Near-Optimal Secret Sharing and Error Correcting Codes in AC^0

Kuan Cheng [*] Yuval Ishai [†] Xin Li [‡]

September 22, 2017

Abstract

We study the question of minimizing the computational complexity of (robust) secret sharing schemes and error correcting codes. In standard instances of these objects, both encoding and decoding involve linear algebra, and thus cannot be implemented in the class AC^0. The feasibility of non-trivial secret sharing schemes in AC^0 was recently shown by Bogdanov et al. (Crypto 2016) and that of (locally) decoding errors in AC^0 by Goldwasser et al. (STOC 2007).

In this paper, we show that by allowing some slight relaxation such as a small error probability, we can construct much better secret sharing schemes and error correcting codes in the class AC^0. In some cases, our parameters are close to optimal and would be impossible to achieve without the relaxation. Our results significantly improve previous constructions in various parameters.

Our constructions combine several ingredients in pseudorandomness and combinatorics in an innovative way. Specifically, we develop a general technique to simultaneously amplify security threshold and reduce alphabet size, using a two-level concatenation of protocols together with a random permutation. We demonstrate the broader usefulness of this technique by applying it in the context of a variant of secure broadcast.

[*]kcheng17@jhu.edu. Department of Computer Science, Johns Hopkins University.
[†]yuvali@cs.technion.ac.il. Department of Computer Science, Technion and UCLA.
[‡]lixints@cs.jhu.edu. Department of Computer Science, Johns Hopkins University.

1 Introduction

The motivation for this paper comes from two different sources. The first is the general theme of improving performance at the price of allowing some small probability of error or failure. This is evident throughout computer science. For example, randomized algorithms tend to be much more efficient than their deterministic counterparts. In cryptography and coding theory, randomization with small failure probability can often be used to amplify security or improve efficiency. This is arguably a good tradeoff in practice.

The second source of motivation is the goal of minimizing the computational complexity of cryptographic primitives and related combinatorial objects. For example, a line of work on the parallel complexity of cryptography [18, 15, 30, 2, 3] successfully constructed one way functions and other cryptographic primitives in the complexity class NC^0 based on different kinds of assumptions, including very standard cryptographic assumptions. Works along this line have found several unexpected applications, most recently in the context of general-purpose obfuscation [25]. The study of low-complexity cryptography is also motivated by the goal of obtaining stronger *negative results*. For instance, low-complexity pseudo-random functions imply stronger hardness results for learning [31] and stronger natural proof barriers [28], and low-complexity decryption [8] implies a barrier for function secret sharing [10].

In this paper, we address the question of minimizing the complexity of secret sharing schemes and error correcting codes by introducing additional randomization and allowing for a small failure probability. We focus on the complexity class AC^0, which is the lowest class for which a secret can be reconstructed or a message be decoded with negligible error probability. We show that the randomization approach can be used towards obtaining much better parameters than previous constructions. In some cases, our parameters are close to optimal and would be impossible to achieve without randomization.

We now give a more detailed account of our results, starting with some relevant background.

1.1 (Robust) secret sharing in AC^0

A secret sharing scheme allows a dealer to randomly split a secret between n parties so that qualified subsets of parties can reconstruct the secret from their shares while unqualified subsets learn nothing about the secret. We consider here a variant of threshold secret sharing (also known as a "ramp scheme"), where any k parties can lean nothing about the secret, whereas all n parties together can recover the secret from their shares. We also consider a robust variant where the secret should be correctly reconstructed even if at most d shares are corrupted by an adversary, possibly in an adaptive fashion. We formalize this below.

Definition 1.1 (secret sharing). *An (n, k) secret sharing scheme with message alphabet Σ_0, message length m, and share alphabet Σ is a pair of functions* (Share, Rec)*, where* Share : $\Sigma_0^m \to \Sigma^n$ *is probabilistic and* Rec : $\Sigma^n \to \Sigma_0^m$ *is deterministic, which satisfy the following properties.*

- *Privacy: For a privacy threshold k, the adversary can choose a sequence $W = (w_1, \dots, w_k) \in [n]^k$ of share indices to observe, either adaptively (where each w_i depends on previously observed shares* $\mathsf{Share}(x)_{w_1}, \dots, \mathsf{Share}(x)_{w_{i-1}}$*) or non-adaptively (where W is picked in one shot). We say that the scheme is ϵ-private if for every such strategy, there is a share distribution $\mathcal{D}$ over Σ^k such that for every secret message $x \in \Sigma_0^m$,* $\mathsf{Share}(x)_W$ *is ϵ-close (in statistical distance) to $\mathcal{D}$. We refer to ϵ as the* privacy error *and say that the scheme has* perfect privacy *if $\epsilon = 0$.*

- *Reconstruction: We say that the scheme has reconstruction error η if for every $x \in \Sigma_0^m$,*

$$\Pr[\mathsf{Rec}(\mathsf{Share}(x)) = x] \geq 1 - \eta.$$

  We say the scheme has perfect reconstruction *if $\eta = 0$.*

We are also interested in robust secret sharing*, where an adversary is allowed to modify at most d shares.*

- *Robustness: For any secret $x \in \Sigma_0^m$, let $Y = \mathsf{Share}(x)$. Consider an arbitrary adversary who (adaptively or non-adaptively) observes d shares and can then arbitrarily change these d shares, transforming Y to Y'. The scheme is d-*robust *if for every such adversary,*

$$\Pr[\mathsf{Rec}(Y') = x] \geq 1 - \eta.$$

If the share alphabet and the message alphabet are both Σ, then we simply say the alphabet of the scheme is Σ.

By saying a secret sharing scheme is in AC^0, we mean that both the sharing function and the reconstruction function can be computed by (uniform) AC^0 circuits.

A recent work of Bogdanov et al. [7] considers the complexity of sharing and reconstructing secrets. The question is motivated by the observation that almost all known secret sharing schemes, including the well known Shamir's scheme [32], require the computation of linear functions over finite fields, and thus cannot be implemented in the class AC^0 (i.e., constant depth circuits). Thus a natural question is whether there exist secret sharing schemes in AC^0 with good parameters. In the case of threshold secret sharing, Bogdanov et. al [7] showed a relation between the approximate degree[1] of a function and the privacy threshold of a secret sharing scheme. Using this and known approximate degree lower bounds, they obtained several secret sharing schemes with sharing and reconstruction functions computable in AC^0. However, to achieve a large privacy threshold (e.g., $k = \Omega(n)$) their construction needs to use a large alphabet (e.g., size $2^{\mathsf{poly}(n)}$). In the case of binary alphabet, they can only achieve privacy threshold $\Omega(\sqrt{n})$ with perfect reconstruction and privacy threshold $\Omega((n/\log n)^{2/3})$ with constant reconstruction error $\eta < 1/2$. This limit is inherent without improving the best known approximate degree of an AC^0 function [11]. Furthermore, their schemes only share one bit, and a naive approach of sharing more bits by repeating the scheme multiple times will lead to a bad information rate. This leaves open the question of improving these parameters. Ideally, we would like to share many bits (e.g., $\Omega(n)$), obtain a large privacy threshold (e.g., $\Omega(n)$), and achieve perfect reconstruction and small alphabet size at the same time.

In order to improve the AC^0 secret sharing schemes from [7], we relax their perfect privacy requirement and settle for the notion of ϵ-privacy from Definition 1.1. (This relaxation was recently considered in [9], see discussion below.) Note that this relaxation is necessary to improve the privacy threshold of AC^0 secret sharing schemes, unless one can obtain better approximate degree lower bounds of an explicit AC^0 function (as [7] showed that an explicit AC^0 secret sharing scheme with privacy threshold k and perfect privacy also implies an explicit function in AC^0 with approximate degree at least k). Like most schemes in [7], we only require that the secret can be reconstructed by all n parties. On the other hand, we always require perfect reconstruction. We show that under this slight relaxation, we can obtain much better secret sharing schemes in AC^0. For an adaptive adversary, we can achieve both a constant information rate and a large privacy threshold ($k = \Omega(n)$) over a binary alphabet. In addition, our privacy error is exponentially small. Specifically, we have the following theorems.

Theorem 1.2 (adaptive adversary)**.** *For every $n \in \mathbb{N}$, every constant $\gamma \in (0, 1/4)$, there exists an explicit $(n, \Omega(n))$ secret sharing scheme in AC^0 with alphabet $\{0,1\}$, secret length $m = \Omega(n)$, adaptive privacy error $2^{-\Omega(n^{\frac{1}{4}-\gamma})}$ and perfect reconstruction.*

Note that again, here by using randomization and allowing for a small error in security, we can significantly improve both the privacy threshold and the information rate, while also making the scheme much more efficient by using a smaller alphabet.

[1]The approximate degree of a Boolean function is the lowest degree of a real polynomial that can approximate the function within, say, an additive difference of $1/3$ on every input.

Remark 1.3. *We note that a recent paper by Bun and Thaler [11], gave improved lower bounds for the approximate degree of* AC^0 *functions. Specifically, for any constant* $\alpha > 0$ *they showed an explicit* AC^0 *function with approximate degree at least* $n^{1-\alpha}$*, and by the relation established in [7] this also gives a secret sharing scheme in* AC^0 *with privacy threshold* $n^{1-\alpha}$*. However, our results are stronger in the sense that we can achieve threshold* $\Omega(n)$*, and furthermore we can achieve perfect reconstruction while the secret sharing scheme in [11] only has constant reconstruction advantage.*

Remark 1.4. *Our construction of* AC^0 *secret sharing schemes is actually a general transformation and can take any such scheme in [7] or [11] as the starting point. The error* $2^{-\Omega(n^{\frac{1}{4}-\gamma})}$ *in Theorem 1.2 comes from our use of the one-in-a-box function [29], which has approximate degree* $n^{1/3}$*. We can also use the new* AC^0 *function of [11] with approximate degree* $n^{1-\alpha}$*, which will give us an error of* $2^{-\Omega(n^{\frac{1}{2}-\gamma})}$ *but the reconstruction error will become a constant. We note that the privacy error of our construction is also close to optimal, without further improvement on the lower bounds of approximate degree of* AC^0 *functions. This is because a privacy error of* 2^{-s} *will imply an* AC^0 *function of approximate degree* $\Omega(s/\log n)$*. Thus if one can achieve a sufficiently small privacy error (e.g.,* $2^{-\Omega(n)}$*), then this will give an improved approximate degree lower bound for an* AC^0 *function. See the more detailed explanation in Appendix of the full version.*

A very recent paper by Bogdanov and Williamson [9] considered a similar relaxation as ours. Specifically, they showed how to construct two distributions over n bits that are (k, ϵ)-wise indistinguishable, but can be distinguished with probability $1-\eta$ by some AC^0 function. Here (k, ϵ)-wise indistinguishable means that if looking at any subset of k bits, the two distributions have statistical distance at most ϵ. Translating into the secret sharing model, this roughly implies an AC^0 secret sharing scheme with privacy threshold k, privacy error ϵ and reconstruction error η. Bogdanov and Williamson [9] obtained several results in this case. Specifically, they showed a pair of such distributions for any $k \leq n/2$ with $\epsilon = 2^{-\Omega(n/k)}$, that can be distinguished with $\eta = \Omega(1)$ by the OR function; or for any k with $\epsilon = 2^{-\Omega((n/k)^{1-1/d})}$, that can be distinguished with $\eta = 0$ by a depth-d AND-OR tree.

We note the following important differences between our results and the results by Bogdanov and Williamson [9]: first, the results in [9], in the language of secret sharing, can only share one bit; while all our results can share $\Omega(n)$ bits. Thus our information rate is much larger than theirs. Second, we can achieve a privacy threshold of $k = \Omega(n)$ while simultaneously achieving an exponentially small privacy error of $\epsilon = 2^{-n^{\Omega(1)}}$ and perfect reconstruction of $\eta = 0$. In contrast, the results in [9], when going into the range of $k = \Omega(n)$, only has constant privacy error. In short, our results are better than the results in [9], in the sense that we can simultaneously achieve asymptotically optimal information rate, privacy threshold, exponentially small privacy error and perfect reconstruction. As a direct corollary, we have the following result, which is incomparable to the results in [9].

Corollary 1.5. *There exists a constant* $\alpha > 0$ *such that for every* n *and* $k \leq \alpha n$*, there exists a pair of* $(k, 2^{-n^{\Omega(1)}})$*-wise indistinguishable distributions* X*,* Y *over* $\{0,1\}^n$ *and an* AC^0 *function* D *such that* $\Pr[D(X)] - \Pr[D(Y)] = 1$.

Next, we extend our AC^0 secret sharing schemes to the robust case, where the adversary can tamper with several parties' shares. Our goal is to simultaneously achieve a large privacy threshold, a large tolerance of error, a large information rate and a small alphabet size. We can achieve a constant information rate, privacy threshold and error tolerance both $\Omega(n)$, with constant size alphabet, exponentially small privacy error and polynomially small reconstruction error. However, here we can only handle non-adaptive adversary. Specifically, we have the following theorem.

Theorem 1.6 (non-adaptive adversary)**.** *For every* $n \in \mathbb{N}$*, every* $\eta = \frac{1}{\mathsf{poly}(n)}$*, there exists an explicit* $(n, \Omega(n))$ *robust secret sharing scheme in* AC^0 *with share alphabet* $\{0,1\}^{O(1)}$*, message alphabet* $\{0,1\}$*,* *message length $m = \Omega(n)$, non-adaptive privacy error $2^{-n^{\Omega(1)}}$, non-adaptive robustness $\Omega(n)$ and reconstruction error η.*

1.2 Error correcting codes for additive channels with AC^0 encoding/decoding

Robust secret sharing schemes are natural generalizations of error correcting codes. Thus our robust secret sharing schemes in AC^0 also give error correcting codes with randomized AC^0 encoding and deterministic AC^0 decoding. The model of our error correcting codes is the same as that considered by Guruswami and Smith [20]: stochastic error correcting codes for additive channels. Here, the code has a randomized encoding function and a deterministic decoding function, while the channel can add an arbitrary error vector $e \in \{0,1\}^n$ of Hamming weight at most ρn to the transmitted codeword of length n. As in [20], the error may depend on the message but crucially does not depend on the randomness used by the encoder. Formally, we have the following definition.

Definition 1.7. *For any $n, m \in \mathbb{N}$, any $\rho, \epsilon > 0$, an (n, m, ρ) stochastic binary error correcting code* $(\mathsf{Enc}, \mathsf{Dec})$ *with randomized encoding function* $\mathsf{Enc} : \{0,1\}^m \to \{0,1\}^n$, *deterministic decoding function* $\mathsf{Dec} : \{0,1\}^n \to \{0,1\}^m$ *and decoding error ϵ, is such that for every $x \in \{0,1\}^m$, every $e = (e_1, \dots, e_m) \in \{0,1\}^m$ with hamming weight at most ρn,*

$$\Pr[\mathsf{Dec}(\mathsf{Enc}(x) + e) = x] \geq 1 - \epsilon.$$

An (n, m, ρ) stochastic error correcting code $(\mathsf{Enc}, \mathsf{Dec})$ *can be computed by* AC^0 *circuits if both* Enc *and* Dec *can be computed by* AC^0 *circuits.*

Previously, Guruswami and Smith [20] constructed such codes that approach the Shannon capacity $1 - H(\rho)$. Their encoder and decoder run in polynomial time and have exponentially small decoding error. Here, we aim at constructing such codes with AC^0 encoder and decoder. In a different aspect, Goldwasser et. al [19] gave a construction of locally decodable codes that can tolerate a constant fraction of errors and has AC^0 decoding. Their code has deterministic encoding but randomized decoding. By repeating the local decoder for each bit for $O(\log n)$ times and taking majority, one can decode each bit in AC^0 with error probability $1/\mathsf{poly}(n)$ and thus by a union bound the original message can also be decoded with error probability $1/\mathsf{poly}(n)$. However we note that the encoding function of [19] is not in AC^0, moreover their message rate is only polynomially small. In contrast, our code has constant message rate and can tolerate a constant fraction of errors when the decoding error is $1/\mathsf{poly}(n)$ or even $2^{-\mathsf{poly}\log(n)}$. The rate and tolerance are asymptotically optimal. We can also achieve even smaller error $(2^{-\Omega(r/\log n)})$ with message rate $1/r$. Furthermore both our encoding and decoding are in AC^0. Specifically, we have the following theorems.

Theorem 1.8 (error-correcting codes)**.** *For any $n \in \mathbb{N}$, any $\epsilon = 2^{-\mathsf{poly}\log(n)}$, there exists an $(n, \Omega(n), \Omega(1))$ stochastic binary error correcting code with decoding error ϵ, which can be computed by* AC^0 *circuits.*

Theorem 1.9 (error-correcting codes with smaller decoding error)**.** *For any $n, r \in \mathbb{N}$, there exists an $(n, m = \Omega(n/r), \Omega(1))$ stochastic binary error correcting code with decoding error $2^{-\Omega(r/\log n)}$, which can be computed by* AC^0 *circuits.*

Note that Theorem 1.9 mainly considers the case where r is significantly larger than $\mathsf{poly}\log n$.

Remark 1.10. *We note that, without randomization, it is well known that deterministic* AC^0 *circuits cannot compute asymptotically good codes [26]. Thus the randomization in our* AC^0 *encoding is necessary here. For deterministic* AC^0 *decoding, only very weak lower bounds are known. In particular, Lee and Viola [24] showed that any depth-c* AC^0 *circuit with parity gates cannot decode beyond error $(1/2 - 1/O(\log n)^{c+2})d$, where d is the distance of the code.*

1.3 Secure message broadcasting with an adversary

We also apply our ideas and approach to the following question, which can be viewed as a generalization of one-time pad and we call it secure message broadcasting with an adversary. In a one-time pad, two parties share a secret key which can be used to transform messages secretly between the parties. In the information theoretic setting, suppose the two parties each wants to send an m-bit string to the other party. If an adversary sees the entire communication, then it is well known that to keep the messages secret the two parties must share a secret key of length at least $2m$. This can be generalized to the case of n parties, where we assume that they have access to a broadcasting network, and each party wants to send an m-bit string to all the other parties secretly. This model is naturally applicable in many situations, for example when the n parties want to secretly compute some joint function on their inputs (such as computing coin tosses by taking the parity of their inputs). Again, here if the adversary gets to see the entire communication, then the parties need to share a secret key of length at least nm.

Now, what if we relax the problem a little bit by restricting the adversary's power? Suppose that instead of seeing the entire communication, the adversary can only see some fraction of the communicated messages, how does this affect our solution? We formally define this model as follows.

Definition 1.11. *For any $n, m \in \mathbb{N}$, any $\alpha, \epsilon > 0$, a communication protocol in a broadcast channel involving n parties each having a message of length m is an $(n, m, \alpha, \epsilon, \eta)$-secure message broadcasting protocol if the following holds.*

- *(Privacy) For any adaptive adversarial observation W which observes at most $1 - \alpha$ fraction of the communication transcripts, there is a distribution $\mathcal{D}$, such that for any message $x = (x_1, \ldots, x_n) \in (\{0,1\}^m)^n$ with the transcripts generated from x being Y, Y_W is ϵ-close to $\mathcal{D}$.*
- *(Robustness) For any adaptive adversary that corrupts at most $1 - \alpha$ fraction of the communication transcripts, for any message $x = (x_1, \ldots, x_n) \in (\{0,1\}^m)^n$, all n parties can reconstruct x correctly with probability at least $1 - \eta$ after the communication.*

Here the transcripts are the binary strings sent by the parties during the communication.

The naive solution of applying one-time pad still requires a shared secret key of length at least nm, since otherwise even if the adversary only sees part of the communication, he may learn some information of the inputs. However, by using randomization and allowing for a small error, we can achieve much better performance. Specifically, we have the following theorem.

Theorem 1.12 (secure message broadcasting)**.** *For any $n, m, r \in \mathbb{N}$ with $r \leq m$, there exists an explicit $(n, m, \alpha = \Omega(1), n2^{-\Omega(r)}, n2^{-\Omega(r)} + nm2^{-\Omega(m/r)})$ secure message broadcasting protocol with communication complexity $O(nm)$, shared secret key length $O(r \log(nr))$.*

1.4 Overview of the Techniques

Secret sharing. Here we give an overview of the techniques used in our constructions of AC^0 secret sharing schemes and error correcting codes. Our constructions combine several ingredients in pseudorandomness and combinatorics in an innovative way, so before describing our constructions, we will first describe the important ingredients used.

The secret sharing scheme in [7]. As mentioned before, Bogdanov et. al [7] were the first to consider secret sharing schemes in AC^0. Our constructions will use one of their schemes as the starting point. Specifically, since we aim at perfect reconstruction, we will use the secret sharing scheme in [29] based on the so called "one-in-a-box function" or Minsky-Papert CNF function. This scheme can share one bit among n parties, with binary alphabet, privacy threshold $\Omega(n^{1/3})$ and perfect reconstruction.

Random permutation. Another important ingredient, as mentioned before, is random permutation. Applying a random permutation, in many cases, reduces worst case errors to random errors, and the latter is much more convenient to handle. This property has been exploited in several previous work, such as the error correcting codes by Guruswami and Smith [20]. We note that a random permutation from $[n]$ to $[n]$ can be computed in AC^0 [27, 22, 34].

K-wise independent generators. The third ingredient of our construction is the notion of k-wise independent pseudorandom generators. This is a function that stretches some r uniform random bits to n bits such that any subset of k bits is uniform. Such generators are well studied, while for our constructions we need such generators which can be computed by AC^0 circuits. This requirement is met by using k-wise independent generators based on unique neighbor expander graphs, such as those constructed by Guruswami et. al [21] which use seed length $r = k\mathsf{poly}\log(n)$.

Secret sharing schemes based on error correcting codes. Using asymptotically good linear error correcting codes, one can construct secret sharing schemes that simultaneously achieve constant information rate and privacy threshold $\Omega(n)$ (e.g., [12]). However, certainly in general these schemes are not in AC^0 since they need to compute linear functions such as parity. For our constructions, we will use these schemes with a small block length (e.g., $O(\log n)$ or $\mathsf{poly}\log(n)$) such that parity with such input length can be computed by constant depth circuits. For robust secret sharing, we will also be using robust secret sharing schemes based on codes, with constant information rate, privacy threshold and tolerance $\Omega(n)$ (e.g., [14]), with a small block length.

The constructions. We can now give an informal description of our constructions. As mentioned before, our construction is a general transformation and can take any scheme in [7] or [11] as the starting point. A specific scheme of interest is the one in [7] based on the one-in-a-box function, which has perfect reconstruction. Our goal then is to keep the property of perfect reconstruction, while increasing the information rate and privacy threshold. One naive way to share more bits is to repeat the scheme several times, one for each bit. Of course, this does not help much in boosting the information rate. Our approach, on the other hand, is to use this naive repeated scheme to share a short random seed R. Suppose this gives us n parties with privacy threshold k_0. We then use R and the k-wise independent generator G mentioned above to generate an n-bit string Y, and use Y to share a secret X by computing $Y \oplus X$.

Note that now the length of the secret X can be as large as n and thus the information rate is increased to $1/2$. To reconstruct the secret, we can use the first n parties to reconstruct R, then compute Y and finally X. Note that the whole computation can be done in AC^0 since the k-wise independent generator G is computable in AC^0. The privacy threshold, on the other hand, is the minimum of k_0 and k. This is because if an adversary learns nothing about R, then Y is k-wise independent and thus by looking at any k shares in $Y \oplus X$, the adversary learns nothing about X. This is the first step of our construction.

In the next step, we would like to boost the privacy threshold to $\Omega(n)$ while decreasing the information rate by at most a constant factor. Our approach for this purpose can be viewed as concatenating a larger outer protocol with a smaller inner protocol, which boosts the privacy threshold while keeping the information rate and the complexity of the whole protocol. More specifically, we first divide the parties obtained from the first step into small blocks, and then for each small block we use a good secret sharing scheme based on error correcting codes. Suppose the adversary gets to see a constant fraction of the shares, then on average for each small bock the adversary also gets to see only a constant fraction of the shares. Thus, by Markov's inequality and adjusting the parameters, the adversary only gets to learn the information from a constant fraction of the blocks. However, this is still not enough for us, since the outer protocol only has threshold $n^{\Omega(1)}$.

We solve this problem by using a threshold amplification technique. This is one of our main innovations, and a key step towards achieving both constant information rate and privacy threshold $\Omega(n)$ without sacrificing the error. On a high level, we turn the inner protocol itself into another concatenated protocol (i.e., a larger outer protocol combined with a smaller inner protocol), and then apply a random permutation. Specifically, we choose the size of the block mentioned above to be something like $O(\log^2 n)$, apply a secret sharing scheme based on asymptotically good error correcting codes and obtain $O(\log^2 n)$ shares. We then divide these shares further into $O(\log n)$ smaller blocks each of size $O(\log n)$ (alternatively, this can be viewed as a secret sharing scheme using alphabet $\{0,1\}^{O(\log n)}$), and now we apply a random permutation of these smaller blocks. If we are to use a slightly larger alphabet, we can now store each block together with its index before the permutation as one share. Note that we need the index information when we try to reconstruct the secret, and the reconstruction can be done in AC^0.

Now, suppose again that the adversary gets to see some small constant fraction of the final shares, then since we applied a random permutation, we can argue that each smaller block gets learned by the adversary only with some constant probability. Thus, in the larger block of size $O(\log^2 n)$, by a Chernoeff type bound, except with probability $1/\mathsf{poly}(n)$, we have that only some constant fraction of the shares are learned by the adversary. Note that here by using two levels of blocks, we have reduced the probability that the adversary learns some constant fraction of the shares from a constant to $1/\mathsf{poly}(n)$, which is much better for the outer protocol as we shall see soon. By adjusting the parameters we can ensure that the number of shares that the adversary may learn is below the privacy threshold of the larger block and thus the adversary actually learns nothing. Now, going back to the outer protocol, we know that the expected number of large blocks the adversary can learn is only $n/\mathsf{poly}(n)$; and again by a Chernoff type bound, except with probability $2^{-n^{\Omega(1)}}$, the outer protocol guarantees that the adversary learns nothing. This gives us a secret sharing scheme with privacy threshold $\Omega(n)$ while the information rate is still constant since we only increased the number of shares by a constant factor. With the $O(\log n)$ size alphabet, we can actually achieve privacy threshold $(1-\alpha)n'$ for any constant $0 < \alpha < 1$, where n' is the total number of final parties.

To reduce to the binary alphabet, we can apply another secret sharing scheme based on error correcting codes to each share of length $O(\log n)$. In this case then we won't be able to achieve privacy threshold $(1-\alpha)n'$, but we can achieve $\beta n'$ for some constant $\beta > 0$. This is because if the adversary gets to see a small constant fraction of the shares, then by Markov's inequality only for some constant fraction of the smaller blocks the adversary can learn some useful information. Thus the previous argument still holds.

As described above, our general construction uses two levels of concatenated protocols, which corresponds to two levels of blocks. The first level has larger blocks of size $O(\log^2 n)$, where each larger block consists of $O(\log n)$ smaller blocks of size $O(\log n)$. We use this two-level structure to reduce the probability that an adversary can learn some constant fraction of shares, and this enables us to amplify the privacy threshold to $\Omega(n)$. We choose the smaller block to have size $O(\log n)$ so that both a share from the larger block with length $O(\log n)$ and its index information can be stored in a smaller block. This ensures that the information rate is still a constant even if we add the index information. Finally, the blocks in the second level are actually the blocks that go into the random permutation. This general strategy is one of our main contributions and we hope that it can find other applications.

The above construction gives an AC^0 secret sharing scheme with good parameters. However, it is not a priori clear that it works for an adaptive adversary. In standard secret sharing schemes, a non-adaptive adversary and an adaptive adversary are almost equivalent since usually we have privacy error 0. More specifically, a secret sharing scheme for a non-adaptive adversary with privacy error ϵ and privacy threshold k is also a secret sharing scheme for an adaptive adversary with privacy error $n^k\epsilon$ and privacy threshold k. However in our AC^0 secret sharing scheme the error ϵ is not small enough to kill the n^k factor. Instead, we use the property of the random permutation to argue that our final distribution is essentially symmetric; and thus informally no matter how the adversary picks the shares to observe adaptively, he will not gain any advantage. This will show that our AC^0 secret sharing scheme also works for an adaptive adversary.

To extend to robust secret sharing, we need to use robust secret sharing schemes instead of normal schemes for the first and second level of blocks. Here we use the nearly optimal robust secret sharing schemes based on various codes by Cheraghchi [14]. Unfortunately since we need to use it on a small block length of $O(\log n)$, the reconstruction error becomes $1/\mathsf{poly}(n)$. Another tricky issue here is that an adversary may modify some of the indices. Note that we need the correct index information in order to know which block is which before the random permutation. Suppose the adversary does not modify any of the indices, but only modify the shares, then the previous argument can go through exactly when we change the secret sharing schemes based on error correcting codes into robust secret sharing schemes. However, if the adversary modifies some indices, then we could run into situations where more than one block have the same index and thus we cannot tell which one is correct (and it's possible they are all wrong). To overcome this difficulty, we store every index multiple times among the blocks in the second level. Specifically, after we apply the random permutation, for every original index we randomly choose $O(\log n)$ blocks in the second level to store it. As the adversary can only corrupt a small constant fraction of the blocks in the second level, for each such block, we can correctly recover its original index with probability $1-1/\mathsf{poly}(n)$ by taking the majority of the backups of its index. Thus by a union bound with probability $1-1/\mathsf{poly}(n)$ all original indices can be correctly recovered. In addition, we use the same randomness for each block to pick the $O(\log n)$ blocks, except we add a different shift to the selected blocks. This way, we can ensure that for each block the $O(\log n)$ blocks are randomly selected and thus the union bound still holds. Furthermore the randomness used here is also stored in every block in the second level, so that we can take the majority to reconstruct it correctly. In the above description, we sometimes need to take majority for n inputs, which is not computable in AC^0. However, we note that by adjusting parameters we can ensure that at least say $2/3$ fraction of the inputs are the same, and in this case it suffices to take *approximate majority*, which can be computed in AC^0 [33].

For our error correcting codes, the construction is a simplified version of the robust secret sharing construction. Specifically, we first divide the message itself into blocks of the first level, and then encode every block using an asymptotically good code and divide the obtained codeword into blocks of the second level. Then we apply a random permutation to the blocks of the second level as before, and we encode every second level block by another asymptotically good code. In short, we replace the above mentioned robust secret sharing schemes by asymptotically good error correcting codes. We use the same strategy as in robust secret sharing to identify corrupted indices. Using a size of $O(\log^2 n)$ for blocks in the first level will result in decoding error $1/\mathsf{poly}(n)$, while using larger block size (e.g., $\mathsf{poly}\log(n)$) will result in decoding error $2^{-\mathsf{poly}\log(n)}$. This gives Theorem 1.8. To achieve even smaller error, we can first repeat each bit of the message r times for some parameter r. This serves as an outer error correcting code, which can tolerate up to $r/3$ errors, and can be decoded in in AC^0 by taking approximate majority. The two-level block structure and the argument we described before can now be used to show a smaller decoding error of $2^{-\Omega(r/\log^2 n)}$. This gives Theorem 1.9.

Secure message broadcasting. Rather than using the naive approach of one-time pad, here a more clever solution is to use secret sharing (assuming that each party also has access to local private random bits). By first applying a secret sharing scheme to the message and then broadcast, a party can ensure that if the adversary only gets to see part of the message (below the secrecy threshold), then the adversary learns nothing. In this case the parties do not even need shared secret key. However, one problem with this solution is that the adversary cannot be allowed to see more than $1/n$ fraction of the messages, since otherwise he can just choose the messages broadcasted from one particular party, and then the adversary learns the entire original message from that party. This is the place where randomization comes into play. If in addition, we allow the parties to share a small number of secret random bits, then the parties can use this secret key to *randomly permute* the order in which the parties broadcast their messages (after applying the secret sharing scheme). Since the adversary does not know the secret key, his observation becomes random to the parties, and we can argue that with high probability each party's secret shares are only observed some small fraction. Therefore by the property of secret sharing we can say that roughly the adversary learns nothing about each party's original message. The point is that in this solution, first, the adversary can see some fixed fraction of messages, which is independent of the number of parties n (and thus can be much larger than $1/n$). Second, the number of shared secret random bits is much smaller than the naive approach of one-time pad, and we show in Theorem 7.11 that to achieve security parameter roughly r it is enough for the parties to share $O(r(\log n + \log r))$ random bits. Finally, by using an appropriate secret sharing scheme, the communication complexity of our protocol for each party is $O(m)$, which is optimal up to a constant factor. Note that here, by applying random permutation and allowing for a small probability of error, we simultaneously improve the security threshold (from $1/n$ to $\Omega(1)$) and the length of the shared secret key (from nm to $O(r(\log n + \log r))$).

Discussions and open problems. In this paper we continued the lines of work in applying randomization in computation to improve performance, and minimizing the computational complexity of cryptographic primitives and related combinatorial objects. Specifically, we achieved substantial improvements in AC^0 secret sharing schemes, by allowing an (exponentially) small privacy error. We note that achieving exponentially small error here is non-trivial. In fact, if we allow for a larger error then (for non-adaptive adversary) there is a simple protocol for AC^0 secret sharing: one can first take a random seed R of length $\Omega(n)$, and then apply a deterministic AC^0 extractor for bit-fixing sources to obtain an output Y of length $\Omega(n)$, and we share the secret X by computing the parity of Y and X. This way, one can still share $\Omega(n)$ bits of secret, and if the adversary only learns some small fraction of the seed, then the output Y is close to uniform by the property of the extractor, and thus X remains secret. However, by the lower bound of [13] the error of such AC^0 extractors (or even for the stronger seeded AC^0 extractors) is at least $2^{-\mathsf{poly}\log(n)}$. Therefore, one has to use additional techniques to achieve exponentially small error. We also extended our techniques to robust AC^0 secret sharing schemes, stochastic error correcting codes for additive channels, and secure message broadcasting. Several intriguing open problems remain.

First, in our robust AC^0 secret sharing scheme, we only achieve reconstruction error $1/\mathsf{poly}(n)$. This is because we need to use existing robust secret sharing schemes on a block size of $O(\log n)$. Is it possible to avoid this and improve the error to exponentially small? Also, again in this case we can only handle non-adaptive adversary, and it would be very interesting to obtain a robust AC^0 secret sharing scheme that can handle adaptive adversary. These questions can also be asked for AC^0 stochastic error correcting codes.

Second, as we mentioned in Remark 1.4, a sufficiently small privacy error in an AC^0 secret sharing scheme would imply an improved approximate degree lower bound for AC^0 functions. Is it possible to improve our AC^0 secret sharing scheme, and use this approach to obtain better approximate degree lower bound for AC^0 functions? This seems like an interesting direction.

In addition, the privacy threshold amplification technique we developed, by using two levels of concatenated protocols together with random permutation, is pretty general and we feel that it should have applications elsewhere. We note that the approach of combining an outer protocol with an inner protocol has been used a lot in previous constructions, e.g., to construct better codes [1, 20] or better secure multi-party computation protocols [16]. However, almost in all these situations, one starts with an outer protocol with very good threshold (e.g., the Reed-Solomon code which has a large distance) and the goal is to use the inner protocol to inherit this good threshold while improving some other parameters (such as alphabet size). Thus one only needs one level of concatenation. In our case, instead, we start with an outer protocol with very weak threshold (e.g., the one-in-a-box function which only has privacy threshold $n^{1/3}$). By using two levels of concatenated protocols together with a random permutation, we can actually amplify this threshold to $\Omega(n)$ while simultaneously reducing the alphabet size. This is an important difference to previous constructions and one of our main contributions. We hope that these techniques can find other applications in similar situations.

Finally, since secret sharing schemes are building blocks of many other important cryptographic applications, it is an interesting question to see if the low complexity secret sharing schemes we developed here can be used to reduce the computational complexity of other cryptographic primitives.

Paper organization. We introduce some notation and useful results in Section 2. In Section 3 we give our privacy threshold amplification techniques. In Section 4, we show how to increase the information rate using k-wise independent generators. Combining all the above techniques, our final construction of AC^0 secret sharing schemes is given in Section 5. Instantiations appear in Section 6. Finally, we give our constructions of robust AC^0 secret sharing schemes, AC^0 error correcting codes, and secure message broadcast protocols in Section 7.

2 Preliminaries

Let $|\cdot|$ denote the size of the input set or the absolute value of an input real number, based on contexts.

For any set I of integers, for any $r \in \mathbb{Z}$, we denote $r + I$ or $I + r$ to be $\{i' : i' = i + r, i \in I\}$.

We use Σ to denote the alphabet. Readers can simply regard Σ as $\{0,1\}^l$ for some $l \in \mathbb{N}$. For $\sigma \in \Sigma$, let $\sigma^n = (\sigma, \sigma, \ldots, \sigma) \in \Sigma^n$. For any sequence $s = (s_1, s_2, \ldots, s_n) \in \Sigma^n$ and sequence of indices $W = (w_1, \ldots, w_t) \in [n]^t$ with $t \leq n$, let s_W be the subsequence $(s_{w_1}, s_{w_2}, \ldots, s_{w_t})$.

For any two sequences $a \in \Sigma^n, b \in \Sigma'^{n'}$ where $a = (a_1, a_2, \ldots, a_n), b = (b_1, b_2, \ldots, b_{n'})$, let $a \circ b = (a_1, \ldots, a_n, b_1, \ldots, b_{n'}) \in \Sigma^n \times \Sigma'^{n'}$.

Let $\mathsf{supp}(\cdot)$ denote the support of the input random variable. Let $I(\cdot)$ be the indicator function.

Definition 2.1 (Statistical Distance)**.** *The statistical distance between two random variables X and Y over Σ^n for some alphabet Σ, is $\mathsf{SD}(X, Y)$ which is defined as follows,*

$$\mathsf{SD}(X, Y) = 1/2 \sum_{a \in \Sigma^n} |\Pr[X = a] - \Pr[Y = a]|.$$

Here we also say that X is $\mathsf{SD}(X, Y)$-close to Y.

Lemma 2.2 (Folklore Properties of Statistical Distance [4])**.**

1. *(Triangle Inequality) For any random variables X, Y, Z over Σ^n, we have*

$$\mathsf{SD}(X, Y) \leq \mathsf{SD}(X, Z) + \mathsf{SD}(Y, Z).$$

2. *$\forall n, m \in \mathbb{N}$, any deterministic function $f : \{0,1\}^n \to \{0,1\}^m$ and any random variables X, Y over Σ^n, $\mathsf{SD}(f(X), f(Y)) \leq \mathsf{SD}(X, Y)$.*

We will use the following well known perfect XOR secret sharing scheme.

Theorem 2.3 (Folklore XOR secret sharing)**.** *For any finite field $\mathbb{F}$, define $\mathsf{Share}_+ : \mathbb{F} \to \mathbb{F}^n$ and $\mathsf{Rec}_+ : \mathbb{F}^n \to \mathbb{F}$, such that for any secret $x \in \mathbb{F}$, $\mathsf{Share}_+(x) = y$ such that y is uniformly chosen in $\mathbb{F}^n$ conditioned on $\sum_{i \in [n]} y_i = x$ and Rec_+ is taking the sum of its input.*

$(\mathsf{Share}_+, \mathsf{Rec}_+)$ is an $(n, n-1)$ secret sharing scheme with share alphabet and message alphabet both being $\mathbb{F}$, message length 1, perfect privacy and reconstruction.

Definition 2.4 (Permutation)**.** *For any $n \in \mathbb{N}$, a permutation over $[n]$ is defined to be a bijective function $\pi : [n] \to [n]$.*

Definition 2.5 (k-wise independence)**.** *For any set S, let $X_1, \dots, X_n$ be random variables over S. They are k-wise independent (and uniform) if any k of them are independent (and uniformly distributed).*

For any $r, n, k \in \mathbb{N}$, a function $g : \{0,1\}^r \to \Sigma^n$ is a k-wise (uniform) independent generator, if for $g(U) = (Y_1, \dots, Y_n)$, $Y_1, \dots, Y_n$ are k-wise independent (and uniform). Here U is the uniform distribution over $\{0,1\}^r$.

Definition 2.6 ([21])**.** *A bipartite graph with N left vertices, M right vertices and left degree D is a (K, A) expander if for every set of left vertices $S \subseteq [N]$ of size K, we have $|\Gamma(S)| > AK$. It is a $(\leq K_{\max}, A)$ expander if it is a (K, A) expander for all $K \leq K_{\max}$.*

Here $\forall x \in [N]$, $\Gamma(x)$ outputs the set of all neighbours of x. It is also a set function which is defined accordingly. Also $\forall x \in [N], d \in [D]$, the function $\Gamma : [N] \times [D] \to [M]$ is such that $\Gamma(x, d)$ is the dth neighbour of x.

Theorem 2.7 ([21])**.** *For all constants $\alpha > 0$, for every $N \in \mathbb{N}$, $K_{\max} \leq N$, and $\epsilon > 0$, there exists an explicit $(\leq K_{\max}, (1-\epsilon)D)$ expander with N left vertices, M right vertices, left degree $D = O((\log N)(\log K_{\max})/\epsilon)^{1+1/\alpha}$ and $M \leq D^2 K_{\max}^{1+\alpha}$. Here D is a power of 2.*

Definition 2.8 (AC^0)**.** AC^0 *is the complexity class which consists of all families of circuits having constant depth and polynomial size. The gates in those circuits are* NOT*,* AND *and* OR*, where* AND *gates and* OR *gates have unbounded fan-in.*

For any circuit C, the size of C is denoted as $\mathsf{size}(C)$. The depth of C is denoted as $\mathsf{depth}(C)$. Usually when we talk about computations computable by AC^0 circuits, we mean uniform AC^0 circuits, if not stated specifically.

Lemma 2.9 (Forklore properties of AC^0 circuits [4, 19])**.** *The following are well known properties of* AC^0 *circuits.*

For every $n \in \mathbb{N}$,

1. *([4] forklore) every boolean function $f : \{0,1\}^{l=\Theta(\log n)} \to \{0,1\}$ can be computed by an* AC^0 *circuit of size* $\mathsf{poly}(n)$ *and depth* 2*.*

2. *([19]) for every $c \in \mathbb{N}$, every integer $l = \Theta(\log^c n)$, if the function $f_l : \{0,1\}^l \to \{0,1\}$ can be computed by a circuit with depth $O(\log l)$ and size* $\mathsf{poly}(l)$*, then it can be computed by a circuit with depth $c+1$ and size* $\mathsf{poly}(n)$*.*

Remark 2.10. *We briefly describe the proof implied in [19] for the second property of our Lemma 2.9. As there exists an* NC^1 *complete problem which is downward self-reducible, the function f_l can be reduced to (*AC^0 *reduction) a function with input length $O(\log n)$. By Lemma 2.9 part 1, and noting that the reduction here is an* AC^0 *reduction, f_l can be computed by an* AC^0 *circuit.*

3 Random Permutation

3.1 Increasing the privacy threshold

The main technique we use here is random permutation.

Lemma 3.1 ([27, 22, 34])**.** *For any constant $c \geq 1$, there exists an explicit AC^0 circuit $C : \{0,1\}^r \to [n]^n$ with size poly(n), depth $O(1)$ and $r = O(n^{c+1}\log n)$ such that with probability $1 - 2^{-n^c}$, $C(U_r)$ gives a uniform random permutation of $[n]$; When this fails the outputs are not distinct.*

In the following we give a black box AC^0 transformation of secret sharing schemes increasing the privacy threshold.

Construction 3.2. *For any $n, k, m \in \mathbb{N}$ with $k \leq n$, any alphabet Σ, Σ_0, let $(\mathsf{Share}, \mathsf{Rec})$ be an (n, k) secret sharing scheme with share alphabet Σ, message alphabet Σ_0, message length m.*

Let $(\mathsf{Share}_+, \mathsf{Rec}_+)$ be a $(t, t-1)$ secret sharing scheme with alphabet Σ by Theorem 2.3.

For any constant $a \geq 1, \alpha > 0$, large enough $b \geq 1$, we can construct the following $(n' = tn\bar{n}, k' = (1-\alpha)n')$ secret sharing scheme $(\mathsf{Share}', \mathsf{Rec}')$ with share alphabet $\Sigma \times [n']$, message alphabet Σ_0, message length $m' = m\bar{n}$, where $t = O(\log n), \bar{n} = bn^{a-1}$.

Function $\mathsf{Share}' : \Sigma_0^{m'} \to (\Sigma \times [n'])^{n'}$ is as follows.

1. *On input secret $x \in \Sigma_0^{m\bar{n}}$, parse x to be $(x_1, x_2, \ldots, x_{\bar{n}}) \in (\Sigma_0^m)^{\bar{n}}$.*
2. *Compute $y = (y_1, \ldots, y_{\bar{n}}) = (\mathsf{Share}(x_1), \ldots, \mathsf{Share}(x_{\bar{n}}))$ and parse it to be $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_{n\bar{n}}) \in \Sigma^{n\bar{n}}$. Note that Share is from Σ_0^m to Σ^n.*
3. *Compute $(\mathsf{Share}_+(\hat{y}_1), \ldots, \mathsf{Share}_+(\hat{y}_{n\bar{n}})) \in (\Sigma^t)^{n\bar{n}}$ and split every entry to be t elements in Σ to get $y' = (y'_1, \ldots, y'_{n'}) \in \Sigma^{n'}$. Note that Share_+ is from Σ to Σ^t.*
4. *Generate π by Lemma 3.1 which is uniformly random over permutations of $[n']$. If it fails, which can be detected by checking element distinctness, set π to be such that $\forall i \in [n'], \pi(i) = i$.*
5. *Let*
$$\mathsf{Share}'(x) = (y'_{\pi^{-1}(1)} \circ \pi^{-1}(1), \ldots, y'_{\pi^{-1}(n')} \circ \pi^{-1}(n')) \in (\Sigma \times [n'])^{n'}.$$

Function $\mathsf{Rec}' : (\Sigma \times [n'])^{n'} \to \Sigma_0^{m'}$ is as follows.

1. *Parse the input to be $(y'_{\pi^{-1}(1)} \circ \pi^{-1}(1), \ldots, y'_{\pi^{-1}(n')} \circ \pi^{-1}(n'))$.*
2. *Compute $y' = (y'_1, \ldots, y'_{n'})$ according to the permutation.*
3. *Apply Rec_+ on y' for every successive t entries to get $\hat{y}$.*
4. *Parse $\hat{y}$ to be y.*
5. *Compute x by applying Rec on every entry of $\hat{y}$.*
6. *Output x.*

Lemma 3.3. *If Share and Rec can be computed by AC^0 circuits, then Share' and Rec' can also be computed by AC^0 circuits.*

Proof. As Share can be computed by an AC^0 circuit, y can be computed by an AC^0 circuit (uniform). By Lemma 2.9 part 1, we know that $(\mathsf{Share}_+, \mathsf{Rec}_+)$ both can be computed by AC^0 circuits. By Lemma 3.1, $(\pi^{-1}(1), \pi^{-1}(2), \ldots, \pi^{-1}(n'))$ can be computed by an AC^0 circuit. Also $\forall i \in [n'], y'_{\pi^{-1}(i)} = \bigvee_{j \in [n']}(y'_j \wedge (j = \pi^{-1}(i)))$. Thus Share' can be computed by an AC^0 circuit.

For Rec', $\forall i \in [n'], y'_i = \bigvee_{j \in [n']}(y'_{\pi^{-1}(j)} \wedge (\pi^{-1}(j) = i))$. As Rec_+ can be computed by an AC^0 circuit, y can be computed by an AC^0 circuit. As Rec can be computed by an AC^0 circuit, Rec' can be computed by an AC^0 circuit. $\square$

Lemma 3.4. *If the reconstruction error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* η*, then the reconstruction error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\eta' = \bar{n}\eta$*.*

Proof. According to the construction, as $(\mathsf{Share}_+, \mathsf{Rec}_+)$ has perfect reconstruction by Lemma 2.3, the y computed in Rec' is exactly $(\mathsf{Share}(x_1), \mathsf{Share}(x_2), \ldots, \mathsf{Share}(x_{\bar{n}}))$. As $\forall i \in [\bar{n}], \Pr[\mathsf{Rec}(\mathsf{Share}(x_i)) = x_i] \geq 1 - \eta$,

$$\Pr[\mathsf{Rec}'(\mathsf{Share}'(x)) = x] = \Pr[\bigwedge_{i \in [\bar{n}]} (\mathsf{Rec}(\mathsf{Share}(x_i)) = x_i)] \geq 1 - \bar{n}\eta,$$

by the union bound. $\square$

In order to show privacy, we need the following Chernoff Bound.

Definition 3.5 (Negative Correlation [5, 6])**.** *Binary random variables* $X_1, X_2, \ldots, X_n$ *are negative correlated, if* $\forall I \subseteq [n]$*,*

$$\Pr[\bigwedge_{i \in I} (X_i = 1)] \leq \prod_{i \in I} \Pr[X_i = 1] \text{ and } \Pr[\bigwedge_{i \in I} (X_i = 0)] \leq \prod_{i \in I} \Pr[X_i = 0].$$

Theorem 3.6 (Negative Correlation Chernoff Bound [5, 6])**.** *Let* $X_1, X_2, \ldots, X_n$ *be negatively correlated random variables with* $X = \sum_{i=1}^{n} X_i$*,* $\mu = \mathbb{E}[X]$*.*

- *For any* $\delta \in (0, 1)$*,*

$$\Pr[X \leq (1 - \delta)\mu] \leq e^{-\delta^2\mu/2} \text{ and } \Pr[X \geq (1 + \delta)\mu] \leq e^{-\delta^2\mu/3}.$$

- *For any* $d \geq 6\mu$*,* $\Pr[X \geq d] \leq 2^{-d}$*.*

Lemma 3.7. *Let* $\pi : [n] \to [n]$ *be a random permutation. For any set* $S, W \subseteq [n]$*, let* $u = \frac{|W|}{n}|S|$*. Then the following holds.*

- *for any constant* $\delta \in (0, 1)$*,*

$$\Pr[|\pi(S) \cap W| \leq (1 - \delta)\mu] \leq e^{-\delta^2\mu/2},$$

$$\Pr[|\pi(S) \cap W| \geq (1 + \delta)\mu] \leq e^{-\delta^2\mu/3}.$$

- *for any* $d \geq 6\mu$*,* $\Pr[|\pi(S) \cap W| \geq d] \leq 2^{-d}$*.*

Proof. For every $s \in S$, let X_s be the indicator such that $X_s = 1$ is the event that $\pi(s)$ is in W. Let $X = \sum_{s \in S} X_s$. So $|\pi(S) \cap W| = X$. Note that $\Pr[X_s = 1] = |W|/n$. So $\mu = \mathbb{E}(X) = \frac{|W|}{n}|S|$.

For any $I \subseteq S$,

$$\Pr[\bigwedge_{i \in I} (X_i = 1)] = \frac{|W|}{n} \cdot \frac{|W| - 1}{n - 1} \cdots \frac{|W| - |I|}{n - |I|}$$

(if $|W| < |I|$, it is 0). This is because the random permutation can be viewed as throwing elements $1, \ldots, n$ into n boxes uniformly one by one, where every box can have at most one element. We know that for $j = 1, \ldots, |I|$, $\frac{|W| - j}{n - j} \leq \frac{|W|}{n}$ as $|W| \leq n$. So $\Pr[\bigwedge_{i \in I} (X_i = 1)] \leq \prod_{i \in I} \Pr[X_i = 1]$. In the same way, for any $I \subseteq [n]$,

$$\Pr[\bigwedge_{i \in I} (X_i = 0)] = \frac{n - |W|}{n} \cdot \frac{n - |W| - 1}{n - 1} \cdots \frac{n - |W| - |I|}{n - |I|}$$

(if $n - |W| < |I|$, it is 0). Thus $\forall I \subseteq [n], \Pr[\bigwedge_{i \in I} (X_i = 0)] \leq \prod_{i \in I} \Pr[X_i = 0]$. By Theorem 3.6, the conclusion follows. $\square$

We can get the following more general result by using Lemma 3.7.

Lemma 3.8. *Let $\pi : [n] \to [n]$ be a random permutation. For any $W \subseteq [n]$ with $|W| = \gamma n$, any constant $\delta \in (0,1)$, any $t, l \in \mathbb{N}^+$ such that $tl \leq \frac{0.9\delta}{1+0.9\delta}\gamma n$, any $\mathcal{S} = \{S_1, \ldots, S_l\}$ such that $\forall i \in [l], S_i \subseteq [n]$ are disjoint sets and $|S_i| = t$, let X_i be the indicator such that $X_i = 1$ is the event $|\pi(S_i) \cap W| \geq (1+\delta)\gamma t$. Let $X = \sum_{i\in[l]} X_i$. Then for any $d \geq 0$,*

$$\Pr[X \geq d] \leq e^{-2d+(e^2-1)e^{-\Omega(\gamma t)}l}.$$

Proof. For any $s > 0$, $\Pr[X \geq d] = \Pr[e^{sX} \geq e^{sd}] \leq \frac{\mathbb{E}[e^{sX}]}{e^{sd}}$ by Markov's inequality. For every $i \in [l]$, $\forall x_1, \ldots, x_{i-1} \in \{0,1\}$, consider $p = \Pr[X_i = 1 | \forall j < i, X_j = x_j]$. Let $\bar{S}_i = \bigcup_{j=1}^{i} S_j$ for $i \in [l]$. Note that the event $\forall j < i, X_j = x_j$ is the union of exclusive events $\pi(\bar{S}_{i-1}) = V, \forall j < i, X_j = x_j$ for $V \subseteq [n]$ with $|V| = (j-1)t$ and $\pi(\bar{S}_{i-1}) = V$ does not contradict $\forall j < i, X_j = x_j$. Conditioned on any one of those events, saying $\pi(\bar{S}_{i-1}) = V, \forall j < i, X_j = x_j$, π is a random bijective mapping from $[n] - \bar{S}_i$ to $[n] - V$. Note that $\frac{|W \cap ([n]-V)|}{n-(i-1)t} \leq \frac{\gamma n}{n - \frac{0.9\delta}{1+0.9\delta}\gamma n} \leq \frac{\gamma n}{n - \frac{0.9\delta}{1+0.9\delta} n} \leq (1+0.9\delta)\gamma n$, since $(i-1)t \leq lt \leq \frac{0.9\delta}{1+0.9\delta}\gamma n$. So $\mathbb{E}[\pi(S_i) \cap W | |\pi(\bar{S}_{i-1}) = V, \forall j < i, X_j = x_j] \leq (1+0.9\delta)\gamma t$. By Lemma 3.7, $\Pr[X_i = 1 | \pi(\bar{S}_{i-1}) = V, \forall j < i, X_j = x_j] = \Pr[|\pi(S_i) \cap W| \geq (1+\delta)\gamma t | \pi(\bar{S}_{i-1}) = V, \forall j < i, X_j = x_j] \leq e^{-\Omega(\gamma t)}$. Thus $p \leq e^{-\Omega(\gamma t)}$. Next note that

$$\begin{aligned}
&\mathbb{E}[e^{s\sum_{k=i}^{l} X_k} | \forall j < i, X_j = x_j] \\
=&pe^s \mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 1] + (1-p)\mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 0] \\
\leq&(pe^s + 1 - p)\max(\mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 1], \mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 0]) \\
\leq&e^{p(e^s-1)}\max(\mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 1], \mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 0]) \\
\leq&e^{e^{-\Omega(\gamma t)}(e^s-1)}\max(\mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 1], \mathbb{E}[e^{s\sum_{k=i+1}^{l} X_k} | \forall j < i, X_j = x_j, X_i = 0]).
\end{aligned} \tag{1}$$

As this holds for every $i \in [l]$ and every $x_1, \ldots, x_{i-1} \in \{0,1\}$, we can iteratively apply the inequality and get the result that there exists $x'_1, \ldots, x'_l \in \{0,1\}$ such that $\mathbb{E}[e^{sX}] \leq e^{e^{-\Omega(\gamma t)}(e^s-1)}\mathbb{E}[e^{s\sum_{k=2}^{l} X_k} | X_1 = x'_1] \leq e^{2e^{-\Omega(\gamma t)}(e^s-1)}\mathbb{E}[e^{s\sum_{k=3}^{l} X_k} | X_1 = x'_1, X_2 = x'_2] \leq \cdots \leq e^{e^{-\Omega(\gamma t)}(e^s-1)l}$. Let's take $s = 2$. So $\Pr[X \geq d] \leq \frac{\mathbb{E}[e^{sX}]}{e^{sd}} \leq e^{-2d+(e^2-1)e^{-\Omega(\gamma t)}l}$.

□

Let's first show the non-adaptive privacy of this scheme.

Lemma 3.9. *If the non-adaptive privacy error of* (Share, Rec) *is ϵ, then the non-adaptive privacy error of* (Share′, Rec′) *is $\bar{n}(\epsilon + 2^{-\Omega(k)})$.*

Proof. We show that there exists a distribution $\mathcal{D}$ such that for any string $x \in \Sigma_0^{m'}$, for any sequence of distinct indices $W = (w_1, w_2, \ldots, w_{k'}) \in [n']^{k'}$ (chosen before observation),

$$\mathsf{SD}(\mathsf{Share}'(x)_W, \mathcal{D}) \leq \bar{n}(\epsilon + 2^{-\Omega(k)}).$$

For every $i \in [n\bar{n}]$, the block $\mathsf{Share}_+(\hat{y}_i)$ has length t. Let the indices of shares in $\mathsf{Share}_+(\hat{y}_i)$ be $S_i = \{(i-1)t+1, \ldots, it\}$.

For every $i \in [\bar{n}]$, let E_i be the event that for at most k of $j \in \{(i-1)n+1, \ldots, in\}$, $\pi(S_j) \subseteq W$. Let $E = \bigcap_{i\in[\bar{n}]} E_i$. We choose b to be such that $tn \leq \frac{0.9\alpha}{1+0.9\alpha}|W|$. So by Lemma 3.8, $\Pr[E_i] \geq 1 -$

$e^{-\Omega(k)+(e^2-1)e^{-\Omega((1-\alpha)t)}n}$. We choose a large enough $t = O(\log n)$ such that $\Pr[E_i] \geq 1 - e^{-\Omega(k)}$. So $\Pr[E] \geq 1 - \bar{n}e^{-\Omega(k)}$ by the union bound.

Let's define the distribution $\mathcal{D}$ to be $\mathsf{Share}'(\sigma)_W$ for some $\sigma \in \Sigma_0^{m'}$. We claim that $\mathsf{Share}'(x)_W|E$ and $\mathcal{D}|E$ have statistical distance at most $\bar{n}\epsilon$. The reason is as follows.

Let's fix a permutation π for which E happens. We claim that $\mathsf{Share}'(x)_W$ is a deterministic function of at most k entries of each y_i for $i \in [\bar{n}]$ and some extra uniform random bits. This is because, as E happens, for those $i \in [n\bar{n}]$ with $\pi(S_i) \nsubseteq W$, the shares in $\pi(S_i) \cap W$ are independent of the secret by the privacy of $(\mathsf{Share}_+, \mathsf{Rec}_+)$. Note that they are also independent of other shares since the construction uses independent randomness for $\mathsf{Share}_+(\hat{y}_i), i \in [n\bar{n}]$. For those $i \in [n\bar{n}]$ with $\pi(S_i) \subseteq W$, the total number of them is at most k. So the claim holds. Hence by the privacy of $(\mathsf{Share}, \mathsf{Rec})$ with noting that $y_i, i \in [\bar{n}]$ are generated using independent randomness,

$$\mathsf{SD}(\mathsf{Share}'(x)_W, \mathcal{D}) \leq \bar{n}\epsilon.$$

So with probability at least $1 - \bar{n}e^{-\Omega(k)}$ over the fixing of π, $\mathsf{Share}'(x)_W$ and $\mathcal{D}$ have statistical distance at most $\bar{n}\epsilon$, which means that

$$\mathsf{SD}(\mathsf{Share}'(x)_W, \mathcal{D}) \leq \bar{n}(\epsilon + 2^{-\Omega(k)}).$$

□

Next we show the adaptive privacy.

Lemma 3.10. *For any alphabet Σ, any $n, k \in \mathbb{N}$ with $k \leq n$, for any distribution $X = (X_1, \ldots, X_n)$ over Σ^n, let $Y = ((X_{\pi^{-1}(1)} \circ \pi^{-1}(1)), \ldots, (X_{\pi^{-1}(n)} \circ \pi^{-1}(n)))$ where π is a random permutation over $[n] \to [n]$. For any adaptive observation W with $|W| = k$, Y_W is the same distribution as $Y_{[k]}$.*

Proof. Let $W = (w_1, \ldots, w_k)$.

We use induction.

For the base step, for any $x \in \Sigma$, any $i \in [n]$,

$$\Pr[Y_{w_1} = (x, i)] = \Pr[X_i = x]/n,$$

while

$$\Pr[Y_1 = (x, i)] = \Pr[X_i = x]/n.$$

So Y_{w_1} and Y_1 are the same distributions.

For the inductive step, assume that $Y_{W_{[i]}}$ and $Y_{[i]}$ are the same distributions. We know that for any $u \in (\Sigma \times [n])^i$,

$$\Pr[Y_{W_{[i]}} = u] = \Pr[Y_{[i]} = u].$$

Fix a $u \in (\Sigma \times [n])^i$. For any $v = (v_1, v_2) \in (\Sigma \times [n])$, where $v_1 \in \Sigma, v_2 \in [n]$, $\Pr[Y_{w_{i+1}} = v | Y_{W_{[i]}} = u] = 0$ if v_2 has already been observed in the previous i observations; otherwise $\Pr[Y_{w_{i+1}} = v | Y_{W_{[i]}} = u] = \frac{\Pr[X_{v_2} = v_1]}{n-i}$. Also $\Pr[Y_{i+1} = v | Y_{[i]} = u] = 0$ if v_2 has already been observed in the previous i observations; otherwise $\Pr[Y_{i+1} = v | Y_{[i]} = u] = \frac{\Pr[X_{v_2} = v_1]}{n-i}$.

Thus $Y_{W_{[i+1]}}$ and $Y_{[i+1]}$ are the same distributions. This finishes the proof.

□

Lemma 3.11. *If* $(\mathsf{Share}, \mathsf{Rec})$ *has non-adaptive privacy error* ϵ*, then* $(\mathsf{Share}', \mathsf{Rec}')$ *has adaptive privacy error* $\bar{n}(\epsilon + 2^{-\Omega(k)})$.

Proof. First we assume that the adaptive observer always observes k' shares. For every observer M which does not observe k' shares, there exists another observer M' which can observe the same shares as M and then observe some more shares. That is to say that if the number of observed shares is less than k', M' will choose more unobserved shares (sequentially in a fixed order) to observe until k' shares are observed. Since we can use a deterministic function to throw away the extra observes of M' to get what M should observe, by Lemma 2.2 part 2, if the privacy holds for M' then the privacy holds for M. As a result, we always consider observers which observe k' shares.

By Lemma 3.10, for any $s \in \Sigma_0^{m'}$, $\mathsf{Share}'(s)_W$, for any adaptive observation W, is the same distribution as $\mathsf{Share}'(s)_{W'}$ where $W = \{w_1, w_2, \ldots, w_{k'}\}$, $W' = [k']$. As W' is actually a non-adaptive observation, by Lemma 3.9, for distinct $s, s' \in \{0,1\}^{m'}$, $\mathsf{SD}(\mathsf{Share}'(s)_{W'}, \mathsf{Share}'(s')_{W'}) \le \bar{n}(\epsilon + 2^{-\Omega(k)})$. So

$$\mathsf{SD}(\mathsf{Share}'(s)_W, \mathsf{Share}'(s')_W) = \mathsf{SD}(\mathsf{Share}'(s)_{W'}, \mathsf{Share}'(s')_{W'}) \le \bar{n}(\epsilon + 2^{-\Omega(k)}).$$

□

Theorem 3.12. *For any $n, m \in \mathbb{N}, m \le n$, any $\epsilon, \eta \in [0,1]$ and any constant $a \ge 1, \alpha \in (0,1]$, if there exists an explicit (n,k) secret sharing scheme in* AC^0 *with share alphabet Σ, message alphabet Σ_0, message length m, non-adaptive privacy error ϵ and reconstruction error η, then there exists an explicit $(n' = O(n^a \log n), (1-\alpha)n')$ secret sharing scheme in* AC^0 *with share alphabet $\Sigma \times [n']$, message alphabet Σ_0, message length $\Omega(mn^{a-1})$, adaptive privacy error $O(n^{a-1}(\epsilon + 2^{-\Omega(k)}))$ and reconstruction error $O(n^{a-1}\eta)$.*

Proof. It immediately follows from Construction 3.2, Lemma 3.3, Lemma 3.4 and Lemma 3.11. □

3.2 Binary alphabet

In this subsection, we construct AC^0 secret sharing schemes with binary alphabet based on some existing schemes with binary alphabets, enlarging the privacy threshold.

If we simply break each share in Construction 3.2 into bits, then we in fact get a secret sharing scheme with non-adaptive privacy. However, the privacy threshold becomes $O(n/\log n)$ which is sublinear, as the observer does not have to observe the indices. To overcome the barrier, we use some coding techniques and secret sharing for small blocks. An even bigger problem is that whether we can achieve adaptive privacy in this case. It seems to be hard since we have to break the indices into pieces. But surprisingly, we are still able to show adaptive privacy.

Lemma 3.13 ([12] Section 4)**.** *For any $n \in \mathbb{N}$, any constant $\delta_0, \delta_1 \in (0,1)$, let $C \subseteq \mathbb{F}_2^n$ be an asymptotically good $(n, k = \delta_0 n, d = \delta_1 n)$ linear code.*

1. *There exists an (n,d) secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *with alphabet $\{0,1\}$, message length k,perfect privacy and reconstruction. Here $\forall x \in \{0,1\}^k$,* $\mathsf{Share}(x) = f(x) + c$ *with c drawn uniform randomly from $C^\perp$ (the dual code of C) and f is the encoding function from $\{0,1\}^k$ to C. For $y \in \{0,1\}^n$,* $\mathsf{Rec}(y)$ *is to find x such that there exists a $c \in C^\perp$ with $f(x) + c = y$.*

2. *For any $p =$* $\mathsf{poly}(n)$*, there exists an explicit (n,d) secret sharing scheme* $(\mathsf{Share}, \mathsf{Rec})$ *with alphabet $\{0,1\}^p$, message length k, perfect privacy and reconstruction.*

3. *If the codeword length is logarithmic (say $n = O(\log N)$ for some $N \in \mathbb{N}$), then both schemes can be constructed explicitly in* AC^0 *(in N).*

Proof. The first assertion is proved in [12].

The second assertion follows by applying the construction of the first assertion in parallel p times.

The third assertion holds because, when the codeword length is $O(\log N)$, both encoding and decoding functions have input length $O(\log N)$. For encoding, we can use any classic methods for generating asymptotically good binary codes. For decoding, we can try all possible messages to uniquely find the correct one. By Lemma 2.9, both functions can be computed by AC^0 circuits.

□

Now we give the secret sharing scheme in AC^0 with a constant privacy rate while having binary alphabet.

Construction 3.14. *For any $n, k, m \in \mathbb{N}$ with $k, m \leq n$, let $(\mathsf{Share}, \mathsf{Rec})$ be an (n, k) secret sharing scheme with alphabet $\{0,1\}$, message length m.*

Let $(\mathsf{Share}_C, \mathsf{Rec}_C)$ be an (n_C, k_C) secret sharing scheme with alphabet $\{0,1\}^{p=O(\log n)}$, message length m_C by Lemma 3.13, where $m_C = \delta_0 n_C$, $k_C = \delta_1 n_C$, $n_C = O(\log n)$ for some constants δ_0, δ_1.

Let $(\mathsf{Share}_0, \mathsf{Rec}_0)$ be an (n_0, k_0) secret sharing scheme with alphabet $\{0,1\}$, message length m_0 by Lemma 3.13, where $m_0 = \delta_0 n_0 = p + O(\log n)$, $k_0 = \delta_1 n_0$.

For any constant $a \geq 1$, we can construct the following $(n' = O(n^a), k' = \Omega(n'))$ secret sharing scheme $(\mathsf{Share}', \mathsf{Rec}')$ with alphabet $\{0,1\}$, message length $m' = m\bar{n}$, where $\bar{n} = \Theta(n^{a-1})$ is large enough.

Function $\mathsf{Share}' : \{0,1\}^{m'} \to \{0,1\}^{n'}$ is as follows.

1. *On input $x \in \{0,1\}^{m\bar{n}}$, parse it to be $(x_1, x_2, \dots, x_{\bar{n}}) \in (\{0,1\}^m)^{\bar{n}}$.*
2. *Compute $y = (y_1, \dots, y_{\bar{n}}) = (\mathsf{Share}(x_1), \dots, \mathsf{Share}(x_{\bar{n}})) \in (\{0,1\}^n)^{\bar{n}}$. Split each entry to be blocks each has length pm_C to get $\hat{y} = (\hat{y}_1, \dots, \hat{y}_{\tilde{n}}) \in (\{0,1\}^{pm_C})^{\tilde{n}}$, where $\tilde{n} = \bar{n}\lceil \frac{n}{pm_C} \rceil$.*
3. *Let $y^* = (\mathsf{Share}_C(\hat{y}_1), \dots, \mathsf{Share}_C(\hat{y}_{\tilde{n}}))$. Parse it to be $y^* = (y^*_1, \dots, y^*_{n^*}) \in (\{0,1\}^p)^{n^*}$, $n^* = \tilde{n} n_C$.*
4. *Generate π by Lemma 3.1 which is uniform random over permutations of $[n^*]$. If it failed, which can be detected by checking element distinctness, set π to be such that $\forall i \in [n^*], \pi(i) = i$.*
5. *Compute*

$$z(x) = \mathsf{Share}'(x) = (\mathsf{Share}_0(y^*_{\pi^{-1}(1)} \circ \pi^{-1}(1)), \dots, \mathsf{Share}_0(y^*_{\pi^{-1}(n^*)} \circ \pi^{-1}(n^*))) \in (\{0,1\}^{n_0})^{n^*}.$$

6. *Parse $z(x)$ to be bits and output.*

Function $\mathsf{Rec}' : \{0,1\}^{n'=n_0 n^*} \to \{0,1\}^{m'}$ is as follows.

1. *Parse the input bits to be $z \in (\{0,1\}^{n_0})^{n^*}$ and compute*

$$(y^*_{\pi^{-1}(1)} \circ \pi^{-1}(1), \dots, y^*_{\pi^{-1}(n^*)} \circ \pi^{-1}(n^*)) = (\mathsf{Rec}_0(z_1), \dots, \mathsf{Rec}_0(z_{n^*})).$$

2. *Compute $y^* = (y^*_1, \dots, y^*_{n^*})$.*
3. *Compute $\hat{y}$ by applying Rec_C on y^* for every successive n_C entries.*
4. *Parse $\hat{y}$ to be y.*
5. *Compute x by applying Rec on every entry of y.*

Lemma 3.15. *If Share and Rec can be computed by AC^0 circuits, then Share' and Rec' can be computed by AC^0 circuits.*

Proof. As Share can be computed by an AC^0 circuit, y can be computed by an AC^0 circuit. By Lemma 2.9 part 2 and 3.13, we know that $(\mathsf{Share}_C, \mathsf{Rec}_C)$ both can be computed by AC^0 circuits. By Lemma 3.1, π can be computed by an AC^0 circuit. Also $\forall i \in [n^*], y^*_{\pi^{-1}(i)} = \bigvee_{j\in[n^*]}(y^*_j \wedge (j = \pi^{-1}(i)))$. Thus Share' can be computed by an AC^0 circuit.

For Rec', $\forall i \in [n^*], y^*_i = \bigvee_{j\in[n^*]}(y^*_{\pi^{-1}(j)} \wedge (\pi^{-1}(j) = i))$. As Rec_C can be computed by an AC^0 circuit, y can be computed by an AC^0 circuit. As Rec can be computed by an AC^0 circuit, Rec' can be computed by an AC^0 circuit. □

Lemma 3.16. *If the reconstruction error of* (Share, Rec) *is* η*, then the reconstruction error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\eta' = \bar{n}\eta$.

Proof. As $(\mathsf{Share}_0, \mathsf{Rec}_0)$ and $(\mathsf{Share}_C, \mathsf{Rec}_C)$ have perfect reconstruction by Lemma 3.13, the y computed in Rec' is exactly $(\mathsf{Share}(x_1), \mathsf{Share}(x_2), \ldots, \mathsf{Share}(x_{\bar{n}}))$. As $\forall i \in [\bar{n}], \Pr[\mathsf{Rec}(\mathsf{Share}(x_i)) = x_i] \geq 1 - \eta$,

$$\Pr[\mathsf{Rec}'(\mathsf{Share}'(x)) = x] = \Pr[\bigwedge_{i\in[\bar{n}]}(\mathsf{Rec}(\mathsf{Share}(x_i)) = x_i)] \geq 1 - \bar{n}\eta,$$

by the union bound. □

Lemma 3.17. *If the non-adaptive privacy error of* (Share, Rec) *is* ϵ*, then the non-adaptive privacy error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\bar{n}(\epsilon + 2^{-\Omega(k/\log^2 n)})$.

Proof. Let $k' = 0.9\delta_1^2 n'$. We show that there exists a distribution $\mathcal{D}$ such that for any string $x \in \{0,1\}^m$, for any $W \subseteq [n']$ with $|W| \leq k'$,

$$\mathsf{SD}(\mathsf{Share}'(x)_W, \mathcal{D}) \leq \bar{n}(\epsilon + 2^{-\Omega(k/\log^2 n)}).$$

Let $\mathcal{D}$ be $\mathsf{Share}'(\sigma)_W$ for some $\sigma \in \{0,1\}^{m'}$.

Consider an arbitrary observation $W \subseteq [n']$, with $|W| \leq k'$. Note that for at least $1 - 0.9\delta_1$ fraction of all blocks $z_i \in \{0,1\}^{n_0}, i = 1, \ldots, n^*$, at most δ_1 fraction of the bits in the block can be observed. Otherwise the number of observed bits is more than $0.9\delta_1 \times \delta_1 n'$. Let W^* be the index set of those blocks which have more than δ_1 fraction of bits being observed.

For every $i \in [n^*]\backslash W^*$, z_i is independent of $y^*_{\pi^{-1}(i)} \circ \pi^{-1}(i)$ by the privacy of $(\mathsf{Share}_0, \mathsf{Rec}_0)$. Note that z_i is also independent of $z_{i'}, i' \in [n^*], i' \neq i$ since it is independent of $y^*_{\pi^{-1}(i)} \circ \pi^{-1}(i)$ (its randomness is only from the randomness of the Share_0 function) and every Share_0 function uses independent randomness. So we only have to show that

$$\mathsf{SD}(z_{W^*}(x), z_{W^*}(\sigma)) \leq \bar{n}(\epsilon + 2^{-\Omega(k/\log^2 n)}).$$

For every $i \in [\tilde{n}]$, let $S_i = \{(i-1)n_C + 1, \ldots, in_C\}$. Let X_i be the indicator that $|\pi(S_i) \cap W^*| > k_C, i \in [\tilde{n}]$. Note that $\mathbb{E}[|\pi(S_i) \cap W^*|] \leq 0.9\delta_1 n_C = 0.9k_C$.

For every $i \in [\bar{n}]$, let E_i be the event that $\sum_{j=(i-1)\lceil \frac{n}{pm_C}\rceil+1}^{i\lceil \frac{n}{pm_C}\rceil} X_j \leq \frac{k}{pm_C}$. Let $E = \bigcap_{i\in[\bar{n}]} E_i$. We take $\bar{n}$ to be large enough such that $n_C\lceil \frac{n}{pm_C}\rceil \leq \frac{0.9\times 0.1}{1+0.9\times 0.1}|W^*|$. For every $i \in [\bar{n}]$, by Lemma 3.8,

$$1 - \Pr[E_i] \leq e^{-2k/(pm_C) + (e^2-1)e^{-\Omega(0.9\delta_1^2 n_C)}\lceil \frac{n}{pm_C}\rceil}.$$

We take $n_C = O(\log n)$ to be large enough such that the probability is at most $e^{-\Omega(k/(pm_C))} \leq e^{-\Omega(k/\log^2 n)}$.

Next we do a similar argument as that in the proof of Lemma 3.9. We know that $\Pr[E] \geq 1 - \bar{n}e^{-\Omega(k/\log^2 n)}$. We claim that $z_{W^*}(x)|E$ and $z_{W^*}(\sigma)|E$ have statistical distance at most $\bar{n}\epsilon$. The reason follows.

Let's fix a permutation π for which E happens. We claim that $z_{W^*}(x)$ is a deterministic function of at most k bits of each y_i for $i \in [\bar{n}]$ and some extra uniform random bits. This is because, as E happens, for those $i \in [\tilde{n}]$ with $|\pi(S_i) \cap W^*| \leq k_C$, the shares in $\pi(S_i) \cap W^*$ are independent of the secret by the privacy of $(\mathsf{Share}_C, \mathsf{Rec}_C)$. Note that they are also independent of other shares since the construction uses independent randomness for $\mathsf{Share}_C(\hat{y}_i), i \in [\tilde{n}]$. For those $i \in [\tilde{n}]$ with $|\pi(S_i) \cap W^*| > k_C$, the total number of them is at most $\frac{k}{pm_C}$. By the construction, $\mathsf{Share}'(x)_{W^*}$ is computed from at most $\frac{k}{pm_C} \times pm_C = k$ bits of each y_i for $i \in [\bar{n}]$ and some extra uniform random bits. Hence by the privacy of $(\mathsf{Share}, \mathsf{Rec})$ and noting that $y_i, \in [\bar{n}]$ are generated using independent randomness,

$$\mathsf{SD}(z_{W^*}(x), z_{W^*}(\sigma)) \leq \bar{n}\epsilon.$$

Thus with probability at least $1 - \bar{n}e^{-\Omega(k/\log^2 n)}$ over the fixing of π, $z_{W^*}(x)$ and $z_{W^*}(\sigma)$ have statistical distance at most $\bar{n}\epsilon$, which means that

$$\mathsf{SD}(z_{W^*}(x), z_{W^*}(\sigma)) \leq \bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}).$$

□

Lemma 3.18. *For any alphabet Σ, any $n \in \mathbb{N}$, Let $X = (X_1, \ldots, X_n)$ be an arbitrary distribution over Σ^n. For any $n_0, k_0 \in \mathbb{N}$ with $k_0 \leq n_0$, let $(\mathsf{Share}_0, \mathsf{Rec}_0)$ be an arbitrary (n_0, k_0)-secret sharing scheme with binary alphabet, message length $m_0 = \log|\Sigma| + O(\log n)$, perfect privacy. Let $Y = (\mathsf{Share}_0(X_{\pi^{-1}(1)} \circ \pi^{-1}(1)), \ldots, \mathsf{Share}_0(X_{\pi^{-1}(n)} \circ \pi^{-1}(n)))$ where π is a random permutation over $[n] \to [n]$. For any $t \leq n \cdot k_0$, let W be an any adaptive observation which observes t shares. Then there exists a deterministic function $f : \{0,1\}^{\mathsf{poly}(n)} \to \{0,1\}^t$ such that Y_W has the same distribution as $f(Y_{W'} \circ S)$, where S is uniform over $\{0,1\}^{\mathsf{poly}(n)}$ and $W' = [t'n_0], t' = \lceil \frac{t}{k_0} \rceil$.*

Proof. For every $i \in [n]$, Let $B_i = \{(i-1)n_0 + 1, \ldots, in_0\}$. Assume the adaptive adversary is M.

Let f be defined as the following.

Algorithm 3.1: $f(\cdot)$

```
Input: $y \in \{0,1\}^{t'n_0}$, $s \in \{0,1\}^{\mathsf{poly}(n)}$
Let $c = 1$;
$\forall i \in [n], l_i \in [n] \cup \{\mathsf{null}\}$ is assigned to be null;
Compute the secrets for the $t'$ blocks $y$, which are
    $$(x_1, \ldots, x_{t'}) \in (\{0,1\}^{m_0})^{t'};$$
Compute $(\mathsf{Share}_0(\sigma), \ldots, \mathsf{Share}_0(\sigma)) \in (\{0,1\}^{n_0})^n$ and parse it to be $r \in \{0,1\}^{n_0 n}$, for an arbitrary $\sigma \in \Sigma$. Here for each $\mathsf{Share}_0$ function, we take some unused bits from $s$ as the random bits used in that function.
Next $f$ does the following computation by calling $M$;
while $M$ wants to observe the ith bit which is not observed previously do
    Find $j \in [n]$ such that $i \in B_j$;
    if the number of observed bits in the $j$th block is less than $k_0$ then
        Let $M$ observe $r_i$;
    else
        Let $I_j$ be the indices of the observed bits in the $j$th block. (The indices here are the relative indices in the $j$th block)
        if $l_j = \mathsf{null}$ then
            $l_j = c$;
            $c = c + 1$;
            Draw a string $v^j$ from $\mathsf{Share}_0(x_c)|_{\mathsf{Share}_0(x_c)_{I_j} = r_{(j-1)n_0 + I_j}}$ by using some unused bits of $s$;
        end
        Let $M$ observe $v^j_{i-(j-1)n_0}$;
    end
end
```

Let $W = (w_1, \ldots, w_t) \in [n \cdot n_0]^t$, $Z = f(Y_{W'} \circ S)$. Let $R \in \{0,1\}^{nn_0}$ be the random variable corresponds to r.

We use induction to show that Y_W has the same distribution as Z.

For the base case, the first bits of both random variables have the same distributions by the perfect privacy of $(\mathsf{Share}_0, \mathsf{Rec}_0)$.

For the inductive step, assume that, projected on the first d bits, the two distributions are the same. Fix the first d observed bits for both Y_W and Z to be $\bar{y} \in \{0,1\}^d$. Assume that the $(d+1)$th observation is to observe the w_dth bit where w_d is in B_j for some j.

If the number of observed bits in the jth block is less than k_0 then $Y_{\{w_1,\ldots,w_{d+1}\} \cap B_j}$ has the same distribution as $R_{\{w_1,\ldots,w_{d+1}\} \cap B_j}$, following the privacy of $(\mathsf{Share}_0, \mathsf{Rec}_0)$. Note that the blocks $Y_{\{w_1,\ldots,w_{d+1}\} \cap B_i}$, $i \in [n]$ are independent. The blocks $R_{\{w_1,\ldots,w_{d+1}\} \cap B_i}, i \in [n]$ are also independent. As f will output $R_{w_{d+1}}$, the conclusion holds for $d+1$.

Else, if the number of observed bits in the jth block is at least k_0, it is sufficient to show that $Y_{w_{d+1}}|_{Y_{\{w_1,\ldots,w_d\}} = \bar{y}}$ has the same distribution as that of $Z_{d+1}|_{Z_{\{1,\ldots,d\}} = \bar{y}}$. Note that there are c blocks such that W observes more than k_0 bits for each of them. Let $q_1, \ldots, q_c$ denote those blocks. Let $I = ((q_1 - 1)n_0 + I_{q_1}, \ldots, (q_c - 1)n_0 + I_{q_c})$, which is the set of indices of all observed bits. Note that $I \subseteq \{w_1, \ldots, w_d\}$.

By the privacy of the secret sharing scheme, for those blocks which have at most k_0 bits being observed, they are independent of the secret and hence independent of other blocks. So $Y_{w_{d+1}}|_{Y_{\{w_1,\ldots,w_d\}} = \bar{y}}$ is in fact $Y_{w_{d+1}}|_{Y_I = y^*}$ where y^* are the corresponding bits from $\bar{y}$ with a proper rearrangement according to I. From the definition of f we know that for $i \in [c]$, the observed bits in the q_ith block is exactly the same distribution as $(Y_{B_{l_{q_i}}})_{I_{q_i}} = \mathsf{Share}_0(x_{l_{q_i}})_{I_{q_i}}$. So for $Z_{d+1}|_{Z_{\{1,\dots,d\}}=\bar{y}}$, it is the same distribution as

$$\begin{aligned} T &= (Y_{B_{l_j}})_{w_d-(j-1)n_0}|_{\bigwedge_{i=1}^{c}((Y_{B_{l_{q_i}}})_{I_{q_i}}=y^*_{(q_i-1)n_0+I_{q_i}})} \\ &= \mathsf{Share}_0(x_{l_j})_{w_d-(j-1)n_0}|_{\bigwedge_{i=1}^{c}(\mathsf{Share}_0(x_{l_{q_i}})_{I_{q_i}}=y^*_{(q_i-1)n_0+I_{q_i}})}. \end{aligned} \tag{2}$$

By Lemma 3.10, $(Y_{B_{q_1}}, \dots, Y_{B_{q_c}})$ has the same distribution as $(Y_{B_{l_{q_1}}}, \dots, Y_{B_{l_{q_c}}})$ as they both are the same distribution as $(\mathsf{Share}_0(x_1), \dots, \mathsf{Share}_0(x_c))$. Thus $Y_{w_{d+1}}|_{Y_I=y^*}$ has the same distribution as T, as $Y_{w_{d+1}}|_{Y_I=y^*}$ is the distribution of some bits in $(Y_{B_{q_1}}, \dots, Y_{B_{q_c}})$ and T is the distribution of the corresponding bits (same indices) in $(Y_{B_{l_{q_1}}}, \dots, Y_{B_{l_{q_c}}})$. So we know that $Y_{w_{d+1}}|_{Y_{\{w_1,\dots,w_d\}}=\bar{y}}$ has the same distribution as $Z_{d+1}|_{Z_{\{1,\dots,d\}}=\bar{y}}$ and this shows our conclusion.

□

Lemma 3.19. *If the non-adaptive privacy error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* ϵ*, then the adaptive privacy error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\bar{n}(\epsilon + 2^{-\Omega(k/\log^2 n)})$.

Proof. Let W be an adaptive observation . Let $W' = [\lceil |W|/k_0 \rceil n_0]$. Let $|W| = \Omega(n')$ be small enough such that $|W'| \le 0.9\delta_1^2 n'$. By Lemma 3.18, there exists a deterministic function f such that for any $x, x' \in \{0,1\}^{m'}$, $\mathsf{SD}(\mathsf{Share}'(x)_W, \mathsf{Share}(x')_W) = \mathsf{SD}(f(\mathsf{Share}'(x)_{W'} \circ S), f(\mathsf{Share}'(x')_{W'} \circ S))$ where S is the uniform distribution as defined in Lemma 3.18 which is independent of $\mathsf{Share}'(x)_{W'}$ or $\mathsf{Share}'(x')_{W'}$. By Lemma 2.2, we know that

$$\mathsf{SD}(f(\mathsf{Share}'(x)_{W'} \circ S), f(\mathsf{Share}'(x')_{W'} \circ S)) \le \mathsf{SD}(\mathsf{Share}'(x)_{W'}, \mathsf{Share}'(x')_{W'}).$$

By Lemma 3.17 we know that

$$\mathsf{SD}(\mathsf{Share}'(x)_{W'}, \mathsf{Share}'(x')_{W'}) \le \bar{n}(\epsilon + 2^{-\Omega(k/\log^2 n)}).$$

Hence

$$\mathsf{SD}(\mathsf{Share}'(x)_W, \mathsf{Share}'(x')_W) \le \bar{n}(\epsilon + 2^{-\Omega(k/\log^2 n)}).$$

□

Theorem 3.20. *For any* $n, m \in \mathbb{N}, m \le n$*, any* $\epsilon, \eta \in [0,1]$ *and any constant* $a \ge 1$*, if there exists an explicit* (n,k) *secret sharing scheme in* AC^0 *with alphabet* $\{0,1\}$*, message length* m*, non-adaptive privacy error* ϵ *and reconstruction error* η*, then there exists an explicit* $(n' = O(n^a), k' = \Omega(n'))$ *secret sharing scheme in* AC^0 *with alphabet* $\{0,1\}$*, message length* $\Omega(mn^{a-1})$*, adaptive privacy error* $O(n^{a-1}(\epsilon + 2^{-\Omega(k/\log^2 n)}))$ *and reconstruction error* $O(n^{a-1}\eta)$.

Proof. It follows from Construction 3.14, Lemma 3.15, 3.16 and 3.19.

□

4 k-wise independent generator in AC0

In this section we focus on increasing the secret length to be linear of the number of shares while keeping the construction in AC^0. The privacy rate is not as good as the previous section. The main technique is to use the following well known k-wise independent generator which is constructed from expander graphs.

Theorem 4.1 ([30])**.** *For any* $N, D, M \in \mathbb{N}$*, any* $\epsilon > 0$*, if there exists a* $(\le K_{\max}, (\frac{1}{2}+\epsilon)D)$ *expander with left set of vertices* $[N]$*, right set of vertices* $[M]$*, left degree* D*, then the function* $g : \{0,1\}^M \to \{0,1\}^N$*, defined by* $g(x)_i = \bigoplus_{j\in[D]} x_{\Gamma(i,j)}, i = 1, 2, \dots, N$*, is a* $K_{\max}$*-wise uniform independent generator.*

Proof. For any subset $S \subseteq [N]$ with $|S| \leq K_{\max}$, there exists a $u \in \Gamma(S)$ such that $\exists v \in S, u \in \Gamma(v)$ while $\forall w \in S$ with $w \neq v$, $u \notin \Gamma(w)$. This is because if not, then $|\Gamma(S)| \leq \frac{1}{2}D|S|$ which contradicts that Γ is a $(\leq K_{\max}, (\frac{1}{2}+\epsilon)D)$ expander.

As

$$\bigoplus_{i \in S} g(x)_i = \bigoplus_{i \in S} \bigoplus_{j \in [D]} x_{\Gamma(i,j)},$$

$\bigoplus_{i \in S} g(U_M)_i$ is uniform.

By the Information Theoretic XOR-Lemma of [17], for every set $S' \subseteq [N]$ of size $K_{\max}$, $g(U_M)_{S'}$ is uniform. Thus g is a $K_{\max}$-wise uniform independent generator. □

Theorem 4.2. *For any $M \in \mathbb{N}$, $N = \mathsf{poly}(M)$, any alphabets Σ_0, Σ, any constant $\gamma \in (0,1]$, there exists an explicit K-wise independent generator $g : \Sigma_0^M \to \Sigma^N$ in AC^0, where $K = (\frac{M \log|\Sigma_0|}{\log|\Sigma|})^{1-\gamma}$.*

Proof. We first consider $\Sigma_0 = \Sigma = \{0,1\}$. By Theorem 2.7 for any constant α and every $\epsilon > 0$ there exists an explicit function $\Gamma : [N] \times [D] \to [M]$ which is the neighbour function of a $(\leq K_{\max}, (\frac{1}{2}+\epsilon)D)$ expander, where $D = O((\log N)(\log K_{\max})/\epsilon)^{1+1/\alpha}$ and $M \leq D^2 K_{\max}^{1+\alpha}$. We take $\epsilon = 0.1$ and take α to be a small enough constant such that $K_{\max} \geq M^{1-\gamma}$. By Theorem 4.1 we get an explicit K-wise independent generator $g : \{0,1\}^M \to \{0,1\}^N$ where $K = M^{1-\gamma}$.

For arbitrary alphabets Σ_0, Σ, we simply apply the generator for $\log|\Sigma|$ times in parallel using independent seeds. Note that the total seed length is $M \log|\Sigma_0|$ by parsing the input symbols into bits. So for every one of the $|\Sigma|$ generator in parallel, its seed length is $\frac{M \log|\Sigma_0|}{\log|\Sigma|}$. Hence each of them is a K-wise uniform independent generator with $K = (\frac{M \log|\Sigma_0|}{\log|\Sigma|})^{1-\gamma}$.

By Theorem 4.1, the construction take XOR over D bits. So by Lemma 2.9, it can be computed in AC^0. Thus the generator can be computed in AC^0.

□

Now we give the construction of secret sharing schemes in AC^0 with large message rate (saying $1 - 1/\mathsf{poly}(n)$).

Construction 4.3. *For any $n, k, m \in \mathbb{N}$ with $k \leq n$, any alphabets Σ_0, Σ, let $(\mathsf{Share}, \mathsf{Rec})$ be an (n,k) secret sharing scheme with share alphabet Σ, message alphabet Σ_0, message length m.*

For any constant $a > 1, \gamma \in (0,1]$, we construct the following $(n' = n + m', k' = \min(k, l))$ secret sharing scheme $(\mathsf{Share}', \mathsf{Rec}')$ with alphabet Σ, message length $m' = \Omega(n^a)$, where $l = \Theta(\frac{m \log|\Sigma_0|}{\log|\Sigma|})^{1-\gamma}$.

The function $\mathsf{Share}' : \Sigma^{m'} \to \Sigma^{n'}$ is as follows.

1. *Let $g_\Gamma : \Sigma_0^m \to \Sigma^{m'}$ be the l-wise independent generator by Theorem 4.2.*

2. *For secret $x \in \Sigma^{m'}$, we draw r uniform randomly from Σ_0^m let*

$$\mathsf{Share}'(x) = (\mathsf{Share}(r), g_\Gamma(r) \oplus x).$$

The function $\mathsf{Rec}' : \Sigma^{n'} \to \Sigma^{m'}$ is as follows.

1. *The input is $y = (y_1, y_2)$ where $y_1 \in \Sigma^n, y_2 \in \Sigma^{m'}$.*

2. *Let*

$$\mathsf{Rec}'(y) = g_\Gamma(\mathsf{Rec}(y_1)) \oplus y_2.$$

Lemma 4.4. *If Share and Rec can be computed by AC^0 circuits, then Share' and Rec' can be computed by AC^0 circuits.*

Proof. As Share can be computed by an AC^0 circuit and $g_\Gamma(r) \oplus x$ can be computed by a CNF or DNF, Share' can be computed by an AC^0 circuit.

Similarly, Rec' can also be computed by an AC^0 circuit.

□

Lemma 4.5. *If the reconstruction error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* η*, then the reconstruction error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\eta' = \eta$*.*

Proof. Let the input for Rec' be $(y_1, y_2) = (\mathsf{Share}(r), g_\Gamma(r) \oplus x)$. If $\mathsf{Rec}(\cdot)$ computes correctly then

$$\mathsf{Rec}'(y) = g_\Gamma(\mathsf{Rec}(y_1)) \oplus y_2 = g_\Gamma(r) \oplus g_\Gamma(r) \oplus x = x,$$

which means that Rec' recovers the correct secret.

So $\eta' = \eta$.

□

Next we show the non-adaptive privacy error of the construction.

Lemma 4.6. *If the non-adaptive privacy error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* ϵ*, then the non-adaptive privacy error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is also* ϵ*.*

Proof. Consider an arbitrary set $W \subseteq [n']$ of size k'. We view W as the union of two disjoint sets $W_1 \subseteq [n]$ and $W_2 \subseteq \{n+1, \ldots, n+m'\}$. Consider any two distinct secrets $x, x' \in \Sigma^{m'}$. As g_Γ is an l-wise independent generator and $k' \le l$, $\mathsf{Share}'(x)_{W_2} = (g_\Gamma(R) \oplus x)_{W_2}$ and $\mathsf{Share}'(x')_{W_2} = (g_\Gamma(R) \oplus x')_{W_2}$ are both uniform distributions where R is uniform over Σ^m. For any string $u \in \Sigma^{m'}$, the statistical distance between the distribution $\mathsf{Share}'(x)_{W_1}|_{\mathsf{Share}'(x)_{W_2}=u}$ and the distribution $\mathsf{Share}'(x')_{W_1}|_{\mathsf{Share}'(x')_{W_2}=u}$ is ϵ, because $(\mathsf{Share}, \mathsf{Rec})$ has privacy error ϵ. So we have that

$$\begin{aligned}
&\mathsf{SD}(\mathsf{Share}'(x)_W, \mathsf{Share}'(x')_W) \\
=&\sum_{u \in \Sigma^{m'}} \frac{1}{|\Sigma|^{m'}} \mathsf{SD}(\mathsf{Share}'(x)_{W_1}|_{\mathsf{Share}'(x)_{W_2}=u}, \mathsf{Share}'(x')_{W_1}|_{\mathsf{Share}'(x')_{W_2}=u}) \qquad (3)\\
=&\epsilon.
\end{aligned}$$

□

Theorem 4.7. *For any* $n, m \in \mathbb{N}, m \le n$*, any* $\epsilon, \eta \in [0,1]$ *, any constant* $\gamma \in (0,1]$*, any* $m' = \mathsf{poly}(n)$ *and any alphabets* Σ_0, Σ*, if there exists an explicit* (n, k) *secret sharing scheme in* AC^0 *with share alphabet* Σ*, message alphabet* Σ_0*, message length* m*, non-adaptive privacy error* ϵ *and reconstruction error* η*, then there exists an explicit* $(n + m', \min(k, (\frac{m \log |\Sigma_0|}{\log |\Sigma|})^{1-\gamma}))$ *secret sharing scheme in* AC^0 *with alphabet* Σ*, message length* m'*, non-adaptive privacy error* ϵ *and reconstruction error* η*.*

Proof. It immediately follows from Construction 4.3, Lemma 4.4, 4.5 and 4.6.

□

5 Final construction

In this section we give our final AC^0 construction of secret sharing schemes which has constant message rate and constant privacy rate.

Our construction will use both random permutation and k-wise independent generator proposed in the previous sections.

5.1 The construction

We first give the construction with a relatively big alphabet.

Construction 5.1. *For any $n, k, m \in \mathbb{N}$ with $k, m \leq n$, any alphabets Σ_0, Σ, let $(\mathsf{Share}, \mathsf{Rec})$ be an (n, k) secret sharing scheme with share alphabet Σ, message alphabet Σ_0, message length m.*

Let $(\mathsf{Share}_C, \mathsf{Rec}_C)$ be an (n_C, k_C) secret sharing scheme from Lemma 3.13 with alphabet Σ, message length m_C, where $m_C = \delta_0 n_C$, $k_C = \delta_1 n_C$, $n_C = O(\log n)$ for some constant δ_0, δ_1.

For any constant $a \geq 1$, $\gamma \in (0, 1]$, we can construct the following $(n' = O(n^a), k' = \Omega(n')$ secret sharing scheme $(\mathsf{Share}', \mathsf{Rec}')$ with share alphabet $\Sigma \times [n']$, message alphabet Σ, message length $m' = \Omega(n')$.

The function $\mathsf{Share}' : \Sigma^{m'} \to (\Sigma \times [n'])^{n'}$ is as follows.

1. *Let $\bar{n} = \Theta(n^{a-1})$ where the constant factor is large enough.*
2. *Let $g_\Gamma : \Sigma_0^{m\bar{n}} \to \Sigma^{m'}$ be the l-wise independent generator by Theorem 4.2, where $l = \Omega(\frac{m\bar{n} \log |\Sigma_0|}{\log |\Sigma|})^{1-\gamma}$.*
3. *For secret $x \in \Sigma^{m'}$, we draw a string $r = (r_1, \ldots, r_{\bar{n}})$ uniformly from $(\Sigma_0^m)^{\bar{n}}$.*
4. *Let $y = (y_s, y_g)$, where $y_s = (\mathsf{Share}(r_1), \ldots, \mathsf{Share}(r_{\bar{n}})) \in (\Sigma^n)^{\bar{n}}$ and $y_g = g_\Gamma(r) \oplus x \in \Sigma^{m'}$.*
5. *Get $\hat{y}_s \in (\Sigma^{m_C})^{n_s}$ from y_s by parsing $y_{s,i}$ to be blocks each having length m_C for every $i \in [\bar{n}]$, where $n_s = \lceil \frac{n}{m_C} \rceil \bar{n}$.*
6. *Get $\hat{y}_g \in (\Sigma^{m_C})^{n_g}$ from y_g by parsing y_g to be blocks each having length m_C, where $n_g = \lceil \frac{m'}{m_C} \rceil$.*
7. *Compute*
$$(\mathsf{Share}_C(\hat{y}_{s,1}), \ldots, \mathsf{Share}_C(\hat{y}_{s,n_s}), \mathsf{Share}_C(\hat{y}_{g,1}), \ldots, \mathsf{Share}'_C(\hat{y}_{g,n_g})).$$
and parse it to be $y' = (y'_1, \ldots, y'_{n'}) \in \Sigma^{n'}$, where $n' = (n_s + n_g) n_C$.
8. *Generate a random permutation $\pi : [n'] \to [n']$ and output*
$$z = ((y'_{\pi^{-1}(1)} \circ \pi^{-1}(1)), (y'_{\pi^{-1}(2)} \circ \pi^{-1}(2)), \ldots, (y'_{\pi^{-1}(n')} \circ \pi^{-1}(n'))) \in (\Sigma \times [n'])^{n'}.$$

The function $\mathsf{Rec}' : (\Sigma \times [n'])^{n'} \to \Sigma^{m'}$ is as follows.

1. *The input is $z = ((y'_{\pi^{-1}(1)} \circ \pi^{-1}(1)), (y'_{\pi^{-1}(2)} \circ \pi^{-1}(2)), \ldots, (y'_{\pi^{-1}(n')} \circ \pi^{-1}(n')))$.*
2. *Compute $y' = (y'_1, \ldots, y'_{n'})$.*
3. *Parse y' to be (y'_s, y'_g) where $y'_s = (y'_{s,1}, \ldots, y'_{s,n_s}) \in (\Sigma^{n_C})^{n_s}$, $y'_g = (y'_{g,1}, \ldots, y'_{g,n_g}) \in (\Sigma^{n_C})^{n_g}$.*
4. *Compute $(\mathsf{Rec}_C(y'_{s,1}), \ldots, \mathsf{Rec}_C(y'_{s,n_g}))$ and $(\mathsf{Rec}_C(y'_{g,1}), \ldots, \mathsf{Rec}_C(y'_{g,n_g}))$. Parse them to get y_s and y_g.*
5. *Compute r by applying Rec on every entry of y_s.*
6. *Output*
$$\mathsf{Rec}'(z) = g_\Gamma(r) \oplus y_g.$$

Lemma 5.2. *If $(\mathsf{Share}, \mathsf{Rec})$ can be computed by AC^0 circuits, then $(\mathsf{Share}', \mathsf{Rec}')$ can be computed by AC^0 circuits.*

Proof. By Theorem 4.2, g_Γ can be computed by an AC^0 circuit. As Share can be computed by an AC^0 circuit, y can be computed by an AC^0 circuit. By Lemma 2.9 part 2, as $n_C = O(\log n)$, $(\mathsf{Share}_C, \mathsf{Rec}_C)$ can be computed by an AC^0 circuit. By Lemma 3.1 the random permutation π can be computed by an AC^0 circuit. Also $\forall i \in [n'], y'_{\pi^{-1}(i)} = \bigvee_{j\in[n']}(y'_j \wedge (j = \pi^{-1}(i)))$. Thus Share' can be computed by an AC^0 circuit.

For Rec', $\forall i \in [n'], y'_i = \bigvee_{j\in[n']}(y'_{\pi^{-1}(j)} \wedge (\pi^{-1}(j) = i))$. As Rec_C and Rec can be computed by an AC^0 circuits, Rec' can be computed by an AC^0 circuit.

□

Lemma 5.3. *If the reconstruction error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* η*, then the reconstruction error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\eta' = \bar{n}\eta$.

Proof. As $(\mathsf{Share}_C, \mathsf{Rec}_C)$ has perfect reconstruction, the error only occurs when we apply Rec. So we can compute each $r_i, i = 1, \dots, \bar{n}$ correctly except with error η. By the union bound, with probability $1 - \bar{n}\eta$, r is correctly computed. Once we can compute r correctly, the secret $x = g_\Gamma(r) \oplus y_g$. Note that the correctness of y_g is guaranteed. This is because from z we can get the value of every entry of y' since each entry of z includes one entry of y' and an index showing which entry of y' it is. So y_g is correct as it is part of y'.

□

Lemma 5.4. *If the non-adaptive privacy error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* ϵ*, then the non-adaptive privacy error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\bar{n}(\epsilon + e^{-\Omega(k/\log n)}) + e^{-\Omega(l/\log n)}$.

Proof. Let $k' = 0.9\delta_1 n'$. Consider an arbitrary $W \subseteq [n']$ with $|W| \le k'$.

For every $i \in [n_s]$, let $S_i = \{n_C(i-1)+1, \dots, n_C i\}$. Let $\mathcal{S} = \{S_1, \dots, S_{n_s}\}$. Let X_i be the indicator such that $X_i = 1$ is the event $|\pi(S_i) \cap W| > k_C$. Note that $\mathbb{E}[|\pi(S_i) \cap W|] \le 0.9\delta_1 n_C = 0.9k_C$.

For every $i \in [\bar{n}]$, let E_i be the event that $\sum_{j=(i-1)\lceil \frac{n}{m_C}\rceil+1}^{i\lceil \frac{n}{m_C}\rceil} X_j \le \frac{k}{m_C}$. Since $\bar{n}$ is large enough, $n_C\lceil \frac{n}{m_C}\rceil \le \frac{0.9\times 0.1}{1+0.9\times 0.1}|W|$. By Lemma 3.8,

$$1 - \Pr[E_i] \le e^{-2k/m_C + (e^2-1)e^{-\Omega(0.9\delta_1 n_C)}\lceil \frac{n}{m_C}\rceil}.$$

We take $n_C = O(\log n)$ to be large enough such that the probability is at most $e^{-\Omega(k/m_C)} \le e^{-\Omega(k/\log n)}$.

Let $\mathcal{T} = \{T_1, \dots, T_{n_g}\}$, where $T_i = n_s n_C + \{n_C(i-1)+1, \dots, n_C i\}, i = 1, \dots, n_g$. Let Y_i be the indicator such that $Y_i = 1$ is the event $|\pi(T_i) \cap W| > k_C$. Let $Y = \sum_{i\in[n_g]} Y_i$. Note that $\mathbb{E}[|\pi(T_i) \cap W|] \le 0.9\delta_1 n_C = 0.9k_C$. Let E_g be the event that $Y \le \frac{l}{m_C}$. Since $\bar{n}$ is large enough, we can have $n_C n_g \le \frac{0.9\times 0.1}{1+0.9\times 0.1}|W|$ when $m' = \Omega(n)$. By Lemma 3.8,

$$1 - \Pr[E_g] \le e^{-2l/m_C + (e^2-1)e^{-\Omega(0.9\delta_1 n_C)} n_g}.$$

We take $n_C = O(\log n)$ to be large enough such that the probability is at most $e^{-\Omega(l/m_C)} \le e^{-\Omega(l/\log n)}$.

Let E be the event that $(\bigcap_{i\in[\bar{n}]} E_i) \cap E_g$. By the union bound, $\Pr[E] \ge 1 - \bar{n}e^{-\Omega(k/\log n)} - e^{-\Omega(l/\log n)}$. We claim that $\mathsf{Share}'(x)_W|E$ and $\mathsf{Share}'(\sigma)_W|E$ have statistical distance at most $\bar{n}\epsilon$, where σ is an arbitrary string in $\Sigma^{m'}$. The reason is as follows.

We fix a permutation π for which E happens. Let $W_s = (\bigcup_{i\in[n_s]} S_i) \cap W$, $W_g = (\bigcup_{i\in[n_g]} T_i) \cap W$. Let R be the random variable which corresponds to the random choice of r.

We claim that $\mathsf{Share}'(x)_{W_g}$ is a deterministic function of at most l entries of y_g and some extra uniform random bits. As E_g happens, for those $i \in [n_g]$ with $|\pi(T_i) \cap W| \le k_C$, the shares indexed by $\pi(T_i) \cap W$ are independent of the secret by the privacy of $(\mathsf{Share}_C, \mathsf{Rec}_C)$. Note that they are also independent of other shares since the construction uses independent randomness for sharing $\hat{y}_{g,i}, i \in [n_g]$ and $\hat{y}_{s,i}, i \in [n_s]$. For those $i \in [n_g]$ with $|\pi(T_i) \cap W| > k_C$, the total number of them is at most $\frac{l}{m_C}$. So $\mathsf{Share}'(x)_{W_g}$ is computed from at most $\frac{l}{m_C} \times m_C = l$ entries of y_g and some extra uniform random bits.

As $g_\Gamma(\cdot)$ is an l-wise independent generator, the distribution of $\mathsf{Share}'(x)_{W_g}$ is independent of the secret. For any $v \in \mathsf{supp}(\mathsf{Share}'(x)_{W_g})$, $\mathsf{Share}'(x)_{W_s}|_{\mathsf{Share}'(x)_{W_g}=v}$ is a convex combination of $\mathsf{Share}'(x)_{W_s}|_{R=r}$ for some different r such that $\mathsf{Share}'(x)_{W_g} = v$ happens.

We claim that $\mathsf{Share}'(x)_{W_s}|_{R=r}$ is a deterministic function of at most k entries of each $y_{s,i}$ for $i \in [\bar{n}]$ and some extra uniform random bits. This is because, as E happens, for those $i \in [n_s]$ with $|\pi(S_i) \cap W| \leq k_C$, the shares in $\pi(S_i) \cap W$ are independent of the secret by the privacy of $(\mathsf{Share}_C, \mathsf{Rec}_C)$. Note that they are also independent of other shares since the construction uses independent randomness for sharing $\hat{y}_{g,i}, i \in [n_g]$ and $\hat{y}_{s,i}, i \in [n_s]$. For those $i \in [n_s]$ with $|\pi(S_i) \cap W| > k_C$, the total number of them is at most $\frac{k}{m_C}$. So $\mathsf{Share}'(x)_{W_s}|_{R=r}$ is computed from at most $\frac{k}{m_C} \times m_C = k$ entries of each $y_{s,i}$ for $i \in [\bar{n}]$ and some extra uniform random bits.

Since the privacy error of $(\mathsf{Share}, \mathsf{Rec})$ is ϵ and $y_{s,i}|_{R=r}, i \in [\bar{n}]$ are computed using independent uniform random bits, for any $r, r' \in (\Sigma_0^m)^{\bar{n}}$,

$$\mathsf{SD}(\mathsf{Share}'(x)_{W_s}|_{R=r}, \mathsf{Share}'(\sigma)_{W_s}|_{R=r'}) \leq \bar{n}\epsilon.$$

So

$$\mathsf{SD}(\mathsf{Share}'(x)_{W_s}|_{\mathsf{Share}'(x)_{W_g}=v}, \mathsf{Share}'(\sigma)_{W_s}|_{\mathsf{Share}'(\sigma)_{W_g}=v}) \leq \bar{n}\epsilon.$$

As a result,

$$\mathsf{SD}(\mathsf{Share}'(x), \mathsf{Share}'(\sigma)) \leq \bar{n}\epsilon.$$

Thus with probability at least $1 - \bar{n}e^{-\Omega(k/\log n)} - e^{-\Omega(l/\log n)}$ over the fixing of π, $\mathsf{Share}'(x)_W$ and $\mathsf{Share}'(\sigma)_W$ have statistical distance at most $\bar{n}\epsilon$, which means that

$$\mathsf{SD}(\mathsf{Share}'(x), \mathsf{Share}'(\sigma)_W) \leq \bar{n}(\epsilon + e^{-\Omega(k/\log n)}) + e^{-\Omega(l/\log n)}.$$

□

Lemma 5.5. *If the non-adaptive privacy error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* ϵ*, then the adaptive privacy error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\bar{n}(\epsilon + e^{-\Omega(k/\log n)}) + e^{-\Omega(l/\log n)}$.

Proof. It follows immediately from Lemma 3.10 and 5.4.

□

Theorem 5.6. *For any* $\epsilon, \eta \in [0,1]$*, any* $n, m \in \mathbb{N}, m \leq n$ *and any constant* $a > 1, \gamma \in (0,1]$*, if there exists an explicit* (n,k) *secret sharing scheme in* AC^0 *with share alphabet* Σ*, message alphabet* Σ_0*, message length* m*, non-adaptive privacy error* ϵ *and reconstruction error* η*, then there exists an explicit* $(n' = O(n^a), \Omega(n'))$ *secret sharing scheme in* AC^0 *with share alphabet* $\Sigma \times [n']$*, message alphabet* Σ *message length* $\Omega(n')$*, adaptive privacy error* $O(n^{a-1}(\epsilon + e^{-\Omega(k/\log n)}) + e^{-\Omega(l/\log n)})$ *and reconstruction error* $O(n^{a-1}\eta)$ *where* $l = \Omega(\frac{mn^{a-1}\log|\Sigma_0|}{\log|\Sigma|})^{1-\gamma}$.

Proof. It follows from Construction 5.1, Lemma 5.2, 5.3, 5.5.

□

In step 5 of Construction 5.1, if we instead using xor based secret sharing scheme (Theorem 2.3) then we can get a even larger privacy threshold, but shorter message length. The proof is similar.

Theorem 5.7. *For any $n, m \in \mathbb{N}, m \leq n$, any $\epsilon, \eta \in [0,1]$ and any constant $a > 1, \gamma \in (0,1]$, if there exists an explicit (n,k) secret sharing scheme in AC^0 with share alphabet Σ, message alphabet Σ_0, message length m, non-adaptive privacy error ϵ and reconstruction error η, then there exists an explicit $(n' = O(n^a \log n), (1-\alpha)n')$ secret sharing scheme in AC^0 with share alphabet $\Sigma \times [n']$, message alphabet Σ, message length $\Omega(n^a)$, adaptive privacy error $O(n^{a-1}(\epsilon + 2^{-\Omega(k)}) + 2^{-\Omega(l)})$ and reconstruction error $n^{a-1}\eta$, where $l = \Omega(\frac{mn^{a-1}\log|\Sigma_0|}{\log|\Sigma|})^{1-\gamma}$.*

5.2 Binary alphabet

Our construction can be modified to have binary alphabet while keeping the message rate and privacy rate to be constant. We again use the tiny secret sharing schemes from asymptotically good codes as in Section 3.

Construction 5.8. *For any $n, k, m \in \mathbb{N}$ with $k, m \leq n$, let $(\mathsf{Share}, \mathsf{Rec})$ be an (n,k) secret sharing scheme with alphabet $\{0,1\}$, message length m.*

Let $(\mathsf{Share}_C, \mathsf{Rec}_C)$ be an (n_C, k_C) secret sharing scheme from Lemma 3.13 with alphabet $\{0,1\}^{p=O(\log n)}$, message length m_C, where $m_C = \delta_0 n_C$, $k_C = \delta_1 n_C$, $n_C = O(\log n)$ for some constants δ_0, δ_1.

Let $(\mathsf{Share}^*_C, \mathsf{Rec}^*_C)$ be an (n^*_C, k^*_C) secret sharing scheme from Lemma 3.13 with alphabet $\{0,1\}$, message length large enough m^*_C, where $m^*_C = \delta_0 n^*_C = p + O(\log n)$, $n^*_C = \delta_1 n^*_C$.

For any constant $a > 1$, $\gamma > 0$, we can construct the following $(n' = O(n^a), k' = \Omega(n'))$ secret sharing scheme $(\mathsf{Share}', \mathsf{Rec}')$ with alphabet $\{0,1\}$, message length $m' = \Omega(n')$.

The function $\mathsf{Share}' : \{0,1\}^{m'} \to \{0,1\}^{n'}$ is as follows.

1. *Let $\bar{n} = \Theta(n^{a-1})$ where the constant factor is large enough.*
2. *Let $g_\Gamma : \{0,1\}^{m\bar{n}} \to \{0,1\}^{m'}$ be the l-wise independent generator by Theorem 4.2, where $l = \Omega(mn^{a-1})^{1-\gamma}$.*
3. *For secret $x \in \{0,1\}^{m'}$, we draw a string $r = (r_1, \ldots, r_{\bar{n}})$ uniform randomly from $(\{0,1\}^m)^{\bar{n}}$.*
4. *Let $y = (y_s, y_g)$, where $y_s = (y_{s,1}, \ldots, y_{s,\bar{n}}) = (\mathsf{Share}(r_1), \ldots, \mathsf{Share}(r_{\bar{n}})) \in (\{0,1\}^n)^{\bar{n}}$ and $y_g = (y_{g,1}, \ldots, y_{g,m'}) = g_\Gamma(r) \oplus x \in \{0,1\}^{m'}$.*
5. *Compute $\hat{y}_s \in ((\{0,1\}^p)^{m_C})^{n_s}$ from y_s by parsing $y_{s,i}$ to be blocks over $(\{0,1\}^p)^{m_C}$ for every $i \in [\bar{n}]$, where $n_s = \lceil \frac{n}{pm_C} \rceil \bar{n}$.*
6. *Compute $\hat{y}_g \in ((\{0,1\}^p)^{m_C})^{n_g}$ from y_g by parsing y_g to be blocks over $(\{0,1\}^p)^{m_C}$, where $n_g = \lceil \frac{m'}{pm_C} \rceil$.*
7. *Let*
$$y' = (\mathsf{Share}_C(\hat{y}_{s,1}), \ldots, \mathsf{Share}_C(\hat{y}_{s,n_s}), \mathsf{Share}_C(\hat{y}_{g,1}), \ldots, \mathsf{Share}_C(\hat{y}_{g,n_g})).$$
Parse y' as $(y'_1, \ldots, y'_{n^*}) \in (\{0,1\}^p)^{n^*}$, where $n^* = (n_s + n_g)n_C$.
8. *Generate a random permutation $\pi : [n^*] \to [n^*]$ and compute*
$$z(x) = (\mathsf{Share}^*_C(y'_{\pi^{-1}(1)} \circ \pi^{-1}(1)), \ldots, \mathsf{Share}^*_C(y'_{\pi^{-1}(n^*)} \circ \pi^{-1}(n^*))) \in (\{0,1\}^{n^*_C})^{n^*}.$$
9. *Parse $z(x)$ to be bits and output.*

The function $\mathsf{Rec}' : \{0,1\}^{n'} \to \{0,1\}^{m'}$ is as follows.

1. *Parse the input bits to be $z = (z_1, \ldots, z_{n^*}) \in (\{0,1\}^{n^*_C})^{n^*}$.*

2. *For every $i \in [n^*]$, let $(y'_{\pi^{-1}(i)} \circ \pi^{-1}(i)) = \mathsf{Rec}^*_C(z_i)$ to get y'.*

3. *Parse $y' = (y'_s, y'_g)$ where $y'_s = (y'_{s,1}, \dots, y'_{s,n_s}) \in (\{0,1\}^{pn_C})^{n_s}$, $y'_g = (y'_{g,1}, \dots, y'_{g,n_g}) \in (\{0,1\}^{pn_C})^{n_g}$.*

4. *Let*
$$\hat{y}_s = (\mathsf{Rec}_C(y'_{s,1}), \dots, \mathsf{Rec}_C(y'_{s,n_s})), \hat{y}_g = (\mathsf{Rec}_C(y'_{g,1}), \dots, \mathsf{Rec}_C(y'_{g,n_g})).$$

5. *Parse $\hat{y}_s$ to get y_s.*

6. *Parse $\hat{y}_g$ to get y_g*

7. *Let $r = (\mathsf{Rec}(y_{s,1}), \dots, \mathsf{Rec}(y_{s,\bar{n}}))$.*

8. *Output*
$$\mathsf{Rec}'(z) = g_\Gamma(r) \oplus y_g.$$

Lemma 5.9. *If* $(\mathsf{Share}, \mathsf{Rec})$ *can be computed by* AC^0 *circuits, then* $(\mathsf{Share}', \mathsf{Rec}')$ *can be computed by* AC^0 *circuits.*

Proof Sketch. The construction is similar to that of Construction 5.1. As $n_C = O(\log n), n'_C = O(\log n)$, $n^*_C = O(\log n)$, we know that $(\mathsf{Share}_C, \mathsf{Rec}_C)$ and $\mathsf{Share}^*_C, \mathsf{Rec}^*_C$ can be computed by AC^0 circuits by Lemma 2.9 part 2.

So the overall construction can be computed by AC^0 circuits.

□

Lemma 5.10. *If the reconstruction error of* $(\mathsf{Share}, \mathsf{Rec})$ *is η, then the reconstruction error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is $\eta' = \bar{n}\eta$.*

Proof. As $(\mathsf{Share}_C, \mathsf{Rec}_C)$ and $(\mathsf{Share}^*_C, \mathsf{Rec}^*_C)$ have perfect reconstructions, the error only occurs when we apply Rec. So we can compute each $r_i, i = 1, \dots, \bar{n}$ correctly except with error η. By the union bound, with probability $1 - \bar{n}\eta$, r is correctly computed. Once we can compute r correctly, the secret $x = g_\Gamma(r) \oplus y_g$. It remains to show the correctness of y_g. From z we can get the value of every entry of y', since for every $i \in [n^*]$, $(y'_{\pi^{-1}(i)} \circ \pi^{-1}(i)) = \mathsf{Rec}^*_C(z_i)$ and we can get y' by putting each value into its correct position. Thus y'_g is correct as it is part of y'. By noting that Rec_C has no reconstruction error, $\hat{y}_g$ is correct. As y_g is from $\hat{y}_g$ by parsing, it is also correct.

□

Lemma 5.11. *If the non-adaptive privacy error of* $(\mathsf{Share}, \mathsf{Rec})$ *is ϵ, then the non-adaptive privacy error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}) + e^{-\Omega(l/\log^2 n)}$.

Proof Sketch. Let $k' = 0.9\delta_1^2 n'$. We need to show that there exists a distribution $\mathcal{D}$ such that for any $W \subseteq [n']$ with $|W| \leq k'$, for every $x \in \{0,1\}^{m'}$,

$$\mathsf{SD}(\mathsf{Share}'(x)_W, \mathcal{D}) \leq \bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}) + e^{-\Omega(l/\log^2 n)}.$$

Let $\mathcal{D}$ be $\mathsf{Share}'(\sigma)_W$ for an arbitrary $\sigma \in \{0,1\}^{m'}$.

Consider an arbitrary observation $W \subseteq [n']$ with $|W| \leq k'$. For at least $1 - 0.9\delta_1$ fraction of the blocks $z_i, i = 1, \dots, n^*$, at most δ_1 fraction of the bits in each block can be observed, because otherwise the number of observed shares is more than $0.9\delta_1 \times \delta_1 n' = 0.9\delta_1^2 n'$. Let W^* be the index sequence of those blocks which have more than δ_1 fraction of their bits observed. Let $|W^*| = k^*$ which is at most $0.9\delta_1 n^*$.

Consider every $i \notin W^*$. The distribution of z_i is independent of $y'_{\pi^{-1}(i)} \circ \pi^{-1}(i)$ by the privacy of $(\mathsf{Share}^*_C, \mathsf{Rec}^*_C)$. Since every Share_0 function uses independent randomness, z_i is also independent of $z_{i'}$ for every $i' \in [n^*]$ with $i' \neq i$. So we only have to show that

$$\mathsf{SD}(z_{W^*}(x), z_{W^*}(\sigma)) \leq \bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}) + e^{-\Omega(l/\log^2 n)}.$$

For every $i \in [n_s]$, let $S_i = \{n_C(i-1)+1, \ldots, n_C i\}$ and X_i be the boolean random variable such that $X_i = 1$ is the event $|\pi(S_i) \cap W^*| > k_C$. Let $\mathcal{S} = \{S_1, \ldots, S_{n_s}\}$. Note that $\forall i \in [n_s], \mathbb{E}[|\pi(S_i) \cap W^*|] \leq 0.9\delta_1 n_C = 0.9k_C$.

For every $i \in [\bar{n}]$, let E_i be the event that $\sum_{j=(i-1)\lceil \frac{n}{pm_C} \rceil + 1}^{i\lceil \frac{n}{pm_C} \rceil} X_j \leq \frac{k}{pm_C}$. We take $\bar{n}$ to be large enough such that $n_C \lceil \frac{n}{pm_C} \rceil \leq \frac{0.9 \times 0.1}{1 + 0.9 \times 0.1} |W^*|$. By Lemma 3.8, for every $i \in [\bar{n}]$.

$$1 - \Pr[E_i] \leq e^{-2k/(pm_C) + (e^2-1)e^{-\Omega(0.9\delta_1^2 n_C)} \lceil \frac{n}{pm_C} \rceil}.$$

We take $n_C = O(\log n)$ to be large enough such that the probability is at most $e^{-\Omega(k/(pm_C))} \leq e^{-\Omega(k/\log^2 n)}$.

For every $i \in [n_g]$, $T_i = \{n'_C(i-1)+1, \ldots, n_C i\}$ and Y_i be the event that $|\pi(T_i) \cap W^*| > k_C$. Let $\mathcal{T} = \{T_1, \ldots, T_{n_g}\}$. Let $Y = \sum_{i \in [n_g]} Y_i$. Note that $\mathbb{E}[|\pi(T_i) \cap W^*|] \leq 0.9\delta_1 n_C = 0.9k_C$. Let E_g be the event such that $Y \leq \frac{l}{pm_C}$. Since $\bar{n}$ is large enough, we have $n_C n_g \leq \frac{0.9 \times 0.1}{1 + 0.9 \times 0.1} |W^*|$. By Lemma 3.8,

$$1 - \Pr[E_g] \leq e^{-2l/(pm_C) + (e^2-1)e^{-\Omega(0.9\delta_1^2 n_C)} n_g}.$$

We take $n_C = O(\log n)$ to be large enough such that the probability is at most $e^{-\Omega(l/(pm_C))} \leq e^{-\Omega(l/\log^2 n)}$.

Let E be the event that $(\bigcap_{i \in [\bar{n}]} E_i) \cap E_g$. By the union bound, $\Pr[E] \geq 1 - \bar{n}e^{-\Omega(k/(\log^2 n))} - e^{-\Omega(l/(\log^2 n))}$. We claim that $z_{W^*}(x)|E$ and $z_{W^*}(\sigma)|E$ have statistical distance at most $\bar{n}\epsilon$. The reason is as follows.

We fix a permutation π for which E happens. Let $W_s = (\bigcup_{i \in [n_s]} S_i) \cap W^*$, $W_g = (\bigcup_{i \in [n_g]} T_i) \cap W^*$. Let R be the random variable which corresponds to the random choice of r.

We claim that $z_{W_g}(x)$ is a deterministic function of at most l entries of y_g and some extra uniform random bits. As E_g happens, for those $i \in [n_g]$ with $|\pi(T_i) \cap W^*| \leq k_C$, the blocks indexed by $\pi(T_i) \cap W^*$ are independent of the secret by the privacy of $(\mathsf{Share}_C, \mathsf{Rec}_C)$. Note that they are also independent of other blocks since the construction uses independent randomness for sharing $\hat{y}_{g,i}, i \in [n_g]$ and $\hat{y}_{s,i}, i \in [n_s]$. For those $i \in [n_g]$ with $|\pi(T_i) \cap W^*| > k_C$, the total number of them is at most $\frac{l}{pm_C}$. So $z_{W_g}(x)$ is computed from at most $\frac{l}{pm_C} \times pm_C = l$ entries of y_g and some extra uniform random bits.

As $g_\Gamma(\cdot)$ is an l-wise independent generator, the distribution of $z_{W_g}(x)$ is independent of the secret. For any $v \in \mathsf{supp}(z_{W_g}(x))$, consider $z_{W_s}(x)|_{z_{W_g}(x)=v}$ and $z_{W_s}(\sigma)|_{z_{W_g}(\sigma)=v}$. Note that $z_{W_s}(x)|_{z_{W_g}(x)=v}$ is a convex combination of $z_{W_s}(x)|_{R=r}$ for some different r such that $z_{W_g}(x) = v$ happens.

We claim that $z_{W_s}(x)|_{R=r}$ is a deterministic function of at most k entries of each $y_{s,i}$ for $i \in [\bar{n}]$ and some extra uniform random bits. This is because, as E happens, for those $i \in [n_s]$ with $|\pi(S_i) \cap W^*| \leq k_C$, the shares in $\pi(S_i) \cap W^*$ are independent of the secret by the privacy of $(\mathsf{Share}_C, \mathsf{Rec}_C)$. Note that they are also independent of other shares since the construction uses independent randomness for sharing $\hat{y}_{g,i}, i \in [n_g]$ and $\hat{y}_{s,i}, i \in [n_s]$. For those $i \in [n_s]$ with $|\pi(S_i) \cap W^*| > k_C$, the total number of them is at most $\frac{k}{pm_C}$. So $z_{W_s}(x)|_{R=r}$ is computed from at most $\frac{k}{pm_C} \times pm_C = k$ entries of each $y_{s,i}$ for $i \in [\bar{n}]$ and some extra uniform random bits.

Since the privacy error of $(\mathsf{Share}, \mathsf{Rec})$ is ϵ and every Share function uses independent uniform random bits, for any $r, r' \in (\Sigma_0^m)^{\bar{n}}$,

$$\mathsf{SD}(z_{W_s}(x)|_{R=r}, z_{W_s}(\sigma)|_{R=r}) \leq \bar{n}\epsilon.$$

So

$$\mathsf{SD}(z_{W_s}(x)|_{z_{W_g}(x)=v}, z_{W_s}(\sigma)|_{z_{W_g}(\sigma)=v}) \leq \bar{n}\epsilon.$$

As a result,

$$\mathsf{SD}(z_{W_s}(x), z_{W_s}(\sigma)) \leq \bar{n}\epsilon.$$

Thus with probability at least $1 - \bar{n}e^{-\Omega(k/(\log^2 n))} - e^{-\Omega(l/(\log^2 n))}$ over the fixing of π, $z_{W^*}(x)$ and $z_{W^*}(\sigma)$ have statistical distance at most $\bar{n}\epsilon$, which means that

$$\mathsf{SD}(z_{W^*}(x), z_{W^*}(\sigma)) \leq \bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}) + e^{-\Omega(l/\log^2 n)}.$$

□

Using Lemma 5.11 and a similar argument as in Lemma 3.19, we can get adaptive privacy as follows.

Lemma 5.12. *If the non-adaptive privacy error of* $(\mathsf{Share}, \mathsf{Rec})$ *is* ϵ*, then the adaptive privacy error of* $(\mathsf{Share}', \mathsf{Rec}')$ *is* $\bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}) + e^{-\Omega(l/\log^2 n)}$.

Proof. Let W be the adaptive observation of length k'. Let $W' = [\lceil k'/k_C^* \rceil]$. By Lemma 3.18, there exists a deterministic function f such that for $x, x' \in \{0,1\}^{m'}$, $\mathsf{SD}(\mathsf{Share}'(x)_W, \mathsf{Share}(x')_W) = \mathsf{SD}(f(\mathsf{Share}'(x)_{W'} \circ R \circ S), f(\mathsf{Share}'(x')_{W'} \circ R \circ S))$ where R, S are as defined in Lemma 3.18 which are independent of $\mathsf{Share}'(x)_{W'}$ and $\mathsf{Share}'(x')_{W'}$. By Lemma 2.2, we know that

$$\mathsf{SD}(f(\mathsf{Share}'(x)_{W'} \circ R \circ S), f(\mathsf{Share}'(x')_{W'} \circ R \circ S)) \leq \mathsf{SD}(\mathsf{Share}'(x)_{W'}, \mathsf{Share}'(x')_{W'}).$$

By Lemma 5.11 we know that

$$\mathsf{SD}(\mathsf{Share}'(x)_{W'}, \mathsf{Share}'(x')_{W'}) \leq \bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}) + e^{-\Omega(l/\log^2 n)}.$$

So

$$\mathsf{SD}(\mathsf{Share}'(x)_W, \mathsf{Share}'(x')_W) \leq \bar{n}(\epsilon + e^{-\Omega(k/\log^2 n)}) + e^{-\Omega(l/\log^2 n)}.$$

□

Theorem 5.13. *For any* $\epsilon, \eta \in [0,1]$*, any* $n, m \in \mathbb{N}, m \leq n$ *and any constant* $a > 1, \gamma > 0$*, if there exists an explicit* (n,k) *secret sharing scheme in* AC^0 *with alphabet* $\{0,1\}$*, message length* m*, non-adaptive privacy error* ϵ *and reconstruction error* η*, then there exists an explicit* $(n' = O(n^a), \Omega(n'))$ *secret sharing scheme in* AC^0 *with alphabet* $\{0,1\}$*, message length* $\Omega(n')$*, adaptive privacy error* $O(n^{a-1}(\epsilon + 2^{-\Omega(k/\log^2 n)}) + 2^{-\Omega((mn^{a-1})^{1-\gamma}/\log^2 n)})$ *and reconstruction error* $O(n^{a-1}\eta)$.

Proof. It follows from Construction 5.8, Lemma 5.9, 5.10, 5.12.

□

6 Instantiation

The Minsky-Papert function [29] gives a secret sharing scheme in AC^0 with perfect privacy.

Theorem 6.1 ([29])**.** *For any* $n \in \mathbb{N}$*, there exists an explicit* $(n, n^{\frac{1}{3}})$ *secret sharing scheme in* AC^0 *with alphabet* $\{0,1\}$*, message length* 1*, perfect privacy and reconstruction.*

Combining our techniques with Theorem 6.1, we have the following results.

Theorem 6.2. *For any $n \in \mathbb{N}$, any constant $\alpha \in (0,1], \beta \in [0,1)$, there exists an explicit $(n,(1-\alpha)n)$ secret sharing scheme in AC^0 with share alphabet $\{0,1\}^{O(\log n)}$, message alphabet $\{0,1\}$, message length $m = n^{\beta}$, adaptive privacy error $2^{-\Omega((\frac{n}{m\log n})^{1/3})}$ and perfect reconstruction.*

Proof. It follows from Theorem 6.1 and 3.12. We use the $(n_0, n_0^{1/3})$ secret sharing scheme from Theorem 6.1 to instantiate Theorem 3.12. So $n = O(n_0^a \log n_0)$ for some constant $a > 1$. The message length is $O(n/(n_0 \log n))$. Since $n_0 = \frac{n}{m\log n}$, the privacy error is $2^{-\Omega((\frac{n}{m\log n})^{1/3})}$. The share alphabet is $\{0,1\} \times [n]$. □

Note that when $\beta = 0$, this is a scheme sharing 1 bit. Next we give our theorem for secret sharing schemes with binary alphabet, constant secret rate and constant privacy rate.

Theorem 6.3. *For any $n \in \mathbb{N}$, for any constant $\gamma \in (0,1/4)$, there exists an explicit $(n,\Omega(n))$ secret sharing scheme in AC^0 with alphabet $\{0,1\}$, message length $m = \Omega(n)$, adaptive privacy error $2^{-\Omega(n^{\frac{1}{4}-\gamma})}$ and perfect reconstruction.*

Proof. It follows from Theorem 6.1 and 5.13.

Let $(n_0, n_0^{1/3})$ be the secret sharing scheme of Theorem 6.1 with message length $m_0 = 1$. Let $n = O(n_0^a)$ for some constant $a > 1$. For any constant $\beta \in (0,1)$, let $n_0^{1/3} = (m_0 n_0^{a-1})^{1-\beta}$. Then $a = \frac{4-3\beta}{3(1-\beta)}$. So $n_0 = O(n^{\frac{3(1-\beta)}{4-3\beta}})$. Hence by Theorem 5.13, we have the desired secret sharing scheme with the privacy error $2^{-\Omega(n^{\frac{1-\beta}{4-3\beta}}/\log^2 n)}$.

□

7 Extensions and other Applications

7.1 Robust secret sharing

Our secret sharing schemes can be made robust by using robust secret sharing schemes and authentication techniques in small blocks.

We will use cyclic shifting of indices in some constructions in this section. For any index i in some index set S, for any $j \in \mathbb{N}$, $i \gg j$ is the index obtained from i by doing right cyclic shifting for j positions, whereas $i \ll j$ is the index obtained from i by doing left cyclic shifting for j positions.

Theorem 7.1 ([14]). *For any $n \in \mathbb{N}$, any constant $\rho < 1/2$, there exists an $(n,\Omega(n))$ robust secret sharing scheme, with alphabet $\{0,1\}^{O(1)}$, message length $\Omega(n)$, perfect privacy, robustness parameter $d = \rho n$ and reconstruction error $2^{-\Omega(n)}$.*

Also we need the following theorem about computing approximate majorities.

Theorem 7.2 ([33]). *For every $n \in \mathbb{N}$, there exists an explicit depth 3 circuit $C_n : \{0,1\}^n \to \{0,1\}$ which decides whether the fraction of 1's in the input is at least 2/3.*

We use concatenations of the schemes from Theorem 7.1 to get the following robust secret sharing scheme in AC^0 with poly-logarithmic number of shares.

Lemma 7.3. *For any $n \in \mathbb{N}$, any constant $a \in \mathbb{N}$, any $\epsilon = 1/\mathsf{poly}(n)$, there exists an $(n_0 = O(\log^a n), k_0 = \Omega(n_0))$ robust secret sharing scheme in AC^0 (in n), with share alphabet $\{0,1\}^{O(1)}$, message alphabet $\{0,1\}$, message length $\Omega(n_0)$, perfect privacy, robustness parameter $\Omega(n_0)$, reconstruction error ϵ.*

Proof. Let $(\mathsf{RShare}_1, \mathsf{RRec}_1)$ be an $(n_1 = O(\log n), k_1 = \Omega(n_1))$ robust secret sharing scheme from Theorem 7.1 with message length m_1, robustness parameter d_1, share alphabet $\{0,1\}^{p=O(1)}$.

We use induction on a.

For $a = 1$, as the output length is $O(\log n)$, by Lemma 2.9, the sharing and reconstruction can both be done in AC^0. Other properties follow from Theorem 7.1.

Assume the conclusion holds for some $a \geq 1$. So there exists an $(n_a = O(\log^a n), k_a = \Omega(n_a))$ robust secret sharing scheme meeting the requirements, with message length m_a, message alphabet $\{0,1\}$, robustness parameter d_a, share alphabet $\{0,1\}^p$. Consider $a+1$. For any $x \in \{0,1\}^{O(\log^{a+1} n)}$, let RShare_{a+1} be constructed as follows. Split x into blocks $(\bar{x}_1, \ldots, \bar{x}_{m_1}) \in (\{0,1\}^{m_a})^{m_1}$. Let $\bar{y}_i = \mathsf{RShare}_1(\bar{x}_{1,i}, \ldots, \bar{x}_{m_1,i})$, $i = 1, \ldots, m_a$, where $\bar{x}_{j,i}$ is the ith bit of $\bar{x}_j$, $j = 1, \ldots, m_1$. Let $\tilde{x}_i = (\bar{y}_{1,i}, \ldots, \bar{y}_{m_a,i})$, $i = 1, \ldots, n_1$, where $\bar{y}_{j,i}$ is the ith bit of $\bar{y}_j$, $j = 1, \ldots, m_a$. Let $y_i = \mathsf{RShare}_a(\tilde{x}_i)$, $i = 1, \ldots, n_1$. Let $\mathsf{RShare}_{a+1}(x) = (y_1, \ldots, y_{n_1})$. The message length is $m_{a+1} = m_1 \cdot m_a$. The privacy is $k_{a+1} = k_1 \cdot k_a = \Omega(n_a)$. This is because, if the adversary can see at most $k_1 \cdot k_a$ shares, then at most k_1 of $\tilde{x}_1, \ldots, \tilde{x}_{n_1}$ are observed. So by the privacy of Share_1, $\bar{x}_1, \ldots, \bar{x}_{m_1}$ are not observed. Thus x is not observed. The robustness is $d_{a+1} = d_1 d_a = \Omega(n_a)$, due to a similar argument as for the privacy. RShare_{a+1} can be computed by AC^0 circuits since both RShare_1 and RShare_a can be computed by AC^0 circuits. The reconstruction is in AC^0 since we can first apply RRec_a on y_i for every $i = 1, \ldots, n_1$. By assumption this is in AC^0. Then we apply RRec_1 on $\bar{y}_i$ for every $i = 1, \ldots, m_a$. This is in AC^0 by the base case. The reconstruction error is still $1/\mathsf{poly}(n)$ since both $(\mathsf{RShare}_1, \mathsf{RRec}_1)$ and $(\mathsf{RShare}_a, \mathsf{RRec}_a)$ have reconstruction error $1/\mathsf{poly}(n)$.

□

Next, we give our construction of robust secret sharing scheme with "asymptotically good" parameters.

Theorem 7.4. *For any $n \in \mathbb{N}$, any $\eta = \frac{1}{\mathsf{poly}(n)}$, there exists an explicit $(n, \Omega(n))$ robust secret sharing scheme in AC^0 with share alphabet $\{0,1\}^{O(1)}$, message alphabet $\{0,1\}$, message length $m = \Omega(n)$, non-adaptive privacy error $2^{-n^{\Omega(1)}}$, non-adaptive robustness $\Omega(n)$ and reconstruction error η.*

Proof Sketch. We modify Construction 5.8. Let δ_0, δ_1, ρ be some proper constants in $(0,1)$.

Let $(\mathsf{RShare}_C, \mathsf{RRec}_C)$ be an (n_C, k_C) secret sharing scheme from Theorem 7.3 with share alphabet $\{0,1\}^{p=O(\log^2 n')}$, message alphabet $\{0,1\}$, message length m_C, where $m_C = \delta_0 n_C$, $k_C = \delta_1 n_C$, $n_C = O(\log n')$, robustness parameter ρn_C.

Also let $(\mathsf{RShare}^*_C, \mathsf{RRec}^*_C)$ be an (n^*_C, k^*_C) secret sharing scheme from Theorem 7.3 with share alphabet $\Sigma = \{0,1\}^{O(1)}$, message alphabet $\{0,1\}$, message length $m^*_C = p + O(\log^2 n)$, where $m^*_C = \delta_0 n^*_C$, $k^*_C = \delta_1 n^*_C$, robustness parameter ρn^*_C.

The robust secret sharing scheme construction is the same as that of Construction 5.1 except the following modifications. We replace $(\mathsf{Share}_C, \mathsf{Rec}_C)$ and $(\mathsf{Share}^*_C, \mathsf{Rec}^*_C)$ by their corresponding robust ones $(\mathsf{RShare}_C, \mathsf{RRec}_C)$ and $(\mathsf{RShare}^*_C, \mathsf{RRec}^*_C)$. For the share function, we replace the last two steps in Construction 5.8 by the following.

- Generate a random permutation $\pi : [n^*] \to [n^*]$.
- Randomly pick $l' = O(\log n^*)$ indices $r'_1, \ldots, r'_{l'} \in [n^*]$ and let $r' = (r'_1, \ldots, r'_{l'})$.
- For each block $y'_{\pi^{-1}(i)} \circ \pi^{-1}(i)$, $i = 1, \ldots, n^*$, we attach r' and $\pi^{-1}(i \gg r'_1), \pi^{-1}(i \gg r'_2), \ldots, \pi^{-1}(i \gg r'_{l'})$ to it. That is, the ith block is
$$\tilde{y}_i = y'_{\pi^{-1}(i)} \circ \pi^{-1}(i) \circ \pi^{-1}(i \gg r'_1) \circ \cdots \circ \pi^{-1}(i \gg r'_{l'}) \circ r'.$$
- Compute $z(x) = (\mathsf{RShare}^*_C(\tilde{y}_1), \ldots, \mathsf{RShare}^*_C(\tilde{y}_{n^*}))$.

- Parse z to be shares over $\Sigma^{n'=n^*_C n^*}$ and output.

For the reconstruction function we replace the first two steps with the following.

- Parse the input to be $z = (\mathsf{RShare}^*_C(\tilde{y}_1), \dots, \mathsf{RShare}^*_C(\tilde{y}_{n^*}))$, apply RRec^*_C on every entry to get $\tilde{y}$.
- Compute r' by taking the approximate majority of the r's in $\tilde{y}_i, i = 1, \dots, n^*$.
- For every $i \in [n^*]$, we do the following. Check that for every $j \in [l']$, the corresponding backup of $\pi^{-1}(i)$ in the $(i - r'_j)$th block is equal to the one stored in the ith block. Take the approximate majority of these l' tests, if the output is true then mark the $\tilde{y}_i$ as good, otherwise mark it as bad.
- Compute the entries of y' from shares that are marked as good. Other entries are left as blank.

We claim that, we get a $(n', \Omega(n'))$ robust secret sharing scheme with share alphabet $\{0,1\}^{O(1)}$, message alphabet $\{0,1\}$, message length $m' = \Omega(n')$, non-adaptive privacy error $2^{-n'^{\Omega(1)}}$, non-adaptive robust parameter $\Omega(n')$ and reconstruction error η.

The non-adaptive privacy can be proved in the same way as that of Lemma 5.11. Note that for each $i \in [n^*]$ we attach additional information about the indices. But this gives no more information about the secret since we are considering the non-adaptive case.

What we need to prove is that the reconstruction works under non-adaptive adversaries. Assume that the adversary corrupts at most $\min(0.9\rho^2, \rho/3)$ fraction of shares. Thus for at most $1/3$ fraction of $z_i, i \in [n^*]$, the fraction of corrupted shares is more than ρ. Because otherwise the total fraction of corrupted shares is more than $\rho/3$. Hence for at least $2/3$ fraction of $z_i, i \in [n^*]$, the fraction of corrupted shares is at most ρ. By the robustness of $(\mathsf{RShare}^*_C, \mathsf{RRec}^*_C)$, at least $2/3$ fraction of $\tilde{y}_i, i \in [n^*]$ can be reconstructed correctly. By Theorem 7.2, we can reconstruct r' correctly.

For any $i \in [n^*]$, we define that $\tilde{y}_i$ is marked correctly if $\tilde{y}_i$ is marked as good while the second entry $\pi^{-1}(i)$ is not corrupted or $\tilde{y}_i$ is marked as bad while the second entry $\pi^{-1}(i)$ is corrupted. For any $i \in [n^*]$, the event $\tilde{y}_i$ is marked correctly happens with probability $1 - e^{-\Theta(\log n')} = 1 - \mathsf{poly}(n')$ by a Chernoff bound, since the l' indices $r'_1, \dots, r'_{l'}$ are independently chosen and at most $\rho/3$ fraction of $\tilde{y}_1, \dots, \tilde{y}_{n^*}$ cannot be reconstructed correctly. By the union bound with probability $1 - \mathsf{poly}(n')$, $\forall i \in [n^*]$, $\tilde{y}_i$ is correctly marked. Note that when this happens, there is no collision of indices for $\tilde{y}_i, i = 1, \dots, n^*$ that are marked as good. Hence the algorithm can compute a large fraction of entries of y' correctly while other entries are left as blank. Before applying RRec_C on every y'_i, we bound the number of corrupted bits (including the blanks) for y'_i. The probability that the fraction of corrupted bits in y'_i is at most ρ is at least $1 - 1/\mathsf{poly}(n')$ by Lemma 3.7. Once for every block $y'_i, i = 1, \dots, n^*$, the corrupted rates are at most ρ , we can finally get the correct secret. By the union bound, we know the probability that this happens is at least $1 - 1/\mathsf{poly}(n')$.

The construction is still in AC^0. We only need to show that the modified parts can be computed in AC^0. For the share function, generating random permutation is by Lemma 3.1. Additions of indices can be computed by AC^0 circuits by Lemma 2.9 since the indices are recorded by $O(\log n')$ bits. Also note that RShare^*_C is from Lemma 7.3. For the reconstruction, note that RRec^*_C is from Lemma 7.3. The approximate majority is from Theorem 7.2. So they all can be computed by AC^0 circuits. The tests which check the equivalence of indices are in AC^0 by Lemma 2.9, as the input length is $O(\log n')$.

We still use Theorem 6.1 to instantiate the scheme.

□

7.2 Stochastic error correcting code

Using our general strategy, we can also construct stochastic error correcting codes in AC^0 which can resist additive errors ([20]).

One important component of our construction is the following "tiny" codes. It is constructed by classic code concatenation techniques.

Lemma 7.5. *For any $n \in \mathbb{N}$, any constant $a \in \mathbb{N}$, there exists an asymptotically good binary $(n_0 = O(\log^a n), m_0, d_0)$ code C such that the encoding and decoding can both be computed by* AC^0 *circuits of size* $\mathsf{poly}(n)$.

Proof. Let C_1 be an $(n_1 = O(\log n), m_1, d_1)$ binary code which is asymptotically good.

We use induction on a.

For $a = 1$, as the code length is $O(\log n)$ and there are plenty of asymptotically good binary codes construction, by Lemma 2.9, the encoding and decoding can both be done in AC^0. So our conclusion holds in this case.

Assume the conclusion holds for some $a \geq 1$. So there exists an asymptotically good binary $(n_a = O(\log^a n), m_a, d_a)$ code C_a. Consider $a+1$. For any $x \in \{0,1\}^{O(\log^{a+1} n)}$, let $C_{a+1}(x)$ be computed as the following codes concatenation. Parse x into blocks of length m_a, which is $(\bar{x}_1, \ldots, \bar{x}_{m_1})$. Let $\bar{y}_i = C_1(\bar{x}_{1,i}, \ldots, \bar{x}_{m_1,i})$, $i = 1, \ldots, m_a$, where $\bar{x}_{j,i}$ is the ith bit of $\bar{x}_j$. Let $\tilde{x}_i = (\bar{y}_{1,i}, \ldots, \bar{y}_{m_a,i})$, $i = 1, \ldots, n_1$, where $\bar{y}_{j,i}$ is the ith bit of $\bar{y}_j$, $j = 1, \ldots, m_a$. Let $y_i = C_a(\tilde{x}_i)$, $i = 1, \ldots, n_1$. Let $C_{a+1}(x) = (y_1, \ldots, y_{n_1})$. The message length is $m_{a+1} = m_1 \cdot m_a$. The distance is $d_{a+1} = d_1 \cdot d_a$. So C_{a+1} is still an asymptotically good code due to that a is a constant. The encoding can be computed by AC^0 circuits since the encoding of both C_1 and C_a can be computed by AC^0 circuits. The decoding is in AC^0 since we can first decode y_i for every $i = 1, \ldots, n_1$. By assumption this is in AC^0. Then we decode $\bar{y}_i$ for every $i = 1, \ldots, m_a$. This is in AC^0 by the base case.

□

Here we give the construction of stochastic error correcting codes in AC^0 which are "asymptotically good".

Construction 7.6. *For any $n \in \mathbb{N}$, we construct the following $(n, m = \Omega(n), \rho = \Omega(1))$ stochastic error correcting code.*

Let δ_0, δ_1 be some proper constants in $(0,1)$.

Let $(\mathsf{Enc}_0, \mathsf{Dec}_0)$ be an asymptotically good (n_0, m_0, d_0) error correcting code with alphabet $\{0,1\}^p$, $n_0 = O(\log n)$, $m_0 = \delta_0 n_0$, $d_0 = \delta_1 n_0$. In fact we can realize this code by applying an asymptotically good binary code, having the same rate, in parallel p times.

Let $(\mathsf{Enc}_1, \mathsf{Dec}_1)$ be an asymptotically good (n_1, m_1, d_1) error correcting code from Lemma 7.5 with alphabet $\{0,1\}$, $n_1 = p + O(\log n)$, $m_1 = \delta_0 n_1 = O(p)$, $d_1 = \delta_1 n_0$.

Encoding function $\mathsf{Enc} : \{0,1\}^{m=\Omega(n)} \to \{0,1\}^n$ is a random function which is as follows.

1. *On input $x \in \{0,1\}^m$, split x into blocks of length pm_0 such that $x = (\bar{x}_1, \ldots, \bar{x}_{m/(pm_0)}) \in (\{0,1\}^{pm_0})^{m/(pm_0)}$.*
2. *Let $y = (y_1, \ldots, y_{n'}) = (\mathsf{Enc}_0(\bar{x}_1), \ldots, \mathsf{Enc}_0(\bar{x}_{m/(pm_0)})) \in (\{0,1\}^p)^{n'}$, $n' = m/(\delta_0 p)$*
3. *Generate a random permutation $\pi : [n'] \to [n']$.*
4. *Randomly pick $l = O(\log n)$ different indices $r_1, \ldots, r_l \in [n']$ and let $r = (r_1, \ldots, r_l)$.*
5. *For every $i \in [n']$, let $\tilde{y}_i = (y_{\pi^{-1}(i)}, \pi^{-1}(i), \pi^{-1}(i \gg r_1), \ldots, \pi^{-1}(i \gg r_l), r)$.*
6. *Output $z = (\mathsf{Enc}_1(\tilde{y}_1), \ldots, \mathsf{Enc}_1(\tilde{y}_{n'})) \in (\{0,1\}^{n_1})^{n'}$.*

Decoding function $\mathsf{Dec} : \{0,1\}^{n=n_1 n'} \to \{0,1\}^m$ is as follows.

1. *On the input z, apply Dec_1 on every block of length n_0 to get $\tilde{y}$.*

2. *Take the majority of the r in every $\tilde{y}_i, i \in [n']$ to get r.*

3. *$\forall i \in [n']$, we do the following. Check that for every $j \in [l]$, the corresponding backup of $\pi^{-1}(i)$ in the $(i \lll r_j)$th block is equal to the one stored in the ith block. Take the approximate majority of these l tests, if the output is true then mark $\tilde{y}_i$ as good, otherwise mark it as bad.*

4. *Compute the entries of y from shares that are marked as good. Other entries are set as blank.*

5. *Apply Dec_0 on every block of y of length pn_0 to get x.*

Theorem 7.7. *For any $n \in \mathbb{N}$, any $\epsilon = 1/\mathsf{poly}(n)$, there exists an explicit $(n, m = \Omega(n), \rho = \Omega(1))$ stochastic binary error correcting code with decoding error ϵ, which can be computed by AC^0 circuits.*

The proof here is similar to the proof of Theorem 7.4.

Proof Sketch. We claim that Construction 7.6 gives such a code.

Let $\rho = 0.9(\delta_1/3)^2$. So for at most $0.9(\delta_1/3)$ fraction of blocks $z_1, \ldots, z'_n$, each of them has at least $(\delta_1/3)$ fraction of bits being corrupted, since otherwise the overall corrupted bits is larger than $0.9(\delta_1/3) \times (\delta_1/3)n_1n' = \rho n$. For blocks of $z_1, \ldots, z_n$ with less than $(\delta_1/3)$ fraction of bits being corrupted, they can be decoded correctly since $d_1/2 = \delta_1 n_1/2$. Thus we know that r can be reconstructed correctly through taking approximate majorities.

For any $i \in [n']$, define that $\tilde{y}_i$ is marked correctly if $\tilde{y}_i$ is marked as as good while the second entry $\pi^{-1}(i)$ is not corrupted or $\tilde{y}_i$ is marked as as bad while the second entry $\pi^{-1}(i)$ is corrupted. As $r_1, \ldots, r_l$ are randomly chosen and at most $0.9(\delta_1/3)$ fraction of $\tilde{y}_1, \ldots, \tilde{y}_{n'}$ can not be decoded correctly, by a Chernoff bound, for every $i \in n'$, the probability that $\tilde{y}_i$ is correctly marked can be at least $1 - 0.5\epsilon/n$ by setting l to be large enough. So by the union bound, the probability that for every $i \in [n']$, $\tilde{y}_i$ is marked correctly is at least $1 - 0.5\epsilon$. Once this event happens, the algorithm can get y with some blank entries, but there are no collision of indices.

Now we bound the number of corrupted bits (including the blanks) for every y_i. The probability that the fraction of corrupted bits in y_i is at most $\delta_1/3$ is at least $1 - 1/\mathsf{poly}(n)$ by Lemma 3.7. Once for every block $y_i, i = 1, \ldots, n'$, the corrupted rates are at most $\delta_1/3$, we can decode correctly. By the union bound, we know the probability that this happens is at least $1 - 1/\mathsf{poly}(n)$.

The Construction can be computed by AC^0 circuits since all components can be computed by AC^0 circuits.

□

Note that if we set both levels of codes in our construction to be from Lemma 7.5 with length $\mathsf{poly}\log n$ and l to be also $\mathsf{poly}\log n$, we can get quasi-polynomially small decoding error following the same proof. The result is stated as the follows.

Theorem 7.8. *For any $n \in \mathbb{N}$, any $\epsilon = 2^{-\mathsf{poly}\log n}$, there exists an explicit $(n, m = \Omega(n), \rho = \Omega(1))$ stochastic binary error correcting code with decoding error ϵ, which can be computed by AC^0 circuits.*

We can use duplicating techniques to make the decoding error to be even smaller, however with a smaller message rate.

Theorem 7.9. *For any $n, r \in \mathbb{N}$, there exists an $(n, m = \Omega(n/r), \rho = \Omega(1))$ stochastic binary error correcting code with decoding error $2^{-\Omega(r/\log n)}$, which can be computed by AC^0 circuits.*

Proof Sketch. For message $x \in \{0,1\}^m$, we simply repeat every bit for r times and then apply the coding scheme in Construction 7.6. When decoding, we apply the circuits from Theorem 7.2 to decide each x_i from r symbols. The error is $2^{-\Omega(r/\log n)}$ since we need to correctly reconstruct $\Omega(r/\log n)$ blocks to have the approximate majority being correct.

□

7.3 Secure message broadcasting

We give a secure protocol for the multi-party message broadcasting model against outside adversaries. The definition is given by Definition 1.11. Here we briefly describe the model again.

There are n parties where every party $i \in [n]$ has a private message $x_i \in \{0,1\}^m$. After communication, they want every party to know all private messages $x_i, i \in [n]$. Usually we assume the communication is conducted in a broadcast channel. The adversary can observe/corrupt some messages appeared in the channel but not all of them. A protocol for this model is secure in the sense that the adversary can learn almost nothing about the private messages and all parties can finally get all private messages.

The major difference between our model and the secure multi-party computation model is that, in our case, all parties are honest. The adversary can only observe/corrupt a constant fraction of messages.

The model is pretty practical in real world applications. For example, in military, we can think of several command-centres willing to exchange their information, while there are enemy radars or receivers which can detect their information frequently (but not always, since our army will find and attack them). Or several players want to have a common guess for the lottery but do not want anybody else occasionally passing by to know their guess.

To achieve our objective, we need to use the almost t-wise independent random permutation. A random variable $\pi : [n] \to [n]$ is an ϵ almost t-wise independent random permutation if for every t elements $i_1, \ldots, i_t \in [n]$, $(\pi(i_1), \ldots, \pi(i_t))$ has statistical distance at most ϵ from $(\pi'(i_1), \ldots, \pi'(i_t))$ where π' is a random permutation over $[n]$. Kaplan, Naor and Reingold [23] give a polynomial time construction generating ϵ almost t-wise independent permutations using $O(t \log n + \log(1/\epsilon))$ random bits.

Our protocol for secure multi-party message broadcasting is as follows. We assume that all parties share a small secret key. This assumption is reasonable since in our protocol the length of the shared secret key is significantly smaller than the message length.

Protocol 7.10. *For any $n, m \in \mathbb{N}$, for any $i \in [n]$, let $x_i \in \{0,1\}^m$ be the message of party i. Let the security parameter be $r \in \mathbb{N}$ with $r \le m$.*

Let $(\mathsf{RShare}_0, \mathsf{RRec}_0)$ be an $(n_0, k_0 = \delta_0 n_0)$ robust secret sharing scheme with share alphabet $\{0,1\}^{p=O(1)}$, message length $m_0 = m = \delta n_0$ and robust parameter $d_0 = \delta_1 n_0$, by Theorem 7.1 for some constant $\delta, \delta_0, \delta_1$ with $\delta_0 \ge \delta_1$.

Let $(\mathsf{RShare}_1, \mathsf{RRec}_1)$ be an $(n_1, k_1 = \delta_0 n_1)$ robust secret sharing scheme with share alphabet $\{0,1\}^{p=O(1)}$, message length $m_1 = pn_0/r = \delta n_1$ and robust parameter $d_1 = \delta_1 n_1$, by Theorem 7.1.

Assume that all parties have a common secret key $s \in \{0,1\}^{O(r\log(nr))}$.

The i-th party does the following.

1. *Generate a $2^{-\Omega(r)}$-almost r-wise independent random permutation π over $[nr]$ using s.*
2. *Compute the secret shares $y_i = \mathsf{RShare}_0(x_i) \in (\{0,1\}^p)^{n_0}$. Split y_i into r blocks each of length pn_0/r such that $y_i = (y_{i,1}, \ldots, y_{i,r})$.*
3. *View the communication procedure as having $[nr]$ time slots. For $j \in [r]$, on the $\pi((i-1)r + j)$'s time slot, send message $z_{i,j} = \mathsf{RShare}_1(y_{i,j})$.*

4. *For every $i \in [n]$, $j \in [r]$, compute $y_{i,j} = \mathsf{RRec}_1(z_{i,j})$, where $z_{i,j}$ is the message received in the $\pi((i-1)r+j)$'s time slot.*

5. *For every $i \in [n]$ get $y_i = (y_{i,1}, \ldots, y_{i,r})$.*

6. *For every $i \in [n]$, $x_i = \mathsf{RRec}_0(y_i)$.*

Theorem 7.11. *For any $n, m, r \in \mathbb{N}$ with $r \leq m$, there exists an explicit $(n, m, \alpha = \Omega(1), n2^{-\Omega(r)}, n2^{-\Omega(r)} + nm2^{-\Omega(m/r)})$ secure message broadcasting protocol with communication complexity $O(nm)$, secret key length $O(r \log(nr))$.*

Proof Sketch. Let $\alpha = 0.1\delta_1^2/p$. We first consider the non-adaptive adversary. For at least $1 - 0.1\delta_1$ fraction of nr blocks, the adversary can only observe/corrupt at most δ_1/p fraction of bits in one block. Because otherwise the total observed/corrupted fraction is more than $0.1\delta_1 \times \delta_1/p$.

For every block $z_{i,j}$ with at most δ_1/p fraction of bits being corrupted, $y_{i,j}$ is hidden and can be reconstructed correctly, since $(\mathsf{RShare}_1, \mathsf{RRec}_1)$ is a robust secret sharing scheme. Let $W \subseteq [nr]$ denote the set of indices of those blocks for which the adversary can tempt more than δ_1/p fraction of bits. So $\frac{|W|}{nr} \leq 0.1\delta_1$.

Let's first assume that π is a perfect random permutation.

Let $X_{i,j}$ be the indicator such that $X_{i,j} = 1$ is the event that $\pi((i-1)n+j) \in W$. Let $X_i = \sum_{j\in[r]} X_{i,j}$. Thus

$$\Pr[X_{i,j} = 1] \leq 0.1\delta_1.$$

By Lemma 3.7,

$$\Pr[X_i > \delta_1 r] \leq 2^{-\Omega(r)}.$$

As $(\mathsf{RShare}_0, \mathsf{RRec}_0)$ and $(\mathsf{RShare}_1, \mathsf{RRec}_1)$ are all robust, once $X_i \leq \delta_1 r$, x_i can be reconstructed correctly. Since $\delta_0 \geq \delta_1$, x_i is also hidden.

Now consider π being a $2^{-\Omega(r)}$-almost r-wise independent random permutation. Since X_i is a deterministic function of π, by Lemma 2.9 the statistical distance between X_i and that in the perfect random permutation case is $2^{-\Omega(r)}$. So

$$\Pr[X_i > \delta_1 r] \leq 2^{-\Omega(r)}.$$

By the union bound, the probability that $\forall i \in [n]$, x_i is hidden and can be reconstructed correctly, is at least $1 - n2^{-\Omega(r)}$.

Note that the adaptive privacy can be achieved from non-adaptive privacy by Lemma 3.10. Also note that we have adaptive robustness if we have adaptive privacy, since the tampering adversary is a function on the observed random variables.

The reconstruction error is from the two reconstruction functions RRec_0 and RRec_1. As we applied them for at most $O(nm)$ times and want them to always compute correctly, the error is at most $n2^{-\Omega(r)} + nm2^{-\Omega(m/r)}$.

The total number of bits in the transmission is $nm(p/\delta)^2 = O(nm)$.

□

Acknowledgements. This collaboration started at the DIMACS Workshop on Cryptography and its Interactions. The first and third authors were supported by NSF Grant CCF-1617713. The second author was supported by NSF-BSF grant 2015782, BSF grant 2012366, ISF grant 1709/14, ERC grant 742754, DARPA/ARL SAFEWARE award, NSF Frontier Award 1413955, NSF grants 1619348, 1228984, 1136174, and 1065276, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the DARPA through the ARL under Contract W911NF-15-C-0205. The views expressed are those of the authors and do not reflect the official policy or position of the DoD, the NSF, or the U.S. Government.

References

[1] Noga Alon, Jehoshua Bruck, Joseph Naor, Moni Naor, and Ron M Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on information theory*, 38(2):509–516, 1992.

[2] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc 0. *SIAM Journal on Computing*, 2006.

[3] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. On pseudorandom generators with linear stretch in nc 0. *Computational Complexity*, 17(1):38–69, 2008.

[4] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[5] Anne Auger and Benjamin Doerr. *Theory of randomized search heuristics: Foundations and recent developments*, volume 1. World Scientific, 2011.

[6] Allison Bishop, Valerio Pastro, Rajmohan Rajaraman, and Daniel Wichs. Essentially optimal robust secret sharing with maximal corruptions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 58–86. Springer, 2016.

[7] Andrej Bogdanov, Yuval Ishai, Emanuele Viola, and Christopher Williamson. Bounded indistinguishability and the complexity of recovering secrets. In *Annual Cryptology Conference*, pages 593–618. Springer, 2016.

[8] Andrej Bogdanov and Chin Ho Lee. Homomorphic evaluation requires depth. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 365–371, 2016.

[9] Andrej Bogdanov and Christopher Williamson. Approximate bounded indistinguishability. In *International Colloquium on Automata, Languages, and Programming*, 2017.

[10] Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 337–367, 2015.

[11] Mark Bun and Justin Thaler. A nearly optimal lower bound on the approximate degree of ac0. In *FOCS*, 2017.

[12] Hao Chen, Ronald Cramer, Shafi Goldwasser, Robbert De Haan, and Vinod Vaikuntanathan. Secure computation from random error correcting codes. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 291–310. Springer, 2007.

[13] Kuan Cheng and Xin Li. Randomness extraction in ac0 and with small locality. *arXiv preprint arXiv:1602.01530*, 2016.

[14] Mahdi Cheraghchi. Nearly optimal robust secret sharing. In *Information Theory (ISIT), 2016 IEEE International Symposium on*, pages 2509–2513. IEEE, 2016.

[15] Mary Cryan and Peter Bro Miltersen. On pseudorandom generators in nc0. In *Mathematical Foundations of Computer Science 2001*, pages 272–284. Springer, 2001.

[16] Ivan Damgård, Yuval Ishai, Mikkel Krøigaard, Jesper Buus Nielsen, and Adam Smith. Scalable multiparty computation with nearly optimal work and resilience. In *Annual International Cryptology Conference*, pages 241–261. Springer, 2008.

[17] Oded Goldreich. Three XOR-lemmas - an exposition. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(56), 1995.

[18] Oded Goldreich. Candidate one-way functions based on expander graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 76–87. Springer, 2011.

[19] Shafi Goldwasser, Dan Gutfreund, Alexander Healy, Tali Kaufman, and Guy N Rothblum. Verifying and decoding in constant depth. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 440–449. ACM, 2007.

[20] Venkatesan Guruswami and Adam Smith. Optimal rate code constructions for computationally simple channels. *Journal of the ACM (JACM)*, 63(4):35, 2016.

[21] Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM*, 56(4), 2009.

[22] Torben Hagerup. Fast parallel generation of random permutations. In *International Colloquium on Automata, Languages, and Programming*, pages 405–416. Springer, 1991.

[23] Eyal Kaplan, Moni Naor, and Omer Reingold. Derandomized constructions of k-wise (almost) independent permutations. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 354–365. Springer, 2005.

[24] Chin Ho Lee and Emanuele Viola. Some limitations of the sum of small-bias distributions. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 22, page 5. Citeseer, 2015.

[25] Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, pages 28–57, 2016.

[26] Shachar Lovett and Emanuele Viola. Bounded-depth circuits cannot sample good codes. In *Computational Complexity (CCC), 2011 IEEE 26th Annual Conference on*, pages 243–251. IEEE, 2011.

[27] Yossi Matias and Uzi Vishkin. Converting high probability into nearly-constant timewith applications to parallel hashing. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 307–316. ACM, 1991.

[28] Eric Miles and Emanuele Viola. Substitution-permutation networks, pseudorandom functions, and natural proofs. *J. ACM*, 62(6):46:1–46:29, 2015.

[29] Marvin Minsky and Seymour Papert. *Perceptrons*. MIT press, 1988.

[30] Elchanan Mossel, Amir Shpilka, and Luca Trevisan. On ε-biased generators in nc0. *Random Structures & Algorithms*, 29(1):56–81, 2006.

[31] Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2):336–375, 1999.

[32] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[33] Emanuele Viola. On approximate majority and probabilistic time. *Computational Complexity*, 18(3):337, 2009.

[34] Emanuele Viola. the complexity of distributions. *SIAM Journal on Computing*, 41:191–218, 2012.

A Secret sharing with small privacy error implies approximation degree lower bound

The following result is a direct corollary of Theorem D.1. and Theorem 1.2 of [7].

Theorem A.1 (Corollary from Theorem D.1. and Theorem 1.2 of [7])**.** *For any $n, k \in \mathbb{N}$, if there exists an (n, k) secret sharing scheme with alphabet $\{0, 1\}$, message length $m \geq 1$, privacy error $\epsilon = O(\frac{1}{n^k})$, reconstruction function f and reconstruction error being constant, then f has α-approximation degree at least k, where α is a constant in $(0, 1)$.*

Proof. Let the secret sharing scheme be (Share, f). For two different message $x, x' \in \{0, 1\}^m$, consider $Y = \mathsf{Share}(x)$ and $Y' = \mathsf{Share}(x')$. By definition of secret sharing scheme, there are no test on k bits which can distinguish them with advantage ϵ. By Theorem D.1. of [7], there exists Z, Z' over $\{0, 1\}^n$, where Z is $2\epsilon n^k$ close to Y and Z' is $2\epsilon n^k$ close to Y'. Also Z and Z' are k-wise indistinguishable. So once ϵ is small enough, f can still distinguish them with a constant advantage. Thus by Theorem 1.2 of [7], f has α-approximation degree at least k with α being a constant in $(0, 1)$.

□