomes spoiled since the beginning of round $t+1$.
- A node is non-spoiled for Bob until it becomes spoiled. In particular, all nodes on leaked chains are always non-spoiled.

Alice's and Bob's simulated adversary. Since Alice doesn't know Bob's processed input $\mathbf{Y}'$, she is not able to simulate the reference adversary. Instead, Alice will simulate a different adversary (called Alice's simulated adversary) using only her local knowledge, as follows:

- The topology in round 0 is the same as in double-type-1 reference adversary.
- For every chain that is not leaked, Alice's simulated adversary removes the top (bottom) edge at the beginning of round $t+1$ if the chain is in the form of $|_*^{2t}$ (in the form of $|_*^{2t-1}$).
- For every leaked chain, Alice's simulated adversary behaves exactly the same as the double-type-1 reference adversary for this chain. Note that for a leaked chain corresponding to index i, Alice knows both $\mathbf{x}'_i$ and $\mathbf{y}'_i$, and hence can do exactly what the reference adversary does for this chain.

Bob's simulated adversary works in a similar way:

- The topology in round 0 is the same as in double-type-1 reference adversary.
- For every chain that is not leaked, Bob's simulated adversary removes the bottom (top) edge at the beginning of round $t+1$ if the chain is in the form of $|_{2t}^*$ (in the form of $|_{2t-1}^*$).
- For every leaked chain, Bob's simulated adversary behaves exactly the same as the double-type-1 reference adversary for this chain.

Alice's and Bob's simulation. Protocol 2 gives the pseudo-code that Alice and Bob execute to simulate the oracle CONSENSUS protocol $\mathscr{P}$. Alice and Bob will feed public coin flips into $\mathscr{P}$ in the simulation. It will be convenient to imagine that such public coin flips has already been done beforehand, with the outcomes being $C_{\mathscr{P}}$, so that $\mathscr{P}$ can be treated as deterministic given $C_{\mathscr{P}}$.

Protocol 2 is executed by both Alice and Bob, separately. We will explain Protocol 2 as it is executed by Alice. In Protocol 2, Alice simulates total $\frac{q-1}{2}$ rounds of $\mathscr{P}$'s execution. For each node in the dynamic network, Alice maintains the state for $\mathscr{P}$ running on that node. In each round r, Alice first checks all nodes that were non-spoiled for her in round $r-1$ and determines whether each of them is sending or receiving in round r. Note that if a node τ was non-spoiled in round $r-1$ but becomes spoiled in round

```
Procedure Simulate_consensus_protocol()
    foreach $r = 1, \ldots, \frac{q-1}{2}$ do
        msg_pool ← $\emptyset$; msg_to_other_party ← $\emptyset$ ;
        foreach node $\tau$ that was non-spoiled for me in round $r-1$ do
            determine whether $\tau$ is sending or receiving in round $r$ ;
        end
        foreach node $\tau$ that was non-spoiled for me in round $r-1$ and is sending in round $r$ do
            out_msg ← Advance_oracle_protocol_by_one_round($\mathscr{P}$, $C_{\mathscr{P}}$, $\tau$, initial input to
              $\tau$);
            add out_msg to msg_pool;
            if $\tau = \alpha$ or $\tau = \beta$ then msg_to_other_party ← out_msg ;
        end
        send msg_to_other_party to the other party ;
        receive msg_to_other_party from the other party, and add to msg_pool ;
        foreach node $\tau$ that remains to be non-spoiled for me in round $r$ and is receiving in round $r$ do
            in_msg ← {msg|msg $\in$ msg_pool and the sender of msg is $\tau$'s neighbor in round $r$ in my
              simulated adversary} ;
            if in_msg is legal then // see text for the definition of legal
                Advance_oracle_protocol_by_one_round($\mathscr{P}$, $C_{\mathscr{P}}$, $\tau$, initial input to $\tau$,
                  in_msg);
            else
                abort;
            end
        end
    end
end
```

Protocol 2: Simulation protocol executed by Alice and Bob to solve GDC.

r, we will later prove that Alice can still i) determine whether τ is sending or receiving in round r, and ii) simulate $\mathscr{P}$ on τ in round r if τ is sending in round r (since such a τ's behavior is not influenced by potential incoming messages in round r).

Next Alice processes all nodes that were non-spoiled for her in round $r-1$ and are sending in round r. For each such node τ, Alice simulates and advances $\mathscr{P}$ running on that node by one round. To do so, Alice will need to know the initial input to τ, which may be used by the protocol. Note that incoming messages to τ in previous rounds have already been captured in the current state of the protocol, and there is no need for Alice to provide those again. $\mathscr{P}$ on τ will then generate an outgoing message (since τ is sending), which Alice adds to the pool of messages to be delivered. If $\tau = \alpha$, then Alice will further send this message to Bob. Note that for Alice, β is always spoiled and hence τ can never be β.

Finally Alice processes all nodes that remain non-spoiled for her in round r and are receiving in round r. For each such node τ, from the pool of messages to be delivered, Alice chooses all those messages that were sent by τ's neighbors to construct a set `in_msg`. When deciding which nodes are τ's neighbors, Alice uses the dynamic network as determined by Alice's simulated adversary during round r. If `in_msg` is *legal* (defined in the next paragraph), then Alice injects `in_msg` into $\mathscr{P}$ running on τ, and advances $\mathscr{P}$ by one round at Line 17.

Checking whether incoming messages are legal. When the processed input $(\mathbf{X}', \mathbf{Y}')$ is of double-type-$\perp$, the simulated CONSENSUS protocol $\mathscr{P}$ on different nodes in Protocol 2 may be inconsistent, and may not correspond to the execution of $\mathscr{P}$ over any dynamic network. In such a case, the set `in_msg` of incoming message as constructed at Line 15 of Protocol 2 may be corrupted — namely, $\mathscr{P}$ never expects to receiving such a set of incoming messages. While we will not be concerned with the correctness of $\mathscr{P}$ when the processed input is of double-type-$\perp$, we do want to ensure that i) Alice can complete simulation of $\mathscr{P}$ on τ at Line 17 within finite amount of time, and ii) τ will not send excessively large messages in later rounds since Alice will need to forward τ's message to Bob when $\tau = \alpha$.

To ensure this, at Line 16, Alice check whether `in_msg` is *legal* in the following way: Alice exhaustively enumerate all possible dynamic networks with no more than $3n + 4$ nodes and no more than r rounds, and all possible initial values of the nodes in the network. Alice next simulates the execution of $\mathscr{P}$ with $C_{\mathscr{P}}$ under each such setting. Note that all such simulations are done unilaterally by Alice and are completely independent of the simulation done by Alice and Bob together. Alice then checks whether `in_msg` matches any set of incoming messages to any node φ in any such simulation in round r, where φ has the same state as τ at the end of round $r - 1$ and has the same initial input as τ. Such checking is possible since communication complexity lower bounds hold irrespective of the computational power of Alice and Bob. If there is no such node φ, then Alice claims that `in_msg` is not legal and will abort Protocol 2.

Note that `in_msg` being legal does not necessarily mean that the simulated CONSENSUS protocol $\mathscr{P}$ on different nodes in Protocol 2 are consistent — it only implies that when feeding `in_msg` into τ in round r, the simulation of $\mathscr{P}$ on τ will, intuitively, "stay on track".

E.2 Properties of Our Simulation

Intuitively, the following lemma proves that regardless of whether the simulation of $\mathscr{P}$ on different nodes are consistent or not, the simulation will always terminate and will not incur too much communication:

Lemma 9. *For any* CONSENSUS *protocol $\mathscr{P}$, there exists positive constant c such that for all n, q, and $C_{\mathscr{P}}$, Protocol 2 always terminates within finite amount of time and incurs at most $cq \log n$ bits of communication between Alice and Bob.*

Proof. For any node τ and any round r, we say that the state of $\mathscr{P}$ running on τ (as maintained by Alice or Bob using Protocol 2) is *legal* if there exists some dynamic network of no more than n nodes, some initial values to the nodes in this dynamic network, and some node φ in this dynamic network whose initial value is the same as τ's, such that when running $\mathscr{P}$ on this dynamic network with $C_{\mathscr{P}}$, the state of $\mathscr{P}$ on φ in round r is exactly the same as the state of $\mathscr{P}$ on τ as maintained by Alice or Bob using Protocol 2.

We next prove via an induction that for all node τ and all round r, the state of $\mathscr{P}$ running on τ in round r is legal. The case for $r = 0$ is trivial. Assume the claim holds for round $r - 1$, and consider any node τ. If τ is sending in round r, it is easy to see that the state of the CONSENSUS protocol running on τ will continue to be legal. If τ is receiving in round r, then Line 16 of Protocol 2 explicitly ensures that the state will be legal, before continuing.

Next since the state of $\mathscr{P}$ on all node τ are always legal in all round r, it immediately means that simulated $\mathscr{P}$ running on τ will complete its execution for round r within finite amount of time at Line 8 and Line 17 of Protocol 2. Furthermore at Line 10, the size of `out_msg` (and hence the size of `msg_to_other_party`) must satisfy the maximum allowed message size (i.e., $O(\log n)$) for a network with $\Theta(n)$ nodes. The lemma follows since in each round, Alice and Bob only communicates once at Line 12 by sending `msg_to_other_party` to the other party. □

Consider any processed input $(\mathbf{X}', \mathbf{Y}')$ that is either double-type-0 or double-type-1, any given set of indices that are leaked by the leaker, the corresponding reference adversary for $(\mathbf{X}', \mathbf{Y}')$ and this set of leaked indices, any given CONSENSUS protocol $\mathscr{P}$, any given public coin flip outcomes $C_{\mathscr{P}}$ that Alice and Bob generate to feed into $\mathscr{P}$, the dynamic network as determined by this reference adversary under such $C_{\mathscr{P}}$, and any given initial values for all the nodes in this dynamic work. We define the *reference execution* be the execution of $\mathscr{P}$ under such coin flips, the above initial values of the nodes, and the above dynamic network. (Such a reference execution will be deterministic since all coins have been flipped.)

Lemma 10. *Consider any given reference execution, any node τ in the reference execution, and any round $1 \leq r \leq \frac{q-1}{2}$. Let S be the set of nodes that are τ's neighbors under Alice's (Bob's) simulated adversary in round r and are sending in the reference execution in round r. Let S' be the set nodes that are τ's neighbors under the reference adversary in round r and are sending in the reference execution in round r. If τ is receiving in the reference execution in round r and if τ is non-spoiled for Alice (Bob) in round r, then:*

- *$S = S'$.*
- *For all $\varphi \in S$, either φ is non-spoiled in round $r-1$ for Alice (Bob) or $\varphi = \beta$ ($\varphi = \alpha$).*

Proof. It suffices to prove the lemma for Alice. Throughout our proof, we will extensively leverage the fact the since we are considering a reference execution, the reference adversary must be ether double-type-0 or double-type-1. Hence whenever we consider φ's neighbors under the reference adversary, we should keep in mind that the reference adversary is not double-type-$\perp$. Define T to be the set of τ's neighbors under Alice's simulated adversary in round r, and T' to be the set of τ's neighbors under the reference adversary in round r. Obviously, $S \subseteq T$ and $S' \subseteq T'$. Furthermore, $T = T'$ implies that $S = S'$.

Since τ is non-spoiled, τ must be a stable node. Such τ can either be the special node α or can be a node on any of the chains consisting of stable nodes. If $\tau = \alpha$, then τ's neighbors under Alice's simulated adversary are always exactly the same as τ's neighbors under the reference adversary, and all these neighbors are never spoiled for Alice. Hence the lemma holds when $\tau = \alpha$.

Next we consider the case where τ is on some chain consisting of stable nodes. If the chain is leaked, then regardless where τ is on the chain, we have $T = T'$. Furthermore, since nodes on a leaked chain are always non-spoiled, a node in T must be either β or some non-spoiled node. Hence the lemma holds. The remainder of our proof covers the case where τ is on some non-leaked chain consisting of stable nodes. Let υ, ν, and ω be the three nodes, from top to the bottom, on any such chain. We exhaustively enumerate all possibilities, depending on what kind of chain it is. Let t be any integer where $0 \leq t \leq \frac{q-1}{2}$:

- For a $|_{2t+1}^{2t}$ chain, υ is always non-spoiled, and ν and ω are non-spoiled iff $r < t+1$:
  - For node υ, we exhaustively enumerate all scenarios: i) If $r < t+1$, then $T = T' = \{\alpha, \nu\}$. By definition, both α and ν are non-spoiled in round $r-1$. ii) If $r > t+1$, then $T = T' = \{\alpha\}$. By definition, α is non-spoiled in round $r-1$. iii) If $r = t+1$ and ν is sending in round r, then $T = T' = \{\alpha\}$ and α is non-spoiled in round $r-1$. iv) If $r = t+1$ and ν is receiving in round r, then $T' = \{\alpha, \nu\}$ and $T = \{\alpha\}$. If α is receiving in round r, we have $S = S' = \emptyset$. Otherwise, $S' = \{\alpha\} = S$. By definition, α is non-spoiled in round $r-1$.
  - For node ν and $r < t+1$, we have $T = T' = \{\upsilon, \omega\}$, and both nodes are non-spoiled in round $r-1$.
  - For node ω and $r < t+1$, we have $T = T' = \{\nu, \beta\}$, and ν is non-spoiled in round $r-1$.
- For a $|_{2t-1}^{2t}$ chain, υ is always non-spoiled, and ν and ω are non-spoiled iff $r < t+1$:
  - For node υ, we exhaustively enumerate all scenarios: i) If $r < t+1$, then $T = T' = \{\alpha, \nu\}$. By definition, both α and ν are non-spoiled in round $r-1$. ii) If $r \geq t+1$, then $T = T' = \{\alpha\}$. By definition, α is non-spoiled in round $r-1$.
  - For node ν and $r < t+1$, we have $T = T' = \{\upsilon, \omega\}$, and both nodes are non-spoiled in round $r-1$.
  - For node ω and $r < t+1$, we have $T = T' = \{\nu, \beta\}$, where ν is non-spoiled in round $r-1$.
- For a $|_{2t}^{2t+1}$ chain, υ and ν are always non-spoiled, and ω is non-spoiled iff $r < t+1$:

- For node υ, $T = T' = \{\alpha, \nu\}$. By definition, both α and ν are non-spoiled in round $r - 1$.
- For node ν, we exhaustively enumerate all scenarios: i) If $r < t + 1$, we have $T = T' = \{\upsilon, \omega\}$, and both nodes are non-spoiled in round $r - 1$. ii) If $r > t + 1$, then $T = T' = \{\upsilon\}$. By definition, υ is non-spoiled in round $r - 1$. iii) If $r = t + 1$, recall that we only need to consider the case where ν is receiving in round r. Thus, $T = T' = \{\upsilon, \omega\}$, and both υ and ω are non-spoiled in round $r - 1$.
- For node ω and $r < t + 1$, we have $T = T' = \{\nu, \beta\}$, where ν is non-spoiled in round $r - 1$.

- For a $|_{2t}^{2t-1}$ chain, υ and ν are always non-spoiled, and ω is non-spoiled iff $r < t$:
  - For node υ, $T = T' = \{\alpha, \nu\}$. By definition, both α and ν are non-spoiled in round $r - 1$.
  - For node ν, we exhaustively enumerate all scenarios: i) If $r \leq t$, we have $T = T' = \{\upsilon, \omega\}$, and both nodes are non-spoiled in round $r - 1$. ii) If $r \geq t + 1$, then $T = T' = \{\upsilon\}$. By definition, υ is non-spoiled in round $r - 1$.
  - For node ω and $r < t$, we have $T = T' = \{\nu, \beta\}$, where ν is non-spoiled in round $r - 1$.
- For a $|_{q-1}^{q-1}$ chain, υ, ν, and ω are always non-spoiled:
  - For node υ, we have $T = T' = \{\alpha, \nu\}$. By definition, both α and ν are non-spoiled in round $r - 1$.
  - For node ν, we have $T = T' = \{\upsilon, \omega\}$. By definition, both υ and ω are non-spoiled in round $r - 1$.
  - For node ω, we have $T = T' = \{\nu, \beta\}$. By definition, ν is non-spoiled in round $r - 1$.

Hence the lemma holds in all above cases. □

Let τ be any node and r be any round where $0 \leq r \leq \frac{q-1}{2}$. We say that an outgoing message from τ (or a set of incoming messages to τ) as determined in round r of Protocol 2 in Line 8 (Line 15) is *correct* if it is exactly the same as τ's outgoing message (incoming messages) in the reference execution in round r.

Lemma 11. *Consider any given reference execution, any node τ in the reference execution, and any r where $1 \leq r \leq \frac{q-1}{2}$.*

- *If τ was non-spoiled for Alice (Bob) in round $r - 1$ and is sending in the reference execution in round r, then i) τ will be determined as sending in round r by Alice (Bob) in Line 5 of Protocol 2, and ii) τ's outgoing message as determined by round r of Alice's (Bob's) Protocol 2 at Line 8 is correct.*
- *If τ was non-spoiled for Alice (Bob) in round $r - 1$ and is receiving in the reference execution in round r, then τ will be determined as receiving in round r by Alice (Bob) in Line 5 of Protocol 2. Furthermore if such a τ continues to be non-spoiled in round r, the set of τ's incoming messages as determined by round r of Alice's (Bob's) Protocol 2 at Line 15 is correct.*

Furthermore, Line 19 in Protocol 2 will not be executed in round r.

Proof. The last claim that Line 19 will not be executed does not need to be proved separately — as long as we can prove the other claims in the lemma, the last claim will directly follow. The reason is that Line 19 can only be executed when `in_msg` is not legal in Line 16. However, if the previous claims in the lemma hold, then `in_msg` must be legal. Thus we will not separately prove the last claim.

It suffices to prove the lemma for Alice. We prove via an induction on r. The induction base for $r = 0$ is trivial since τ by definition is in the receiving state in that round, and the set of incoming messages is empty. For the inductive step, suppose that the lemma holds for all rounds before round r. We prove the lemma for round r.

First, consider the case where τ is non-spoiled for Alice in round $r-1$ and is sending in the reference execution in round r. By definition, τ must be non-spoiled in round 0 through round $r-1$. By the inductive hypothesis, in Protocol 2 for all previous rounds where τ is receiving, Alice was able to feed the correct sets of incoming messages into τ. Since everything is deterministic, in Line 5 Alice can determine that τ must be sending in round r, and the outgoing message from τ in round r as generated by Protocol 2 must be correct as well.

Next consider the case where τ is non-spoiled for Alice in round $r-1$ and is receiving in the reference execution in round r. By same argument as earlier, in Line 5 Alice must be able to determine that τ is in a receiving state in round r.

If τ continues to be non-spoiled in round r, let S be the set of node that are τ's neighbors in Alice's simulated adversary in round r and are sending in the reference execution in round r. Consider the set `in_msg` of messages that Alice constructs as τ's incoming messages in Line 15. We claim that `in_msg` is the same as the set of the messages sent by all the nodes in S in the reference execution. It is easy to see that for any node $\varphi \notin S$, the outgoing message from φ will not be added to `in_msg` in Line 15 since by definition of S, φ is not τ's neighbor in Alice's simulated adversary in round r. Hence to prove the claim, we only need to show that for any node $\varphi \in S$, Alice gets the correct outgoing message from φ and adds this message to `msg_pool` at either Line 8 or Line 13. As long as this message is in `msg_pool`, it will be later added to `in_msg` in Line 15 since φ is τ's neighbor in Alice's simulated adversary. If $\varphi \in S$ and $\varphi = \beta$, then φ is sending in round r in the reference execution and φ is non-spoiled for Bob in round $r-1$. By our earlier argument, at Line 8, Bob will generate the correct outgoing message from φ in round r. Such a message will then be forwarded to Alice at Line 12, and then added to `msg_pool` at Line 13. If $\varphi \in S$ and $\varphi \neq \beta$, by Lemma 10, φ must be non-spoiled in round $r-1$. Again by our earlier arguments, at Line 8, Alice will generate the correct outgoing message from φ in round r, and add such a message to `msg_pool` at Line 8.

So far we have proved that `in_msg` is the same as the set of the messages sent by all the nodes in S in the reference execution. Let S' be the set of nodes that are τ's neighbors in the reference adversary in round r and are sending in the reference execution in round r. Lemma 10 tells us that $S = S'$, which immediately implies that `in_msg` is correct and hence completes the proof. □

E.3 Prove Theorem 4 via the Simulation

Theorem 4. *If the nodes only know a poor estimate m' for m such that $|\frac{m'-m}{m}|$ reaches $\frac{1}{3}$ or above, then a $\frac{1}{10}$-error* CONSENSUS *protocol for dynamic networks with oblivious adversaries must have a time complexity of $\Omega(d + m^{\frac{1}{12}})$ rounds.*

Proof. Consider any given $\frac{1}{10}$-error CONSENSUS protocol $\mathscr{P}$ with time complexity of $\text{tc}(d, m)$ rounds over average coin flips, when running over dynamic networks controlled by oblivious adversaries and with d diameter and m nodes. We aim to prove that $\text{tc}(d, m) = \Omega(d + m^{\frac{1}{12}})$. To do so, we will prove that $\text{tc}(8, m) \geq m^{\frac{1}{12}}$ for all sufficiently large m. This proof will trivially extend to $\text{tc}(d, m)$ for all $d > 8$. Combining with the fact that $\text{tc}(d, m) = \Omega(d)$ then completes the proof.

Consider the constants c_1 and c_2 in Theorem 3 (for $\delta = \frac{2}{5}$), the constant c in Lemma 9, and the following inequalities:

$$\frac{m-4}{3} \geq 60(20m^{\frac{1}{12}} + 20)\ln(20m^{\frac{1}{12}} + 20) \quad (6)$$

$$\frac{c_1\sqrt{\frac{m-4}{3}}}{15(20m^{\frac{1}{12}} + 20)^{4.5}\log^3 m} \geq 2c(20m^{\frac{1}{12}} + 20) + c_2 \quad (7)$$

It is easy to see that there must exist constant $c_3 > 0$ such that for all $m \geq c_3$, both inequalities hold. We will prove that $\text{tc}(8, m) \geq m^{\frac{1}{12}}$ for all $m \geq c_3$.

Assume by contradiction that there exists some $m \geq c_3$ such that $\text{tc}(8, m) < m^{\frac{1}{12}}$. We will proceed with the reduction from GDC and eventually obtain a contradiction. Let $n = \frac{m-4}{3}$, $q = 20\text{tc}(8, m) + 20$, and $g = 15q \ln q$. We later will need to invoke Lemma 8. Note that these parameters do satisfy the requirements in Lemma 8, since by Equation 6:

$$n \ = \ \frac{m-4}{3} \geq 60(20m^{1/12} + 20)\ln(20m^{1/12} + 20) > 60q\ln q = 4g$$

Also note that since $n > 4g$, we have $n > q$, and hence the $\text{GDC}_n^{g,q}$ problem is well-defined.

To solve the $\text{GDC}_n^{g,q}(X, Y)$ problem with our leaker, Alice and Bob will simulate the execution of $\mathscr{P}$. Alice and Bob will first generate public coin flip outcomes (denoted as $\mathbf{C}_{\mathscr{P}}$) to feed into $\mathscr{P}$. This effectively makes $\mathscr{P}$ deterministic. Alice and Bob set $\hat{m} = \frac{2}{3}m = \frac{2}{3}(3n + 4)$, and feeds $\hat{m}$ into $\mathscr{P}$ as an estimate of the total number of nodes, if $\mathscr{P}$ needs such an estimate. As we will quickly see, the number of nodes in the dynamic network will be either m or $m/2$. Hence obviously, such $\hat{m}$ satisfies both $|\frac{\hat{m}-m}{m}| = \frac{1}{3}$ and $|\frac{\hat{m}-m/2}{m/2}| = \frac{1}{3}$.

Alice and Bob will simulate $\mathscr{P}$ twice on two different dynamic networks, using the same $\mathbf{C}_{\mathscr{P}}$. The ids of the nodes in the dynamic network will be determined by the adversary and then given to $\mathscr{P}$ as inputs.

- **First simulation.** The first simulation is based on the processed input $(\mathbf{X}', \mathbf{Y}')$. We first assign initial values and ids to the nodes under the corresponding reference adversary. All stable nodes has initial values 0. Order all the stable nodes into a total order by some arbitrary criterion, and then assign them ids from 1 to $\frac{3}{2}n + 2$. Note that Alice and Bob can determine the initial values and the ids of all the stable nodes without the need of communication, since these initial values and ids do not depend $(\mathbf{X}', \mathbf{Y}')$.

  If there are unstable nodes (i.e., when $(\mathbf{X}', \mathbf{Y}')$ is of double-type-0), then they will all have initial values 1. The unstable nodes will have ids from $\frac{3}{2}n + 3$ to $3n + 4$, by the total ordering as described later for the stable nodes in the second simulation. Note that by our definition, a non-spoiled node must be stable, and hence as explained in the previous paragraph, Alice and Bob know the initial values and ids of all their respective non-spoiled nodes. Alice and Bob then proceed with the first simulation using Protocol 2. By Lemma 9, such simulation must complete within finite time.

- **Second simulation.** For the second simulation, we construct a second processed input $(\mathbf{X}'', \mathbf{Y}'')$ by swapping the first half and second half of $(\mathbf{X}', \mathbf{Y}')$. Specifically, we set $\mathbf{X}''_i = \mathbf{X}'_{i+\frac{n}{2}}$ and $\mathbf{Y}''_i = \mathbf{Y}'_{i+\frac{n}{2}}$ for $1 \leq i \leq \frac{n}{2}$, and $\mathbf{X}''_i = \mathbf{X}'_{i-\frac{n}{2}}$ and $\mathbf{Y}''_i = \mathbf{Y}'_{i-\frac{n}{2}}$ for $\frac{n}{2} + 1 \leq i \leq n$. It is trivial to see that $(\mathbf{X}'', \mathbf{Y}'')$ and $(\mathbf{X}', \mathbf{Y}')$ must be of the same double-type. The second simulation is based on the processed input $(\mathbf{X}'', \mathbf{Y}'')$. In particular, if $(\mathbf{X}'', \mathbf{Y}'')$ is of double-type-1, then the reference adversary will use the first half $(\mathbf{X}'', \mathbf{Y}'')$ to construct the topology. For clarity, we rename the nodes α, β, γ, and λ to be α', β', γ', and λ' in the second simulation.

  We still need to assign initial values and ids to the nodes under the corresponding reference adversary. All stable nodes have initial values of 1, and all unstable nodes have initial values of 0. Order all the stable nodes into a total order by some arbitrary criterion. These nodes are then assigned ids from $\frac{3}{2}n + 3$ to $3n + 4$. Note that the initial topology among these stable nodes will be exactly the same as the initial topology among the unstable nodes in the first simulation. As mentioned earlier, we used the same total ordering used here to order the unstable nodes in the first simulation, if there were unstable nodes there.

  If there are unstable nodes (i.e., when $(\mathbf{X}'', \mathbf{Y}'')$ is of double-type-0), then again, the initial topology among these unstable nodes will be exactly the same as the initial topology among the stable nodes in the first simulation. We will use the same total ordering used in the first simulation to order these unstable nodes, and assign them ids from 1 to $\frac{3}{2}n + 2$.

Again, in the second simulation, Alice and Bob know the initial values and ids of all their respective non-spoiled nodes. Alice and Bob then proceed with the second simulation using Protocol 2. By Lemma 9, such simulation must complete within finite time.

- **Generating an output.** Alice monitors when α decides in the first simulation and when α' decides in the second simulation. If they *both* decide by round $10\text{tc}(8, m)$, Alice outputs 1 for the original GDC problem. Otherwise Alice outputs 0. Note that if either of the simulation aborts at Line 19 of Protocol 2, Alice will output 0 as well.

- **Correctness of Alice's output.** If $\text{GDC}(X, Y) = 1$, then Lemma 8 tells us that $(\mathbf{X}', \mathbf{Y}')$ is of double-type-1 with probability at least $1 - \frac{1}{q}$. Since $(\mathbf{X}'', \mathbf{Y}'')$ and $(\mathbf{X}', \mathbf{Y}')$ must be of the same double-type, with at least such probability, both of them are of double-type-1. When both of them are of double-type-1, Lemma 12 later proves that with probability at least $1 - \frac{1}{5}$, α in the first simulation and α' in the second simulation both decide within $10\text{tc}(8, m)$ rounds. This will make Alice generate the correct output 1. Hence Alice generates the correct output 1 with probability at least $(1 - \frac{1}{q})(1 - \frac{1}{5}) \geq (1 - \frac{1}{20})(1 - \frac{1}{5}) > 1 - \frac{2}{5}$.

  If $\text{GDC}(X, Y) = 0$, then by Lemma 8 and similar argument as before, we know that with at least $1 - \frac{1}{q}$ probability, both $(\mathbf{X}', \mathbf{Y}')$ and $(\mathbf{X}'', \mathbf{Y}'')$ are of double-type-0. When both of them are of double-type-0, Lemma 13 later proves that with probability at most $\frac{3}{10}$, α in the first simulation and α' in the second simulation both decide within $10\text{tc}(8, m)$ rounds. Hence Alice's output is correct with probability at least $(1 - \frac{1}{q})(1 - \frac{3}{10}) \geq (1 - \frac{1}{20})(1 - \frac{3}{10}) > 1 - \frac{2}{5}$.

- **From communication complexity to time complexity.** We have proved so far that Alice and Bob can solve $\text{GDC}_n^{g,q}$ with $\frac{2}{5}$ error, by simulating $\mathscr{P}$ twice. Lemma 9 tells us that there exists some constant $c > 0$, such that in each simulation, Alice and Bob never incur more than $cq \log n$ bits of communication. Hence Alice and Bob can solve $\text{GDC}_n^{g,q}$ with no more than $2cq \log n$ bits of communication. By the lower bound in Theorem 3, we know that there exist constants c_1 and c_2 such that all $\frac{2}{5}$-error protocols for solving $\text{GDC}_n^{g,q}$ have a communication complexity of at least $\frac{c_1\sqrt{n}}{gq^{3.5}\log q} - c_2 \log \frac{\sqrt{n}}{gq^{1.5}\log q}$ bits, over average coin flips. This implies:

$$\begin{aligned}
& 2cq \log n \geq \frac{c_1\sqrt{n}}{gq^{3.5}\log q} - c_2 \log \frac{\sqrt{n}}{gq^{1.5}\log q} \\
\Rightarrow\quad & 2cq \geq \frac{c_1\sqrt{n}}{gq^{3.5}\log q \log n} - \frac{c_2}{\log n}(\frac{1}{2}\log n - \log(gq^{1.5}\log q)) > \frac{c_1\sqrt{n}}{gq^{3.5}\log q \log n} - c_2 \\
\Rightarrow\quad & 2cq + c_2 > \frac{c_1\sqrt{n}}{15q^{4.5}\ln q \log q \log n} \geq \frac{c_1\sqrt{n}}{15q^{4.5}\log^3 m} \quad \text{(since } q < n < m\text{)} \\
\Rightarrow\quad & 2c(20\text{tc}(8, m) + 20) + c_2 > \frac{c_1\sqrt{\frac{m-4}{3}}}{15(20\text{tc}(8, m) + 20)^{4.5}\log^3 m} \\
\Rightarrow\quad & 2c(20m^{\frac{1}{12}} + 20) + c_2 > \frac{c_1\sqrt{\frac{m-4}{3}}}{15(20m^{\frac{1}{12}} + 20)^{4.5}\log^3 m} \quad \text{(since } m^{\frac{1}{12}} > \text{tc}(8, m)\text{)}
\end{aligned}$$

The last inequality contradicts with Equation 7, which completes our proof by contradiction.

□

Lemma 12. *Consider any given processed inputs (X', Y') and (X'', Y'') in the proof of Theorem 4, and the corresponding first simulation and second simulation. If both processed inputs are of double-type-1, then α in the first simulation and α' in the second simulation will both decide within $10tc(8, m)$ rounds with probability at least $1 - \frac{1}{5}$, where the probability is taken over the coin flips of both the protocol and the adversary. Furthermore, neither the first simulation nor the second simulation will abort at Line 19 of Protocol 2.*

Proof. Consider the first simulation where the reference adversary $\mathscr{A}$ is based on (X', Y'). Since (X', Y') is of double-type-1, it is easy to verify that the dynamic network as generated by $\mathscr{A}$ has a diameter of no more than 8, under all possible coin flips of the CONSENSUS protocol $\mathscr{P}$ and of the reference adversary $\mathscr{A}$. We want to increase the diameter of the dynamic network to exactly 8. Recall that $\mathscr{P}$ is only simulated for round 1 through $\frac{q-1}{2}$. Given this, increasing the diameter to exactly 8 is trivial: Starting from round $\frac{q-1}{2} + 1$, we let the dynamic network's topology to be some fixed topology such that the resulting (dynamic) diameter of the dynamic network is exactly 8. Since the simulation has already stopped by round $\frac{q-1}{2}$, whatever we do after that will not impact the simulation in any way. (If we want to reason about $\text{tc}(d, m)$ for $d > 8$, then we should increase the diameter to exactly d, which is also trivial to achieve using the above approach.) In the next, when we refer to $\mathscr{A}$ (which was originally defined only for the first $\frac{q-1}{2}$ rounds), we will include the above topology starting from round $\frac{q-1}{2} + 1$ as well.

Directly followed from the definition, $\mathscr{A}$ is a sanitized adaptive adversary. Let the *cost* of $\mathscr{P}$ be the number of rounds before termination. By Theorem 2, we know that there exists some deterministic oblivious adversary $\mathscr{B}$ such that $\mathscr{P}$'s expected cost under $\mathscr{B}$ is no smaller than its expected cost under $\mathscr{A}$. Furthermore also by Theorem 2, we know that for any coin flip outcomes of $\mathscr{P}$, there exist coin flip outcomes of $\mathscr{A}$, such that the decisions made by $\mathscr{B}$ are the same as the decisions made by $\mathscr{A}$ under those coin flip outcomes. Thus since the dynamic network constructed by $\mathscr{A}$ always has a diameter of 8, we know that the dynamic network constructed by $\mathscr{B}$ has a diameter of 8 as well.

When running against any given oblivious adversary where the corresponding dynamic network has a diameter of 8 and has m nodes, $\mathscr{P}$ promises to terminate within $\text{tc}(8, m)$ rounds over average coin flips. Hence $\mathscr{P}$ must terminate within $\text{tc}(8, m)$ rounds over average coin flips when running against $\mathscr{B}$. In turn, $\mathscr{P}$ must terminate within $\text{tc}(8, m)$ rounds over average coin flips (of both $\mathscr{P}$ and $\mathscr{A}$) when running against $\mathscr{A}$. By Markov inequality, $\mathscr{P}$ terminates within $10\text{tc}(8, m)$ rounds with probability at least $\frac{9}{10}$ when running against $\mathscr{A}$.

Since $10\text{tc}(8, m) \leq \frac{q-1}{2}$ and since α is always non-spoiled for Alice, Lemma 11 tells us that at Line 8 of Protocol 2, the outgoing message of α as determined by Alice must be correct (i.e., the same as the corresponding outgoing message in the reference execution). Without loss of generality, assume that when α decides, it sends a special message. Hence if α decides within $10\text{tc}(8, m)$ rounds in the reference execution, Alice must be able to observe that.

By same argument, since (X'', Y'') is of double-type-1, $\mathscr{P}$ must terminate within $10\text{tc}(8, m)$ rounds with probability at least $\frac{9}{10}$ when running against our reference adversary in the second simulation. Again by Lemma 11, Alice can observe when α' decides. A simple union bound shows that with probability at least $1 - \frac{1}{5}$, Alice will be able to observe that both α and α' decide within $10\text{tc}(8, m)$ rounds.

Finally, Lemma 11 also confirms that neither the first simulation nor the second simulation will abort at Line 19 of Protocol 2. □

Lemma 13. *Consider any given processed inputs (X', Y') and (X'', Y'') in the proof of Theorem 4, and the corresponding first simulation and second simulation. If both processed inputs are of double-type-0, then α in the first simulation and α' in the second simulation will both decide within $10tc(8, m)$ rounds with probability at most $\frac{3}{10}$, where the probability is taken over the coin flips of both the protocol and the adversary. Furthermore, neither the first simulation nor the second simulation will abort at Line 19 of Protocol 2.*

Proof. We first prove that when the oracle CONSENSUS protocol $\mathscr{P}$ runs against our reference adversary in the first simulation, α and γ both decide within $10\text{tc}(8, m)$ rounds with probability at most $\frac{3}{10}$.

Let $\mathscr{A}$ be our reference adversary in the first simulation, which is a sanitized adaptive adversary. We construct another sanitized adaptive adversary $\mathscr{B}$, in the following way. Under the given initial values assigned in the first simulation, under any given coin flip outcomes $C_{\mathscr{P}}$ of $\mathscr{P}$, and under any given coin flip outcomes $C_{\mathscr{A}}$ of $\mathscr{A}$, let $\mathcal{G}$ be the resulting (unique) dynamic network. Under *all* possible initial values to the nodes, when $\mathscr{P}$'s coin flip outcomes are $C_{\mathscr{P}}$ and when $\mathscr{B}$'s coin flip outcomes are $C_{\mathscr{A}}$, $\mathscr{B}$ constructs the dynamic network the same as $\mathcal{G}$. It is easy to verify that since $\mathscr{A}$ is a sanitized adaptive adversary, $\mathscr{B}$ must be a sanitized adaptive adversary as well.

For coin flip outcomes $C_{\mathscr{P}}$ of $\mathscr{P}$ and coin flip outcomes $C_{\mathscr{A}}$ of $\mathscr{A}$, define $\text{cost}(\mathscr{P}, \mathscr{A}, C_{\mathscr{P}}, C_{\mathscr{A}})$ to be 0 if the protocol's output is correct and 1 otherwise. Since $\mathscr{A}$ is a sanitized adaptive adversary, Theorem 2 tells us that there exists some deterministic oblivious adversary such that the protocol's cost under this deterministic oblivious adversary is no smaller than its cost under $\mathscr{A}$. On the other hand, when executing against any given oblivious adversary and with any initial values, $\mathscr{P}$ promises to have at most $\frac{1}{10}$ error over average coin flips. Hence when running against $\mathscr{A}$ and with any initial values, $\mathscr{P}$ must have at most $\frac{1}{10}$ error over average coin flips (of both $\mathscr{P}$ and $\mathscr{A}$). By same argument, when running against $\mathscr{B}$ and with any initial values, $\mathscr{P}$ must have at most $\frac{1}{10}$ error.

Let $\mathcal{I}$ denote the CONSENSUS instance in the first simulation. We will construct two additional CONSENSUS instances, in the following way. The CONSENSUS instance $\mathcal{I}_0$ is the same as $\mathcal{I}$ except that i) all nodes in $\mathcal{I}_0$ have initial values of 0, and ii) $\mathcal{I}_0$ is under adversary $\mathscr{B}$ instead of $\mathscr{A}$. We similarly construct $\mathcal{I}_1$ under adversary $\mathscr{B}$ where all nodes have initial values of 1. Now consider any given coin flip outcomes $C_{\mathscr{P}}$ of $\mathscr{P}$ and coin flip outcomes $C_{\mathscr{A}}$ of the adversary (which is either $\mathscr{A}$ or $\mathscr{B}$). Note that under given $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$, the dynamic networks in the three instances as determined by their respective adversaries are exactly the same. We claim that if α and γ both decide within $10\text{tc}(8, m)$ rounds, then under $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$, $\mathscr{P}$ must err in either $\mathcal{I}$ or $\mathcal{I}_0$ or $\mathcal{I}_1$.

To see why, we consider two cases. If $\mathscr{P}$ err in $\mathcal{I}$, we are done. If $\mathscr{P}$ does not err in $\mathcal{I}$, without loss of generality, let the decision value be 1. This means that both α and γ decide on 1 within $10\text{tc}(8, m)$ rounds in $\mathcal{I}$. Next consider α's behavior in $\mathcal{I}_0$. Note that $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$ have all been fixed, and also that $\mathcal{I}$ and $\mathcal{I}_0$ have exactly the same dynamic network. The only difference between $\mathcal{I}$ and $\mathcal{I}_0$ is the initial values. Since $q \geq 10\text{tc}(8, m)$, by the way we construct $\mathscr{A}$ and $\mathscr{B}$, it is easy to verify that for all nodes τ where $(\tau, 0) \rightsquigarrow (\alpha, 10\text{tc}(8, m))$, τ has the same initial value of 0 in both $\mathcal{I}$ and $\mathcal{I}_0$. Only a node τ such that $(\tau, 0) \rightsquigarrow (\alpha, 10\text{tc}(8, m))$ may influence α's behavior by round $10\text{tc}(8, m)$. Thus for every node τ that can influence α's behavior by round $10\text{tc}(8, m)$, τ has the same initial value in $\mathcal{I}$ and $\mathcal{I}_0$. Hence α's behavior in $\mathcal{I}$ and $\mathcal{I}_0$ must be the same. Since α decides on 1 by round $10\text{tc}(8, m)$ in $\mathcal{I}$, it must also decide on 1 by round $10\text{tc}(8, m)$ in $\mathcal{I}_0$. But such a decision value is wrong in $\mathcal{I}_0$.

We have proved that for every $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$, if α and γ in the first simulation both decide within $80\text{tc}(m)$ rounds, then $\mathscr{P}$ must err in one of the 3 instances. On the other hand, as shown earlier, in each of the instances, $\mathscr{P}$ must have at most $\frac{1}{10}$ error, over average $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$. Hence α and γ in the first simulation both decide within $10\text{tc}(8, m)$ rounds with probability at most $\frac{3}{10}$.

So far we have proved that when $\mathscr{P}$ runs against our reference adversary in the first simulation, α and γ both decide within $10\text{tc}(8, m)$ rounds with probability at most $\frac{3}{10}$. We call this as the *first reference execution*. Next we consider running $\mathscr{P}$ against our reference adversary in the second simulation (which we call the *second reference execution*), and consider the node α' there. One can verify that when both (X', Y') and (X'', Y'') are of double-type-0, then under the same $C_{\mathscr{P}}$ and $C_{\mathscr{A}}$, the first reference execution and the second reference execution are "isomorphic": A node with a certain id in the first reference execution must have exactly the same behavior as the node with that id in the second reference execution. This means that the behavior of node α' in the second reference execution must be exactly the same as the behavior of node γ in the first reference execution. Together with our earlier arguments, this means that with probability at most $\frac{3}{10}$, α in the first reference execution and α' in the second reference execution both decide within $10\text{tc}(8, m)$ rounds.

Finally, since $10\text{tc}(8, m) \leq \frac{q-1}{2}$ and since α and α' are always non-spoiled for Alice, Lemma 11 tells us that at Line 8 of Protocol 2, the outgoing messages of α and α' as determined by Alice must be correct (i.e., the same as the corresponding outgoing messages in the reference execution). Hence if α and α' decide within $10\text{tc}(8, m)$ rounds in the respective reference executions, Alice will observe that. Lemma 11 also confirms that neither the first simulation nor the second simulation will abort at Line 19 of Protocol 2. □

F Proof for Theorem 5

The section proves Theorem 5. Theorem 5 is not a surprising result, and we do not claim it as a major contribution. We include this proof here mainly for completeness — while the overall approach is quite natural, the proof does involve some tedious and complicated steps, in order to get a relatively strong result. In particular, a weaker form Theorem 5 (i.e., by requiring μ in the theorem to be much larger) can be proved via a less complicated approach. But this weaker form would negatively impact the final asymptotic results in this paper.

Notations. We introduce some additional notations to be used in this section. For any μ where 2μ is a positive integer, recall that $\mathbb{B}(\mu)$ is defined to be the binomial distribution describing the number of heads obtained when flipping 2μ independent fair coins. Define continuous distribution $\widetilde{\mathbb{B}}(\mu)$ to be the distribution whose density function is $\frac{1}{2^{2\mu}}\binom{2\mu}{\lfloor x \rfloor}$ for $0 \leq x < 2\mu + 1$, and 0 for other x values. It is easy to verify that $\widetilde{\mathbb{B}}(\mu)$ is indeed a distribution. Intuitively, $\widetilde{\mathbb{B}}(\mu)$ is the continuous version of $\mathbb{B}(\mu)$. Define $\mathbb{N}(\mu)$ to be the normal distribution whose mean is μ and whose variance is $\mu/2$. For any distribution $\mathbb{D}$, $f_{\mathbb{D}}$ is the distribution's density function.

Recall that for any two given distributions $\mathbb{D}$ and $\mathbb{D}'$, with $\mathcal{D}$ being the sample space of $\mathbb{D}$ and $\mathbb{D}'$, we use $||\mathbb{D} - \mathbb{D}'||$ to denote their L_1 *distance*. The L_1 distance is defined as $\int_{x\in\mathcal{D}} |f_{\mathbb{D}}(x) - f_{\mathbb{D}'}(x)|dx$ if $\mathcal{D}$ is continuous, and $\sum_{x\in\mathcal{D}} |f_{\mathbb{D}}(x) - f_{\mathbb{D}'}(x)|$ if $\mathcal{D}$ is discrete. We use $D_{KL}(\mathbb{D}||\mathbb{D}')$ to denote their *KL Distance*, defined as $\int_{x\in\mathcal{D}} f_{\mathbb{D}}(x) \ln \frac{f_{\mathbb{D}}(x)}{f_{\mathbb{D}'}(x)}dx$ if $\mathcal{D}$ is continuous, and $\sum_{x\in\mathcal{D}} f_{\mathbb{D}}(x) \ln \frac{f_{\mathbb{D}}(x)}{f_{\mathbb{D}'}(x)}$ if $\mathcal{D}$ is discrete.

Overview of the proof. Theorem 5 is concerned with $||\mathbb{D}-\mathbb{D}'||$, where $\mathbb{D} = \mathbb{B}(\mu_1)\times\mathbb{B}(\mu_2)\times\cdots\times\mathbb{B}(\mu_k)$ and $\mathbb{D}' = \mathbb{B}(\mu_1')\times\mathbb{B}(\mu_2')\times\cdots\times\mathbb{B}(\mu_k')$. The overall approach in our proof is to first show that $||\mathbb{D}-\mathbb{D}'||$ is close to $||\vec{\mathbb{N}}-\vec{\mathbb{N}}'||$, where $\vec{\mathbb{N}} = \mathbb{N}(\mu_1)\times\mathbb{N}(\mu_2)\times\cdots\times\mathbb{N}(\mu_k)$ and $\vec{\mathbb{N}}' = \mathbb{N}(\mu_1')\times\mathbb{N}(\mu_2')\times\cdots\times\mathbb{N}(\mu_k')$. Next we will use existing results to upper bound $D_{KL}(\vec{\mathbb{N}}||\vec{\mathbb{N}}')$, which translates to an upper bound on $||\vec{\mathbb{N}}-\vec{\mathbb{N}}'||$, and in turn an upper bound on $||\mathbb{D}-\mathbb{D}'||$.

It is not surprising that $||\mathbb{D}-\mathbb{D}'||$ is close to $||\vec{\mathbb{N}}-\vec{\mathbb{N}}'||$, since normal distribution can be used to approximate binomial distribution. For our proof, however, the complexity arises from need to quantify the approximation error. Since $\mathbb{D}$ and $\mathbb{D}'$ are not continuous distributions, we cannot directly compare them with $\vec{\mathbb{N}}$ and $\vec{\mathbb{N}}'$. Hence we first consider $\widetilde{\mathbb{D}}$ and $\widetilde{\mathbb{D}}'$, where $\widetilde{\mathbb{D}} = \widetilde{\mathbb{B}}(\mu_1)\times\widetilde{\mathbb{B}}(\mu_2)\times\cdots\times\widetilde{\mathbb{B}}(\mu_k)$ and $\widetilde{\mathbb{D}}' = \widetilde{\mathbb{B}}(\mu_1')\times\widetilde{\mathbb{B}}(\mu_2')\times\cdots\times\widetilde{\mathbb{B}}(\mu_k')$. In other words, they are the continuous versions of $\mathbb{D}$ and $\mathbb{D}'$. It is easy to show that $||\mathbb{D}-\mathbb{D}'|| = ||\widetilde{\mathbb{D}}-\widetilde{\mathbb{D}}'||$. We then show that the continuous distributions $\widetilde{\mathbb{D}}$ and $\widetilde{\mathbb{D}}'$ are close to $\vec{\mathbb{N}}$ and $\vec{\mathbb{N}}'$, respectively. To do so, we will prove that $f_{\widetilde{\mathbb{B}}}(x)$ is close to $f_{\mathbb{N}}(x)$, and in turn that $\widetilde{\mathbb{B}}(\mu)$ is close to $\mathbb{N}(\mu)$.

F.1 Basic Technical Lemmas

We first cite a strong form of Stirling's formula, and then prove a few basic technical lemmas. All these will be useful later.

Lemma 14. [7] *For all positive integer i,*

$$i^i e^{-i}\sqrt{2i\pi}\sqrt{\frac{i}{i-0.149}} < i! < i^i e^{-i}\sqrt{2i\pi}\sqrt{\frac{i}{i-1/6}}$$

Lemma 15. *For all real number $x \in [0,1)$, $\ln(1-x) \geq \frac{-x}{1-x}$.*

Proof. Let $f(x) = \ln(1-x)$. From Taylor's theorem, we have $f(x) = f(0) + f'(\xi)x = \frac{-x}{1-\xi}$, where ξ is a real number between 0 and x. The lemma follows since $\frac{-x}{1-\xi} \geq \frac{-x}{1-x}$.

□

Lemma 16. *For all real number $x \in (-1,1)$:*

$$(1+x)\ln(1+x) + (1-x)\ln(1-x) \geq x^2 + \frac{1}{6}x^4$$

For all real number $x \in [-0.5, 0.5]$:

$$(1+x)\ln(1+x) + (1-x)\ln(1-x) \leq x^2 + \frac{1}{3}x^4$$

Proof. For the first equation, define $f(x) = (1+x)\ln(1+x) + (1-x)\ln(1-x) - x^2 - \frac{1}{6}x^4$. We have:

$$\begin{aligned} f'(x) &= \ln(\frac{1+x}{1-x}) - 2x - \frac{4}{6}x^3 \\ f''(x) &= \frac{-2x^4}{x^2-1} \end{aligned}$$

It is easy to verify that $\lim_{x\to -1} f'(x) \to -\infty$, $\lim_{x\to 1} f'(x) \to \infty$, $f''(0) = 0$, and $f''(x) > 0$ for $x \in (-1,0)$ and for $x \in (0,-1)$. This means that $f'(x) = 0$ has a unique root, which is $f'(0) = 0$. Next since $f(0) = 0$ and $\lim_{x\to -1} f(x) = \lim_{x\to 1} f(x) = 2\ln(2) - 1 - \frac{1}{6} > 0$, we know that $f(x) \geq 0$ for $x \in (-1,1)$.

For the second equation, define $f(x) = (1+x)\ln(1+x) + (1-x)\ln(1-x) - x^2 - \frac{1}{3}x^4$. We have:

$$\begin{aligned} f'(x) &= \ln(\frac{1+x}{1-x}) - 2x - \frac{4}{3}x^3 \\ f''(x) &= \frac{2x^2 - 4x^4}{x^2-1} \end{aligned}$$

It is easy to verify that $f'(-0.5) > 0$, $f'(0.5) < 0$, $f''(0) = 0$, and $f''(x) < 0$ for $x \in (-1,0)$ and for $x \in (0,-1)$. This means that $f'(x) = 0$ has a unique root, which is $f'(0) = 0$. Next since $f(0) = 0$ and $f(-0.5) = f(0.5) < 0$, we know that $f(x) \leq 0$ for $x \in [-0.5, 0.5]$.

□

F.2 Proof for $f_{\widetilde{\mathbb{B}}}(x)$ Being Close to $f_{\mathbb{N}}(x)$

Lemma 17 below proves that $f_{\widetilde{\mathbb{B}}}(x)$ and $f_{\mathbb{N}}(x)$ are close to easy other, under certain conditions.

Lemma 17. *Let $f_{\widetilde{\mathbb{B}}}(x)$ and $f_{\mathbb{N}}(x)$ be the probability density function for $\widetilde{\mathbb{B}}(\mu)$ and $\mathbb{N}(\mu)$, respectively. For all integer i where $1 \leq i \leq 2\mu - 1$:*

$$\frac{f_{\widetilde{\mathbb{B}}}(i)}{f_{\mathbb{N}}(i)} < exp(-\frac{\delta^4}{6\mu^3} + \frac{\delta^2}{2(\mu^2 - \delta^2)}) \quad \textit{where } \delta = i - \mu$$

If $\mu \geq 16$, then for all integer i where $\mu - 2\sqrt{\mu} \leq i \leq \mu + 2\sqrt{\mu}$:

$$\frac{f_{\widetilde{\mathbb{B}}}(i)}{f_{\mathbb{N}}(i)} > (1 - \frac{6}{\mu})$$

Proof. Let $\delta = i - \mu$, $a = (\frac{\mu}{\mu+\delta})^{\mu+\delta}(\frac{\mu}{\mu-\delta})^{\mu-\delta}$, and $b = 1/\sqrt{1-\frac{\delta^2}{\mu^2}}$. We first derive a simple equation:

$$\begin{aligned}
& \frac{1}{2^{2\mu}}\frac{(2\mu)^{2\mu}e^{-2\mu}\sqrt{2\pi\cdot 2\mu}}{i^i e^{-i}\sqrt{2\pi i}(2\mu-i)^{(2\mu-i)}e^{-(2\mu-i)}\sqrt{2\pi(2\mu-i)}} \\
= \quad & (\frac{2\mu}{2i})^i(\frac{2\mu}{4\mu-2i})^{(2\mu-i)}\sqrt{\frac{2\mu}{2\pi i(2\mu-i)}} \\
= \quad & (\frac{\mu}{\mu+\delta})^{\mu+\delta}(\frac{\mu}{\mu-\delta})^{\mu-\delta}\sqrt{\frac{\mu}{\pi(\mu+\delta)(\mu-\delta)}} \\
= \quad & (\frac{\mu}{\mu+\delta})^{\mu+\delta}(\frac{\mu}{\mu-\delta})^{\mu-\delta}\frac{1}{\sqrt{1-\frac{\delta^2}{\mu^2}}}\frac{1}{\sqrt{\pi\mu}} \\
= \quad & \frac{ab}{\sqrt{\mu\pi}}
\end{aligned}$$

Next, we upper bound $\frac{f_{\widetilde{\mathbb{B}}}(i)}{f_{\mathbb{N}}(i)}$ for $1 \le i \le 2\mu - 1$. Apply Lemma 14, and we have:

$$\begin{aligned}
f_{\widetilde{\mathbb{B}}}(i) &= \frac{1}{2^{2\mu}}\binom{2\mu}{i} = \frac{1}{2^{2\mu}}\frac{(2\mu)!}{i!(2\mu-i)!} \\
&< \left(\frac{1}{2^{2\mu}}\frac{(2\mu)^{2\mu}e^{-2\mu}\sqrt{2\pi\cdot 2\mu}}{i^i e^{-i}\sqrt{2\pi i}(2\mu-i)^{(2\mu-i)}e^{-(2\mu-i)}\sqrt{2\pi(2\mu-i)}}\right)\times \\
&\quad \sqrt{\frac{2\mu}{2\mu-1/6}\cdot\frac{i-0.149}{i}\cdot\frac{2\mu-i-0.149}{2\mu-i}} \\
&= \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{\frac{2\mu}{2\mu-1/6}\cdot\frac{\min(i,2\mu-i)-0.149}{\min(i,2\mu-i)}\cdot\frac{\max(i,2\mu-i)-0.149}{\max(i,2\mu-i)}} \\
&< \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{\frac{2\mu}{2\mu-1/6}\frac{\min(i,2\mu-i)-0.149}{\min(i,2\mu-i)}} \\
&\le \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{\frac{2\mu}{2\mu-1/6}\cdot\frac{\mu-0.149}{\mu}} < \frac{ab}{\sqrt{\mu\pi}}
\end{aligned}$$

Lemma 16 tells us that:

$$\begin{aligned}
\ln a &= (\mu+\delta)\ln\left(\frac{\mu}{\delta+\mu}\right) + (\mu-\delta)\ln\left(\frac{\mu}{\mu-\delta}\right) \\
&= -(\mu+\delta)\ln(1+\frac{\delta}{\mu}) - (\mu-\delta)\ln(1-\frac{\delta}{\mu}) \\
&\le -\mu\left((\frac{\delta}{\mu})^2 + \frac{1}{6}(\frac{\delta}{\mu})^4\right) = -\frac{\delta^2}{\mu} - \frac{\delta^4}{6\mu^3}
\end{aligned}$$

For b, applying Lemma 15 yields $\ln b = -\frac{1}{2}\ln(1-\frac{\delta^2}{\mu^2}) \le \frac{1}{2}\cdot\frac{\frac{\delta^2}{\mu^2}}{1-\frac{\delta^2}{\mu^2}}$. Therefore:

$$f_{\widetilde{\mathbb{B}}}(i) < \frac{ab}{\sqrt{\mu\pi}} \le exp(-\frac{\delta^2}{\mu} - \frac{\delta^4}{6\mu^3})\exp(\frac{\frac{\delta^2}{\mu^2}}{2(1-\frac{\delta^2}{\mu^2})})\frac{1}{\sqrt{\mu\pi}} = f_{\mathbb{N}}(i)\exp(-\frac{\delta^4}{6\mu^3} + \frac{\delta^2}{2(\mu^2-\delta^2)})$$

Finally, we lower bound $\frac{f_{\widetilde{\mathbb{B}}}(i)}{f_{\mathbb{N}}(i)}$ for $\mu - 2\sqrt{\mu} \leq i \leq \mu + 2\sqrt{\mu}$. Applying Lemma 14 again in a similar way as before, we have:

$$\begin{aligned}
f_{\widetilde{\mathbb{B}}}(i) &= \frac{1}{2^{2\mu}}\binom{2\mu}{i} = \frac{1}{2^{2\mu}}\frac{(2\mu)!}{i!(2\mu-i)!} \\
&> \left(\frac{1}{2^{2\mu}}\frac{(2\mu)^{2\mu}e^{-2\mu}\sqrt{2\pi\cdot 2\mu}}{i^i e^{-i}\sqrt{2\pi i}(2\mu-i)^{(2\mu-i)}e^{-(2\mu-i)}\sqrt{2\pi(2\mu-i)}}\right)\times \\
&\quad \sqrt{\frac{2\mu}{2\mu-0.149}\cdot\frac{\mu+\delta-1/6}{\mu+\delta}\cdot\frac{\mu-\delta-1/6}{\mu-\delta}} \\
&= \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{\frac{2\mu}{2\mu-0.149}\cdot\frac{\mu+\delta-1/6}{\mu+\delta}\cdot\frac{\mu-\delta-1/6}{\mu-\delta}} \\
&> \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{\frac{\mu+\delta-1/6}{\mu+\delta}\cdot\frac{\mu-\delta-1/6}{\mu-\delta}} = \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{1+\frac{1-12\mu}{36\mu^2-36\delta^2}} \\
&\geq \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{1+\frac{1-12\mu}{36\mu^2-144\mu}} \quad \text{(since } \delta^2 \leq 4\mu \text{ and since } \mu^2 > 4\mu) \\
&> \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{1-\frac{12\mu}{36\mu^2-18\mu^2}} \quad \text{(since } u \geq 8) \\
&= \frac{ab}{\sqrt{\mu\pi}}\times\sqrt{1-\frac{2}{3\mu}} > \frac{ab}{\sqrt{\mu\pi}}\times(1-\frac{2}{3\mu})
\end{aligned}$$

Next because $\delta \leq 2\sqrt{\mu}$ and $\mu \geq 16$, we have $|\delta/\mu| \leq 0.5$. Lemma 16 tells us that:

$$\begin{aligned}
\ln a &= (\mu+\delta)\ln\left(\frac{\mu}{\delta+\mu}\right) + (\mu-\delta)\ln\left(\frac{\mu}{\mu-\delta}\right) \\
&= -(\mu+\delta)\ln(1+\frac{\delta}{\mu}) - (\mu-\delta)\ln(1-\frac{\delta}{\mu}) \\
&\geq -\mu\left((\frac{\delta}{\mu})^2 + \frac{1}{3}(\frac{\delta}{\mu})^4\right) = -\frac{\delta^2}{\mu} - \frac{\delta^4}{3\mu^3}
\end{aligned}$$

Therefore:

$$\begin{aligned}
f_{\widetilde{\mathbb{B}}}(i) &> \frac{ab}{\sqrt{\mu\pi}}\times(1-\frac{2}{3\mu}) \geq exp(-\frac{\delta^2}{\mu}-\frac{\delta^4}{3\mu^3})\cdot\frac{1}{\sqrt{1-\frac{\delta^2}{\mu^2}}}\cdot(1-\frac{2}{3\mu})\cdot\frac{1}{\sqrt{\mu\pi}} \\
&> exp(-\frac{\delta^2}{\mu}-\frac{\delta^4}{3\mu^3})\cdot(1-\frac{2}{3\mu})\cdot\frac{1}{\sqrt{\mu\pi}} \geq f_{\mathbb{N}}(i)\cdot(1-\frac{\delta^4}{3\mu^3})\cdot(1-\frac{2}{3\mu}) \\
&\geq f_{\mathbb{N}}(i)\cdot(1-\frac{16\mu^2}{3\mu^3})\cdot(1-\frac{2}{3\mu}) \quad \text{(since } \delta^2 \leq 4\mu) \\
&\geq f_{\mathbb{N}}(i)(1-\frac{6}{\mu})
\end{aligned}$$

□

F.3 Upper Bound $||\widetilde{\mathbb{B}}(\mu) - \mathbb{N}(\mu)||$

This section shows that $\widetilde{\mathbb{B}}(\mu)$ and $\mathbb{N}(\mu)$ are close, by leveraging the lemma proved in the previous section.

Lemma 18. *For $\mu \geq 50$ where 2μ is an integer,* $||\widetilde{\mathbb{B}}(\mu) - \mathbb{N}(\mu)|| < \frac{4.3}{\sqrt{\mu}}$.

Proof. Let $f_{\widetilde{\mathbb{B}}}(x)$ and $f_{\mathbb{N}}(x)$ be the probability density function for $\widetilde{\mathbb{B}}(\mu)$ and $\mathbb{N}(\mu)$, respectively. We also define function $f_{\widetilde{\mathbb{N}}}(x) = f_{\mathbb{N}}(\lfloor x \rceil)$. Note that $f_{\widetilde{\mathbb{N}}}$ is not necessarily a probability density function. Define $S = \{\text{integer } i \mid \mu - 2\sqrt{\mu} \leq i \leq \mu + 2\sqrt{\mu}\}$. We have:

$$\begin{aligned}
& ||\widetilde{\mathbb{B}}(\mu) - N(\mu)|| \\
= & \int_{-\infty}^{\infty} |f_{\widetilde{\mathbb{B}}}(x) - f_{\mathbb{N}}(x)| dx = \int_{-\infty}^{\infty} |(f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)) + (f_{\widetilde{\mathbb{N}}}(x) - f_{\mathbb{N}}(x))| dx \\
\leq & \int_{-\infty}^{\infty} |f_{\widetilde{\mathbb{N}}}(x) - f_{\mathbb{N}}(x)| dx + \int_{-\infty}^{\infty} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx \\
= & \int_{-\infty}^{\infty} |f_{\widetilde{\mathbb{N}}}(x) - f_{\mathbb{N}}(x)| dx + \int_{\lfloor x \rceil \in S} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx + \int_{\lfloor x \rceil \notin S} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx
\end{aligned}$$

We will prove the following three equations, which will complete the proof:

$$\int_{-\infty}^{\infty} |f_{\widetilde{\mathbb{N}}}(x) - f_{\mathbb{N}}(x)| dx < \frac{1.13}{\sqrt{\mu}} \tag{8}$$

$$\int_{\lfloor x \rceil \in S} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx < \frac{1}{\sqrt{\mu}} \tag{9}$$

$$\int_{\lfloor x \rceil \notin S} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx < \frac{2.13}{\sqrt{\mu}} \tag{10}$$

- Proof for Equation 8.

$$\begin{aligned}
\int_{-\infty}^{\infty} |f_{\widetilde{\mathbb{N}}}(x) - f_{\mathbb{N}}(x)| dx & = \int_{-\infty}^{\infty} |f_{\mathbb{N}}(\lfloor x \rceil) - f_{\mathbb{N}}(x)| dx \\
& \leq \sum_{i=-\infty}^{\infty} (\max_{x \in [i, i+1)} f_{\mathbb{N}}(x) - \min_{x \in [i, i+1)} f_{\mathbb{N}}(x)) \\
& \leq 2 \times (\max_{x \in (-\infty, \infty)} f_{\mathbb{N}}(x) - \min_{x \in (-\infty, \infty)} f_{\mathbb{N}}(x)) \\
& \quad \text{(since } f_{\mathbb{N}}(x) \text{ is first increasing and then decreasing)} \\
& < \frac{2}{\sqrt{\pi \mu}} < \frac{1.13}{\sqrt{\mu}}
\end{aligned}$$

- Proof for Equation 9. We first show that $f_{\widetilde{\mathbb{B}}}(\lfloor x \rceil)$ is very close to $f_{\widetilde{\mathbb{N}}}(\lfloor x \rceil)$. Let $\delta = \lfloor x \rceil - \mu$. For $\lfloor x \rceil \in S$, invoke the first equation in Lemma 17 with $\mu \geq 9$ (implying that $\lfloor x \rceil \in S \Rightarrow 1 \leq \lfloor x \rceil \leq 2\mu - 1$), and we have:

$$\begin{aligned}
& f_{\widetilde{\mathbb{B}}}(\lfloor x \rceil) \\
< & f_{\mathbb{N}}(\lfloor x \rceil) \cdot \exp\left(-\frac{\delta^4}{6\mu^3} + \frac{\delta^2}{2(\mu^2 - \delta^2)}\right) = f_{\widetilde{\mathbb{N}}}(\lfloor x \rceil) \cdot \exp\left(-\frac{\delta^4}{6\mu^3} + \frac{\delta^2}{2(\mu^2 - \delta^2)}\right) \\
\leq & f_{\widetilde{\mathbb{N}}}(\lfloor x \rceil) \cdot \exp\left(\frac{\delta^2}{2(\mu^2 - \delta^2)}\right) \\
\leq & f_{\widetilde{\mathbb{N}}}(\lfloor x \rceil) \cdot \exp\left(\frac{4\mu}{2(\mu^2 - 4\mu)}\right) \qquad \text{(since } |\delta| \leq 2\sqrt{\mu} \text{ for } \lfloor x \rceil \in S, u > 4, \text{ and } \mu^2 > 4\mu) \\
= & f_{\widetilde{\mathbb{N}}}(\lfloor x \rceil) \cdot \exp\left(\frac{2}{\mu - 4}\right) \leq f_{\widetilde{\mathbb{N}}}(\lfloor x \rceil)\left(1 + \frac{5}{\mu}\right) \qquad \text{(since } u \geq 10)
\end{aligned}$$

Since $\mu \geq 16$, together with the second equation in Lemma 17, this means that for $\lfloor x \rfloor \in S$:

$$\begin{aligned} f_{\widetilde{\mathbb{N}}}(\lfloor x \rfloor)(1 - \frac{6}{\mu}) < f_{\widetilde{\mathbb{B}}}(\lfloor x \rfloor) < f_{\widetilde{\mathbb{N}}}(\lfloor x \rfloor)(1 + \frac{5}{\mu}) \\ \Rightarrow \quad |f_{\widetilde{\mathbb{B}}}(\lfloor x \rfloor) - f_{\widetilde{\mathbb{N}}}(\lfloor x \rfloor)| < \frac{6}{\mu} f_{\widetilde{\mathbb{N}}}(\lfloor x \rfloor) < \frac{6}{\mu - 6} f_{\widetilde{\mathbb{B}}}(\lfloor x \rfloor) \end{aligned}$$

This enables us to prove Equation 9:

$$\int_{\lfloor x \rfloor \in S} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx \quad < \quad \int_{\lfloor x \rfloor \in S} \frac{6}{\mu - 6} f_{\widetilde{\mathbb{B}}}(\lfloor x \rfloor) dx < \frac{6}{\mu - 6} < \frac{1}{\sqrt{\mu}} \qquad \text{(since } \mu \geq 50\text{)}$$

- Proof for Equation 10. We will later prove that $f_{\widetilde{\mathbb{B}}}(i) < f_{\mathbb{N}}(i)$ for all $i \notin S$. If such a claim does hold, then we have:

$$\begin{aligned} \int_{\lfloor x \rfloor \notin S} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx &= \sum_{i \notin S} |f_{\widetilde{\mathbb{B}}}(i) - f_{\widetilde{\mathbb{N}}}(i)| \\ &= \sum_{i \notin S} |f_{\widetilde{\mathbb{B}}}(i) - f_{\mathbb{N}}(i)| = \sum_{i \notin S} (f_{\mathbb{N}}(i) - f_{\widetilde{\mathbb{B}}}(i)) \\ &= (\sum_{i=-\infty}^{\infty} f_{\mathbb{N}}(i) - \sum_{i \in S} f_{\mathbb{N}}(i)) - (\sum_{i=-\infty}^{\infty} f_{\widetilde{\mathbb{B}}}(i) - \sum_{i \in S} f_{\widetilde{\mathbb{B}}}(i)) \\ &= \sum_{i \in S} f_{\widetilde{\mathbb{B}}}(i) - \sum_{i \in S} f_{\mathbb{N}}(i) \\ &= (\sum_{i \in S} f_{\widetilde{\mathbb{B}}}(i) - \sum_{i \in S} f_{\widetilde{\mathbb{N}}}(i)) + (\sum_{i \in S} f_{\widetilde{\mathbb{N}}}(i) - \sum_{i \in S} f_{\mathbb{N}}(i)) \\ &\leq \int_{\lfloor x \rfloor \in S} |f_{\widetilde{\mathbb{B}}}(x) - f_{\widetilde{\mathbb{N}}}(x)| dx + \int_{\lfloor x \rfloor \in S} |f_{\widetilde{\mathbb{N}}}(x) - f_{\mathbb{N}}(x)| dx \\ &< \frac{1}{\sqrt{\mu}} + \frac{1.13}{\sqrt{\mu}} = \frac{2.13}{\sqrt{\mu}} \qquad \text{(by Equation 8 and Equation 9)} \end{aligned}$$

The only thing left now is to show that $f_{\widetilde{\mathbb{B}}}(i) < f_{\mathbb{N}}(i)$ for all $i \notin S$. If $i \notin S$ and $i < 0$ (or $i \notin S$ and $i > 2\mu$), then $f_{\widetilde{\mathbb{B}}}(i) = 0 < f_{\mathbb{N}}(i)$. If $i \notin S$ and $i = 0$ (or $i \notin S$ and $i = 2\mu$), then:

$$f_{\widetilde{\mathbb{B}}}(i) = \frac{1}{2^{2\mu}} = \frac{1}{4^{\mu}} < \frac{1}{\sqrt{\pi\mu}} exp(-\mu) = f_{\mathbb{N}}(i) \qquad \text{(since } \mu \geq 50\text{)}$$

If $i \notin S$ and $i = 1$ (or $i \notin S$ and $i = 2\mu - 1$), then:

$$f_{\widetilde{\mathbb{B}}}(i) = \frac{1}{2^{2\mu}} 2\mu = \frac{2\mu}{4^{\mu}} < \frac{1}{\sqrt{\pi\mu}} exp(-\frac{(\mu-1)^2}{\mu}) = f_{\mathbb{N}}(i) \qquad \text{(since } \mu \geq 50\text{)}$$

Finally, if $i \notin S$ and $2 \leq i \leq 2\mu - 2$, then let $\delta = i - \mu$ and leverage the first equation in Lemma 17:

$$\begin{aligned} f_{\widetilde{\mathbb{B}}}(i) &< f_{\mathbb{N}}(i) \cdot exp(-\frac{\delta^4}{6\mu^3} + \frac{\delta^2}{2(\mu^2 - \delta^2)}) \\ &= f_{\mathbb{N}}(i) \cdot exp(\frac{\delta^2}{6\mu^3(\mu^2 - \delta^2)}(3\mu^3 - \delta^2(\mu^2 - \delta^2))) \end{aligned}$$

We will prove that $\delta^2(\mu^2 - \delta^2) > 3\mu^3$ when $\mu \geq 50$. Since $i \notin S$ and $2 \leq i \leq 2\mu - 2$, we know that $4\mu \leq \delta^2 \leq (\mu - 2)^2$. Define $f(\delta^2) = \delta^2(\mu^2 - \delta^2)$ where $\delta^2 \in [4\mu, (\mu - 2)^2]$, and it is easy to verify that since $4\mu \leq \mu^2/2$ and $(\mu-2)^2 \geq \mu^2/2$ (for $\mu \geq 8$), the minimum of $f(\delta^2)$ is reached at $f((\mu-2)^2)$. Hence we have $f(\delta^2) \geq f((\mu-2)^2) = (\mu-2)^2(\mu^2-(\mu-2)^2) = 4(\mu-2)^2(\mu-1) > 3\mu^3$ for $\mu \geq 50$. Hence we have $f_{\widetilde{\mathbb{B}}}(i) < f_{\mathbb{N}}(i) \cdot exp(0) = f_{\mathbb{N}}(i)$.

□

F.4 Upper Bound $||\vec{\mathbb{N}} - \vec{\mathbb{N}}'||$

This section will prove an upper bound on $||\vec{\mathbb{N}} - \vec{\mathbb{N}}'||$. We do so by using some existing result to upper bound $D_{KL}(\vec{\mathbb{N}}||\vec{\mathbb{N}}')$, and then using another existing result to convert this upper bound to an upper bound on $||\vec{\mathbb{N}} - \vec{\mathbb{N}}'||$.

Define $\vec{\mathbb{N}}(\vec{\mu}, \Sigma)$ to be the multi-variate normal distribution whose mean vector is $\vec{\mu}$ and whose covariance matrix is Σ. The following is a known result on the KL distance between two multi-variate normal distributions:

Lemma 19. [27] *Let $\vec{\mathbb{N}}(\vec{\mu}, \Sigma)$ and $\vec{\mathbb{N}}'(\vec{\mu'}, \Sigma')$ be two arbitrary k-variate normal distributions. If Σ and Σ' are both non-singular matrices, then:*

$$D_{KL}(\vec{\mathbb{N}}||\vec{\mathbb{N}}') = \frac{1}{2}\left(\operatorname{tr}\left(\Sigma'^{-1}\Sigma\right) + \left(\vec{\mu'} - \vec{\mu}\right)^{\top} \Sigma'^{-1}(\vec{\mu'} - \vec{\mu}) - k + \ln\left(\frac{\det \Sigma'}{\det \Sigma}\right)\right)$$

Applying this lemma to our setting yields:

Lemma 20. *For any given positive integers μ_1 through μ_k and μ'_1 through μ'_k, define distributions $\vec{\mathbb{N}} = \mathbb{N}(\mu_1) \times \mathbb{N}(\mu_2) \times \cdots \times \mathbb{N}(\mu_k)$ and $\vec{\mathbb{N}}' = \mathbb{N}(\mu'_1) \times \mathbb{N}(\mu'_2) \times \cdots \times \mathbb{N}(\mu'_k)$. We have:*

$$D_{KL}(\vec{\mathbb{N}}||\vec{\mathbb{N}}') = \frac{1}{2}\left(\sum_{i=1}^{k} \frac{\mu_i - \mu'_i}{\mu'_i} + 2\sum_{i=1}^{k} \frac{(\mu_i - \mu'_i)^2}{\mu'_i} + \sum_{i=1}^{k} \ln \frac{\mu'_i}{\mu_i}\right)$$

Proof. Obviously, $\vec{\mathbb{N}}$ is a multi-variate normal distribution. Let $\vec{\mu}$ be its mean vector and Σ be its covariance matrix. Similarly define $\vec{\mu'}$ and Σ'. We have:

$$\vec{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \cdots \\ \mu_k \end{bmatrix}, \vec{\mu'} = \begin{bmatrix} \mu'_1 \\ \mu'_2 \\ \cdots \\ \mu'_k \end{bmatrix}, \Sigma = \frac{1}{2}\begin{bmatrix} \mu_1 & 0 & \cdots & 0 \\ 0 & \mu_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_k \end{bmatrix}, \Sigma' = \frac{1}{2}\begin{bmatrix} \mu'_1 & 0 & \cdots & 0 \\ 0 & \mu'_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu'_k \end{bmatrix}$$

Since Σ and Σ' are both diagonal matrices of size $k \times k$, we denote $\Sigma = \operatorname{diag}(\mu_1, \mu_2, \ldots, \mu_k)$ and $\Sigma' = \operatorname{diag}(\mu'_1, \mu'_2, \ldots, .\mu'_k)$. Obviously, both Σ and Σ' are non-singular. Following are some useful properties of diagonal matrices:

$$\begin{aligned}
(\operatorname{diag}(\mu'_1, \mu'_2, \ldots, \mu'_k))^{-1} &= \operatorname{diag}(\mu_1'^{-1}, \mu_2'^{-1}, \ldots, \mu_k'^{-1}) \\
\operatorname{diag}(\mu'_1, \mu'_2, \ldots, \mu'_k) \cdot \operatorname{diag}(\mu_1, \mu_2, \ldots, \mu_k) &= \operatorname{diag}(\mu_1\mu'_1, \mu_2\mu'_2, \ldots, \mu_k\mu'_k) \\
(\mu'_1, \mu'_2, \ldots, \mu'_k) \cdot \operatorname{diag}(\mu_1, \mu_2, \ldots, \mu_k) &= (\mu_1\mu'_1, \mu_2\mu'_2, \ldots, \mu_k\mu'_k) \\
\operatorname{tr}(\operatorname{diag}(\mu_1, \mu_2, \ldots, \mu_k)) &= \sum_{i=1}^{k} \mu_i \\
\det(\operatorname{diag}(\mu_1, \mu_2, \ldots, \mu_k)) &= \Pi_{i=1}^{k} \mu_i
\end{aligned}$$

The last 2 equations are directly from the definition of the trace and the determinant of a matrix. We therefore have:

$$\begin{aligned}
\operatorname{tr}\left(\Sigma'^{-1}\Sigma\right) &= \operatorname{tr}((\frac{1}{2}\operatorname{diag}(\mu'_1,\mu'_2,\ldots,\mu'_k))^{-1}\cdot\frac{1}{2}\operatorname{diag}(\mu_1,\mu_2,\ldots,.\mu_k))\\
&= \operatorname{tr}(\operatorname{diag}(\mu_1'^{-1},\mu_2'^{-1},\ldots,\mu_k'^{-1})\cdot\operatorname{diag}(\mu_1,\mu_2,\ldots,.\mu_k))\\
&= \operatorname{tr}(\operatorname{diag}(\mu_1\mu_1'^{-1},\mu_2\mu_2'^{-1},\ldots,\mu_k\mu_k'^{-1})) = \sum_{i=1}^{k}\frac{\mu_i}{\mu'_i}\\
\left(\vec{\mu'}-\vec{\mu}\right)^{\top}\Sigma'^{-1}(\vec{\mu'}-\vec{\mu}) &= (\mu'_1-\mu_1,\mu'_2-\mu_2,\ldots,\mu'_k-\mu_k)(\frac{\operatorname{diag}(\mu'_1,\mu'_2,\ldots,\mu'_k)}{2})^{-1}(\vec{\mu'}-\vec{\mu})\\
&= 2(\frac{\mu'_1-\mu_1}{\mu'_1},\frac{\mu'_2-\mu_2}{\mu'_2},\ldots,\frac{\mu'_k-\mu_k}{\mu'_k})\begin{bmatrix}\mu'_1-\mu_1\\ \mu'_2-\mu_2\\ \cdots\\ \mu'_k-\mu_k\end{bmatrix} = 2\sum_{i=1}^{k}\frac{(\mu'_i-\mu_i)^2}{\mu'_i}\\
\det\Sigma &= \det(\frac{1}{2}\operatorname{diag}(\mu_1,\mu_2,\ldots,\mu_k)) = \Pi_{i=1}^{k}\frac{1}{2}\mu_i\\
\det\Sigma' &= \det(\frac{1}{2}\operatorname{diag}(\mu'_1,\mu'_2,\ldots,\mu'_k)) = \Pi_{i=1}^{k}\frac{1}{2}\mu'_i
\end{aligned}$$

Plug in all the above equations into Lemma 19, and we have:

$$\begin{aligned}
D_{KL}(\vec{\mathbb{N}}||\vec{\mathbb{N}'}) &= \frac{1}{2}\left(\sum_{i=1}^{k}\frac{\mu_i}{\mu'_i}+2\sum_{i=1}^{k}\frac{(\mu_i-\mu'_i)^2}{\mu'_i}-k+\sum_{i=1}^{k}\ln\frac{\mu'_i}{\mu_i}\right)\\
&= \frac{1}{2}\left(\sum_{i=1}^{k}\frac{\mu_i-\mu'_i}{\mu'_i}+2\sum_{i=1}^{k}\frac{(\mu_i-\mu'_i)^2}{\mu'_i}+\sum_{i=1}^{k}\ln\frac{\mu'_i}{\mu_i}\right)
\end{aligned}$$

□

Next, the following lemma is the well-known *Pinsker's inequality*:

Lemma 21. [30] *For all distributions $\mathbb{D}$ and $\mathbb{D}'$, $\frac{1}{2}||\mathbb{D}-\mathbb{D}'||\le\sqrt{\frac{1}{2}D_{KL}(\mathbb{D}||\mathbb{D}')}$.*

From Lemma 20 and 21, we trivially have:

Lemma 22. *For any given positive integers μ_1 through μ_k and μ'_1 through μ'_k, define distributions $\vec{\mathbb{N}}=\mathbb{N}(\mu_1)\times\mathbb{N}(\mu_2)\times\cdots\times\mathbb{N}(\mu_k)$ and $\vec{\mathbb{N}'}=\mathbb{N}(\mu'_1)\times\mathbb{N}(\mu'_2)\times\cdots\times\mathbb{N}(\mu'_k)$. We have:*

$$||\vec{\mathbb{N}}-\vec{\mathbb{N}'}||\le\sqrt{\sum_{i=1}^{k}\frac{\mu_i-\mu'_i}{\mu'_i}+2\sum_{i=1}^{k}\frac{(\mu_i-\mu'_i)^2}{\mu'_i}+\sum_{i=1}^{k}\ln\frac{\mu'_i}{\mu_i}}$$

F.5 Putting Everything Together

In this section, we first show a simple connection between the L_1 distance between two product distributions and the L_1 distances between the respective component distributions in the two product distributions. Next we will put everything together to prove Theorem 5.

Lemma 23. *Consider any positive integer k, and any two product distributions $\mathbb{D}=\mathbb{D}_1\times\mathbb{D}_2\times\cdots\times\mathbb{D}_k$ and $\mathbb{D}'=\mathbb{D}'_1\times\mathbb{D}'_2\times\cdots\times\mathbb{D}'_k$, where $\mathbb{D}_1$ through $\mathbb{D}_k$ and $\mathbb{D}'_1$ through $\mathbb{D}'_k$ are arbitrary continuous distributions. We have $||\mathbb{D}-\mathbb{D}'||\le\sum_{i=1}^{k}||\mathbb{D}_i-\mathbb{D}'_i||$.*

Proof. First, for all real numbers $a, b, c, d \in [0, 1]$, we always have:

$$|ab - cd| = |(a - c)b + (b - d)c| \le |a - c||b| + |b - d||c| \le |a - c| + |b - d|$$

Let $f_{\mathbb{D}}$ and $f_{\mathbb{D}'}$ be the probability density function for $\mathbb{D}$ and $\mathbb{D}'$, respectively. Similarly define $f_{\mathbb{D}_1}$ through $f_{\mathbb{D}_k}$ and $f_{\mathbb{D}'_1}$ through $f_{\mathbb{D}'_k}$. For all vector $(x_1, x_2, \dots x_k)$ in the domain of $\mathbb{D}$ and $\mathbb{D}'$, we have:

$$\begin{aligned}
& |f_{\mathbb{D}}(x_1, x_2, \dots, x_k) - f'_{\mathbb{D}}(x_1, x_2, \dots, x_k)| \\
= \; & |f_{\mathbb{D}_1}(x_1) f_{\mathbb{D}_2}(x_2) \cdots f_{\mathbb{D}_k}(x_k) - f_{\mathbb{D}'_1}(x_1) f_{\mathbb{D}'_2}(x_2) \cdots f_{\mathbb{D}'_k}(x_k)| \\
\le \; & |f_{\mathbb{D}_1}(x_1) - f_{\mathbb{D}'_1}(x_1)| + |f_{\mathbb{D}_2}(x_2) f_{\mathbb{D}_3}(x_3) \cdots f_{\mathbb{D}_k}(x_k) - f_{\mathbb{D}'_2}(x_2) f_{\mathbb{D}'_3}(x_3) \cdots f_{\mathbb{D}'_k}(x_k)| \\
\le \; & |f_{\mathbb{D}_1}(x_1) - f_{\mathbb{D}'_1}(x_1)| + |f_{\mathbb{D}_2}(x_2) - f_{\mathbb{D}'_2}(x_2)| + \\
& |f_{\mathbb{D}_3}(x_3) f_{\mathbb{D}_4}(x_4) \cdots f_{\mathbb{D}_k}(x_k) - f_{\mathbb{D}'_3}(x_3) f_{\mathbb{D}'_4}(x_4) \cdots f_{\mathbb{D}'_k}(x_k)| \\
\le \; & \sum_{i=1}^{k} |f_{\mathbb{D}_i}(x_i) - f_{\mathbb{D}'_i}(x_i)|
\end{aligned}$$

Thus we have:

$$\begin{aligned}
||\mathbb{D} - \mathbb{D}'|| &= \int_{x_1, x_2, \dots, x_k} |f_{\mathbb{D}}(x_1, x_2, \dots, x_k) - f'_{\mathbb{D}}(x_1, x_2, \dots, x_k)| d(x_1, x_2, \dots, x_k) \\
&\le \sum_{i=1}^{k} \int_{x_i} |f_{\mathbb{D}_i}(x_i) - f_{\mathbb{D}'_i}(x_i)| dx_i = \sum_{i=1}^{k} ||\mathbb{D}_i - \mathbb{D}'_i||
\end{aligned}$$

□

Theorem 5. *Consider any positive integer k, any μ_i and μ'_i where $2\mu_i$ and $2\mu'_i$ are all integers ($1 \le i \le k$). Let $\mu = \min_{1 \le i \le k}(\min(\mu_i, \mu'_i))$. Let δ and δ' be any given constants where $0 < \delta' < \delta < 0.5$. Let product distribution $\mathbb{D} = \mathbb{B}(\mu_1) \times \mathbb{B}(\mu_2) \times \dots \times \mathbb{B}(\mu_k)$ and $\mathbb{D}' = \mathbb{B}(\mu'_1) \times \mathbb{B}(\mu'_2) \times \dots \times \mathbb{B}(\mu'_k)$. If $\mu \ge \frac{250}{(\delta - \delta')^2}(k^2 + k \max_{1 \le i \le k}(\mu_i - \mu'_i)^2)$, then $||\mathbb{D} - \mathbb{D}'|| \le \frac{2(\delta - \delta')}{3}$.*

Proof. Define $\widetilde{\mathbb{D}} = \widetilde{\mathbb{B}}(\mu_1) \times \widetilde{\mathbb{B}}(\mu_2) \times \cdots \times \widetilde{\mathbb{B}}(\mu_k)$ and $\widetilde{\mathbb{D}'} = \widetilde{\mathbb{B}}(\mu'_1) \times \widetilde{\mathbb{B}}(\mu'_2) \times \cdots \times \widetilde{\mathbb{B}}(\mu'_k)$. Note that for all vector $(x_1, x_2, \dots x_k)$ where x_1 through x_k are integers, the probability density under $\mathbb{D}$ is exactly the same as the probability density under $\widetilde{\mathbb{D}}$. The same property holds for $\mathbb{D}'$ and $\widetilde{\mathbb{D}'}$. Hence $||\mathbb{D} - \mathbb{D}'|| = ||\widetilde{\mathbb{D}} - \widetilde{\mathbb{D}'}||$. Next, define product distribution $\vec{\mathbb{N}} = \mathbb{N}(\mu_1) \times \mathbb{N}(\mu_2) \times \cdots \times \mathbb{N}(\mu_k)$ and $\vec{\mathbb{N}'} = \mathbb{N}(\mu'_1) \times \mathbb{N}(\mu'_2) \times \cdots \times \mathbb{N}(\mu'_k)$. We have:

$$||\mathbb{D} - \mathbb{D}'|| = ||\widetilde{\mathbb{D}} - \widetilde{\mathbb{D}'}|| \le ||\widetilde{\mathbb{D}} - \vec{\mathbb{N}}|| + ||\widetilde{\mathbb{D}'} - \vec{\mathbb{N}'}|| + ||\vec{\mathbb{N}} - \vec{\mathbb{N}'}||$$

In the next, we upper bound each of the three terms.

Since $\mu \ge \frac{250}{(\delta - \delta')^2}(k^2 + k \max_{1 \le i \le k}(\mu_i - \mu'_i)^2) > 50$, by Lemma 18 and Lemma 23, we have:

$$\begin{aligned}
||\widetilde{\mathbb{D}} - \vec{\mathbb{N}}|| &\le \sum_{i=1}^{k} ||\widetilde{\mathbb{B}}(\mu_i) - \mathbb{N}(\mu_i)|| < \sum_{i=1}^{k} \frac{4.3}{\sqrt{\mu_i}} \le \sum_{i=1}^{k} \frac{4.3}{\sqrt{\mu}} \\
&\le 4.3k / \sqrt{\frac{250}{(\delta - \delta')^2}(k^2 + k \max_{1 \le i \le k}(\mu_i - \mu'_i)^2)} < \frac{4.3}{\sqrt{250}}(\delta - \delta')
\end{aligned}$$

Similarly we have $||\widetilde{\mathbb{D}'} - \vec{\mathbb{N}'}|| < \frac{4.3}{\sqrt{250}}(\delta - \delta')$.

Let $a = \max_{1 \le i \le k} |\mu_i - \mu'_i|$. By Lemma 22, we have:

$$\begin{aligned}
||\vec{\mathbb{N}} - \vec{\mathbb{N}}'|| &\le \sqrt{\sum_{i=1}^{k} \frac{\mu_i - \mu'_i}{\mu'_i} + 2\sum_{i=1}^{k} \frac{(\mu_i - \mu'_i)^2}{\mu'_i} + \sum_{i=1}^{k} \ln \frac{\mu'_i}{\mu_i}} \\
&\le \sqrt{\sum_{i=1}^{k} \frac{a}{\mu'_i} + 2\sum_{i=1}^{k} \frac{a^2}{\mu'_i} + \sum_{i=1}^{k} \ln(1 + \frac{a}{\mu_i})} \\
&\le \sqrt{\sum_{i=1}^{k} \frac{a}{\mu} + 2\sum_{i=1}^{k} \frac{a^2}{\mu} + \sum_{i=1}^{k} \frac{a}{\mu}} = \sqrt{\frac{k}{\mu}(2a^2 + 2a)} \\
&\le \sqrt{\frac{k}{\frac{250}{(\delta-\delta')^2}(k^2 + ka^2)}(2a^2 + 2a)} = \frac{\delta - \delta'}{\sqrt{250}} \sqrt{\frac{2a^2 + 2a}{a^2 + k}} \le \frac{\delta - \delta'}{\sqrt{250}} \sqrt{\frac{2a^2 + 2a}{a^2 + 1}}
\end{aligned}$$

It is easy to verify that $\frac{2a^2+2a}{a^2+1}$ is always smaller that 2.5 for $a \in [0, \infty)$. Hence $||\vec{\mathbb{N}} - \vec{\mathbb{N}}'|| < \frac{\sqrt{2.5}}{\sqrt{250}}(\delta - \delta')$.

Finally, put everything together:

$$\begin{aligned}
||\mathbb{D} - \mathbb{D}'|| &= ||\widetilde{\mathbb{D}} - \widetilde{\mathbb{D}}'|| \le ||\widetilde{\mathbb{D}} - \vec{\mathbb{N}}|| + ||\widetilde{\mathbb{D}}' - \vec{\mathbb{N}}'|| + ||\vec{\mathbb{N}} - \vec{\mathbb{N}}'|| \\
&< \frac{4.3}{\sqrt{250}}(\delta - \delta') + \frac{4.3}{\sqrt{250}}(\delta - \delta') + \frac{\sqrt{2.5}}{\sqrt{250}}(\delta - \delta') < \frac{2(\delta - \delta')}{3}
\end{aligned}$$

□