The Dissertation Committee for Xin Li
certifies that this is the approved version of the following dissertation:

Distributed Computing and Cryptography with General Weak Random Sources

Committee:

David Zuckerman, Supervisor

Lorenzo Alvisi

Yael Kalai

Adam Klivans

Brent Waters

Distributed Computing and Cryptography with General Weak Random Sources

by

Xin Li, B.E.; M.E.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2011

Dedicated to my family.

Acknowledgments

First and foremost, I am grateful to my advisor, David Zuckerman, who has helped me during all stages of my PhD study. It was David who first suggested to me the problem of doing distributed computing with imperfect randomness, which has led to a significant part of this thesis. Apart from this, David has been a constant resource for providing me with great advice and useful feedback about my ideas. Without his encouragement and support, I would not have gone this far in my research.

I am also grateful to many other people who have helped me during my study at UT-Austin. Anup Rao taught me a lot about extractors for independent sources. Yael Kalai helped me with many ideas in cryptography and arranged an internship for me at Microsoft Research New England. Adam Klivans taught a very interesting class about computational learning theory. I would also like to thank my co-authors for their contributions in the research that this thesis is about.

Finally, I would like to thank the other professors and graduate students in the theory group for providing me with both technical and non-technical advice, and for making my life at UT-Austin much more enjoyable.

Distributed Computing and Cryptography with General Weak Random Sources

Xin Li, Ph.D.
The University of Texas at Austin, 2011

Supervisor: David Zuckerman

The use of randomness in computer science is ubiquitous and important. Randomized protocols have turned out to be much more efficient than their deterministic counterparts. In addition, many problems in distributed computing and cryptography are impossible to solve without randomness. However, these applications typically require uniform random bits, while in practice it is not clear how to obtain random sources with such high quality. In fact, almost all natural random phenomena are biased. Moreover, even originally uniform random bits can be damaged if an adversary learns some partial information about these bits.

In this thesis, we study how to run randomized protocols in distributed computing and cryptography with imperfect randomness. We model imperfect randomness by the most general model where the weak random source is only required to have a certain amount of min-entropy.

One important tool here is the *randomness extractor*. A randomness extractor is a function that takes as input one or more weak random sources, and outputs a distribution that is close to uniform in statistical distance. Randomness extractors are interesting in their own right and are closely related to many other problems in computer science. Giving efficient constructions of randomness extractors with optimal parameters is one of the major open problems in the area of pseudorandomness.

The main results of this thesis are:

Distributed Computing with Imperfect Randomness We construct *network extractor* protocols that extract private random bits for parties in a communication network, assuming that they each start with an independent weak random source, and some parties are corrupted by an adversary with unlimited computational power who sees all communication in the network. These protocols imply fault-tolerant distributed computing protocols where only imperfect randomness is available.

Cryptography with Imperfect Randomness This is the analogous problem in the computational setting. Here we construct *computational network extractor* protocols where the adversary is computationally bounded and the parties extract computationally private random bits. These protocols imply secure multi-party computation protocols with only imperfect randomness, under some computational assumptions.

Extractors for Independent Sources We study the problem of constructing efficient extractors for independent weak random sources. The probabilistic method shows that there exists an extractor for two independent sources with each having only logarithmic min-entropy, while known constructions are far from achieving these parameters. In this thesis we make progress on this problem. We construct extractors for two independent sources with each having any linear min-entropy, based on a computational assumption. We also construct the best known extractor for three independent sources.

Extractors for Affine Sources over F_2 Here the weak random source is modeled by the uniform distribution over some unknown affine subspace of a vector space. Again known constructions are far from achieving the parameters given by the probabilistic method. We also make progress on this problem and construct the best affine extractor to date.

Non-malleable Extractors and Privacy Amplification with an Active Adversary In this model, two parties share a private weak random source in the beginning, and they wish to achieve a shared uniform random string through communications in a channel controlled by an adversary. The adversary has unlimited computational power and can change the messages sent through the channel in arbitrary ways. All previous results assume that the two parties have access to local uniform random bits. We show that this problem can be solved even if the two parties only have access to local weak random sources. We also improve previous results in various ways by constructing the first explicit non-malleable extractor and giving protocols for privacy amplification based on this extractor.

Table of Contents

Acknowledgments **v**

Abstract **vi**

Chapter 1. Introduction **1**

1.1 Examples 2

1.1.1 Byzantine Agreement and Leader Election 2

1.1.2 Secure Multiparty Computation 3

1.1.3 Privacy Amplification with an Active Adversary 3

1.2 General Weak Random Sources and Extractors 4

1.3 Main Results 5

1.3.1 Distributed Computing and Cryptography with Weak Random Sources 5

1.3.2 Improved Constructions of Extractors 6

1.3.2.1 Extractors for Independent Sources 7

1.3.2.2 Extractors for Affine Sources 8

1.3.2.3 Non-malleable Extractors 8

1.3.3 Privacy Amplification with an Active Adversary 8

1.4 Organization of this Thesis 9

Chapter 2. Basic Definitions **10**

2.1 Statistical Distance 10

2.2 Convex Combinations 10

2.3 Min-Entropy and Sources of Randomness 11

2.3.1 Block Sources and Conditional entropy. 12

2.3.2 Somewhere Random Sources 17

2.3.3 Affine Sources 18

2.4 Cryptographic Definitions 20

Chapter 3. Building Blocks 22
3.1 Extractors, Dispersers, Mergers and Condensers . . . 22
3.1.1 Strong Linear Seeded Extractors . . . 23
3.1.2 Reconstructive Extractors . . . 24
3.2 Primes in Arithmetic Progressions . . . 26
3.3 Fourier Analysis . . . 27
3.4 A Non-Uniform XOR Lemma. . . . 28
3.5 Previous Works that We Use . . . 29

Chapter 4. Distributed Computing with General Weak Random Sources 35
4.1 Previous Results . . . 36
4.2 Our Results . . . 37
4.3 The Constructions . . . 39
4.3.1 Synchronous Network Extractors . . . 39
4.3.1.1 First Attempts . . . 39
4.3.1.2 Protocol for Low Entropy . . . 42
4.3.1.3 Protocol for High Entropy Rate and Block Sources . . . 46
4.3.1.4 Lower bounds . . . 49
4.3.1.5 Protocol for Even Smaller Min-Entropy . . . 50
4.3.1.6 High-Level Ideas of the synchronous extractor . . . 53
4.3.1.7 Synchronous network extractor protocol . . . 54
4.3.1.8 Proof of the theorem . . . 57
4.3.2 Asynchronous Network Extractors . . . 62
4.3.2.1 High-Level Ideas of the Asynchronous Extractor . . . 67
4.3.2.2 Asynchronous Network Extractor Protocol . . . 68
4.3.3 Proof of the Theorems . . . 72
4.4 Applications in Distributed Computing . . . 78
4.4.1 Collective Coin-Flipping and Leader Election . . . 78
4.4.2 Byzantine Agreement . . . 79

Chapter 5. Cryptography with General Weak Random Sources **82**
5.1 The Constructions . . . 85
5.1.1 Computational Network Extractors under Standard Assumptions . . . 85
5.1.1.1 High Level Ideas . . . 85
5.1.1.2 The Protocol . . . 88
5.1.1.3 The Analysis . . . 90
5.1.2 Computational Network Extractors under Stronger Assumptions . . . 98
5.1.2.1 Computational Network Extractors for Linear Min-Entropy . . . 98
5.1.2.2 Network Extractor for Polynomially Small Min-Entropy . . . 106

Chapter 6. Improved Constructions of Extractors **119**
6.1 A Two-Source Extractor under Computational Assumptions . . . 119
6.1.1 Overview of Our Construction . . . 119
6.1.2 The Formal Analysis . . . 122
6.2 Three Source Extractors . . . 130
6.2.1 Overview of the Construction . . . 130
6.2.2 The Construction . . . 134
6.2.3 Analysis of the Extractor . . . 136
6.3 Affine Extractors and Dispersers . . . 139
6.3.1 Overview of Our Constructions . . . 140
6.3.1.1 Extractors for one affine block source . . . 140
6.3.1.2 Obtain one output . . . 141
6.3.1.3 Extractors that are low-degree polynomials . . . 142
6.3.1.4 Affine Extractors . . . 143
6.3.1.5 A word about the affine extractor for entropy rate 1/2 . . . 144
6.3.2 The Constructions . . . 145
6.3.2.1 Low Degree Strong Linear Seeded Extractors . . . 145
6.3.2.2 Extractors for Affine Somewhere Random Sources with Few Rows . . . 147
6.3.2.3 Affine Dispersers for Linear Entropy Sources . . . 150
6.3.2.4 Affine Extractors for Linear Entropy Sources . . . 150
6.3.3 Analysis of the Affine Disperser . . . 151
6.3.4 Analysis of the Affine Extractor . . . 155

6.3.5 Affine Extractors and Dispersers for Sub-Linear Entropy Sources 158
6.4 Non-malleable Extractors 160
6.4.1 The Construction of Non-malleable Extractors 161
6.4.2 A Character Sum Estimate 164

Chapter 7. Privacy Amplification with an Active Adversary **167**
7.1 Some Previous Results that We Need 169
7.2 Privacy Amplification with Local Weak Random Sources 170
7.2.1 Overview of the Constructions 171
7.2.2 The Constructions and the Analysis 174
7.2.2.1 Non Constructive Results 174
7.2.2.2 Weak random sources with entropy rate $> 1/2$ 176
7.2.2.3 Weak random sources with linear min-entropy 177
7.3 Privacy Amplification with Local Uniform Random Bits 186
7.3.1 Overview of the Protocol for Privacy Amplification 186
7.3.2 The Construction and the Analysis 189
7.3.2.1 Security Proof of Our Protocol (Proof of Theorem 7.3.1) 194

Bibliography **204**

Chapter 1

Introduction

This thesis is about the study of using imperfect randomness in distributed computing and cryptography. The motivation comes from the following two aspects.

First, randomness is a very useful and valuable resource in computer science. It has found applications in algorithm design, distributed computing, cryptography and many other areas. These applications generally either lead to much more efficient solutions than their deterministic counterparts (e.g., in the algorithm case), or solve problems that would otherwise be impossible in the deterministic case (e.g., in distributed computing and cryptography). We refer the reader to [MR95] for many of these examples.

Unfortunately, these applications typically require many uniform random bits, while in practice it is not clear how to obtain such high quality resources. Thus it is natural to ask what is the minimum requirement of the randomness used in these applications. In some cases, for example algorithm design, a combinatorial object called a pseudorandom generator can be used to reduce the number of random bits required, or even completely remove the need of randomness and obtain a deterministic algorithm. However, in many other areas such as distributed computing and cryptography, true randomness is provably necessary, in the sense that there are tasks that cannot be done without randomness (for example asynchronous Byzantine Agreement and encryption). Given the fact that almost all physical sources of randomness are biased (e.g., coin flips, white noise), it is important to study how imperfect randomness can be used in these applications.

Second, imperfect randomness often arises in cryptography for another reason: under many situations, a party's secret random bits (even if they are uniform originally) can be compromised, for example as a result of side channel attacks. Thus again in this case we need to consider using imperfect randomness in applications without damaging the desired security properties. This problem is becoming more and more important nowadays because of the enormous amount of information exchanged through internet.

1.1 Examples

Here we give three basic problems in distributed computing and cryptography, which are the focus of this thesis. We will later show how to use imperfect randomness in these problems. We remark that the techniques developed in this thesis are very general and apply to many other problems. However these problems are very basic and important, so we choose them to illustrate the applications of our techniques.

1.1.1 Byzantine Agreement and Leader Election

The Byzantine Agreement problem is a fundamental problem in distributed computing. In this problem, p processors each have an initial value, and they communicate with each other in order to reach an agreement on a value b. Note that in this setting there are no broadcast channels and the processors have to transfer messages to each other in point to point channels. The task is made even more complicated by the existence of an adversary, who may control some of the processors and thus make them collaborate to try to prevent an admissible agreement. The processors, however, must try to reach an admissible agreement that satisfy the following two conditions.

1. All good processors (processors that are not controlled by the adversary) must agree on the value b.
2. If all the good processors have the same initial value, then b must be equal to that value.

Essentially, the task of Byzantine Agreement is to try to simulate a broadcast channel with point to point channels. In this thesis we assume that the adversary is computationally unbounded. Note that the network can be synchronous or asynchronous, thus the Byzantine Agreement problem also has the synchronous case and the asynchronous case. It is well known that both cases can be solved if the adversary controls less than $1/3$ fraction of the processors. In the synchronous case, there is a deterministic solution, but the randomized solution takes much fewer rounds. In the asynchronous case, there is no deterministic solution.

Once we have a broadcast channel, other problems in distributed computing arise. One basic problem is leader election, and the related collective coin flipping. Again, in both problems there are p processors, and some processors may be controlled by an adversary. The goal of leader election is to have the processors communicate with each other to try to select a leader, such that the probability that the leader is a faulty processor is bounded away from 1. The goal of collective coin flipping is to have the processors communicate with each other to provide a random coin flip, such that the probability that the coin takes head or tail is bounded away from 1. It is clear that in both these problems randomness is necessary. Also, note that the collective coin flipping problem can be reduced to the leader election problem, since the processors can first select a leader and then have the leader flip a coin.

1.1.2 Secure Multiparty Computation

This is a basic problem in cryptography. One simple example is the millionaire problem, where two millionaires wish to compute which one is richer, but without revealing their net worth. Generally, the setting is that, p processors each have private data, and they want to jointly compute a function of their data. However, the protocol to compute the function is required to have the property that no processor can learn more from the description of the function and the result of the computation than what he or she can learn from his or her own entry. Also, again there is an adversary who may control some processors, and the protocol must guarantee that the computation does not leak information about a good processor's data to the adversary. Essentially, the processors want to compute the function as well as maintaining the privacy of their own data.

This problem has several different models. In this thesis, we focus on the case where the adversary is computationally bounded (i.e., runs in polynomial time) and is passive. Under the assumption that each processor has access to local uniform random bits, this problem can be solved by using now standard cryptographic primitives [GMW87].

1.1.3 Privacy Amplification with an Active Adversary

This is an important problem in symmetric information-theoretic cryptography. Imagine that two parties (Alice and Bob) want to exchange information through a public channel. However, there is an adversary Eve who is watching the channel. Alice and Bob would like to exchange information secretly without leaking information to Eve. Generally, this task is impossible, since Eve knows everything exchanged through the channel. Luckily, in the case where Alice and Bob share a secret n-bit uniform random string w, the task becomes possible. For example, if Alice wants to send a string x to Bob, then she can send $y = x \oplus w$ instead. Without knowing w, Eve knows no information about x even seeing y. On the other hand, Bob can recover x by computing $x = y \oplus w$.

The above solution is nice, except that often w is not uniform. Thus the problem of privacy amplification arises. Basically, in this case Alice and Bob try to agree on a secret nearly uniform random key, through communications over the public channel. Note that again the channel is controlled by an adversary Eve. Eve could be either passive or active. She is passive if she can only see the communications through the channel without modifying the messages, and she is active if she can modify the messages in arbitrary ways. Of course the problem with an active adversary is much harder than the problem with a passive adversary. In this thesis, we assume that the adversary Eve is *active* with unlimited computational power.

1.2 General Weak Random Sources and Extractors

To formalize our discussion of imperfect randomness, we need a mathematical model. In this thesis, unless otherwise stated, we use a very general model: general weak random sources. A general weak random source is some arbitrary probability distribution, and the only constraint is that it contains a certain amount of entropy. Here we use the standard notion of *min-entropy*– a general weak random source is said to have min-entropy k if the probability of getting any particular string from the source is at most 2^{-k}. A weak random source on n bits with min-entropy k is called an (n, k)-source. In the past, the problem of using general weak random sources in algorithms has been studied extensively, and efficient solutions are provided even for weak random sources with very small min-entropy. The focus of this thesis, however, is the study of using general weak random sources in distributed computing and cryptography.

A very useful tool in dealing with weak random sources is the *extractor*. An extractor is an algorithm that takes as input one or more weak random sources, and outputs a distribution that is close to uniform in statistical distance. Given such an algorithm, we can first convert the weak random sources into distributions that are close to being uniform, and then use these distributions in standard applications that require uniform random bits.

Another reason for constructing explicit extractors is that these are functions that resemble many properties of *random* functions. For example, under certain conditions it can often be shown that a random function is a good extractor with high probability. Other objects that resemble the properties of random functions include for example expander graphs, error-correcting codes, hard functions and pseudo-random generators. In fact, these objects are related to each other and since they somehow look like random functions, they can often be used to *derandomize* applications that require many uniform random bits. The problem of derandomization is also related to some central open questions in complexity theory. For example, giving a certain kind of explicit hard function (again, a random function is such a hard function with high probability, but is not explicit) would imply that $BPP = P$ [NW94, IW97]. Therefore, finding explicit constructions of extractors would help us gain insights into the larger area of derandomization. In the past the problem of constructing certain kinds of efficient extractors has been studied extensively. In this thesis we also make improvements over previous results on extractor constructions.

1.3 Main Results

We summarize our main results in the following three sections.

1.3.1 Distributed Computing and Cryptography with Weak Random Sources

The problem of doing distributed computing with imperfect randomness was first studied by Goldwasser, Sudan, and Vaikuntanathan [GSV05]. There they showed that the task is possible, but the weak random sources they considered are fairly restricted and they only achieved entropy δn, assuming that each process starts with an n-bit weak random string. They then raised the question of whether the task is possible with general weak random sources. In this thesis we provide a positive answer to this question.

The problem of using weak random sources in cryptography has been studied by many researchers [MW97b, DS02, DO03, DOPS04, CPS07]. The most impressive work is probably the negative result of Dodis et al. [DOPS04], where they showed that almost all classical cryptographic tasks are impossible with just *one* general weak random source. However, in the setting of several independent weak random sources, a lot more can be done.

In this thesis, we provide a general framework to solve the problem of using weak random sources in these applications. Our solution is similar to extractors: we first develop a way to convert the weak random sources into distributions that are close to uniform in statistical distance. While the correct tool for a single weak random source is the extractor, here we have a network of players where each of them has a weak random source, and the correct tool turns out to be *network extractors.*

In this thesis, we define *network extractors*, which are information theoretic objects similar to extractors, and *computational network extractors*, which are the counterparts in the computational setting. Informally speaking, assume that we have a network of p parties, such that t of them are corrupted by an adversary and each of the honest parties has access to an (independent) weak random source. Our goal is to build a network extractor protocol such that at the end of the protocol, some g honest parties end up with private random bits that are close to uniform. Ideally we want $g = p - t$; however, this is not generally achievable.

In the information theoretic setting, we assume that the adversary has unlimited computational power and can see whatever message that is transmitted in the network. Thus we require the outputs to be statistically close to uniform and private. In the computational setting, we instead assume that the adversary is computationally bounded (i.e., a polynomial time Turing machine or a polynomial size circuit), and we only require that the outputs are computationally indistinguishable from uniform and private.

In the information theoretic setting, we design several efficient network extractor protocols. Using our protocols, we show that if the weak random source has min-entropy $> n/2$, then in synchronous networks we can do essentially as well as the case where each party has truly uniform random bits. We then show that both in synchronous networks and asynchronous networks, a linear fraction of corrupted parties can be tolerated even when the weak random sources only have min-entropy $2^{\log^\delta n}$ for an arbitrary constant $0 < \delta < 1$. These results imply protocols for Byzantine Agreement and Leader Election where each player only has a (independent) weak random source.

In the computational setting, under standard computational assumptions, we construct computational network extractor protocols such that even if 99% of the parties are corrupted, in the end 99% of the honest parties end up with private random bits. Under the assumption that strong one way permutations exist, we design computational network extractor protocols where all honest parties end up with private random bits. Moreover, we show that if the weak random sources have linear entropy, then it suffices to have just two honest parties. If the weak random sources have entropy n^δ, then it suffices to have just a constant number of honest parties. As a corollary, we show that secure multi-party computation is possible under these two circumstances. Our result is essentially optimal in the case of linear entropy since the negative result of [DOPS04] implies that the task is impossible with just one honest party.

This part of work is based on joint work with Yael Kalai, Anup Rao and David Zuckerman [KLRZ08, KLR09]. The detailed results appear in Chapter 4 and Chapter 5.

1.3.2 Improved Constructions of Extractors

As mentioned earlier, a very useful and standard tool in these applications is the extractor. In this thesis we also give improved constructions of certain kinds of extractors.

Ideally, we want an algorithm $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ such that given any (n,k)-source X, $\mathsf{Ext}(X)$ is close to being uniform. However, it is not hard to show that this is impossible even if k is as large as $n-1$. Given this negative result, several different directions have been explored by researchers.

One possibility is to give the extractor an additional independent short random seed. With the help of the seed the task becomes possible. Such extractors are thus called seeded extractors. In fact, most such extractors are "*strong*", in the sense that the output is close to being uniform even given the random seed. Strong seeded extractors provide an optimal solution to the problem of simulating randomized algorithms using weak random sources, and a long line of research has resulted in seeded extractors with almost optimal parameters [LRVW03, GUV09, DW08, DKSS09].

Another direction is to study extractors for special classes of sources, which are sources that have certain structures. These include for example samplable sources [TV00], bit-fixing sources [KZ07, GRS04], affine sources [GR05, Bou07], independent sources [BIW04, BKS+05, Raz05, Rao06, BRSW06] and small space sources [KRVZ06].

The results in this thesis fall into both kinds. Specifically, in this thesis we construct extractors for independent sources and affine sources, as well as a special case of strong seeded extractors known as *non-malleable* extractors. The detailed results appear in Chapter 6.

1.3.2.1 Extractors for Independent Sources

In this model, the extractor is given as input several independent weak random sources, and is supposed to output a distribution that is close to being uniform. Using the probabilistic method, it can be shown that there exists an extractor for just two independent weak random sources, with each having only logarithmic min-entropy. However, despite considerable efforts on this problem [CG88, BIW04, BKS+05, Raz05, Bou05, Rao06, BRSW06], the known constructions are far from achieving these parameters. Previously, the best explicit extractor for two independent (n, k) sources only achieves min-entropy $k \geq 0.499n$ [Bou05], the best known extractor for three independent sources achieves min-entropy $k \geq n^{0.9}$ [Rao06], and the best explicit extractor for independent (n, n^{α}) sources requires $O(1/\alpha)$ sources [Rao06, BRSW06].

In this thesis, we improve these results. Based on joint work with Kalai and Rao [KLR09], we give an efficient construction of two-source extractors for any linear entropy δn, based on a non-standard but reasonable computational assumption. The assumption is that there exist one-way permutations that are very hard to invert. A candidate one-way permutation is given in [Gol09]. In fact, our construction can even work for entropy n^{α} for some constant $1/2 < \alpha < 1$ if the one-way permutation is sufficiently hard to invert. Our construction also has other nice properties that enable its applications in cryptography. In the case of three independent sources, we give an unconditional explicit extractor that works for min-entropy $k = n^{1/2+\alpha}$ for any constant $0 < \alpha < 1/2$.

1.3.2.2 Extractors for Affine Sources

An affine source is the uniform distribution over some unkown affine subspace of a vector space. In this thesis we focus on the case where the underlying field of the vector space is $\mathbb{F}_2$. Here the entropy of the source is equal to the dimension of the affine subspace.

An affine extractor is a deterministic function such that given any affine source as the input, the output of the function is statistically close to the uniform distribution. A weaker object, called an affine disperser, only requires that the output distribution has a large support size.

Using the probabilistic method, it can again be shown that there exists a deterministic affine extractor for any (n, k) affine source, as long as $k > 2 \log n$ and $m < k - O(1)$. However, again the known constructions are far from achieving these parameters. In this thesis we make progress towards constructing optimal affine extractors. We give an affine extractor that works for entropy $k = n/\sqrt{\log \log n}$ and an affine disperser for entropy $k = n/\sqrt{\log n}$ that both output $n^{\Omega(1)}$ bits.

1.3.2.3 Non-malleable Extractors

A non-malleable extractor is a strong seeded extractor, with some additional properties. In fact, it dramatically strengthens the property of a strong extractor in the following sense. For a strong extractor, the output must be close to the uniform distribution, even given the random seed. For a non-malleable extractor, the output must be close to the uniform distribution, even given the random seed as well as the output of the extractor with the given input and an arbitrarily correlated random seed.

This kind of extractors was first proposed by Dodis and Wichs [DW09] to construct protocols for privacy amplification. Using the probabilistic method, they showed that non-malleable extractors exist with good parameters. However, they were not able to construct such non-malleable extractors. In this thesis, we construct the first explicit non-malleable extractors, based on a widely-believed conjecture about the distribution of prime numbers in arithmetic progressions. This part of work is based on joint work with Yevgeniy Dodis, Trevor Wooley and David Zuckerman [DLWZ11].

1.3.3 Privacy Amplification with an Active Adversary

This problem has also been studied by many researchers [MW97b, DKRS06, RW03, KR09a, DW09, CKOR10], and again typical assumptions are that the two parties have access to local private uniform random bits.

In this thesis we make two improvements over previous results. First, we show that the two parties can use local weak random sources instead of truly uniform random bits. In the case where the two parties have two independent n-bit sources with min-entropy $> n/2$, they can do essentially as well as the case where they have truly random bits. In the case where the two parties only have independent n-bit sources with entropy δn for an arbitrary constant $0 < \delta < 1$, they can achieve privacy amplification with security parameter up to $\Omega(\log k)$, where k is the entropy of the shared weak random source. These results give the first protocols for privacy amplification with local weak random sources.

Second, even in the case where the two parties have access to local uniform random bits, we improve the parameters of previous protocols. The two parameters that matter most in such protocols are the round complexity and the entropy loss. Assume we want to achieve a security parameter of ℓ. Non-constructively, Dodis and Wichs [DW09] showed that there exists a two round protocol that achieves entropy loss $O(\ell)$. However, previously, only one of the two optima could be achieved explicitly: we can either achieve a round complexity of 2 at the price of entropy loss $O(\ell^2)$ [DW09] or achieve entropy loss $O(\ell)$ at the price of round complexity $O(\ell)$ [CKOR10]. In this thesis we improve these results. In the case where the shared weak random source has min-entropy rate $> 1/2$, we give the first protocol that simultaneously achieves both optima– round complexity 2 and entropy loss $O(\ell)$. In the case where the shared weak random source has any linear min-entropy, we give a protocol that runs in $O(1)$ rounds and achieves optimal entropy loss $O(\ell)$. A key ingredient in our protocols is the non-malleable extractor that we construct.

Part of this work is based on joint work with Yevgeniy Dodis,Trevor Wooley and David Zuckerman [DLWZ11]. The detailed results appear in Chapter 7.

1.4 Organization of this Thesis

In Chapter 2 we give some basic definitions to set up the mathematical model that we will study. In Chapter 3 we give some previous works that will be used in our results. Since some part of this chapter is pretty technical and complicated, readers are encouraged to skip this chapter on the first reading, and to return to it whenever necessary. Following this, in Chapter 4 we discuss our constructions for distributed computing with weak random sources, in Chapter 5 we discuss our constructions for cryptography with weak random sources. Chapter 6 gives our improved constructions of extractors. Finally in Chapter 7 we present our protocols for privacy amplification.

Chapter 2

Basic Definitions

This chapter consists of a comprehensive listing of the formal definitions and basic facts that we will use throughout the thesis. We also include here many technical lemmas that will be used later in this thesis.

Throughout this thesis, we use common notations such as $\circ$ for concatenation and $[n]$ for $\{1, 2, \cdots, n\}$. All logarithms are to the base 2, unless otherwise stated. We often use capital letters for random variables and corresponding small letters for their instantiations.

We will use the convention that $N = 2^n, M = 2^m$ and $K = 2^k$. We will use U_m to denote the uniform distribution over $\{0, 1\}^m$.

2.1 Statistical Distance

Definition 2.1.1 (statistical distance)**.** Let D and F be two distributions on a set S. Their **statistical distance** is

$$|D - F| \overset{def}{=} \max_{T \subseteq S}(|D(T) - F(T)|) = \frac{1}{2}\sum_{s \in S}|D(s) - F(s)|$$

If $|D - F| \leq \epsilon$ we say that D is ϵ*-close* to F.

This measure of distance is nice because it is robust in the sense that if two distributions are close in this distance, then applying any functions to them cannot make them go further apart.

Proposition 2.1.2. *Let D and F be any two distributions over a set S s.t. $|D - F| \leq \epsilon$. Let g be any function on S. Then $|g(D) - g(F)| \leq \epsilon$.*

2.2 Convex Combinations

A very useful concept in constructing extractors is the *convex combination.* We have the following definition.

Definition 2.2.1. A distribution X is a *convex combination* of distributions $X_1, X_2, \cdots, X_\ell$ if there exist positive constants $\alpha_1, \alpha_2, \cdots, \alpha_\ell$ such that

- $\sum_i \alpha_i = 1$.
- For every $x \in \mathsf{supp}(X)$, $\Pr[X = x] = \sum_i \alpha_i \Pr[X_i = x]$.

A simple example of this concept, and one that we will often use in this thesis is the following. Suppose X and Y are two random variables in the same probability space, then the distribution of X is a convex combination of the distributions $X|Y = y$, for every $y \in \mathsf{supp}(Y)$.

The reason why the concept of convex combination is useful is that, in many situations, when a distribution is a convex combination of several distributions with some nice property, the distribution itself will also have this nice property. For example, we have the following proposition.

Proposition 2.2.2. *[Rao07b](Preservation of properties under convex combination). Let A, B, Q be distributions over the same finite probability space such that*

$$\Pr_{q \leftarrow_R Q}[|(A|q = q) - (B|Q = q)| \geq \epsilon_1] < \epsilon_2,$$

then $|A - B| < \epsilon_1 + \epsilon_2$.

The above proposition is often used to show that the output of an extractor is close to the uniform distribution.

2.3 Min-Entropy and Sources of Randomness

Definition 2.3.1. The *min-entropy* of a random variable X is defined as

$$H_\infty(X) = min_{x \in \mathsf{supp}(X)}\{-\log_2 \Pr[X = x]\}.$$

We say X is an (n, k)-source if X is a random variable on $\{0, 1\}^n$ and $H_\infty(X) \geq k$. When n is understood from the context we simply say that X is a k-source.

Lemma 2.3.2. *Let X be an (n, k) source. Let $S \subseteq [n]$ with $|S| = n'$. Let $X_{|S}$ denote the projection of X to the bit locations in S. Then for every l, $X_{|S}$ is 2^{-l}-close to a $(n', k - (n - n') - l)$ source.*

Proof. Let $\overline{S}$ be the complement of S.

Then $X_{|S}$ is a convex combination over $X_{|\overline{S}}$. For each setting of $X_{|\overline{S}} = h$, we induce the distribution $X_{|S}|X_{|\overline{S}} = h$.

Define $H = \{h \in \{0,1\}^{n-n'} | H_\infty(X_{|S}|X_{|\overline{S}} = h) < n' - n + k - l\}$. Notice that $H_\infty(X_{|S}|X_{|\overline{S}} = h) = H_\infty(X|X_{|\overline{S}} = h)$. Then by Proposition 2.3.8, for every $h \in H$, $\Pr[X_{|\overline{S}} = h] < 2^{k-(n-n')-l-k} = 2^{-(-n'+n+l)}$. Since $|H| \leq 2^{n-n'}$, by the union bound we get that $\Pr[X_{|\overline{S}} \in H] \leq 2^{-l}]$. $\square$

2.3.1 Block Sources and Conditional entropy.

A block source is a source broken up into a sequence of blocks, with the property that each block has min-entropy even conditioned on previous blocks.

Definition 2.3.3 (Conditional Min-Entropy)**.** Given random variables A, B in the same probability space, we define the conditional min-entropy

$$H_\infty(A|B) = \min_b H_\infty(A|B = b)$$

Definition 2.3.4 (Block sources)**.** A distribution $X = X_1, X_2, \cdots, X_{\mathsf{C}}$ is called a $(k_1, k_2, \ldots, k_{\mathsf{C}})$-block source if for all $i = 1, \ldots, \mathsf{C}$, we have that $H_\infty(X_i|X_{i-1}, \ldots, X_1) \geq k_i$. If $k_i = k$ for every i, we say that X is a k-block source.

If $X = X_1, \cdots, X_{\mathsf{t}}$ is a random variable (not necessarily a block source) over $\{0,1\}^n$ divided into t blocks in some way, and $x_1, \ldots, x_i$ are some strings with $0 \leq i < \mathsf{t}$, we use the notation $X|x_1, \ldots, x_i$ to denote the random variable X conditioned on $X_1 = x_1, \ldots, X_i = x_i$. For $1 \leq i < j \leq \mathsf{t}$, we denote by $X_{i,\ldots,j}$ the projection of X into the blocks $X_i, \ldots, X_j$. The following lemma is useful to prove that a distribution is close to a block source.

Lemma 2.3.5. *Let $X = X_1, \ldots, X_t$ be t dependent random variables. For every $i = 1, 2, \ldots, t$, let X^i denote the concatenation of the first i variables. Suppose each X^i takes values in $\{0,1\}^{n_i}$ and for every $i = 1, 2, \ldots, t$, X^i is ϵ_i-close to k_i-light, with $\sum_i \epsilon_i < 1/10$. Then for every $\ell > 10 \log t$ we must have that X is $\sum_{i=1}^t \epsilon_i + t2^{-\ell}$-close to a block source, where each block X_i has min-entropy $k_i - n_{i-1} - 1 - \ell$.*

Proof. We will need to define the notion of a *submeasure*. Let $n = n_t$. Say that $M : \{0,1\}^n \to [0,1]$ is a submeasure on $\{0,1\}^n$ if $\sum_{m \in \{0,1\}^n} M(m) \leq 1$. Note that every probability measure is a submeasure. We abuse notation and let $M(x^i)$ denote the marginal measure induced on the first i coordinates.

We say a submeasure on $\{0,1\}^n$ is ϵ-close to k-light if

$$\sum_{m \in \{s : M(s) > 2^{-k}\}} M(m) \leq \epsilon.$$

Note that when M is a probability measure, the above corresponds to saying that M is ϵ-close to having min-entropy k.

As usual, for any event $A \subset \{0,1\}^n$, we denote $\Pr[M \in A] = \sum_{m \in A} M(m)$.

We now define the submeasures $M_{t+1} = X$, and for $i = t, t-1, t-2, \ldots, 1$,

$$M_i(m) = \begin{cases} 0 & M^i_{i+1}(m^i) > 2^{-k_i} \vee M^i_{i+1}(m^i) < 2^{-n_i - \ell} \\ M_{i+1}(m) & \text{otherwise} \end{cases}$$

Let $M = M_1$. Now note that for every $j < i$, M^j_i is ϵ_j-close to k_j-light, since we only made points lighter in the above process. Further, for all m and $i \leq j$, $M^j_i(m^j) \leq 2^{-k_j}$, since we reduced the weight of all m's that violated this to 0. We also have that for every m, i, $M^i(m^i) = 0$ or $M^i(m^i) \geq 2^{-n_i - \ell}$ by our construction.

Now define the sets $B_i = \{m \in \{0,1\}^n : M_i(m) \neq M_{i+1}(m)\}$. Set $B = \cup_i B_i$. Then note that $\Pr[X \in B] \leq \sum_{i=2}^t \Pr[M_{i+1} \in B_i]$. Each B_i, contains two types of points: points that were removed when moving from M_{i+1} to M_i because they were too heavy, and points that were removed because they were too light. We set $C_i = \{m : M_{i+1}(m^i) > 2^{-k_i}\}$, namely the "too heavy" points. We see that $\Pr[M_{i+1} \in C_i] \leq \epsilon_i$, since M^i_{i+1} is ϵ_i-close to k_i-light. Set $D_i = \{m : M_{i+1}(m^i) < 2^{-n_i - \ell}\}$, namely the "too light" points. We get $\Pr[M_{i+1} \in D_i] < 2^{-\ell}$ by the union bound. Using both these estimates, we get that $\Pr[X \in B] \leq \sum_{i=1}^t \Pr[M_{i+1} \in B_i] \leq \sum_{i=1}^t \Pr[M_{i+1} \in C_i] + \Pr[M_{i+1} \in D_i] \leq \sum_i \epsilon_i + t2^{-\ell}$.

Now define the distribution $Z = X | X \notin B$. Then Z is $\sum_i \epsilon_i + t2^{-\ell}$-close to X. For every i and $z \in \mathsf{supp}(Z)$, we have that $\Pr[Z^i = z^i | Z^{i-1} = z^{i-1}] = \Pr[Z^i = z^i] / \Pr[Z^{i-1} = z^{i-1}] \leq 2^{-k_i+1} / 2^{-n_{i-1} - \ell}$ (since every point at most doubles in weight over M), which proves the lemma. $\square$

Let $X = X_1, \cdots, X_{\mathsf{t}}$ be a random variable over $\{0,1\}^n$ divided into t blocks in some way, and $x_1, \ldots, x_i$ are some strings with $0 \leq i < \mathsf{t}$. We use the notation $X | x_1, \ldots, x_i$ to denote the random variable X conditioned on $X_1 = x_1, \ldots, X_i = x_i$. For $1 \leq i < j \leq \mathsf{t}$, we denote by $X_{i,\ldots,j}$ the projection of X onto the blocks $X_i, \ldots, X_j$.

Next we show that any weak source with linear min-entropy can be divided into a constant number of blocks, such that the source is close to a convex combination of block sources.

Lemma 2.3.6. *Let X be an $(n, \alpha n)$ source for some constant $0 < \alpha < 1$. Let $t = \frac{4}{\alpha}$. Divide X evenly into t blocks $X = X_1 \circ X_2 \circ \cdots \circ X_t$. Then X is $2^{-\Omega(n)}$-close to being a convex combination of sources $\{X^j\}_{j \in J}$ such that for every j there exists $g \in [t]$ for which*

- *$X_1^j, \ldots, X_{g-1}^j$ is fixed.*
- *$H_\infty(X_g^j) \geq \frac{\alpha^2}{6}$.*
- *$H_\infty(X | X_g^j) \geq \frac{\alpha^2}{6}$.*

To prove this lemma, first we need the definition of a subsource.

Definition 2.3.7 (Subsource)**.** Given random variables X and X' on $\{0,1\}^n$ we say that X' is a *deficiency-d subsource* of X and write $X' \subseteq X$ if there exits a set $A \subseteq \{0,1\}^n$ such that $(X|A) = X'$ and $\Pr[x \in A] \geq 2^{-d}$.

Proposition 2.3.8. *Let X be a random variable with $H_\infty(X) = k$. Let $X' \subset X$ be a subsource of deficiency d corresponding to some set $A \subset \{0,1\}^n$. Then $H_\infty(X') = k - d$.*

Lemma 2.3.9 ([BRSW06])**.** *Let X be a random variable over $\{0,1\}^n$ such that X is ϵ-close to an (n, k) source with $\epsilon \leq 1/4$. Then there is a deficiency 2 subsource $X' \subseteq X$ such that X' is a $(n, k-3)$ source.*

More generally, we have the statement that conditioning on *typical* values of any function cannot reduce the min-entropy of our source by much more than we expect.

Lemma 2.3.10 (Fixing a function)**.** *[BRSW06] Let X be a distribution over $\{0,1\}^n$, $F : \{0,1\}^n \to \{0,1\}^m$ be a function, and $\ell \geq 0$ some number. For every $s \in \mathsf{supp}(F(X))$, define X_s to be the subsource $X|F(X) = s$. Then there exists $s \in \{0,1\}^m$ for which X_s has deficiency at most m. Furthermore, we have that*

$$\Pr_{s \leftarrow_R F(X)}[\textit{deficiency of } X_s \leq m + \ell] \geq 1 - 2^{-\ell}$$

Proof. Let S be the set of $s \in \{0,1\}^m$ such that $\Pr[F(x) = s] < 2^{-m-\ell}$. Since $|S| \leq 2^m$, we have that $\Pr[F(X) \in S] < 2^{-\ell}$. If we choose $s \leftarrow_{\mathrm{R}} F(X)$ and $s \notin S$, we get that $X|F(X) = s$ has deficiency $\leq m + \ell$. Choosing $\ell = 0$ we get the first part of the proposition. $\square$

We next give a lemma that is used to prove Lemma 2.3.6.

Lemma 2.3.11 (Fixing Entropies)**.** *Let $X = X_1 \circ X_2 \circ \cdots \circ X_t$ be a t-block random variable over $\{0,1\}^n$. Fix any $s > 0$ and let $0 = \tau_1 < \tau_2 < \cdots < \tau_{c+1} = n$ be some numbers. There exists a universe U such that for every X there exists a set of random variables $\{X^j\}_{j \in U}$ and a random variable J over U, such that $X = X^J$ (i.e., X is a convex combination of $\{X^j\}_{j \in U}$). $\{X^j\}$ has the following properties:*

- *For every $j \in U$ s.t. $\Pr[J = j] > 0$, there exists a sequence $\bar{e^j} = e^j_1, \cdots, e^j_t \in [c]^t$ such that for every $0 < i \le t$ and every sequence $x_1, \cdots, x_{i-1} \in \mathsf{Supp}(X^j_{1,\cdots,i-1})$;*

$$\tau_{e^j_i} < H_\infty(X^j_i | x_1, \cdots, x_{i-1}) \le \tau_{e^j_i + 1}$$

- *with probability $1 - t2^{-s}$ over J, X^j is a subsource of X with deficiency $< t^2 \log c + ts$.*

Proof. We prove this by induction on t. The base case where $t = 1$ is trivially true. Now suppose this is true for up to $t-1$ blocks and we'll prove it for t blocks. For every $x_1 \in \mathsf{Supp}(X_1)$ define the source $Y(x_1)$ to be $X_{2,\cdots,t} | x_1$. By the induction hypothesis, there exists a universe U' and a random variable J' over U' such that $Y(x_1) = Y^{J'}$. For every $j' \in U'$ s.t. $\Pr[J' = j'] > 0$ there exists a sequence $\bar{e^{j}}'(x_1) \in [c]^{t-1}$ such that $Y^{j'}$ satisfies the first property with respect to $\bar{e^{j}}'(x_1)$. Define the function $F_{j'} : X_1 \to [c]^{t-1}$ that maps x_1 to $\bar{e^{j}}'(x_1)$.

Now let the new universe be $U = \mathbf{Range}(F(X_1)) \times U'$. Note that U is the same for all X. Define the new random variable J over U such that the event $J = (\bar{e}, j')$ stands for $(J' = j', F_{j'}(X_1) = \bar{e})$. Then the convex combination $X = X^J$ satisfies property 1. Moverover, by Lemma 2.3.10, with probability $1 - 2^{-s}$, $X_1 | F_{j'}(X_1) = \bar{e}$ is a deficiency $(t-1)\log c + s$ subsource of X_1, and by the induction hypothesis with probability $1 - (t-1)2^{-s}$ over J', $Y^{j'}$ is a deficiency $(t-1)^2 \log c + (t-1)s$ subsource of $Y(x_1)$. Thus with probability at least $1 - (t-1)2^{-s} - 2^{-s} = 1 - t2^{-s}$, the deficiency of X^j is at most $(t-1)^2 \log c + (t-1)s + (t-1)\log c + s < t^2 \log c + ts$. ☐

Corollary 2.3.12. *If in the lemma above X has min-entropy k, and X^j is a deficiency $t^2 \log c + ts$ subsource of X as in property 2 with $\bar{e^j}$ the sequence corresponding to X^j as in property 1, then $\sum_{i=1}^t \tau_{e^j_i + 1} \ge k - t^2 \log c - ts$.*

Proof. If this was not the case, we could find some string in the support of X that is too heavy. Specifically, we take the heaviest string allowed in each successive block to get $x = x_1 \circ x_2 \circ \cdots \circ x_t$. Then it must be $\Pr[X_i = x_i | x_1, \cdots, x_{i-1}] \ge 2^{-\tau_{e^j_i + 1}}$ for any $0 < i \le t$. Together with the fact that X^j has deficiency $< t^2 \log c + ts$ we get $\Pr[X = x] > 2^{-(t^2 \log c + ts)} \Pi_{i=1}^t 2^{-\tau_{e^j_i + 1}} = 2^{-(t^2 \log c + ts)} 2^{-\sum_{i=1}^t \tau_{e^j_i + 1}} > 2^{-k}$. This contradicts the fact that X has min-entropy k. ☐

Proof of Lemma 2.3.6. We'll use Lemma 2.3.11. Let the parameters in that lemma be $s = \sqrt{k}$, $c = \frac{6}{\alpha^2}$ and $\tau_i = \frac{i-1}{c}n$ for $0 < i \leq c+1$. Then Lemma 2.3.11 shows that X is a convex combination of $\{X^j\}_{j \in U}$ and with probability $1 - t2^{-s} = 1 - 2^{-n^{\Omega(1)}}$, X_j is a subsource with deficiency $< t^2 \log c + ts < 0.01k$. Now Corollary 2.3.12 says that for such a X^j, we must have $\sum_{i=1}^{t} \tau_{e_i^j+1} \geq k - t^2 \log c - ts > 0.99k$. We now show that there must exist at least two e_i^j's s.t. $e_i^j \geq 2$. Otherwise suppose there is at most one e_i^j s.t. $e_i^j \geq 2$. For $e_i^j = 1$ we have $\tau_{e_i^j+1} = \tau_2 = \frac{n}{c}$. For $e_i^j \geq 2$ we have the min-entropy of the block X_i^j conditioned on any fixing of previous blocks is at most $\frac{n}{t}$. Assume for the sake of simplicity that $\frac{n}{ct} = \frac{1.5}{\alpha}$ is an integer, thus $\frac{n}{t}$ appears in set $\{\tau_i\}$ and we must have $\tau_{e_i^j+1} \leq \frac{n}{t}$. Therefore $\sum_{i=1}^{t} \tau_{e_i^j+1} \leq \frac{n}{c}(t-1) + \frac{n}{t} < \frac{tn}{c} + \frac{n}{t} = \frac{2\alpha}{3}n + \frac{\alpha}{4}n < 0.99\alpha n = 0.99k$, which is a contradiction.

Thus, there must exist at least two e_i^j's s.t. $e_i^j \geq 2$, so $\tau_{e_i^j} \geq \frac{n}{c} = \frac{\alpha^2}{6}n$. Let $0 < i_1 < i_2 \leq t$ be the two corresponding i's. Let $g = i_1$ and further condition on any fixing of $X_1^j, \ldots, X_{g-1}^j$. Now by Lemma 2.3.11, we see X is $2^{-\Omega(n)}$-close to being a convex combination of sources $\{X^j\}_{j \in J}$ that satisfy the properties in Lemma 2.3.6. □

We use the following standard lemma about conditional min-entropy. (For a proof, we refer the reader to the proof of Lemma 5 in [MW97a]).

Lemma 2.3.13. *Let X and Y be random variables and let $\mathcal{Y}$ denote the range of Y. Then for all $\epsilon > 0$*

$$\Pr_Y \left[H_\infty(X|Y=y) \geq H_\infty(X) - \log|\mathcal{Y}| - \log\left(\frac{1}{\epsilon}\right) \right] \geq 1 - \epsilon$$

Sometimes a random variable may only be close to having a certain amount of min-entropy, and we have the following lemma, which can be viewed as a generalization of Lemma 3.5.14.

Lemma 2.3.14. *Let X be an (n,k)-source and X' be a random variable such that $|X - X'| < \epsilon$. Let Z be another random variable and $\mathcal{Z}$ denote the range of Z. Then for all $\epsilon_1 > 0$*

$$\Pr_Z \left[(X'|Z=z) \text{ is } \frac{\epsilon|\mathcal{Z}|}{\epsilon_1} \text{ close to having min-entropy } k - \log|\mathcal{Z}| - \log\left(\frac{1}{\epsilon_1}\right) \right] \geq 1 - \epsilon_1$$

Proof. For a particular $Z = z$, $\Pr[X' = x'|Z = z] = \frac{\Pr[X'=x', Z=z]}{\Pr[Z=z]}$. Since X is an (n,k)-source X must have size of support at least 2^k. Choose a subset S in the support of size $\epsilon_1 2^k/|\mathcal{Z}|$ and let $\bar{X}$ be the source that is uniformly distributed on S. Then $H_\infty(\bar{X}) = k - \log|\mathcal{Z}| - \log(1/\epsilon_1)$. Let R be the set $\{r \in S : \Pr[X' = r|Z = z] > |\mathcal{Z}|/(\epsilon_1 2^k)\}$, then

$$|(X'|Z=z) - \bar{X}| = \sum_{r \in R} (\Pr[X' = r|Z = z] - |\mathcal{Z}|/(\epsilon_1 2^k)).$$

If $\Pr[Z = z] > \frac{\epsilon_1}{|\mathcal{Z}|}$, then

$$|X'|(Z = z) - \bar{X}| < \sum_{r \in R} \frac{\Pr[X' = r, Z = z] - 2^{-k}}{\epsilon_1/|\mathcal{Z}|} \leq \frac{\sum_{r \in R} \Pr[X' = r] - \Pr[X = r]}{\epsilon_1/|\mathcal{Z}|}$$
$$\leq \frac{\epsilon}{\epsilon_1/|\mathcal{Z}|} = \frac{\epsilon|\mathcal{Z}|}{\epsilon_1}.$$

The probability this does not happen is at most $\frac{\epsilon_1}{|\mathcal{Z}|}|\mathcal{Z}| = \epsilon_1$. □

Sometimes we need the definition of *average conditional min-entropy.*

Definition 2.3.15. The *average conditional min-entropy* is defined as

$$\widetilde{H}_\infty(X|W) = -\log\left(\mathrm{E}_{w \leftarrow W}\left[\max_x \Pr[X = x|W = w]\right]\right) = -\log\left(\mathrm{E}_{w \leftarrow W}\left[2^{-H_\infty(X=x|W=w)}\right]\right).$$

Average conditional min-entropy tends to be useful for cryptographic applications. It is an essentially weaker notion than min-entropy, as min-entropy implies average conditional min-entropy with a small loss in parameters. We have the following lemmas.

Lemma 2.3.16 ([DORS08])**.** *For any $s > 0$, $\Pr_{w \leftarrow W}[H_\infty(X|W = w) \geq \widetilde{H}_\infty(X|W) - s] \geq 1 - 2^{-s}$.*

Lemma 2.3.17 ([DORS08])**.** *If a random variable B has at most 2^ℓ possible values, then $\widetilde{H}_\infty(A|B) \geq H_\infty(A) - \ell$.*

2.3.2 Somewhere Random Sources

Definition 2.3.18 (Somewhere Random sources)**.** A source $X = (X_1, \cdots, X_t)$ is $(t \times r)$ *somewhere-random* (SR-source for short) if each X_i takes values in $\{0,1\}^r$ and there is an i such that X_i is uniformly distributed.

Definition 2.3.19. An elementary somewhere-k-source is a vector of sources $(X_1, \cdots, X_t)$, such that some X_i is a k-source. A somewhere k-source is a convex combination of elementary somewhere-k-sources.

Definition 2.3.20. (aligned SR-source) [Rao06] We say that a collection of SR-sources $X_1, \cdots, X_u$ is *aligned* if there is some i such that the i'th row of every SR-source in the collection is uniformly distributed.

2.3.3 Affine Sources

Definition 2.3.21. (affine source) Let $\mathbb{F}_q$ be the finite field with q elements. Denote by $\mathbb{F}_q^n$ the n-dimensional vector space over $\mathbb{F}_q$. A distribution X over $\mathbb{F}_q^n$ is an $(n,k)_q$ affine source if there exist linearly independent vectors $a_1, \cdots, a_k \in \mathbb{F}_q^n$ and another vector $b \in \mathbb{F}_q^n$ s.t. X is sampled by choosing $x_1, \cdots, x_k \in \mathbb{F}$ uniformly and independently and computing

$$X = \sum_{i=1}^{k} x_i a_i + b.$$

In this thesis we focus on the case where $q = 2$ and we will just write (n,k) affine sources instead of $(n,k)_2$ affine sources.

In the case of affine sources, the min-entropy coincides with the standard Shannon entropy, and we will just call it entropy.

The following lemma explains the behavior of a linear function acting on an affine source.

Lemma 2.3.22. *(Affine Conditioning). Let X be any affine source on $\{0,1\}^n$. Let $L : \{0,1\}^n \to \{0,1\}^m$ be any linear function. Then there exist independent affine sources A, B such that:*

- *$X = A + B$.*
- *For every $b \in \mathsf{Supp}(B)$, $L(b) = 0$.*
- *$H(A) = H(L(A))$ and there exists an affine function $L^{-1} : \{0,1\}^m \to \{0,1\}^n$ such that $A = L^{-1}(L(A))$.*

Proof. Without loss of generality, assume the support of X is a linear subspace (if not, we can do the analysis for the corresponding linear subspace, and then add the affine shift). Consider the set $\{x \in \mathsf{Supp}(X) : L(x) = 0\}$. Note that this set is a linear subspace. Let B be the affine source whose support is this subspace and let $b_1, ..., b_t$ be a basis for this subspace. Next we complete the basis to get a basis for the support of X. Let A be the affine source whose support is the span of the basis vectors in the completed basis that are not in B. Thus $X = A + B$.

Note that $H(A) \leq H(L(A))$ since $L(a) \neq 0$ for every $a \in \mathsf{Supp}(A)$. On the other hand, since L is a deterministic function we have $H(L(A)) \leq H(A)$. Thus $H(A) = H(L(A))$. In other words, L is a bijection between $\mathsf{Supp}(A)$ and $\mathsf{Supp}(L(A))$. Let $Y = L(A)$. Since A is an affine source there exists a vector $a \in \{0,1\}^n$ such that $A = a + \bar{A}$ where $\bar{A}$ is the uniform distribution over a linear subspace. Thus

$$Y = L(A) = L(a) + L(\bar{A}).$$

Let $\bar{Y} = Y - L(a) = L(\bar{A})$. Since L is a linear function and $\bar{A}$ is uniform distributed over a linear subspace, $\bar{Y}$ is also uniformly distributed over a linear subspace. Note that $H(\bar{Y}) = H(Y) = H(L(A)) = H(L(\bar{A}))$, thus L is a linear bijection between the linear subspaces of $\mathsf{Supp}(\bar{A})$ and $\mathsf{Supp}(\bar{Y})$. Therefore there exists a linear function L' such that $\bar{A} = L'(\bar{Y})$. Thus

$$A = a + \bar{A} = a + L'(\bar{Y}) = a + L'(Y - L(a)) = L'(Y) + a - L'(L(a)).$$

Take L^{-1} to be the affine function $L' + a - L'(L(a))$. Then $A = L^{-1}(L(A))$. $\square$

Now we have the following lemma that exhibits a nice structure of affine sources.

Lemma 2.3.23. *Let X be any affine source on $\{0,1\}^n$. Divide X into t arbitrary blocks $X = X_1 \circ X_2 \circ ... \circ X_t$. Then there exist positive integers $k_1, ..., k_t$ such that,*

- *$\forall j, 1 \leq j \leq t$ and $\forall (x_1, .., x_{j-1}) \in \mathsf{Supp}(X_1, .., X_{j-1})$, $H(X_j|_{X_1=x_1,...,X_{j-1}=x_{j-1}}) = k_j$.*
- *$\sum_{i=1}^{t} k_i = H(X)$.*

Proof. For any $i, 1 \leq i \leq t$, let $Y_i = X_1 \circ X_2 \circ ... \circ X_i$. Note Y_i is a linear function of X, thus Y_i is also an affine source. Now for any j, note that Y_{j-1} is a linear function L_j of Y_j. Thus by Lemma 2.3.22, there exist independent affine sources A_j, B_j such that $Y_j = A_j + B_j$, $H(L_j(Y_j)) = H(A_j)$ and for every $b \in \mathsf{Supp}(B)$, $L_j(b) = 0$. This implies that $Y_{j-1} = L_j(Y_j) = L_j(A_j + B_j) = L_j(A_j)$. Now since $H(L_j(Y_j)) = H(A_j)$, we have that $\forall (x_1, .., x_{j-1}) \in \mathsf{Supp}(X_1, .., X_{j-1})$, there exists a unique $a_j \in \mathsf{Supp}(A_j)$ such that $L_j(a_j) = (x_1, .., x_{j-1})$.

Let $k_j = H(B_j)$. Then,

$$H(X_j|_{X_1=x_1,...,X_{j-1}=x_{j-1}}) = H(X_j|_{A_j=a_j}) = H(Y_j|_{A_j=a_j}) = H(A_j + B_j|_{A_j=a_j}) = H(B_j) = k_j$$

where the last equality holds because A_j, B_j are independent.

Next, observe that $H(Y_j) = H(A_j) + H(B_j) = H(L_j(Y_j)) + H(B_j) = H(Y_{j-1}) + k_j$. A simple induction thus gives that

$$\sum_{i=1}^{t} k_i = H(X).$$

$\square$

This lemma essentially says that if we divide an affine source into several blocks, then a block has the same entropy conditioned on any fixing of the previous blocks. Moreover, the sum of these entropies equals the entropy of the original source. Thus we can view each block as carrying some fixed additional entropy, regardless of what the previous blocks are. We note that this is very different from general weak random sources.

2.4 Cryptographic Definitions

Definition 2.4.1. A function $\mu(\cdot)$ is **negligible** if for every polynomial $q(\cdot)$ there exists a value N such that for all $n > N$ it holds that $\mu(n) < 1/q(n)$.

Definition 2.4.2. Let $\mathcal{D} = \{D_n\}_{n\in\mathbb{N}}$ and $\mathcal{F} = \{F_n\}_{N\in\mathbb{N}}$ be two distribution ensembles. We say that $\mathcal{D}$ and $\mathcal{F}$ are **computationally indistinguishable**, denoted by $\mathcal{D} \approx \mathcal{F}$, if for every non-uniform algorithm $\mathcal{A}$ running in time poly(n) there exists a negligible function ϵ such that for every $n \in \mathbb{N}$,

$$|\Pr[\mathcal{A}(D_n) = 1] - \Pr[\mathcal{A}(F_n) = 1]| \leq \epsilon(n).$$

Remark. Often we abuse notation, and let $D_n \approx F_n$ denote the fact that the two ensembles are computationally indistinguishable.

Definition 2.4.3. If $\mathcal{D} = \{D_n\}_{n\in\mathbb{N}}$ and $\mathcal{F} = \{F_n\}_{N\in\mathbb{N}}$ are two distribution ensembles, and there exists a negligible function $\epsilon(n)$ such that for every $n \in \mathbb{N}$,

$$|D_n - F_n| \leq \epsilon(n),$$

then we say that $\mathcal{D}$ and $\mathcal{F}$ are **statistically close**, and denote it by

$$\mathcal{D} \equiv \mathcal{F}.$$

Remark. Often we abuse notation, and let $D_n \equiv F_n$ denote the fact that the two ensembles are statistically close.

Lemma 2.4.4. *Let $\{X_n\}$ and $\{Y_n\}$ be two distribution ensembles. Let $E = \{E_n\}$ be a sequence of events for which there exists a negligible function ϵ such that $\Pr[E_n] = 1 - \epsilon(n)$. Then $\{X_n|E_n\} \approx \{Y_n|E_n\}$ implies that $\{X_n\} \approx \{Y_n\}$.*

We will also use the the following generalization of Lemma 2.4.4.

Lemma 2.4.5. *Let $\{X_n\}$ and $\{Y_n\}$ be two distribution ensembles. Let J be a set such that for every $j \in J$, $E^j = \{E_n^j\}$ is a sequence of events for which $\{X_n|E_n^j\} \approx \{Y_n|E_n^j\}$. Then, if there exists a negligible function ϵ such that $\Pr[\cup_{j\in J} E_n^j] = 1 - \epsilon(n)$, then $\{X_n\} \approx \{Y_n\}$.*

Chapter 3

Building Blocks

In this chapter we describe some of the tools and previous works that we will use in this thesis. Readers can skip this chapter in a first reading, and return for more details when clarification is needed in later chapters.

3.1 Extractors, Dispersers, Mergers and Condensers

Definition 3.1.1. A function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a *strong seeded extractor* for min-entropy k and error ϵ if for every min-entropy k source X,

$$|(\mathsf{Ext}(X,R),R) - (U_m,R)| < \epsilon,$$

where R is the uniform distribution on d bits independent of X, and U_m is the uniform distribution on m bits independent of R.

Definition 3.1.2. A function $\mathsf{TExt} : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \{0,1\}^m$ is a *strong two source extractor* for min-entropy k_1, k_2 and error ϵ if for every independent (n_1,k_1) source X and (n_2,k_2) source Y,

$$|(\mathsf{TExt}(X,Y),X) - (U_m,X)| < \epsilon$$

and

$$|(\mathsf{TExt}(X,Y),Y) - (U_m,Y)| < \epsilon,$$

where U_m is the uniform distribution on m bits independent of (X,Y).

Definition 3.1.3. (affine extractor) A function $\mathsf{AExt} : \mathbb{F}_q^n \to \{0,1\}^m$ is a deterministic (k,ϵ)-affine extractor if for every $(n,k)_q$ affine source X,

$$|\mathsf{AExt}(X) - U_m| \le \epsilon.$$

Here U_m is the uniform distribution over $\{0,1\}^m$ and $|\cdot|$ stands for the statistical distance.

Definition 3.1.4. (affine disperser) A function $\mathsf{ADisp} : \mathbb{F}_q^n \to \{0,1\}^m$ is a deterministic (k,ϵ)-affine disperser if for every $(n,k)_q$ affine source X,

$$|\mathsf{Supp}(\mathsf{ADisp}(X))| \geq (1-\epsilon)2^m.$$

The function is called a zero-error disperser if $\epsilon = 0$.

Definition 3.1.5. A function $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a $(k \to l, \epsilon)$-condenser if for every k-source X, $C(X, U_d)$ is ϵ-close to some l-source. When convenient, we call C a rate-$(k/n \to l/m, \epsilon)$-condenser.

Definition 3.1.6. A function $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a $(k \to l, \epsilon)$-somewhere-condenser if for every k-source X, the vector $(C(X,y)_{y\in\{0,1\}^d})$ is ϵ-close to a somewhere-l-source. When convenient, we call C a rate-$(k/n \to l/m, \epsilon)$-somewhere-condenser.

Definition 3.1.7 (Merger)**.** A function $M : (\{0,1\}^n)^s \times \{0,1\}^d \to \{0,1\}^n$ is called an (m,ϵ)-merger (of (n,s)-somewhere-random sources), if for every (n,s)-somewhere random source $X = (X_1, \cdots, X_s)$, and for R being uniformly distributed over $\{0,1\}^d$, the distribution of $M(X,R)$ is ϵ-close to having min-entropy m. We say that the merger is strong if the average over $r \in \{0,1\}^d$ of the statistical distance between $M(X,r)$ and an (n,m)-source is $\leq \epsilon$.

3.1.1 Strong Linear Seeded Extractors

We need the following definition and property of a specific kind of extractors.

Definition 3.1.8. A function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a *strong seeded extractor* for min-entropy k and error ϵ if for every min-entropy k source X,

$$\Pr_{u \leftarrow_R U_d}[|\mathsf{Ext}(X,u) - U_m| \leq \epsilon] \geq 1-\epsilon,$$

where U_m is the uniform distribution on m bits. We say that the function is a *linear strong seeded extractor* if the function $\mathsf{Ext}(\cdot, u)$ is a linear function over $\mathsf{GF}(2)$, for every $u \in \{0,1\}^d$.

Proposition 3.1.9 ([Rao09])**.** *Let* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a linear strong seeded extractor for min-entropy* k *with error* $\epsilon < 1/2$*. Let* X *be any affine source with entropy* k*. Then,*

$$\Pr_{u \leftarrow_R U_d}[|\mathsf{Ext}(X,u) - U_m| = 0] \geq 1-\epsilon$$

3.1.2 Reconstructive Extractors

In this thesis we are going to use a special kind of extractors, called reconstructive extractors. Informally, the word "reconstructive" means that, if some algorithm $\mathcal{A}$ can distinguish the output of the extractor from the uniform distribution with some sufficiently large probability, then there is another algorithm that can use $\mathcal{A}$ to reconstruct the input source. Following [Uma05], we define reconstructive extractors:

Definition 3.1.10. A $(n, t, m, d, a, \epsilon, \delta)$-reconstructive extractor is a triple of functions:

- A polynomial time computable extractor function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$
- An advice function $\mathsf{A} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$
- A $\text{poly}(n, 1/\epsilon)$ time randomized oracle reconstruction procedure $\mathcal{R} : \{0,1\}^a \to \{0,1\}^n$

That satisfy the property that for every $x \in \{0,1\}^n$ and $\mathcal{D} : \{0,1\}^m \to \{0,1\}$ for which

$$|\Pr[\mathcal{D}(\mathsf{Ext}(x, U_t), U_t) = 1] - \Pr[\mathcal{D}(U_m, U_t) = 1]| \geq \epsilon,$$

we must have that

$$\Pr_w[\mathcal{R}^{\mathcal{D}}(A(x, w)) = x] \geq \delta.$$

Note that Ext as above must be a seeded extractor for n bit sources with entropy larger than a, since any function that distinguishes the output from uniform can be used to get a procedure that guesses x with probability roughly 2^{-a}.

We have the following theorem, which follows from the discussion in section 6 of [Uma05]:

Theorem 3.1.11 ([Uma05])**.** *There is a constant $\beta > 0$ such that for every n, a, ϵ with $a = n^{\Omega(1)}$, there exists $(n, t = O(\log(n/\epsilon))), m = n^{\beta}, d = O(\log(n/\epsilon)), a, \epsilon, 1/2)$ reconstructive extractor.*

An almost immediate consequence of this theorem is the following theorem.

Theorem 3.1.12. *For every n, k, ϵ with $k = n^{\Omega(1)}$ and $\epsilon = \frac{1}{\text{poly}(n^{\log n})}$, there is a polynomial time computable function $\mathsf{RExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that $d = O(\log(n/\epsilon))$, $m = k^{\Omega(1)}$ and if f is a one way function for $k/3$ sources and X is an (n, k) source, then for every distinguisher $\mathcal{A}$ of size $\text{poly}(n^{\log n})$,*

$$|\Pr[\mathcal{A}(f(X), \mathsf{RExt}(X, U_d), U_d) = 1] - \Pr[\mathcal{A}(f(X), U_m, U_d) = 1]| = \text{negl}(n).$$

Proof. We set RExt to be the reconstructive extractor promised by Theorem 3.1.11, set up so that $a = k/2$.

Assume for the sake of contradiction that there was a distinguishing circuit D of size $\mathrm{poly}(n^{\log n})$ and a polynomial q such that for infinitely many n's,

$$|\Pr[\mathcal{A}(f(X), \mathsf{RExt}(X, U_d), U_d) = 1] - \Pr[\mathcal{A}(f(X), U_m, U_d) = 1]| \geq \frac{1}{q(n)}.$$

By a standard averaging argument we have

$$\Pr_{x \leftarrow X}[|\Pr[\mathcal{D}(f(x), \mathsf{RExt}(x, U_t), U_t) = 1] - \Pr[\mathcal{D}(f(x), U_m, U_t) = 1]| \geq \frac{1}{2q(n)}] \geq \frac{1}{2q(n)}$$

Note $\frac{1}{2q(n)} \geq \epsilon$ since $\epsilon = \frac{1}{\mathrm{poly}(n^{\log n})}$. Thus by the definition, there is a circuit R^D of size $\mathrm{poly}(n, 1/\epsilon)\mathsf{size}(D) = \mathrm{poly}(n^{\log n})$ such that for every x s.t. $|\Pr[\mathcal{D}(f(x), \mathsf{RExt}(x, U_t), U_t) = 1] - \Pr[\mathcal{D}(f(x), U_m, U_t) = 1]| \geq \frac{1}{2q(n)} \geq \epsilon$, $\Pr[R^D(f(x), A(x, W)) = x] = 1/2$. Thus we have

$$\Pr[R^D(f(X), A(X, W)) = X] \geq \frac{1}{4q(n)},$$

where the probability is over X and W.

By Lemma 3.5.14,

$$\Pr_{a \leftarrow_{\mathrm{R}} A(X,W)}[H_\infty(X|A(X, W) = a) \geq k/3] \geq 1 - 2^{a+k/3-k} = 1 - 2^{-\Omega(k)}.$$

Also, by averaging, we have that

$$\Pr_{a \leftarrow_{\mathrm{R}} A(X,W)}[\Pr[R^D(f(X), A(X, W)) = X|A(X, W) = a] \geq \frac{1}{8q(n)}] \geq \frac{1}{8q(n)}.$$

Note $\frac{1}{8q(n)} \geq 2^{-\Omega(k)}$ since $k = n^{\Omega(1)}$. Thus, by a union bound, there is some fixing of $A(X, W) = a$ for which $H_\infty(X|A(X, W) = a) \geq k/3$ and $\Pr[R^D(f(X), A(X, W)) = X|A(X, W) = a] \geq \frac{1}{8q(n)}$. This contradicts the fact that f is one-way for $k/3$-sources. ■

We also need the following lemma.

Lemma 3.1.13. *Let X, Y be two independent random variables on $\{0,1\}^n$ and Z be a random variable on $\{0,1\}^m$ that is independent of (X, Y). Let $f : \{0,1\}^n \to \{0,1\}^d$ and $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be two deterministic functions. Let $R = f(X)$. If there exists a non-uniform adversary $\mathcal{A}$ that distinguishes between $(g(Y,R), R, X, Y)$ and (Z, R, X, Y) with probability ϵ, then there exists another non-uniform adversary $\mathcal{B}$ of size $2^d \cdot n \cdot Size(\mathcal{A})$ that distinguishes between $(g(Y,R), R, Y)$ and (Z, R, Y) with probability at least ϵ.*

Proof. Assume without loss of generality that

$$\Pr[\mathcal{A}(g(Y,R), R, X, Y) = 1] - \Pr[\mathcal{A}(Z, R, X, Y) = 1] \geq \epsilon.$$

Note that R is a deterministic function of X, thus for any fixing of $R = r$, $(g(Y,R), Z, Y)|(R = r)$ is independent of $X|(R = r)$. Therefore, for every fixing of $R = r$, there exists a fixing of $X|(R = r)$ and a non-uniform adversary $\mathcal{A}_r$, that has this fixing hardwired into it and emulates $\mathcal{A}$ w.r.t. this fixing, s.t.

$$\Pr[\mathcal{A}_r(g(Y,R), Y) = 1|R = r] - \Pr[\mathcal{A}_r(Z, Y) = 1|R = r] \geq$$
$$\Pr[\mathcal{A}(g(Y,R), R, X, Y) = 1|R = r] - \Pr[\mathcal{A}(Z, R, X, Y) = 1|R = r].$$

Let $\mathcal{B}$ be an adversary that on input $(g(Y,r), r, Y)$ emulates $\mathcal{A}_r(g(Y,r), Y)$. Then we have

$$\Pr[\mathcal{B}(g(Y,R), R, Y) = 1] - \Pr[\mathcal{B}(Z, R, Y) = 1] =$$
$$\sum_r \Pr[R = r] \left(\Pr[\mathcal{A}_r(g(Y,R), Y) = 1|R = r] - \Pr[\mathcal{A}_r(Z, Y) = 1|R = r]\right) \geq$$
$$\sum_r \Pr[R = r] \left(\Pr[\mathcal{A}(g(Y,R), R, X, Y) = 1|R = r] - \Pr[\mathcal{A}(Z, R, X, Y) = 1|R = r]\right) =$$
$$\Pr[\mathcal{A}(g(Y,R), R, X, Y) = 1] - \Pr[\mathcal{A}(Z, R, X, Y) = 1] \geq \epsilon.$$

Moreover, $\mathcal{B}$ is of size $2^d \cdot n \cdot Size(\mathcal{A})$. □

3.2 Primes in Arithmetic Progressions

Definition 3.2.1. Let $p(r, a)$ be the least prime in the arithmetic progression a modulo r.

We can now state a special case of a well-known conjecture.

Conjecture 3.2.2. *There exists a constant $c > 0$, such that for r a power of 2 and $a = 1$, one has $p(r, a) = O(r \log^c r)$.*

For the applications in this thesis, we don't really need r to be a power of 2; it would suffice if the conjecture held for integers r_n, where r_n is a smooth integer of about n bits computable in time polynomial in n. This conjecture is widely believed for $c = 2$, all r, and all a relatively prime to r. For more on this conjecture, see, for example, the discussion following equation (1) of [HB78]. The best unconditional conclusion is substantially weaker. Thus, one has $p(r, a) = O(r^{5.2})$ (see [Xyl09, HB92].)

3.3 Fourier Analysis

The following definitions from Fourier analysis are standard (see e.g., [Ter99]) , although we normalize differently than in many computer science papers, such as [Rao07a]. For functions f, g from a set S to $\mathbb{C}$, we define the inner product $\langle f, g\rangle = \sum_{x\in S} f(x)\overline{g(x)}$. Let D be a distribution on S, which we also view as a function from S to $\mathbb{R}$. Note that $\mathrm{E}_D[f(D)] = \langle f, D\rangle$. Now suppose we have functions $h : S \to T$ and $g : T \to \mathbb{C}$. Then

$$\langle g \circ h, D\rangle = \mathrm{E}_D[g(h(D))] = \langle g, h(D)\rangle.$$

Let G be a finite abelian group, and let ϕ a character of G, i.e., a homomorphism from G to $\mathbb{C}^\times$. We call the character that maps all elements to 1 the trivial character. Define the Fourier coefficient $\widehat{f}(\phi) = \langle f, \phi\rangle$. We let $\widehat{f}$ denote the vector with entries $\widehat{f}(\phi)$ for all ϕ. Note that for a distribution D, one has $\widehat{D}(\phi) = \mathrm{E}_D[\phi(D)]$.

Since the characters divided by $\sqrt{|G|}$ form an orthonormal basis, the inner product is preserved up to scale: $\langle \widehat{f}, \widehat{g}\rangle = |G|\langle f, g\rangle$. As a corollary, we obtain Parseval's equality:

$$\|\widehat{f}\|_{\ell^2}^2 = \langle \widehat{f}, \widehat{f}\rangle = |G|\langle f, f\rangle = |G|\|f\|_{\ell^2}^2.$$

Hence by Cauchy-Schwarz,

$$\|f\|_{\ell^1} \le \sqrt{|G|}\|f\|_{\ell^2} = \|\widehat{f}\|_{\ell^2} \le \sqrt{|G|}\|\widehat{f}\|_{\ell^\infty}. \tag{3.1}$$

For functions $f, g : S \to \mathbb{C}$, we define the function $(f, g) : S \times S \to \mathbb{C}$ by $(f, g)(x, y) = f(x)g(y)$. Thus, the characters of the group $G \times G$ are the functions (ϕ, ϕ'), where ϕ and ϕ' range over all characters of G. We abbreviate the Fourier coefficient $\widehat{(f,g)}((\phi,\phi'))$ by $\widehat{(f,g)}(\phi,\phi')$. Note that

$$\widehat{(f,g)}(\phi,\phi') = \sum_{(x,y)\in G\times G} f(x)g(y)\phi(x)\phi'(y) = \left(\sum_{x\in G} f(x)\phi(x)\right)\left(\sum_{y\in G} g(x)\phi'(x)\right) = \widehat{f}(\phi)\widehat{g}(\phi').$$

3.4 A Non-Uniform XOR Lemma.

We'll need the following extension of Vazirani's XOR lemma. We can't use traditional versions of the XOR lemma, because our output may not be uniform. Our statement and proof parallels Rao [Rao07a].

Lemma 3.4.1. *Let (W,W') be a random variable on $G\times G$ for a finite abelian group G, and suppose that for all characters ϕ,ϕ' on G with ϕ nontrivial, one has*

$$|\operatorname{E}_{(W,W')}[\phi(W)\phi'(W')]| \le \alpha.$$

Then the distribution of (W,W') is $\alpha|G|$ close to (U,W'), where U is the uniform distribution on G which is independent of W'. Moreover, for $f: G\times G\to\mathbb{R}$ defined as the difference of distributions $(W,W')-(U,W')$, we have $\|f\|_{\ell^\infty}\le\alpha$.

Proof. As implied in the lemma statement, the value of $f(a,b)$ is the probability assigned to (a,b) by the distribution of (W,W') minus that assigned by (U,W'). First observe that

$$\widehat{f}(\phi,\phi') = \langle f,(\phi,\phi')\rangle = \operatorname{E}_{(W,W')}[\phi(W)\phi'(W')] - \operatorname{E}_{(U,W')}[\phi(U)\phi'(W')].$$

Since U and W' are independent, this last term equals

$$\operatorname{E}_{(U,W')}[\phi(U)]\operatorname{E}_{(U,W')}[\phi'(W')] = \operatorname{E}_U[\phi(U)]\operatorname{E}_{W'}[\phi'(W')] = 0,$$

since ϕ is nontrivial. Therefore, by hypothesis, when ϕ is nontrivial, one finds that $|\widehat{f}(\phi,\phi')|\le\alpha$.

When ϕ is trivial, we get

$$\widehat{f}(\phi,\phi') = \operatorname{E}_{(W,W')}[\phi'(W')] - \operatorname{E}_{(U,W')}[\phi'(W')] = 0.$$

Hence $\|f\|_{\ell^1} \le \sqrt{|G\times G|}\|\widehat{f}\|_{\ell^\infty} \le |G|\alpha.$ □

3.5 Previous Works that We Use

We are going to use condensers recently constructed based on the sum-product theorem. The following construction is due to Zuckerman [Zuc07].

Construction 3.5.1. Let $F = \mathbb{F}_q$ be a field where $q = 2^p$ for p prime. Define the point-line incidence graph as the bipartite graph $G = (V, W, E)$ with vertices $V = F^2$ the set of points, and W the set of lines over F, and (p, l) is an edge in G iff p and l are incident. Let the function $h : E \to V \times W$ map an edge to its two endpoints. Equivalently, h is the map from F^3 to $(F^2)^2$ such that $h(a, b, c) = ((b, ab + c), (a, c))$.

The condenser $C : F^3 \times \{0, 1\} \to F^2$ is $C(e, i) = h(e)_i$.

The following theorem is proved in [Zuc07].

Theorem 3.5.2. *[Zuc07] Suppose $\delta < 0.9$ and $q^\delta = \omega(1)$. The function C above is a rate-$(\delta \to (1 + \alpha/2)\delta, \epsilon)$-somewhere-condenser, where $\epsilon = q^{-\alpha\delta/20}$ for some constant $\alpha > 0$.*

Note that each bit of the output of the condenser is a degree 2 polynomial of the bits of the input. Repeating the condenser for a constant number of times, we get the following theorem:

Theorem 3.5.3 ([BKS+05, Zuc07])**.** *For any constant $\beta, \delta > 0$, there is an efficient family of rate-$(\delta \to 1 - \beta, \epsilon = 2^{-\Omega(n)})$-somewhere condensers $\mathsf{Zuc} : \{0, 1\}^n \to (\{0, 1\}^m)^D$ where $D = O(1)$ and $m = \Omega(n)$.*

We now show that this condenser actually works even when the min-entropy of the source is very high. First we need the following improved theorem about line point incidences in finite fields.

Theorem 3.5.4 (Incidence Theorem)**.** *[Vin07] Let $F = \mathbb{F}_q$, where q is either prime or 2^p for p prime. Let P, L be sets of points and lines in F^2 and $|P|, |L| \le N = q^\alpha$ with $1 + \gamma \le \alpha \le 2 - \gamma$ for some $\gamma > 0$. Then the number of incidences*

$$I(P, L) \le 2N^{\frac{3}{2} - \frac{\gamma}{4}}.$$

The following lemma is from [Zuc07].

Lemma 3.5.5. *[Zuc07] If X, Y is not ϵ-close to a somewhere-k-source, then there exists sets $S \subseteq \mathsf{supp}(X), T \subseteq \mathsf{supp}(Y), |S|, |T| < 2^{k+1}/\epsilon$, such that*

$$\Pr[X \in S, Y \in T] > \epsilon/2.$$

Theorem 3.5.6. *Suppose $\frac{1}{2} < \delta \leq 1 - \gamma$ for some $\gamma > 0$. The function C above is a rate-$(\delta \to (1+\gamma/12)\delta, \epsilon)$ somewhere-condenser, where $\epsilon = q^{-\gamma\delta/20}$.*

Proof. We essentially follow the proof in [Zuc07]. As in that proof, we analyze the equivalent function h. We may assume that the input to h is uniform on a set of edges of size $K = 2^k = q^{3\delta}$, and set $k' = (1+\gamma/12)(2k/3)$. Suppose the output (X, Y) of h is not ϵ-close to a somewhere-k'-source. Let $P = S$ and $L = T$ be the sets of size less than $K_0 = 2^{k'+1}/\epsilon$ given by Lemma 3.5.5. Note that $K_0 \leq 2q^{2\delta(1+\gamma/12)+\delta(\gamma/20)} < q^{2-\gamma}$.

Now we calculate the number of incidences $I(P, L)$ in two ways. On the one hand, since each edge is an incident point-line pair, and at least $\epsilon/2$ fraction of these pairs lie in $P \times L$, the number of incidences $I(P, L) \geq \epsilon K/2$. On the other hand, by Theorem 3.5.4,

$$I(P, L) \leq 2K_0^{3/2-\gamma/4} = O(K^{(1+\gamma/12)(3/2-\gamma/4)2/3}/\epsilon^2) = O(K^{1-\gamma/12}/\epsilon^2).$$

This gives a contradiction for $\epsilon = K^{-\gamma/60}$, and the theorem is proved. ■

We need the following two source extractor from [Raz05].

Theorem 3.5.7 ([Raz05])**.** *For any n_1, n_2, k_1, k_2, m and any $0 < \delta < 1/2$ with*

- $n_1 \geq 6 \log n_1 + 2 \log n_2$
- $k_1 \geq (0.5 + \delta)n_1 + 3 \log n_1 + \log n_2$
- $k_2 \geq 5 \log(n_1 - k_1)$
- $m \leq \delta \min[n_1/8, k_2/40] - 1$

There is a polynomial time computable strong 2-source extractor $\mathsf{Raz} : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \{0,1\}^m$ *for min-entropy k_1, k_2 with error $2^{-1.5m}$.*

We need the following theorem from [Rao06].

Theorem 3.5.8 ([Rao06]). *For every constant $\gamma < 1$ and integers n, n', t s.t. $t < n^{\gamma}$ and $t < n'^{\gamma}$ there exists a constant $\alpha < 1$ and a polynomial time computable function $\mathsf{2SRExt} : \{0,1\}^{tn} \times \{0,1\}^{tn'} \to \{0,1\}^{m}$ s.t. if X is a $(t \times n)$ SR-source and Y is an independent aligned $(t \times n')$ SR-source,*

$$|(\mathsf{2SRExt}(X,Y), Y) - (U_m, Y)| < \epsilon$$

and

$$|(\mathsf{2SRExt}(X,Y), X) - (U_m, X)| < \epsilon,$$

where U_m is independent of X, Y, $m = min[n,n'] - min[n,n']^{\alpha}$ and $\epsilon = 2^{-min[n,n']^{\Omega(1)}}$.

Theorem 3.5.9 ([Rao06, BRSW06]). *There exist constants $c > 0$ and c' and a polynomial time computable function $\mathsf{IExt} : (\{0,1\}^{n})^{u} \to \{0,1\}^{k}$ such that for every n, k with $k = k(n) = \Omega(\log^4 n)$, if $X^1, X^2, \ldots, X^u$ are independent (n,k) sources then*

$$|\mathsf{IExt}(X^1, \ldots, X^u) - U_k| < 2^{-k^{c}}$$

Theorem 3.5.10 (Somewhere random vs Independent Source Extractor [Rao06, BRSW06]). *There exists constants $0 < \gamma < 1, c$ and a polynomial time computable function $\mathsf{SRIExt} : (\{0,1\}^{n})^{\mathsf{C}} \times \{0,1\}^{tk} \to \{0,1\}^{k}$ such that for every n, k, t with $k > \log^{10} t, k > \log^{10} n$ and $\mathsf{C} > c\frac{\log t}{\log k})$, if $X = X^1, ..., X^{\mathsf{C}}$ is the concatenation of C independent (n,k) source and Y is an independent $t \times k$-SR-source,*

$$|(X, \mathsf{SRIExt}(X,Y)) - (X, U_k)| < \epsilon$$

$$|(Y, \mathsf{SRIExt}(X,Y)) - (Y, U_k)| < \epsilon$$

where U_k is independent of X and Y, $\epsilon = 2^{-k^{\gamma}}$.

Theorem 3.5.11 (General Source vs Somewhere random source with few rows Extractor [BRSW06]). *There exist constants $\alpha, \beta < 1$ and a polynomial time computable function $\mathsf{BasicExt} : \{0,1\}^{n} \times \{0,1\}^{k^{\gamma+1}} \to \{0,1\}^{m}$ such that for every $n, k(n)$ with $k > \log^{10} n$ and constant $0 < \gamma < 1/2$, if X is an (n,k) source and Y is a $(k^{\gamma} \times k)$ $(k - k^{\beta})$-SR-source,*

$$|(Y, \mathsf{BasicExt}(X, Y)) - (Y, U_m)| < \epsilon$$

and

$$|(X, \mathsf{BasicExt}(X, Y)) - (XU_m)| < \epsilon$$

where U_m is independent of X, Y, $m = k - k^{\Omega(1)}$ and $\epsilon = 2^{-k^{\alpha}}$.

We use the following lossless condenser constructed in [GUV09].

Theorem 3.5.12 ([GUV09])**.** *For all constants $\alpha > 0$, and every $n \in \mathbb{N}$, $k \leq n$ and $\epsilon > 0$, there is an explicit $(k \to k + d, \epsilon)$ (lossless) condenser $\mathsf{Cond} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = (1 + 1/\alpha) \cdot (\log n + \log k + \log(1/\epsilon)) + O(1)$ and $m \leq 2d + (1 + \alpha)k$.*

We use the following strong seeded extractor in [GUV09].

Theorem 3.5.13 ([GUV09])**.** *For every constant $\alpha > 0$, and all positive integers n, k and $\epsilon > \exp(-n/2^{O(\log^* n)})$, there is an explicit construction of a strong (k, ϵ) extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log(1/\epsilon))$ and $m \geq (1 - \alpha)k$.*

We need the following simple lemma about statistical distance.

Lemma 3.5.14 ([MW97b])**.** *Let X and Y be random variables and let $\mathcal{Y}$ denote the range of Y. Then for all $\epsilon > 0$*

$$\Pr_Y \left[H_\infty(X|Y = y) \geq H_\infty(X) - \log|\mathcal{Y}| - \log\left(\frac{1}{\epsilon}\right) \right] \geq 1 - \epsilon$$

We need the following lemma about conditioning on the seed of a condenser.

Lemma 3.5.15. *Let $\mathsf{Cond} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a $(k \to l, \epsilon)$-condenser. For any (n, k)-source X, let R be the uniform distribution over d bits independent of X. With probability $1 - 2\sqrt{\epsilon}$ over the fixings of $R = r$, $\mathsf{Cond}(X, r)$ is $\sqrt{\epsilon}$-close to being an $l - 2d$ source.*

Proof. Let $W = \mathsf{Cond}(X, R)$. We know that W is ϵ close to having min-entropy l. Now for a fixed $R = r$, let $S_r = \{w \in \mathsf{Supp}(W) : \Pr[W = w|_{R=r}] > 2^{-l+2d}\}$. Note that if $\Pr[W = w|_{R=r}] > 2^{-l+2d}$ then $\Pr[W = w] \geq \Pr[W = w|_{R=r}] \Pr[R = r] > 2^{-l+d}$. Pick $\epsilon_1 > 0$ and let $\Pr_R[\Pr[W|_{R=r} \in S_r] > \epsilon_1] = \epsilon_2$, then $\Pr_W[\Pr[W = w] > 2^{-l+d}] > \epsilon_1\epsilon_2$. Thus the statistical distance between W and any l-source is at least $(1 - 2^{-d})\epsilon_1\epsilon_2 > \epsilon_1\epsilon_2/2$. Therefore $\epsilon_1\epsilon_2 < 2\epsilon$.

Therefore with probability $1 - 2\sqrt{\epsilon}$ over R, $\epsilon_1 < \sqrt{\epsilon}$. This implies that $W|_{R=r}$ is $\sqrt{\epsilon}$-close to having min-entropy $l - 2d$. □

Our extractor for affine sources use strong linear seeded extractors as ingredients. Specifically, we use the construction of Trevisan [Tre01] and the improvement by Raz et al. [RRV02].

Theorem 3.5.16 ([Tre01, RRV02])**.** *For every $n, k \in \mathbb{N}$ with $k < n$ and any $0 < \epsilon < 1$ there is an explicit (k, ϵ)-strong linear seeded extractor* $\mathsf{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{\Omega(k)}$ *with* $d = O(\log^2(n/\epsilon))$.

We need to use the following extractor for an affine somewhere random source.

Theorem 3.5.17 ([Rao09])**.** *For every constant $\gamma < 1$ and integers n, t s.t. $t < n^\gamma$ there exists a constant $\alpha < 1$ and a polynomial time computable function* $\mathsf{ASRExt} : \{0,1\}^{tn} \to \{0,1\}^{n-n^\alpha}$ *s.t. for every $(t \times n)$ affine-SR-source X,* $\mathsf{ASRExt}(X)$ *is $2^{-n^{\Omega(1)}}$-close to uniform.*

Lemma 3.5.18. *[ILL89][Leftover Hash Lemma] For any $0 < k < n$, let X be an (n, k)-source and R be the uniform distribution on $\{0,1\}^n$ independent of X. Let $l > 0$ and $m = k - 2l$. Treat x and r as elements in the field F_{2^n} and define the function* $\mathsf{Hash}(x, r)$ *to be the last m bits of $x \cdot r$. Then* $(\mathsf{Hash}(X, R), R)$ *is 2^{-l}-close to uniform.*

We are going to use a simple linear merger given in [LRVW03].

Construction 3.5.19. Let n, s be integers. Define the function

$$\mathsf{Merg} : (\{0,1\}^n)^s \times \{0,1\}^d \to \{0,1\}^n$$

in the following way: Let $\mathbb{F}_q$ be a finite field with q elements where q is a power of 2. Map each element in $\{0,1\}^n$ into $\mathbb{F}_q^\ell$ and each element in $\{0,1\}^d$ into $\mathbb{F}_q^s$, using some injective mapping. Let $x = (x_1, \cdots, x_s) \in (\mathbb{F}_q^\ell)^s$ and $z = (z_1, \cdots, z_s) \in \mathbb{F}_q^s$. The value of $\mathsf{Merg}(x, z)$ is computed as

$$\mathsf{Merg}(x, z) = \sum_{i=1}^{s} x_i \cdot z_i$$

where the operations are performed in the vector space $\mathbb{F}_q^\ell$.

The following theorem is proved in [LRVW03], by using a field of size $O(1/\epsilon)$ in the above construction.

Theorem 3.5.20. *[LRVW03] For every $\epsilon > 0$ and integers n, s, there exists an explicit (m, ϵ)-merger of (n, s)-somewhere-random sources* $\mathsf{Merg} : (\{0,1\}^n)^s \times \{0,1\}^d \to \{0,1\}^n$ *with $d = O(s \log(1/\epsilon))$ and $m = n/2 - O(d)$. Moreover, for any (n, s)-somewhere-random source X, with probability $1 - O(\epsilon)$ over $z \in \{0,1\}^d$,* $\mathsf{Merg}(X, z)$ *is ϵ-close to having min-entropy m.*

Theorem 3.5.21. *[Bou07] There is a polynomial time computable function* $\mathsf{BAExt} : \{0,1\}^n \rightarrow \{0,1\}^m$ *such that* $m = \Omega(n)$ *and for every affine source* X *of entropy* $n/2$*,* $\mathsf{BAExt}(X)$ *is* $2^{-\Omega(n)}$*-close to uniform. Moreover, each bit of the output is a degree 3 polynomial of the bits of the input.*

We need one last ingredient, the simple inner product function as a two source extractor when the sum of the entropy rates of the two independent sources is greater than 1. For a finite field $\mathbb{F}$, let $\mathsf{Had} : \mathbb{F}^l \times \mathbb{F}^l \rightarrow \mathbb{F}$ be the inner product function, i.e., $\mathsf{Had}(x, y) = x \cdot y$.

Theorem 3.5.22. *[CG88, Vaz85] For every constant* $\delta > 0$*, there exists a polynomial time algorithm* $\mathsf{Had} : (\{0,1\}^n)^2 \rightarrow \{0,1\}^m$ *such that if* X *is an* (n, k_1) *source,* Y *is an independent* (n, k_2) *source and* $k_1 + k_2 \geq (1 + \delta)n$*, then*

$$|(Y, \mathsf{Had}(X, Y)) - (Y, U_m)| < \epsilon$$

with $m = \Omega(n)$ *and* $\epsilon = 2^{-\Omega(n)}$*.*

To prove our construction is an extractor, we need the following definition and lemma.

Definition 3.5.23. (ϵ-biased space) A random variable Z over $\{0,1\}$ is ϵ-biased if $|\Pr[Z = 0] - \Pr[Z = 1]| \leq \epsilon$. A sequence of 0-1 random variables $Z_1, \cdots, Z_m$ is *ϵ-biased for linear tests* if for any nonempty set $S \subset \{1, \cdots, m\}$, the random variable $Z_S = \bigoplus_{i \in S} Z_i$ is ϵ-biased.

The following lemma is due to Vazirani. For a proof see for example [Gol95]

Lemma 3.5.24. *Let* $Z_1, \cdots, Z_m$ *be 0-1 random variables that are* ϵ*-biased for linear tests. Then, the distribution of* $(Z_1, \cdots, Z_m)$ *is* $\epsilon \cdot 2^{m/2}$*-close to uniform.*

Chapter 4

Distributed Computing with General Weak Random Sources

We first describe the model of the network that we consider. We assume that p processors communicate with each other via point-to-point channels in order to perform a task. However, an unknown t of the processors are *faulty.* We allow Byzantine faults: faulty processors may behave arbitrarily and even collude maliciously. We call the set of faulty processors the *adversary*, and we only consider a non-adaptive adversary – the set of faulty processors is fixed in advance and does not change. We assume that the communication channels are not private, so the adversary can see all communication. This is called the *full information model.* We note that we could obtain stronger results in a network with private channels, however in the interest of space we focus on the full information model.

Most of our results are for *synchronous* networks: communication between processors takes place in rounds and every message transmitted at the beginning of a round is guaranteed to reach its destination at the end of the round. In this case we allow rushing: the faulty processors may wait for all good processors to transmit their messages for a particular round, before transmitting their own messages. We also have results for *asynchronous* networks– here the only guarantee is that every message will eventually be received.

We assume each processor has access to an unknown, arbitrary (n, k)-source of randomness, and that these sources are mutually independent. This independence assumption seems justifiable if we view the processors as being physically far away from each other. Such sources may also arise if the adversary manages to acquire (say via a virus) a small amount of information about each of the honest processors' (truly random) sources. In this case, conditioning on the adversary's information leaves each of the processors with independent weak sources.

For the case of distributed computing, we mainly focus on two basic problems: Byzantine agreement and leader election/collective coin-flipping.

Byzantine Agreement. The goal of a protocol for *Byzantine agreement* is for the processors to agree on the result of some computation, even if some t of them are faulty. Byzantine agreement is fundamental because it can be used to simulate broadcast and maintain consistency of data.

Following the work of Ben-Or [BO83], a series of randomized protocols for asynchronous as well as synchronous networks appeared, some of which assume the existence of private communication channels between pairs of processors (for instance [Rab83]) while others do not require secret communication (for instance [CC85]). In the full information model, Goldwasser et al. gave an $O(\log p)$ round protocol which tolerates $t < p/3$ faulty processors [GPV06] in the synchronous model. In the asynchronous model, the best known Byzantine agreement protocol still requires $2^{O(t^2/p)}$ rounds [BO83, BT85] if it succeeds with probability 1 and requires polylog(p) rounds if it succeeds with probability $1 - o(1)$ [KKK+08].

Leader Election and Collective Coin Flipping. The goal of a protocol for *leader election* is to select a uniformly random leader from a distributed network of n processors. In the presence of faulty processors, we would like to bound the probability that one of the faulty processors gets selected as the leader. Another related problem, called the collective coin flipping, aims to produce a random coin whose bias is bounded in a network which may consists of faulty processors. under the assumption that the processors have access to uniformly random bits, collective coin flipping can be reduced to the leader election problem: if a leader is successfully elected, then we can have the leader flip a coin and broadcast the coin flip to all the other processors.

Ben-Or and Linial [BOL78] were the first to study collective coin-flipping under what we call the BL model: the full information model with reliable broadcast in a synchronous network. A long sequence of work [Sak89, AN93, BN00, CL95, ORV94, Zuc97, RZ01, Fei99] has resulted in a protocol which tolerates $(1/2 - \alpha)p$ faulty processors and requires only $\log^*(p) + O(1)$ rounds to elect a leader (and hence perform a collective coin flip) in the BL model [RZ01, Fei99].

4.1 Previous Results

For protocols using weak sources, the only results are due to Goldwasser et al. [GSV05]. They require all weak sources to have min-entropy rate at least 1/2. In the full information model, they only obtain results for weak sources that are more restricted than block sources[1]. Under the assumption that the processors have access to general (n, k)-sources, they give results only for the case of private channels. They posed the open question of whether protocols can be designed in the full information model assuming only that each processor has access to general (n, k)-sources.

[1]They refer to these sources as block sources, though they are not as general as block sources are defined in this paper and the extractor literature.

4.2 Our Results

We answered the above question in the affirmative by defining and constructing network extractors. As briefly mentioned in the introduction, these are protocols where the processors interact with each other, and at the end of the protocol some (ideally all) of the honest processors end up with private random bits that are close to uniform. In this chapter we define and construct such objects in the information-theoretic setting, and in the next chapter we study these objects in the computational setting. The results in this chapter are based on work with Yael Kalai, Anup Rao and David Zuckerman [KLRZ08].

In order to define network extractor, we need some notation. Processor i begins with a sample from a weak source $x_i \in \{0,1\}^n$ and ends in possession of a hopefully uniform sample $z_i \in \{0,1\}^m$. Let b be the concatenation of all the messages that were sent during the protocol. Capital letters such as Z_i and B denote these strings viewed as random variables.

Definition 4.2.1 (Network Extractor)**.** A protocol for p processors is a (t, g, ϵ) *network extractor* for min-entropy k if for any min-entropy k independent sources $X_1, \ldots, X_p$ over $\{0,1\}^n$ and any choice of t faulty processors, after running the protocol, the number of processors i for which $|(B, Z_i) - (B, U_m)| < \epsilon$ is at least g. Here U_m is the uniform distribution on m bits, independent of B, and the absolute value of the difference refers to variation distance. We say that a protocol is a *synchronous extractor* if it is a network extractor that operates over a synchronous network. We say that it is an *asynchronous extractor* if it is a network extractor that operates over an asynchronous network.

We now have the following results about network extractors.

As long as the min-entropy rate of the sources is greater than 1/2, we give nearly optimal network extractors. In particular, as long as the fraction of faulty processors t is bounded by a constant less than 1, we show how to build a one round synchronous network extractor which leaves almost every non-faulty processor with private randomness.

Theorem 4.2.2 (High Entropy Synchronous Extractor)**.** *For all $p, t, n, \alpha, \beta > 0$, there exists a constant $c = c(\alpha)$ and a two-round $(t, p - (1+\alpha)t - c, 2^{-k^{\Omega(1)}})$ synchronous extractor for min-entropy $k \geq (\frac{1}{2} + \beta)n$ in the full-information model. The protocol is one round for $t = \Omega(p)$.*

If the min-entropy of the general sources is much smaller, we can still design a good network extractor, though fewer processors end up with private random bits. The new protocol ensures roughly $p - (2 + \frac{\log\log n}{\log\log k})t$ honest processors end up with private randomness, thus tolerating a linear fraction of faulty processors. This protocol runs in a constant number of rounds even with min-entropy $k = 2^{(\log n)^{\Omega(1)}}$:

Theorem 4.2.3 (Low Entropy Synchronous Extractor)**.** *For all $p, t, \beta > 0$, $k > \log p$, and $n \leq 2^{O(t)}$, there exists a constant $c = c(\beta)$ and a $(1/\beta + 1)$ round $(t, p - (1.1 + 1/\beta)t - c, 2^{-k^{\Omega(1)}})$ synchronous extractor for min-entropy $k \geq 2^{\log^\beta n}$ in the full-information model.*

In the asynchronous setting, we get slightly weaker results:

Theorem 4.2.4 (High Entropy Asynchronous Extractor)**.** *For all $p, t, n, \beta > 0$, there exists a one-round $(t, p - 3t - 1, 2^{-k^{\Omega(1)}})$ asynchronous extractor for min-entropy $k \geq (\frac{1}{2} + \beta)n$ in the full-information model.*

Theorem 4.2.5 (Low Entropy Asynchronous Extractor)**.** *There exist constants $c_1, c_2 > 0$ such that for all $p, t, \beta > 0$, $k > \log p$, and $\mathrm{poly}(t) \leq n \leq 2^{O(t)}$, there exists a $(1/\beta + 1)$ round $(t, p - c_1 t/\beta - c_2, 2^{-k^{\Omega(1)}})$ asynchronous extractor for min-entropy $k \geq 2^{\log^\beta n}$ in the full-information model.*

Applying our network extractors to Byzantine Agreement and Leader Election, we obtain the following results.

In the synchronous setting, we essentially match the perfect-randomness case [GPV06] when the min-entropy rate is greater than 1/2, and we can tolerate a linear fraction of faults even with min-entropy $2^{(\log n)^{\Omega(1)}}$.

Theorem 4.2.6 (Synchronous Byzantine Agreement)**.** *Let $\alpha, \beta > 0$ be any constants. For p large enough, assuming each processor has access to an independent (n, k)-source, there exists synchronous $O(\log p)$ expected round protocols for Byzantine Agreement in the full information model with the following properties.*

1. *The protocol for $k \geq (1/2 + \beta)n$ tolerates $(1/3 - \alpha)p$ faulty processors.*
2. *The protocol for $k \geq n^\beta$ tolerates $(1/4 - \alpha)p$ faulty processors.*
3. *The protocol for $k \geq 2^{\log^\beta n}$ tolerates $p/(3.1 + 1/\beta)$ faulty processors.*

In the asynchronous case, we can tolerate a linear fraction of faults in only a polylogarithmic number of rounds, as is the case with perfect randomness [KKK+08].

Theorem 4.2.7 (Asynchronous Byzantine Agreement)**.** *Let $\alpha, \beta > 0$ be any constants. For p large enough, assuming each processor has access to an independent (n, k)-source, there exists a constant $0 < \gamma < 1$ and asynchronous* polylog(p) *expected round protocols for Byzantine Agreement in the full information model with the following properties.*

1. *The protocol for $k \geq (1/2 + \beta)n$ tolerates $(1/8 - \alpha)p$ faulty processors.*
2. *The protocol for $k \geq 2^{\log^{\beta} n}$ tolerates $\beta\gamma p$ faulty processors.*

We obtain essentially the same results if the processors' min-entropy rate is above 1/2, and we can tolerate a linear fraction of faults with min-entropy $2^{(\log n)^{\Omega(1)}}$.

Theorem 4.2.8 (Leader Election)**.** *Let $\alpha, \beta > 0$ be any constants. For p large enough, assuming each processor has access to an independent (n, k)-source, there exists $\log^* p + O(1)$ round protocols for leader election in the BL model with the following properties.*

1. *The protocol for $k \geq (1/2 + \beta)n$ tolerates $(1/2 - \alpha)p$ faulty processors.*
2. *The protocol for $k \geq n^{\beta}$ tolerates $(1/3 - \alpha)p$ faulty processors.*
3. *The protocol for $k \geq 2^{\log^{\beta} n}$ tolerates $(1/(2 + 1/\beta) - \alpha)p$ faulty processors.*

4.3 The Constructions

In this section we discuss how to build network extractors (Definition 4.2.1) in a full-information network.

4.3.1 Synchronous Network Extractors

4.3.1.1 First Attempts

To show how extractors for independent sources can be used to construct network extractors, we start with some simple protocols. We shall just sketch the arguments for why these protocols are good network extractors, reserving formal proofs for our best protocols.

Protocol 4.3.1. For a synchronous network

Player Inputs: Player i has $x_i \in \{0,1\}^n$
Player Outputs: Player i ends up with $z_i \in \{0,1\}^m$

Sub-Routines and Parameters:

- Let IExt be a 2 source extractor for entropy $0.1tk$ sources of length $1.1tn$.

We break up the players into two sets, $A = [1, 1.1t]$ and the rest of the players in B.

Communication in Round 1 : Every player $i \in A$ announces their string x_i.

Computation :

1. Let $y^j = x_1, \ldots, x_{1.1t}$ denote the concatenation of these strings received by j.
2. For every $j \in B$, j'th player computes $z_j = \mathsf{IExt}(y^j, x_j)$.

A first protocol one might think of is something along the lines of Protocol 4.3.1. The idea is to partition the players into two subsets A, B such that A is large enough to guarantee that at least one of the players in A is honest. We see that for every j, the distribution Y^j (note that Y^j may be different for different j, since the faulty players may transmit different strings to the non-faulty players) in the running of the above protocol has min-entropy at least $0.1tk$. Further, Y^j is independent of X_j for every $j \in B$. Thus Z_j is in fact ϵ-close to uniform and independent of Y^j by the properties of IExt. This means that every non-faulty player in the set B ends up with private randomness. By adjusting the constants, we can get a similar protocol that is a $(t, p - (2+\gamma)t, \epsilon)$ synchronous network extractor in the full-information model, as long as $t < p/(2+\gamma)$, for every constant $\gamma > 0$.

There are two problems with the above protocol. First, the best known polynomial time 2-source extractor constructions at the time of this writing [Bou05, Raz05] require that at least one of the sources has min-entropy rate close to half. This means we only get explicit protocols for such high entropy. Second, the above network extractor only guarantees that at most $p - 2t$ players get private randomness, while we hope that as many as $p - t$ players can get private randomness. We shall improve our results on both these fronts. In Protocol 4.3.2, we show how to use Theorem 3.5.11 to get results for low entropy.

Protocol 4.3.2. For a synchronous network

Player Inputs: Player i has $x_i \in \{0,1\}^n$
Player Outputs: Player i ends up with $z_i \in \{0,1\}^m$

Sub-Routines and Parameters:

- Let SRExt be as in Theorem 3.5.11 with parameters $n_1, m_1, \epsilon_1, k_1$.
- Let IExt be a C source extractor with parameters $n_2, k_2, m_2, \epsilon_2$ as in Theorem 3.5.9.
- We assume that $m_1^{0.9} \geq \binom{t+\mathsf{C}}{\mathsf{C}}$.

We break up the players into two sets, $A = [1, t+\mathsf{C}]$ and the rest of the players in B.

Communication in Round 1 : Every player $i \in A$ announces their string x_i.

Computation :

1. Let y^i be the $\binom{t+\mathsf{C}}{\mathsf{C}} \times m_1$ matrix whose j'th row is obtained by computing $y^i_j = \mathsf{IExt}(x^i_{i_1}, x^i_{i_2}, \ldots, x^i_{i_\mathsf{C}})$, where $x^i_1, \ldots, x^i_{\mathsf{C}+t}$ are the strings received by player i.
2. For every $j \in B$, player j computes $z_j = \mathsf{SRExt}(x_j, y^j)$.

Again, the analysis is quite simple. Since the set A contains $t + \mathsf{C}$ players, at least C of them must be non-faulty. Thus, after the first round, Y^j is ϵ_1 close to being a somewhere random source, for every $j \in B$. Thus, for every $j \in B$, Y^j independent of X_j and by the properties of SRExt, all non-faulty players in B get truly random bits. The above protocol is a $(t, p-2t, \epsilon_1+\epsilon_2)$ synchronous extractor in the full-information model, as long as $t < p/2$.

The drawback of this approach is that our extractor SRExt from Theorem 3.5.11 only works when the somewhere random source has much fewer rows than the length of each row. This protocol only succeeds in the case that the entropy of the sources is larger than $\binom{t+\mathsf{C}}{\mathsf{C}}$. On the other hand, this protocol does work for polynomially small entropy, since Theorem 3.5.11 and Theorem 3.5.9 can handle polynomially small entropy. In particular, this protocol works as long as $n, k \gg p$. We shall have to work harder to get a protocol that does not require this much entropy.

4.3.1.2 Protocol for Low Entropy

We have seen that in the case that the entropy is significantly larger than the number of players, or the entropy rate is larger than half, we have very simple network extractor protocols. In this section we describe better results for the case of low entropy sources. We shall start by describing a network extractor protocol that is good enough to get the following theorem:

Theorem 4.3.3 (Polynomial Entropy Synchronous Extractor)**.** *There exists a constant $c > 0$ such that for every $\gamma > \delta > 0$, $\beta > 0$ and p large enough, there exists a 1 round $(\delta p, (1-2\gamma)p, 2^{-k^c})$ synchronous extractor for min-entropy $k \geq n^\beta$ in the full-information model.*

Building on the ideas that go into proving the above theorem, we can give a $O(\log\log n/\log\log k)$ round protocol that works even when the entropy k is as small as $\log^{10} n$. This result appears in section 4.3.1.5.

Theorem 4.3.4. *If $k > \log p$ and $n \leq \exp(t)$ then for t large enough there exists a $(t, p - 2t - (\frac{1.1\log\log n}{\log\log k})t, 2^{-k^{\Omega(1)}})$ synchronous network extractor for min-entropy $k > \log^{10} n$ that runs in $O(\log\log n/\log\log k)$ rounds in the full-information model.*

Our protocol will be a variation on Protocol 4.3.2. Instead of trying *every* possible C-tuple of strings from the set A, we shall use a *derandomized* sample of these tuples.

We shall need the concept of an AND-*disperser*:

Definition 4.3.5 (AND-disperser)**.** An $(l, r, d, \delta, \gamma)$ AND-disperser is a bipartite graph with left vertex set $[l]$, right vertex set $[r]$, left degree d s.t. for every set $V \subset [r]$ with $|V| = \delta r$, there exists a set $U \subset [l]$ with $|U| \geq \gamma l$ whose neighborhood is contained in V.

Each vertex on the left identifies a d-tuple of vertices on the right. Thus when $l = \binom{r}{d}$, we can easily build an AND-disperser with great performance, just by considering every possible such tuple. We shall construct a much better AND disperser, i.e., one where l, r are much closer to each other.

In our application, we shall need a $(l, r, \mathsf{C}, \delta, \gamma)$ AND-disperser with l as small as possible, δ as small as possible and γ as large as possible. We shall prove the following lemma:

Lemma 4.3.6. *For every $\mathsf{C}, \delta > 0$, there exist constants $h, \gamma > 0$ and a polynomial time constructible family of $(hr, r, \mathsf{C}, \delta, \gamma)$ AND-dispersers.*

Before we see how to prove this lemma, we describe the rest of our construction.

Another well studied object that we need is a construction of a bipartite expander.

Definition 4.3.7 (Bipartite Expander)**.** A (l, r, d, β) bipartite expander is a bipartite graph with left vertex set $[l]$, right vertex set $[r]$, left degree d and the property that for any two sets $U \subset [l], |U| = \beta l$ and $V \subset [r], |V| = \beta r$, there is an edge from U to V.

Pippenger proved the following theorem:

Theorem 4.3.8 (Explicit Bipartite Expander [Pip87, LPS88])**.** *For every $\beta > 0$, there exists a constant $d(\beta) < O(1/\beta^2)$ and a family of polynomial time constructible $(l, l, d(\beta), \beta)$ bipartite expanders.*

We shall actually need unbalanced expanders, which can be easily obtained just by deleting vertices from the above graph. We get the following corollary:

Corollary 4.3.9. *For every $1 > \beta > 0$ and constant $h > 0$, there exists a constant $d(\beta, h)$ and a family of polynomial time constructible $(r, hr, d(\beta, h), \beta)$ bipartite expanders.*

We use these objects to design Protocol 4.3.10, which is the protocol in Theorem 4.3.3. We can show that Protocol 4.3.10 is a network extractor for entropy k.

Protocol 4.3.10. For a synchronous network

Player Inputs: Player i has $x_i \in \{0,1\}^n$
Player Outputs: Player i ends up with $z_i \in \{0,1\}^m$

Sub-Routines and Parameters:

- Let $1 > \gamma > \delta > 0$ be any constants.
- Let $\mathsf{SRExt}, n, m, \epsilon_1, k$ be an extractor with parameters as in Theorem 3.5.11. Let IExt be a C source extractor with parameters $n, k, m_2 = k, \epsilon_2$ as in Theorem 3.5.9.
- Set $r = \gamma p$.
- Let G_1, γ', h be such that there is a $(hr, r, \mathsf{C}, \frac{\gamma-\delta}{\gamma}, \gamma')$-$\mathsf{AND}$-disperser promised by Lemma 4.3.6.
- Set $\lambda = \min\{\gamma', \frac{\gamma-\delta}{1-\gamma}\}$.
- Let G_2 denote the $(p-r, hr, d, \lambda)$ bipartite expander given by Corollary 4.3.9.

We break up the players into two sets, $A = [1, r]$ and the rest of the players in B. We identify every player in A with a vertex in the right vertex set of the graph G_1 and identify every player in B with a vertex in the left vertex set of the graph G_2. We identify the left vertex set of G_1 with the right vertex set of G_2.

Communication in Round 1 : Every player $i \in A$ announces his string x_i.

Computation :

1. For every vertex g in the left vertex set of G_1, every remaining player j computes the string $y_g^j = \mathsf{IExt}(x_{g_1}^j, x_{g_2}^j, \ldots, x_{g_\mathsf{C}}^j)$, where here $x_{g_1}^j, x_{g_2}^j, \ldots, x_{g_\mathsf{C}}^j$ are the strings announced by the C neighbors of g.
2. Every player $j \in B$ computes the $d \times k$ matrix s^j whose w'th row is $y_{j_w}^j$, where here j_w is the w'th neighbor of j in G_2.
3. Every player $j \in B$ computes the private string $\mathsf{SRExt}(x_j, s^j)$.

Proof of Theorem 4.3.3. Let SRExt be as in Theorem 3.5.11, set up to extract from an $(n, k = n^\gamma)$ source and an independent $k^{0.9} \times k$ somewhere random source with error $2^{-k^{\Omega(1)}}$. Let $\mathsf{IExt}, \mathsf{C}$ be as in Theorem 3.5.9, set up to extract k random bits from C independent (n, k) sources with error

$2^{-k^{\Omega(1)}}$.

Let $X_1, \dots, X_p$ be any independent (n,k) sources. Since there are at most $t = \delta p$ faulty players in the set A, at least a $\frac{\gamma p - \delta p}{r} = \frac{\gamma - \delta}{\gamma}$ fraction of the strings x_i for $i \in A$ must be samples from an (n,k) source. Since G_1 is a $(hr, r, \mathsf{C}, \frac{\gamma-\delta}{\gamma}, \gamma')$ AND-disperser, we must have that at least a γ' fraction of the vertices g in the left vertex set of G_1 are such that Y_g is ϵ_2 close to uniform.

Now every non-faulty player $j \in B$ who has at least one such g as a neighbor, ends up with a distribution S^j that is ϵ_2 close to being a $d \times k$ somewhere random source. Let H denote the set of non-faulty players in B that don't get such a somewhere random source. Then we see that $|H| < \lambda(p-r) = \lambda(1-\gamma)p < (\gamma - \delta)p$, since G_2 is a $(p-r, hr, d, \lambda, \gamma'\})$ expander and by the definition of λ. Thus, all but $(\gamma-\delta)p + t = \gamma p$ of the players in B compute a somewhere random source. Then, by the properties of the extractor SRExt, each of these players computes a private random string with an additional error of ϵ_1. Since both of these errors are $2^{-k^{\Omega(1)}}$, we get that the final error is also $2^{-k^{\Omega(1)}}$. ∎

Next, we complete the proof by showing how to prove Lemma 4.3.6.

Proof of Lemma 4.3.6. We break up $[r]$ into equally sized disjoint sets $S_1, \dots, S_{\frac{\delta r}{2\mathsf{C}}}$, so that for every i, $|S_i| = 2\mathsf{C}/\delta$. Then consider all subsets $T \subset S_i$, with $|T| = \mathsf{C}$. The number of such subsets is $\binom{2\mathsf{C}/\delta}{\mathsf{C}} \frac{\delta r}{2\mathsf{C}} = hr$ for some constant h.

We define the bipartite graph with left vertex set $[hr]$, right vertex set $[r]$ and left degree C, by connecting every vertex on the left with the corresponding subset of elements of $[r]$. To see that this graph is an AND-disperser, let $V \subset [r]$ be any subset of density δ. Then, by averaging, we must have that V is at least $\delta/2$-dense in at least a $\delta/2$ fraction of the S_i's. But every S_i in which V is $\delta/2$ dense has at least $\frac{2\mathsf{C}}{\delta}\frac{\delta}{2} = \mathsf{C}$ elements of V. For every such S_i, there is a vertex in the left vertex set of the graph whose neighbors all lie in V.

Thus, there must be at least $\frac{\delta}{2}\frac{\delta r}{2\mathsf{C}} = \gamma hr$ such vertices. □

Protocol 4.3.10 addresses the issue of getting network extractors with low entropy (we can at least handle polynomially small entropy). However, it only guarantees that close to $p - 2t$ of the $p - t$ non-faulty players end up with useable randomness. We shall soon see that we cannot hope to give a one round protocol which does better than this, for low min-entropy.

4.3.1.3 Protocol for High Entropy Rate and Block Sources

Next we show that in the case that each player has access to a block source with just 2 blocks (Definition 2.3.4) or a source with entropy rate greater than half, we can give protocols that guarantee that almost all non-faulty players end up with useable randomness. The idea is that in this case, we can essentially run multiple copies of the above protocol at the same time. We partition the players into a constant number of sets. We can argue that most of the partitions must have a significant number of non-faulty players. We then run the previous protocol on every set in the partition.

Protocol 4.3.11. For a synchronous network

Player Inputs: Player i has $x_i, x'_i \in \{0,1\}^n$
Player Outputs: Player i ends up with $z_i \in \{0,1\}^m$

Sub-Routines and Parameters:

- Let $1 > \gamma > \delta > 0$ be any constants.
- Let $\mathsf{SRExt}, n, m, \epsilon_1, k$ be an extractor with parameters as in Theorem 3.5.11. Let IExt be a C source extractor with parameters $n, k, m_2 = k, \epsilon_2$ as in Theorem 3.5.9.
- Set $\alpha = (1-\delta)/2$. Set $r = \alpha p$.
- Let G_1, γ', h be such that there is a $(hr, r, \mathsf{C}, \frac{1-\delta}{1+\delta}, \gamma')$-AND-disperser promised by Lemma 4.3.6.
- Set $\lambda = \min\{\gamma', \gamma - \delta\}$.
- Let G_2 denote the $(p-r, hr, d, \lambda)$ bipartite expander given by Corollary 4.3.9.

We partition the players into $1/\alpha$ equally sized sets $B_1, \dots, B_{1/\alpha}$, each of size r. Let $A_1, \dots, A_{1/\alpha}$ denote the corresponding complements, i.e., $A_i = [p] \setminus B_i$.

Communication in Round 1 : Every player i announces x_i.

Computation :

1. For $i = 1, 2, \dots, 1/\alpha$,
   (a) We identify every player in A_i with a vertex in the right vertex set of the graph G_1 and identify every player in B_i with a vertex in the left vertex set of the graph G_2. We identify the left vertex set of G_1 with the right vertex set of G_2.
   (b) For every vertex g in the left vertex set of G_1, each player $j \in B_i$ compute the string $y_g^j = \mathsf{IExt}(x_{g1}^j, x_{g2}^j, \dots, x_{g\mathsf{C}}^j)$, where here $x_{g1}^j, x_{g2}^j, \dots, x_{g\mathsf{C}}^j$ are the strings received by j for the C neighbors of g.
   (c) Every player $j \in B_i$ computes the $d \times k$ matrix s^j whose w'th row is $y_{j_w}^j$, where here j_w is the w'th neighbor of j in G_2.
   (d) Every player $j \in B_i$ computes the private string $\mathsf{SRExt}(x_j^2, s^j)$.

We can prove the following theorem:

Theorem 4.3.12 (Polynomial Entropy Synchronous Extractor for Block Sources)**.** *There exists a constant $c > 0$ such that for every $\gamma > \delta > 0$, $\beta > 0$ and p large enough, there exists a 1 round $(\delta p, (1-\gamma)p, 2^{-k^c})$ synchronous extractor for (k, k) block sources with min-entropy $k \geq n^\beta$ in the full-information model.*

Proof. We shall analyze Protocol 4.3.11. Let SRExt be as in Theorem 3.5.11, set up to extract from an $(n, k = n^\gamma)$ source and an independent $k^{0.9} \times k$ somewhere random source with error $2^{-k^{\Omega(1)}}$. Let $\mathsf{IExt}, \mathsf{C}$ be as in Theorem 3.5.9, set up to extract k random bits from C independent (n, k) sources with error $2^{-k^{\Omega(1)}}$.

Let $X_1, \ldots, X_p$ be any independent (n, k) sources. Note that for every i, there are at least $(1-\alpha-\delta)p = (1-\delta)p/2$ non-faulty players in the set A_i. This is at least a $(1-\alpha-\delta)/(1-\alpha) = (1-\delta)/(1+\delta)$ fraction of the number of players in this set.

By the properties of G_1 and G_2, this means that at most a λ fraction of the players in each of the B_i's wouldn't compute strings that are close to uniformly random, if each of them computed these strings correctly. However, a δ fraction of the players are faulty. Thus we get that at least $1-\lambda-\delta \geq 1-\gamma$ fraction of the players end up with randomness that is $\epsilon_1+\epsilon_2$ close to uniform. ■

A special case of this above protocol is when the players all have access to a source with min-entropy rate greater than half. In this case, we can show that the players can easily get a block source, just by splitting their sources into two equal parts. This gives us the following theorem:

Theorem 4.3.13 (High Entropy Synchronous Extractor)**.** *There exists a constant $c > 0$ such that for every $\gamma > \delta > 0$, constant $\beta > 0$ and p large enough, there exists a 1 round $(\delta p, (1-\gamma)p, 2^{-k^c} + 2^{-c\beta n}p)$ synchronous extractor for min-entropy $k \geq (\frac{1}{2}+\beta)n$ in the full-information model.*

Proof. Let X be any $(n, (1/2+\beta)n)$ source. Let X_1 be the first $n/2$ bits of X and X_2 be the remaining bits.

Then we have that:

Claim 4.3.14. *X_1, X_2 is $2^{-\Omega(\beta n)}$ close to being a block source with min-entropy $3\beta n/5$ in each block.*

To see this, first observe that by Lemma 2.3.2 (setting $l = \beta n/10$), we get that X_1 is $2^{-\beta n/10}$ close to having min-entropy $(1/2+\beta)n - n/2 - \beta n/10 = 9\beta n/10$. Then, by Lemma 2.3.5, setting

$\ell = \beta n/10$, we get that X_1, X_2 is $2(2^{-\beta n/10} + 2^{-\beta n/10+1})$-close to being a block source with min-entropy $9\beta n/10 - 1 - 2\beta n/10 \geq 6\beta n/10$ in the first block and $(1/2+\beta)n - n/2 - 1 - 2\beta n/10 \geq 3\beta n/5$ in the second block.

Thus, all of the sources are simultaneously $2^{-\Omega(\beta n)}p$-close to being block sources. We can now run Protocol 4.3.11 to get random bits. ■

4.3.1.4 Lower bounds

In this section, we show that there is no one round network extractor protocol that can do much better than our construction for the case of general sources with $k = n^{\delta}$ over synchronous networks in the full information model. Namely, for general weak random source with min-entropy rate $< \frac{1}{2}$, there is no one round network extractor that can tolerate $p/2$ faulty players or guarantee $p - 2t$ honest players end up with private random bits.

Theorem 4.3.15. *There is no one round $(t, p-2t, 1/4)$ synchronous extractor protocol for general $(n, n/2-1)$ sources, in the full information model.*

Proof. For the purpose of contradiction, let us assume that such a protocol exists for min-entropy $k < n/2$.

This protocol must call for some number of players to transmit messages in the first round of the protocol. Let us assume that each player starts with strings $x_i \in \{0,1\}^n$ and that in the first round player i transmits some function $f_i(x_i)$ of the input, where $f_i : \{0,1\}^n \to \{0,1\}^{m_i}$.

We say that i *transmits k bits* if the size of the image $|f_i(\{0,1\}^n)| \geq 2^k$.

We note that if i does not transmit k bits, then there must be some point $a \in \{0,1\}^{m_i}$ such that $|f_i^{-1}(a)| \geq 2^{n-k} \geq 2^k$. Setting X_i to be the flat distribution over $f_i^{-1}(a)$, we get a source X_i with min-entropy at least k s.t. $f_i(X_i)$ is a constant.

On the other hand, if i transmits k bits, then we pick 2^k points $\{x^1, \ldots, x^{2^k}\}$ such that f_i is injective on this set. If we set X_i to be the flat distribution on this set, we get a source with min-entropy k for which for every $a \in \mathsf{supp}(f_i(X_i))$, $H_\infty(X_i | f_i(X_i) = a) = 0$, i.e. the source has no entropy left over after conditioning on the output of f_i.

There are now two cases:

At most t players transmit $n/2$ bits. In this case, by our discussion above, the adversary can replace every player that transmits at least $n/2$ bits with a faulty player and choose min-entropy $n/2$ weak sources X_i for every other player, in such a way that the transcript of the first round transmissions is a constant. The private random string that player i generates is then just a deterministic function of X_i. We can then find a deficiency 1 subsource $X_i' \subset X_i$ such that the first bit of this private string is constant. Note that X_i' has min-entropy at least $n/2-1$, which means the protocol must fail in this case.

More than t players transmit $n/2$ bits. In this case, by our discussion above, for each player i that transmits $n/2$ bits, we can pick a k-source X_i such that the entropy of the source conditioned on the first round transcript is 0. Thus every such player cannot generate any private randomness. We pick some other t players to be faulty. Thus at most $p-2t-1$ players will end up with private randomness.

■

4.3.1.5 Protocol for Even Smaller Min-Entropy

Now we prove the general theorem of synchronous extractor, Theorem 4.3.4. In fact we prove a stronger version, which gives a generic way to transform an independent source extractor that needs C sources into a network extractor protocol which runs in roughly $\frac{\log \mathsf{C}}{\log\log k}+3$ rounds and ensures $p-(3+\frac{\log \mathsf{C}}{\log\log k})t$ honest players end up with private random bits.

We relax the requirement that C is a constant in Assumption 4.3.16, and only require $\mathsf{C}=o(\log n)$:

Assumption 4.3.16. We assume we have access to a strong C-Source extractor $\mathsf{IExt} : (\{0,1\}^n)^{\mathsf{C}} \to \{0,1\}^k$ with error ϵ for (n,k) sources, where $\mathsf{C}=o(\log n)$. Throughout this section we reserve C for the number of sources that IExt needs to function.

Remark 4.3.17. Theorem 3.5.9 gives such an extractor with $\mathsf{C}=o(\log n)$.

First we shall construct a more sophisticated AND-disperser.

Towards this we need the following theorem:

Theorem 4.3.18. *[AFWZ95] Let H be a d-regular graph on n vertices and A be the probability transition matrix of the random walk on H. Let $1=\lambda_0>\lambda_1\geq \ldots \geq \lambda_{n-1}$ be the eigenvalues of A. Let W be a set of w vertices in H and put $\mu=w/n$. Let τ denote the fraction of random walks of length $D-1$ that stay in W. Assume (for the lower bound only) that k is odd and that $\mu+\lambda_{n-1}(1-\mu)\geq 0$. Then*

$$\mu(\mu + \lambda_{n-1}(1-\mu))^{k-1} \leq \tau \leq \mu(\mu + \lambda_1(1-\mu))^{k-1}.$$

Lemma 4.3.19 (AND-disperser)**.** *There exists a constant $c > 0$ such that if $D = o(\log M)$ then for every constant $0 < \alpha < 1$ and large enough M, there exists an explicit construction of an (N, M, D, α, β) AND-disperser G such that $M < N \leq Md^D$ and $\beta > \mu^D$. Here $d = c\alpha^{-8}, \mu = \alpha^2/3$.*

Proof. We use random walks on expander graphs to construct the AND-disperser. Take any d_0-regular expander graph G_0 on M vertices, let $1 = \lambda_0 > \lambda_1 \geq ... \geq \lambda_{M-1}$ be the eigenvalues of the probability transition matrix of the random walk on G_0. For any subset of vertices W with $|W| = \alpha M$, let τ denote the fraction of random walks of length $D-1$ that stay in W. If $\mu_0 = \alpha + \lambda_{M-1}(1-\alpha) > 0$, then Theorem 4.3.18 gives a lower bound of τ as $\tau \geq \alpha\mu_0^{D-1} \geq \mu_1^D$, where $\mu_1 = min\{\alpha, \mu_0\}$. Take all the walks of length $D-1$ as the vertices in $[N]$ and have each vertex in $[N]$ connect to the D vertices in $[M]$ that are in the corresponding walk, we get an AND-disperser as desired.

The problem with the above construction is that some walks may have repeated vertices, thus some vertices in $[N]$ may have degree less than D. On the other hand we need every vertex in $[N]$ to have degree exactly D(Think of $D = \mathsf{C}$ and we have to use C independent sources for the extractor).

To deal with this problem, we make a slight modification. Instead of using walks of length $D-1$, we take all the walks of length $l = 2D-1$ on G_0. Among these walks, we delete all the walks that have more than D repeated vertices. The remaining walks then have at least D distinct vertices. We then take all the remaining walks as vertices of $[N]$ and for each vertex in $[N]$, connect it to the first D distinct vertices in $[M]$ that are in the corresponding walk. We bound the number of walks which have more than D repeated vertices to show that there are still enough walks left. This then gives us the desired AND-disperser. Note as $D = o(\log M)$ the AND-disperser can be constructed in polynomial time.

We use a special family of Ramanujan graphs X on M vertices as constructed in [LPS88]. The graph X is regular with degree $d_0 = p+1$, p prime and has large girth: $g(X) = \Omega(\log |X|) = \Omega(\log M)$.

For a walk $(w_0, .., w_l)$, call w_i a *repeat* if there exists a $j < i$ such that $w_j = w_i$, and *new* otherwise. Note $l = 2D - 1 = o(\log M) = o(g(X))$. Thus for sufficiently large M, $l < g(X)$ and the only possible way to get repeats is to backtrack in the walk.

Define a "backtracking pair" in a walk to be a sub-walk of length $2m-2$ for some m such that the order of the vertices in the sub-walk is $v_1, v_2, ..., v_{m-1}, v_m, v_{m-1}, ..., v_2, v_1$. Now consider any walk with more than D repeats. If we remove all the backtracking pairs in it, it becomes a walk without repeats and the new walk will have length $y \le D-1$. For a fixed y, the length of all the backtracking pairs is $l-y$. Thus there are $\frac{l-y}{2}$ backtracking steps and at most $\binom{l}{\frac{l-y}{2}}$ choices for the positions of these steps. For each such choice there are at most $Md_0^{\frac{l+y}{2}}$ choices for the other $\frac{l+y}{2}$ steps. Therefore the total number of walks with more than D repeats is at most

$$\begin{aligned} P &\le \sum_{y=0}^{D-1} \binom{l}{\frac{l-y}{2}} M d_0^{\frac{l+y}{2}} \\ &\le \sum_{y=0}^{D-1} (\frac{2el}{l-y})^{\frac{l-y}{2}} M d_0^{\frac{l+y}{2}} \le \sum_{y=0}^{D-1} (4e)^{\frac{l-y}{2}} M d_0^{\frac{l+y}{2}} \\ &< M \sum_{y=0}^{D-1} (4ed_0)^{\frac{l+y}{2}} \le MD(4ed_0)^{\frac{3D}{2}-1} \end{aligned}$$

Thus the fraction of walks with more than D repeats is at most

$$\eta = MD(4ed_0)^{\frac{3D}{2}-1} / Md_0^{2D-1} < D\left(\frac{(4e)^3}{d_0}\right)^{\frac{D}{2}}$$

While by Theorem 4.3.18 the fraction of walks that stay in W is at least $\tau \ge \mu_1^{2D} = (\mu_1^4)^{\frac{D}{2}}$, where $\mu_1 = min\{\alpha, \alpha + \lambda_{M-1}(1-\alpha)\}$. By the property of the expander X, $|\lambda_{M-1}| \le \frac{2\sqrt{d_0-1}}{d_0}$ [LPS88]. Let $d_0 = c_1 \alpha^{-4}$ for a sufficiently large global constant $c_1 > 0$, we have

$$\alpha + \lambda_{M-1}(1-\alpha) > \alpha - \frac{2\alpha^2}{\sqrt{c_1}} > 0.9\alpha.$$

Thus $\mu > 0.9\alpha$ and

$$\tau \ge (\mu_1^4)^{\frac{D}{2}} > (0.9^4\alpha^4)^{\frac{D}{2}} = \left(\frac{0.9^4 c_1}{(4e)^3} \cdot \frac{(4e)^3}{d_0}\right)^{\frac{D}{2}} > 2D\left(\frac{(4e)^3}{d_0}\right)^{\frac{D}{2}} = 2\eta.$$

Therefore the fraction of random walks that stay in W and have at least D distinct vertices is at least

$$\beta = \tau - \eta > \frac{1}{2}\tau \geq \frac{1}{2}\mu_1^{2D} > (\alpha^2/3)^D = \mu^D.$$

Note $N < Md_0^{2D-1} \leq Md^D$ with $d = d_0^2 \leq c\alpha^{-8}$ for a global constant $c > 0$, and $N > Md_0^{2D-1} - MD(4ed_0)^{\frac{3D}{2}-1} > M$.

□

The second ingredient we'll use is a family of "m-expanding" graphs.

Definition 4.3.20. [TUZ01] An undirected graph is m-expanding if every two disjoint sets of vertices of size at least m are joined by an edge.

In [TUZ01] almost optimal parameters for explicit m-expanding graphs are achieved:

Theorem 4.3.21. *[TUZ01] For any $N, m > 0$ there exists an explicit m-expanding d-regular graph on N vertices with $d = O(\frac{N}{m}\text{polylog}N)$*

Notice that in the case $\frac{N}{m}$ is a constant, it suffices to use a constant degree expander as in Theorem 4.3.8. In our protocols we'll often view these graphs as bipartite graphs, with N vertices on both sides.

The last ingredient we need is the extractor for somewhere random source and independent sources in Theorem 3.5.10. This theorem says that it suffices to take a $r \times k$ somewhere random source and another $O(\frac{\log r}{\log k})$ independent (n, k) sources to extract almost uniform random bits.

4.3.1.6 High-Level Ideas of the synchronous extractor

We now outline the high-level ideas of the synchronous extractor, Protocol 4.3.24.

Idea 1 : The synchronous extractor is a multiple roundprotocol. Each round is a generalization of Protocol 4.3.10. The difference is that the number of independent sources needed may no longer be a constant. Thus we use the more sophisticated AND-disperser as in Lemma 4.3.19. Note that the reason we need a protocol like Protocol 4.3.10 is that we need more than one independent sources to extract random bits (If there is a one-source extractor, then we don't need such protocols).

Idea 2 : By the end of the first round, all but a small fraction of the receivers obtain a somewhere random source. If the number of rows in the SR-source is small, then by Theorem 3.5.11 a player can extract random bits using the SR-source and his own weak source. This is just Protocol 4.3.10. If the number of rows in the SR-source is not small enough, then we need more than one independent sources together with the SR-source to extract random bits. Thus we can run the one-round protocol again. The key observation is that the number of independent sources needed will decrease in each round.

Idea 3 : Assume at the beginning of round l, the number of independent sources needed (except the SR-source) is C_l. Then, the AND-disperser gives a set of C_l-tuples of which a fraction of roughly $1/2^{\Theta(\mathsf{C}_l)}$ is "good" (consisting of all honest players). The m-expanding graph of Theorem 4.3.21 gives a bipartite graph with degree roughly $O(2^{\Theta(\mathsf{C}_1)})$ (except for a poly-logarithmic factor, which won't affect our result much). The property of the m-expanding graph guarantees that in each round all but a small fraction of the receivers obtain an SR-source.

Idea 4 : The degree of the m-expanding graph is exactly the number of rows in the SR-source obtained by the end of round l. Thus by Theorem 3.5.10 the number of independent sources we need now is roughly $\mathsf{C}_{l+1} = O(\frac{\log 2^{\Theta(\mathsf{C}_l)}}{\log k}) = O(\frac{\mathsf{C}_l}{\log k})$, which says the number of independent sources needed will decrease by a factor of roughly $\log k$ in each round. We then iterate until this number decreases to 1, at which time a player can extract random bits that are close to uniform using the SR-source and his own weak source. This will take roughly $\frac{\log \mathsf{C}}{\log \log k}$ rounds.

4.3.1.7 Synchronous network extractor protocol

Now we describe our protocol for synchronous extractor. First we need a sub protocol, Protocol 4.3.22, a generalization of Protocol 4.3.10, for one round in the whole protocol.

Protocol 4.3.22. For a synchronous one round network

Player Inputs: There are two sets of players, A and B. Every player $u_i \in A$ has a weak random string $x_i \in \{0,1\}^n$ and an optional $r_1 \times k$ somewhere random string $y_i \in \{0,1\}^{r_1 k}$
Player Outputs: For every player $u_j \in B$, u_j ends up with a supposed $r_2 \times k$ somewhere random string $y_j \in \{0,1\}^{r_2 k}$

Sub-Routines and Parameters:

- Let IExt be a C source extractor as in Assumption 4.3.16. Let SRIExt be the Somewhere Random source vs. independent source extractor as in Theorem 3.5.10.
- $|A| = (1+\alpha)t$ for some given constant $\alpha > 0$.
- If this is the first round of the whole protocol, let $D = \mathsf{C}$ be the number of (n,k) sources IExt needs. Otherwise let $D = O(\frac{\log r_1}{\log k})$ be the number of independent sources SRIExt needs, when the somewhere random source has r_1 rows. Construct an $(N, M = |A|, D, \alpha_1 = \frac{0.9\alpha}{1+\alpha}, \beta_1)$ AND-disperser G with $N \leq M d_1^D, \beta_1 \geq \mu_1^D$ as in Lemma 4.3.19.
- If D is a constant, let $m = min\{\mu_1^D N, 0.1\alpha t\}$ and construct the bipartite expander H on $2N$ vertices promised by Theorem 4.3.8 with degree $d_2 = O((N/m)^2)$. Otherwise let $m = min\{\mu_1^D N, \frac{|A|}{2^D}\}$ and construct an m-expanding graph H on N vertices promised by Theorem 4.3.21 with degree $d_2 = O(\frac{N}{m}\text{polylog}N)$. View H as a bipartite graph with N vertices on each side. Identify each player in A with a vertex in $[M]$ and identify each player in B with a vertex in the left vertex set of H. Identify $[N]$ with the right vertex set of H.

Round 1 :

1. Every player $u_i \in A$ sends his random string x_i and his somewhere random string y_i(if u_i has such a string) to all the players in B.
2. For every player $v_j \in B$, his corresponding vertex in H has d_2 neighbors in the right vertex set of H: $w_{j1}, ..., w_{jd_2}$. Each of these neighbors w_{jq} in turn has D neighbors in $[M]$. Let the D neighbors be $u_{q1}, ..., u_{qD}$ and without loss of generality assume $q1 < q2 < ... < qD$.
    - If this is the first round of the whole protocol, compute $s_{jq} = \mathsf{IExt}(x_{q1}, x_{q2}, ..., x_{qD})$. Otherwise compute $s_{jq} = \mathsf{SRIExt}(x_{q1}, x_{q2}, ..., x_{qD}, y_{q1})$.
    - Compute s_{jq} for every neighbor w_{jq} and form a $d_2 \times k$ somewhere random string $y_j = s_{j1} \circ ... \circ s_{jd_2}$

    .

We also need the following sub protocol, Protocol 4.3.23, which guarantees that if at least $3t$ honest players already get private random bits, then in the next round all the other honest players who haven't announced their strings will get private random bits.

Protocol 4.3.23. For a synchronous one round network

Player Inputs: There are two sets of players, A and B. Every player $u_i \in A$ has private random bits $z_i \in \{0,1\}^k$
Player Outputs: For every honest player $u_j \in B$, u_j ends up with private random bits $z_j \in \{0,1\}^m$, while every honest player $u_i \in A$ still has $0.9k$ private random bits left.

Sub-Routines and Parameters:

- Let Raz be the strong 2-sources extractor in Theorem 3.5.7.
- $|A| > 3t$.

Round 1 :

1. Every player $u_i \in A$ takes 0.1 fraction of his private random bits z_i, let the fraction be y_i and sends y_i to all the players in B.
2. For every player $u_j \in B$, let s_j be the concatenation of all the y_i received from the players in A. Compute $z_j = \mathsf{Raz}(s_j, x_j)$.

The synchronous extractor is now described as Protocol 4.3.24.

Protocol 4.3.24. For a synchronous network

Player Inputs: Every player u_i has a weak random string $x_i \in \{0,1\}^n$
Player Outputs: For some players u_j, u_j ends up with private random bits $z_j \in \{0,1\}^m$

Sub-Routines and Parameters:

- Protocol 4.3.22 and Protocol 4.3.23.
- Let $\mathsf{BasicExt}$ be the extractor in Theorem 3.5.11.
- $\alpha > 0$ is a given constant.
- R is given as the number of rounds of the protocol.

Divide p players into $R+1$ disjoint sets $A_1, ..., A_{R+1}$, where $|A_1| = |A_2| = ... = |A_{R-1}| = (1+\alpha)t$ and $|A_R| = min\{p-(1+\alpha)(R-1)t, 3(1+\alpha)t\}$. A_{R+1} is the set of remaining players. It's possible that $A_{R+1} = \Phi$.

Round l, $l = 1, ..., R-1$: Run Protocol 4.3.22 with $A = A_l$, $B = A_{l+1}$ and parameter α. In round 1, players in A don't have the optional somewhere random strings y_i. In subsequent rounds, players in A have the somewhere random strings y_i obtained by the end of the previous round.

Round $R-1$: At the end of round $R-1$, player u_j in A_R computes $z_j = \mathsf{BasicExt}(x_j, y_j)$. Here x_j is u_j's weak random string, y_j is the somewhere random string obtained by the end of this round.

Round R(Last Round) : If $A_{R+1} \neq \Phi$, run Protocol 4.3.23 with $A = A_R$, $B = A_{R+1}$.

4.3.1.8 Proof of the theorem

We prove the following stronger theorem:

Theorem 4.3.25. *If $k > \log p$ and $n \leq \mathrm{poly}(p)$ then as long as $t > p^{0.1}$, for sufficiently large p there exists a $(t, p - 1.1(3 + \frac{\log \mathsf{C}}{\log\log k})t, \epsilon + 2^{-k^{\Omega(1)}})$ synchronous extractor that runs in at most $\frac{1.1 \log \mathsf{C}}{\log\log k} + 3$ rounds in the full-information model. Here ϵ is the error of the extractor in Assumption 4.3.16.*

Remark 4.3.26. The parameter $p^{0.1}$ can be replaced by p^{δ} for any constant $\delta > 0$ and the constant 1.1 can be replaced by $1 + \alpha$ for any constant $\alpha > 0$.

In all the analysis below we call a supposed somewhere random string y "valid" if it contains at least one row that is ϵ-close to being truly random for some small error ϵ.

To analyze Protocol 4.3.24 we first establish the following lemma:

Lemma 4.3.27. *Let C_l be the number of independent (n, k) sources needed for* IExt *or* SRIExt *in round l of the synchronous extractor. Define Bad_l as the number of honest players in A_l that don't have a valid somewhere random string for $l \geq 2$ and let $Bad_1 = 0$. There exist constants $c_1 > 0, c_2 > 0$ such that for every constant $\alpha > 0$, the following claims hold.*

1. *$\forall l, Bad_l \leq 0.1\alpha t$ and $\mathsf{C}_l = o(\log n)$*
2. *If $\mathsf{C}_l > \log k$, then*
$$\mathsf{C}_{l+1} \leq \frac{c_1 \log(c_2/\alpha)}{\log k} \mathsf{C}_l$$
3. *If $\mathsf{C}_l \leq \log k$ and $\mathsf{C}_l = \omega(1)$, then C_{l+1} is a constant $C(\alpha)$*
4. *If C_l is a constant, then by the end of round l, all but Bad_l honest players in A_l obtain random bits that are $\epsilon + l2^{-k^{\Omega(1)}}$-close to uniform and independent of the whole transcript.*

Proof. We prove 1 by induction. For $l = 1$, the number of independent (n, k) sources needed for IExt is $\mathsf{C}_1 = \mathsf{C} = o(\log n)$ by Assumption 4.3.16. Also $Bad_1 = 0 < 0.1\alpha t$.

Now assume for round l, $\mathsf{C}_l = o(\log n)$ and $Bad_l \leq 0.1\alpha t$. Note $\mathsf{C}_l = o(\log n) = o(\log t)$ as $n = O(\text{poly}(p))$ and $t > p^{0.1}$. Let U_l denote the subset of honest players in A_l that have a valid somewhere random source for $l \geq 2$, and the subset of honest players in A_l for $l = 1$, then $|U_l| \geq (1+\alpha)t - t - Bad_l \geq 0.9\alpha t$. Thus $\frac{|U_l|}{|A_l|} \geq \frac{0.9\alpha}{1+\alpha}$. Consider Protocol 4.3.22 for round l. An AND-disperser $G = (N, M, D, \alpha_1, \beta_1)$ is constructed with $M = |A_l| = (1+\alpha)t, D = \mathsf{C}_l = o(\log t), \alpha_1 = \frac{0.9\alpha}{1+\alpha}$. By Lemma 4.3.19, $N \leq M d_1^D = M d_1^{\mathsf{C}_l}$ for some constant $d_1 \leq c_1' \alpha_1^{-8}$, and there exists a subset $V \subset [N]$ with $|V| = \beta_1 N > \mu_1^D N$ s.t. $\Gamma(V) \subset U_l$. V is the set of "good" tuples that consist of all honest players.

As $D = o(\log t)$, $\frac{|A_l|}{2^D} > t^{0.9}$ and $\mu_1^D N > \mu_1^D M > t^{0.9}$. Thus $min\{\mu_1^D N, \frac{|A_l|}{2^D}\} > t^{0.9}$ and $min\{\mu_1^D N, 0.1\alpha t\} > t^{0.9}$. Also, if D is super-constant, then $m = min\{\mu_1^D N, \frac{|A_l|}{2^D}\} \leq \frac{|A_l|}{2^D} < 0.1\alpha t$. Otherwise $m = min\{\mu_1^D N, 0.1\alpha t\} \leq 0.1\alpha t$. Therefore $t^{0.9} < m \leq 0.1\alpha t$. Also $m < \beta_1 N = |V|$. By the property of the m-expanding graph and the bipartite expander, for any subset of players $W \subset A_{l+1}$ with $|W| \geq m$, there is an edge between W and V. Now if an honest player $u_j \in A_{l+1}$ has a neighbor $v_{jq} \in V$, then all the neighbors of v_{jq} in $[M]$, $u_{q1}, ..., u_{qD}$ are honest players.

Moreover if $l \geq 2$, then all these honest players have a valid somewhere random string y_{qi}. Note that each somewhere random string y_i is only a function of the strings broadcasted so far, and thus is independent of any x_j of an honest player j who has not announced his random string. Therefore $s_{jq} = \mathsf{IExt}(x_{q1}, x_{q2}, ..., x_{qD})$ or $s_{jq} = \mathsf{SRIExt}(x_{q1}, x_{q2}, ..., x_{qD}, y_{q1})$ is ϵ-close to uniform and $y_j = s_{j1} \circ ... \circ s_{jd_2}$ is a valid somewhere random source. Therefore $Bad_{l+1} \leq m \leq 0.1\alpha t$. By induction $Bad_l \leq 0.1\alpha t$ for all l.

Now consider the number of independent (n, k) sources needed for a player in round $l+1$. The somewhere random string y_j obtained by player $u_j \in A_{l+1}$ at the end of round l is of size $d_2 \times k$, where d_2 is the degree of the m-expanding graph or bipartite expander H in round l. If $\mathsf{C}_l = \omega(1)$ then in round l we use an m-expanding graph. By Theorem 4.3.21, $d_2 = O(\frac{N}{m}\text{polylog}(N))$, where $\frac{N}{m} = max\{(2d_1)^D, (\frac{1}{\mu_1})^D\} \leq c_3^{8D}\alpha^{-8D}$ for some constant $c_3 > 0$ and $D = \mathsf{C}_l$ as $d_1 \leq c_1'\alpha_1^{-8}, \mu_1 > \alpha_1^2/3$. Thus

$$\mathsf{C}_{l+1} = O(\frac{\log d_2}{\log k}) = O(\frac{\log(N/m)}{\log k} + \frac{\log\log N}{\log k})$$

Now $N \leq Md_1^D = (1+\alpha)td_1^D$, therefore $\log N \leq D\log d_1 + \log(1+\alpha)t \leq D\log d_1 \log(1+\alpha)t$ as we can safely assume $d_1 \geq 4, D \geq 1, t \geq 4$. Thus

$$\frac{\log\log N}{\log k} \leq \frac{\log D + \log\log d_1 + \log\log((1+\alpha)t)}{\log k} = \frac{\log D}{\log k} + O(1)$$

as $k > \log p > \log t$.

On the other hand

$$\frac{\log(N/m)}{\log k} \leq \frac{8D\log(c_3/\alpha)}{\log k}$$

Therefore

$$\mathsf{C}_{l+1} = O(\frac{D\log(c_3/\alpha)}{\log k} + 1) = O(\frac{\mathsf{C}_l\log(c_3/\alpha)}{\log k} + 1)$$

As long as $\mathsf{C}_l = \omega(1)$, the above gives that $\mathsf{C}_{l+1} < \mathsf{C}_l$. As $\mathsf{C}_1 = o(\log n)$, we have $\mathsf{C}_l = o(\log n)$ for all l. Thus claim 1 holds.

Now if $\mathsf{C}_l \leq \log k$, then $\mathsf{C}_{l+1} = O(\log(c_3/\alpha) + 1) = C(\alpha)$ is a constant. Thus claim 3 holds.

If $\mathsf{C}_l > \log k$, then $\frac{\mathsf{C}_l \log(c_3/\alpha)}{\log k} + 1 < \frac{\mathsf{C}_l(\log(c_3/\alpha)+1)}{\log k} = \frac{\mathsf{C}_l \log(2c_3/\alpha)}{\log k}$. Thus

$$\mathsf{C}_{l+1} = O(\frac{\mathsf{C}_l \log(c_3/\alpha)}{\log k} + 1) \leq c_1 \frac{\mathsf{C}_l \log(2c_3/\alpha)}{\log k} = \frac{c_1 \log(c_2/\alpha)}{\log k} \mathsf{C}_l$$

for $c_2 = 2c_3$. Thus claim 2 holds.

If C_l is a constant $C(\alpha)$, then in round l of the synchronous extractor, we use the bipartite expander H with $d_2 = O((N/m)^2)$ by Theorem 4.3.8. Note $N/m \leq max\{\frac{1}{\mu_1^{\mathsf{C}_l}}, \frac{(1+\alpha)td_1^{\mathsf{C}_l}}{0.1\alpha t}\}$ is a constant, therefore $d_2 = O((N/m)^2) = d_2(\alpha)$ is a constant. For sufficiently large p this gives $d_2 < k^\gamma$, where γ is the parameter in Theorem 3.5.11. Therefore, at the end of round l, all but Bad_l honest players u_j in A_l obtain private random bits by computing $z_j = \mathsf{BasicExt}(x_j, y_j)$. The fact that $\mathsf{BasicExt}$ is strong implies that z_j is also close to independent of the whole transcript. As in each round of the protocol the error increases by $2^{-k^{\Omega(1)}}$, the error of the random bits obtained by honest players in A_l is at most $\epsilon + l2^{-k^{\Omega(1)}}$. Thus claim 4 holds. □

Proof of Theorem 4.3.25. Run Protocol 4.3.24 with parameter α and R to be chosen later. Let C_l be the number of independent (n,k) sources needed in round l and let l_0 be the first round where $\mathsf{C}_{l_0} \leq \log k$. By Lemma 4.3.27, at the end of round $l = l_0+1$, C_l becomes a constant and by the end of round $R = l_0 + 2$, all but Bad_{l_0+2} honest players in A_{l_0+2} obtain private random bits. Now by Lemma 4.3.27 there exist constants $c_1 > 0, c_2 > 0$ such that if $\mathsf{C}_l \geq \log k$, then $\mathsf{C}_{l+1} \leq \frac{c_1 \log(c_2/\alpha)}{\log k}\mathsf{C}_l$.

By the recursion $\mathsf{C}_{l+1} \leq \frac{c_1 \log(c_2/\alpha)}{\log k}\mathsf{C}_l$ we have

$$\mathsf{C}_{l_0} \leq \left(\frac{c_1 \log(c_2/\alpha)}{\log k}\right)^{l_0-1} \mathsf{C}_1 = \left(\frac{c_1 \log(c_2/\alpha)}{\log k}\right)^{l_0-1} \mathsf{C}$$

To make $\mathsf{C}_{l_0} \leq \log k$ it suffices to have

$$\left(\frac{c_1 \log(c_2/\alpha)}{\log k}\right)^{l_0-1} \mathsf{C} \leq \log k$$

we get

$$l_0 \geq \frac{\log \mathsf{C} - \log\log k}{\log\log k - \log c_1 - \log\log(c_2/\alpha)} + 1 = \frac{\log \mathsf{C} - \log c_1 - \log\log(c_2/\alpha)}{\log\log k - \log c_1 - \log\log(c_2/\alpha)}.$$

Thus it suffices to take

$$R = l_0 + 2 = \left\lceil \frac{\log \mathsf{C} - \log c_1 - \log\log(c_2/\alpha)}{\log\log k - \log c_1 - \log\log(c_2/\alpha)} \right\rceil + 2 \leq (1+o(1))\frac{\log \mathsf{C}}{\log\log k} + 3. \tag{4.1}$$

Recall that the beginning of round R is the end of round $R-1$, thus at the end of round $R-1$ all but $Bad_R \leq 0.1\alpha t$ of the honest players in A_R get private random bits.

If $A_{R+1} \neq \phi$, then in round R, we run Protocol 4.3.23. As $|A_R| \geq 3(1+\alpha)t$ and there can be at most $(1+0.1\alpha)t$ players in A_R that don't send out a string y_i that is ϵ'-close to uniform and independent of each other, the concatenated string s_j is ϵ'-close to a weak random source with min entropy rate at least $\frac{2}{3}$ and is independent of x_j. Therefore by Theorem 3.5.7 z_j is close to being uniform and private.

Now there are at most $(1+\alpha)(R-1)t$ honest players in the first $R-1$ rounds. In round R there can be at most $0.1\alpha t$ honest players in A_R that don't get private random bits. Therefore the number of honest players that get private random bits is at least $p - t - (1+\alpha)(R-1)t - 0.1\alpha t > p - (1+\alpha)Rt$. By Lemma 4.3.27 the error is at most $\epsilon + R2^{-k^{\Omega(1)}}$.

Note that $R \leq (1+o(1))\frac{\log \mathsf{C}}{\log\log k} + 3$ and $\mathsf{C} = o(\log n)$. Together with the fact $k > \log p = \Omega(\log n)$ this implies that the error $R2^{-k^{\Omega(1)}} = 2^{-k^{\Omega(1)}}$.

Choose $\alpha = 0.03$, we see $(1+\alpha)R = 1.03((1+o(1))\frac{\log \mathsf{C}}{\log\log k} + 3) < 1.1(\frac{\log \mathsf{C}}{\log\log k} + 3)$, and $R < 1.1\frac{\log \mathsf{C}}{\log\log k} + 3$ for p large enough.

Therefore Protocol 4.3.24 is a $(t, p - 1.1(3 + \frac{\log \mathsf{C}}{\log\log k})t, \epsilon + 2^{-k^{\Omega(1)}})$ synchronous extractor that runs in at most $\frac{1.1\log \mathsf{C}}{\log\log k} + 3$ rounds.

■

By using the extractor in Theorem 3.5.9 and arithmetic manipulation of (4.1), we have the following corollaries for special (n,k) sources, which tolerate a linear fraction of faulty players:

Corollary 4.3.28. *For every constant $\alpha, \delta > 0$ and p large enough, there is a $(t < \frac{p}{2+\alpha}, p - (2+\alpha)t, 2^{-k^{\Omega(1)}})$ synchronous extractor that runs in at most 2 rounds for $k = n^\delta$ in the full-information model.*

Corollary 4.3.29. *There exists a constant $c > 0$ such that for every constant $\alpha > 0$ and p large enough, there is a $(p^{0.1} < t < \frac{p}{3+\alpha}, p - (3+\alpha)t, 2^{-k^{\Omega(1)}})$ synchronous extractor that runs in at most 3 rounds for $k \geq 2^{c\sqrt{\log n}}$ in the full-information model.*

Corollary 4.3.30. *For every constant $\alpha, \delta > 0$, there exists a constant $c > 0$ such that for p large enough, there is a $(p^{0.1} < t < \frac{\delta p}{(1+\alpha)(1+\delta)}, p - (1+\alpha)(1/\delta + 1)t, 2^{-k^{\Omega(1)}})$ synchronous extractor that runs in at most $1/\delta + 1$ rounds for $k \geq 2^{c \log^{\delta} n}$ in the full-information model.*

Remark 4.3.31. Corollary 4.3.28 is just Theorem 4.3.3, which gives a synchronous extractor that achieves a tolerance of roughly $t < p/2$ and guarantees $p - 2t$ honest players end up with private random bits for $k \geq n^{\delta}$. If $t = \Theta(p)$, then the protocol runs in one round. Moreover, in this case we don't need $t > p^{0.1}$ or $n \leq \mathrm{poly}(p)$, since C is a constant.

Corollary 4.3.29 and Corollary 4.3.30 give synchronous extractors that run in constant rounds and tolerate a linear fraction of faulty players, even for min-entropy as low as $2^{\log^{\delta} n}$.

4.3.2 Asynchronous Network Extractors

In this section we discuss asynchronous network extractors. Recall that in an asynchronous network the only guarantee is that a message sent will eventually be received. Thus to design network extractors in an asynchronous full-information network is much harder than in a synchronous full-information network. However, we manage to obtain only slightly weaker results in this case.

First we have the following protocol for weak sources with high min-entropy ($k > n/2$).

Protocol 4.3.32. For an asynchronous network

Player Inputs: Every player u_i has $x_i \in \{0,1\}^n$
Player Outputs: For some honest players u_j, u_j ends up with $z_j \in \{0,1\}^m$.

Sub-Routines and Parameters:
Let Raz be the strong 2-sources extractor in Theorem 3.5.7.

Round 1 : Divide p players into 2 sets A and B, with $|A| = 2t+1$, $|B| = p - 2t - 1$. Each player u_i does the following:

1. Every player $u_i \in A$ sends his random string x_i to all players in B.
2. For every player $u_j \in B$, u_j waits to receive $t+1$ strings. Let y_j be the concatenation of all these strings. Compute $z_j = \mathsf{Raz}(y_j, x_j)$.

Theorem 4.3.33 (High Entropy Asynchronous Extractor)**.** *As long as $n \geq \log^2 p$, Protocol 4.3.32 is a $(t < p/3, p - 3t - 1, 2^{-k^{\Omega(1)}})$ asynchronous extractor for min entropy $k \geq (1/2 + \gamma)n$, where $\gamma > 0$ is any constant.*

Proof. As $|A| = 2t+1$, every honest player u_j in B will eventually receive $t+1$ strings from A. At least one of these strings is from a honest player, thus y_j has min-entropy k (the min-entropy can be poly-logarithmic in the length of y_j if $n = \text{polylog}(p)$). Note the random string x_j has min-entropy $k \geq (1/2+\gamma)n$. As long as $n \geq \log^2 p$ it's easy to check that x_j and y_j satisfy the conditions of Theorem 3.5.7. Thus $z_j = \mathsf{Raz}(y_j, x_j)$ is $2^{-k^{\Omega(1)}}$-close to being uniform and independent of the transcript. Moreover, there are at least $|B| - t = p - 3t - 1$ honest players in B. ■

Note that if we have a strong 2-source extractor for poly-logarithmic min-entropy, then Protocol 4.3.32 actually works for small min-entropy. However currently the best known 2-source extractor requires one source to have min-entropy rate $> 1/2$ if the other has only poly-logarithmic min-entropy.

For smaller min-entropy, a simple protocol for asynchronous full-information network is Protocol 4.3.34, which is a generalization of the protocol given in [GSV05].

Protocol 4.3.34. For an asynchronous network

Player Inputs: Every player u_i has $x_i \in \{0,1\}^n$
Player Outputs: For some players $\{u_j\}$, u_j ends up with $z_j \in \{0,1\}^m$

Sub-Routines and Parameters:
Let IExt be a strong C source extractor with parameters $n, k, m = k, \epsilon$ as in Theorem 3.5.9.

Divide the players into p/C disjoint sets, $S_1, ...S_{p/\mathsf{C}}$, each of size C. For set S_i, let the players in that set be $u_{i1}, ...u_{i\mathsf{C}}$.

Round 1 : For each set S_i, $i = 1, ..., p/\mathsf{C}$, each player u_{ij} in S_i does the following:

1. If $j \neq 1$, send x_{ij} to u_{i1}.
2. If $j = 1$, wait to receive $\mathsf{C}-1$ strings $\{x_{il}, l \neq 1\}$ from the other $\mathsf{C}-1$ players.
    - If u_{i1} successfully receives $\mathsf{C}-1$ strings $\{x_{il}, l \neq 1\}$, compute $z_{i1} = \mathsf{IExt}(x_{i1}, ..., x_{i\mathsf{C}})$ and send a message "complete" to all the other players $\{u_{l1}, l \neq i\}$.
    - If u_{i1} receives $p/\mathsf{C} - t$ "complete" from the players $\{u_{l1}, l \neq i\}$ before he receives $\mathsf{C}-1$ strings $\{x_{il}, l \neq 1\}$, then u_{i1} sends a "complete" to all the other players $\{u_{l1}, l \neq i\}$ and aborts.

Theorem 4.3.35. *Protocol 4.3.34 is a $(t < p/(2\mathsf{C}), (p/\mathsf{C} - 2t), \epsilon)$ asynchronous network extractor for min-entropy k, where ϵ is the error of the extractor* IExt.

Proof. We say a set S_i is "good" if there's no faulty player in S_i, and "bad" otherwise. If S_i is good, then u_{i1} will eventually receive $\mathsf{C}-1$ strings from the other players in S_i, or he will receive $p/\mathsf{C}-t$ "complete" messages before that. Either way, u_{i1} will end or abort and send a "complete" to all $u_{j1}, j \neq i$.

If S_i is bad, then u_{i1} may never get all the $\mathsf{C}-1$ strings. However, eventually he'll receive $p/\mathsf{C}-t$ "complete" messages, as there are at least $p/\mathsf{C}-t$ good sets. Thus u_{i1}'s procedure will also end eventually. Therefore the whole protocol will eventually end.

Now consider the time the protocol ends. If no u_{i1} in a good set S_i aborts, then every u_{i1} gets private random bits by computing $z_{i1} = \mathsf{IExt}(x_{i1}, ..., x_{i\mathsf{C}})$. The number of such u_{i1}s is at least $p/\mathsf{C}-t$.

Otherwise, some u_{i1} in a good set S_i aborts. Consider the logically first u_{i1} that aborts, he aborts because he receives $p/\mathsf{C}-t$ "complete"s. At least $p/\mathsf{C}-2t$ of these "complete"s are from u_{j1}s in good sets, who don't abort as u_{i1} is the first one that aborts. Therefore these u_{j1}s have obtained private random bits by computing z_{j1}s. It follows Protocol 4.3.34 is a $(t < p/(2\mathsf{C}), (p/\mathsf{C}-2t), \epsilon)$ asynchronous network extractor for min-entropy k. ■

Note if we use the independent source extractor in Theorem 3.5.9, then for min-entropy $k = n^{\beta}$ we have $\mathsf{C} = O(1/\beta)$. Thus the tolerance of Protocol 4.3.34 depends on β, while in the synchronous case for min-entropy $k = n^{\beta}$ we can tolerate $t < p/2$ faulty players. A natural question arises: can we use the same ideas in synchronous extractor to improve the asynchronous extractor?

A problem here is that in an asynchronous network the faulty players may not send out their strings as expected, thus an honest player may wait forever, which causes the protocol to fail. However, we manage to obtain similar results as the synchronous case, using additional ideas.

Specifically, we have the following theorem.

Theorem 4.3.36. *There exists a constant $c > 0$ such that if $\frac{\log\log\log n}{\log\log k} = o(1)$ and $\mathrm{poly}(t) \leq n \leq \exp(t)$, then for sufficiently large t there exists a $(t, p - c\frac{\log\log n}{\log\log k}t, 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in $O(\log\log n/\log\log k)$ rounds in the full-information model.*

We also get the following corollaries:

Corollary 4.3.37. *There exists a constant $c > 0$ such that for every constant $\delta > 0$ and t large enough, as long as $t < p/c$, there is a $(t, p-ct, 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most 2 rounds for $k = n^{\delta}$ in the full-information model.*

Corollary 4.3.38. *There exists a constant $c > 0$ such that if $\text{poly}(t) \le n \le \exp(t)$, then for every constant $\delta > 0$ and t large enough, as long as $t < \frac{\delta p}{c}$, there is a $(t, p - ct/\delta, 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most $1/\delta + 1$ rounds for $k = 2^{\log^{\delta} n}$ in the full-information model.*

To construct protocols for these theorems, we need some modifications to the ingredients used in the synchronous protocol. First we need a modified AND-disperser:

Lemma 4.3.39. *There exist global constants $b, d > 0$ and $0 < \mu < 1$ s.t. if $D = o(\log M)$, then for sufficiently large M there exists an $(N, M, bD, \alpha, \beta)$ AND-disperser with the following properties:*

1. *$\alpha = \frac{b-1}{b}$, $M < N \le Md^{bD}$ and $\beta > \mu^{bD}$.*
2. *For any subset $W \subset [M]$ with $|W| = \alpha M$, let $Q = \{v \in [N], |\Gamma(v) \cap W| < D\}$ and $\gamma = |Q|/N$, then $\gamma < \frac{\mu^{bD}}{3}$.*

Proof. Similar as in the proof of Lemma 4.3.19, take a Ramanujan graph X on M vertices with degree d_0 as constructed in [LPS88]. Take all the random walks on X with length $2bD - 1$ and delete the walks with more than bD repeated vertices. Take the remaining walks to be vertices of $[N]$ and for each vertex in $[N]$, connect it to the first bD distinct vertices in $[M]$.

Now β is the fraction of walks that stay in W and have at least bD distinct vertices. Let $\mu_1 = min\{\alpha, \alpha + \lambda_{M-1}(1-\alpha)\}$. By the same analysis as in Lemma 4.3.19, if we choose $d_0 = d_0(\alpha) = d_0(b)$ large enough, we can make $\beta > \frac{1}{2}\mu_1^{2bD} \ge \mu^{bD}$ for some constant $0 < \mu = \mu_1^2/2^{\frac{1}{b}} = \mu(b) < 1$ and $M < N < Md_0^{2bD-1} \le Md^{bD}$, where $d = d_0^2$.

Now for any vertex $v \in [N]$, if $|\Gamma(v) \cap W| < D$, then the walk of length $2bD - 1$ corresponding to v must have less than $(b+1)D$ vertices in W. Let the random variable Y denote the number of vertices in W for a random walk of length $2bD - 1$, then $EY = \alpha 2bD = 2(b-1)D$. By the chernoff bound of random walks on expander graph [Gil98],

$$\gamma = \Pr(Y < (b+1)D) \le \Pr(|Y - EY| > (b-3)D) \le 2^{-cbD}$$

for some constant $c > 0$.

Now we have

$$\mu^{bD}/\gamma > \mu^{bD}/2^{-cbD} = (\mu^b 2^{cb})^D = (\frac{1}{2}(2^c \mu_1^2)^b)^D$$

Note $\mu_1 = min\{\alpha, \alpha + \lambda_{M-1}(1-\alpha)\}$ and $\alpha = \frac{b-1}{b}$ goes to 1 as b goes to infinity. Thus for a sufficiently large constant b we have $2^c\mu_1^2 > 1$ and $\frac{1}{2}(2^c\mu_1^2)^b > 3$. Therefore $(\frac{1}{2}(2^c\mu_1^2)^b)^D > 3$ and $\gamma < \frac{\mu^{bD}}{3}$.

□

Another modification is that instead of using an m-expanding graph, we now need to use an *extractor* graph. For convenience we need the following definition of an extractor.

Definition 4.3.40. $\mathsf{Ext} : [N] \times [D] \to [M]$ is a (K, ϵ)-*extractor* if for every subset $S \subseteq [N]$ of size K, $|\mathsf{Ext}(U_S, U_{[D]}) - U_{[M]}| \leq \epsilon$, where U_X denotes the uniform distribution on the set X.

Such an extractor can be viewed naturally as a bipartite graph $\mathsf{Ext}(N, M, D)$ where the left vertex set is $[N]$, right vertex set is $[M]$ and every left vertex has degree D. It can be equivalently viewed as a function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ where $n = \log N, d = \log D, m = \log M$ that works for min-entropy $k = \log K$. We'll use the graph version of Ext in the protocol.

Note that we don't need a *strong* extractor here. However we need an extractor with large output. For a (k, ϵ) extractor that uses d truly random bits and has output length m, what we care about here is the *entropy-loss* defined by $\Delta = k + d - m$ for an extractor and $\Delta = k - m$ for a strong extractor. Nonconstructively, one can show that, for any n and $k \leq n$, there exist strong extractors $\mathsf{Ext}_{n,k} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with $d = \log(n-k) + 2\log(1/\epsilon) + O(1)$ and entropy loss $\Delta = 2\log(1/\epsilon) + O(1)$[RTS97].

Definition 4.3.41. An optimal strong extractor is a strong extractor $\mathsf{Ext}_{n,k} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with $d = \log(n-k) + 2\log(1/\epsilon) + O(1)$ and entropy loss $\Delta = 2\log(1/\epsilon) + O(1)$.

We have the following lemma:

Lemma 4.3.42. *Assume we can explicitly construct optimal strong extractors, then for any $N > K > 0$ and $M > 0$, there is a polynomial time computable (K, ϵ) extractor $\mathsf{Ext} : [N] \times [D] \to [M]$ with $D = O(\frac{M}{K\epsilon^2}\text{polylog}N)$.*

Proof. First construct the strong extractor $\mathsf{Ext}_s : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with $d = O(\log n) + 2\log(1/\epsilon) + O(1)$ and entropy loss $\Delta = 2\log(1/\epsilon) + O(1)$, where $n = \log N$ and $k = \log K$. This gives us output length $m = k - \Delta$. We want the output length to be $\log M$, thus we need another $\log M - m$ bits. Notice Ext_s is a strong extractor, thus we can add the bits of the seed to the output while still keeping the error ϵ. We'll add bits of the seed to the output until the output length is $\log M$. If

$m+d < \log M$, then we add another $\log M - m - d$ random bits to both the seed and the output, i.e., $\mathsf{Ext}(x, y \circ z) = \mathsf{Ext}_s(x, y) \circ y \circ z$. Here y is the seed of length d and z is the additional $\log M - m - d$ random bits. The new seed length in this case will be $d = \log M - m = \log M - \log K + \Delta$. Thus $D = 2^d = max\{O(\frac{1}{\epsilon^2}\text{polylog}N), O(\frac{M}{K\epsilon^2})\} = O(\frac{M}{K\epsilon^2}\text{polylog}N)$. Now $\mathsf{Ext} : [N] \times [D] \to [M]$ is a (K, ϵ)-extractor with $D = O(\frac{M}{K\epsilon^2}\text{polylog}N)$.

□

In fact currently we don't have an explicit construction of the optimal strong extractor. Currently the best known explicit extractor that achieves optimal entropy loss requires the seed length to increase by a poly-logarithmic factor:

Theorem 4.3.43. *[RRV99] For every $n, k \in N$, and $\epsilon > 0$ such that $k \leq n$, there are explicit strong (k, ϵ)-extractors $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-\Delta}$ with entropy loss $\Delta = 2\log(1/\epsilon) + O(1)$ and $d = O(\log^2 n \log(1/\epsilon) \log k)$.*

Using this extractor, we get slightly weaker results. Note $d = O(\log^3 n \log(1/\epsilon))$. Thus in the graph representation the left degree will be $D = 2^d = 2^{O((\log\log N)^3 \log(1/\epsilon))}$. As a result, we will not be able to handle min-entropy as low as $k = \text{polylog}(n)$. However, when the min-entropy is larger, we'll be able to improve the number of honest players who end up with private random bits.

The following proposition is well known.

Proposition 4.3.44. *Let $\mathsf{Ext}(N, M, D)$ be a bipartite graph with left vertex set $[N]$, right vertex set $[M]$ and left degree D. For any subset $S \subset [M]$, let $Bad_S = \{u \in [N], |\frac{|\Gamma(u) \cap S|}{D} - \frac{|S|}{M}| > \epsilon\}$. If Ext is a (K, ϵ) extractor, then $|Bad_S| \leq K$.*

4.3.2.1 High-Level Ideas of the Asynchronous Extractor

We now outline the high-level ideas of the asynchronous extractor. In the highest level the asynchronous extractor does the same thing as the synchronous extractor. The goal is to have the number of independent sources needed decrease in each round by a factor of roughly $\log k$, with the help of somewhere random source of fewer and fewer rows and the extractor in Theorem 3.5.10. Once the number of independent sources needed decreases to 1, a player can extract random bits using the SR-source and his own weak source[2]. However, additional ideas are needed to ensure that this can work:

[2]In some cases the number of independent sources needed cannot decrease to 1, but it will decrease to a constant. In this case we'll use Protocol 4.3.34

Idea 1 : In one round of the protocol, where players in A send their strings to players in B, we can no longer put only $(1+\alpha)t$ players in A. Otherwise there could be a majority of faulty players in A and the players in B could wait forever or receive strings only from faulty players. Similarly we can no longer only use tuples of size C, as if there's only one faulty player in a tuple, then a player in B will never receive that tuple. Therefore, we need to increase the size of A and the size of the tuple.

Idea 2 : The AND-disperser in Lemma 4.3.39 deals with the above problem. More specifically, consider tuples of size $b\mathsf{C}$ for some constant b. Let the total number of such tuples be s_1, the number of tuples that have less than C honest players be t_1 and the number of tuples that consist of all honest players be h_1. Lemma 4.3.39 shows that for a large enough b, $h_1 > t_1$. Now imagine we have the players in B wait to receive C strings from a tuple, and receive such strings from $s_1 - t_1$ tuples. The key observation is that every player in B will eventually receive so many strings, and at least one tuple that sends the strings consists of all honest players. This is because every tuple that consists of at least C honest players will eventually send out C strings, and the number of such tuples is $s_1 - t_1$. Also $h_1 > t_1$ so at least one of the tuples must consist of all honest players. Now the players in B can compute a valid somewhere random source.

Idea 3 : One problem with the above idea is that the number $s_1 - t_1$ is too large, which will cause the number of rows of the SR-source to be too large. Recall in the synchronous extractor we use an m-expanding graph to reduce this number, here instead we use an extractor graph. This is because an extractor graph roughly keeps the right fraction of "good" and "bad" tuples in the neighbors for most vertices, so that the argument in **Idea2** can still work.

Idea 4 : The use of extractor graph will leave us a small fraction of honest players in B that don't have the nice property discussed above. These players could wait forever in the protocol. To deal with this we use the same idea as in Protocol 4.3.34, where the players send "Complete" messages to each other. This will only increase the fraction of unlucky players by a little bit.

4.3.2.2 Asynchronous Network Extractor Protocol

Now we describe our asynchronous network extractor. First we need the following sub protocols:

Protocol 4.3.45. For an asynchronous one round network

Player Inputs: There are two sets of players, A and B. Each player $u_i \in A$ has a string $x_i \in \{0,1\}^n$ and an optional $r_1 \times k$ somewhere random string $y_i \in \{0,1\}^{r_1 k}$.
Player Outputs: For every player $u_j \in B$, u_j ends up with a supposed $r_2 \times k$ somewhere random string $y_j \in \{0,1\}^{r_2 k}$.

Sub-Routines and Parameters:

- Let IExt be the C source extractor as in Assumption 4.3.16. Let SRIExt be the Somewhere Random source vs. independent source extractor as in Theorem 3.5.10.
- $|A| = 3bt$ where b is the constant promised to exist in Lemma 4.3.39. $|B| = at$ where a is another constant.
- If this is the first round of the whole protocol, let $D = \mathsf{C}$ be the number of (n,k) sources IExt needs. Otherwise let $D = O(\frac{\log r_1}{\log k})$ be the number of independent sources SRIExt needs, when the somewhere random source has r_1 rows. Construct an $(N, M = |A|, bD, \alpha = \frac{b-1}{b}, \beta)$ AND-disperser G with $N \le M d^{bD}, \beta \ge \mu^{bD}$ as in Lemma 4.3.39.
- Construct a (K, ϵ_0) extractor graph $\mathsf{Ext} : [N_1] \times [D_1] \to [M_1]$ of Lemma 4.3.42 with $N_1 = |B|, M_1 = N, K = t/2^{bD}, \epsilon_0 = \frac{\mu^{bD}}{3}$. Let $D_2 = (1-2\epsilon_0)D_1 + 1$. Identify each player in A with a vertex in $[M]$ and identify each player in B with a vertex in $[N_1]$. Identify $[M_1]$ with $[N]$.

Round 1 :

1. Every player $u_i \in A$ sends his string x_i and his somewhere random string y_i(if u_i has such a string) to all the players in B.
2. For every player $v_j \in B$, his corresponding vertex in $[N_1]$ has D_1 neighbors in $[M_1]$: $w_{j1}, ..., w_{jD_1}$. Each of these neighbors w_{jq} in turn has bD neighbors in $[M]$. Let the bD neighbors be $u_{q1}, ..., u_{qbD}$ and we call $(u_{q1}, ..., u_{qbD})$ a tuple. Thus each $v_j \in B$ is connected to D_1 tuples. For each of these tuples, v_j waits to receive D strings sent from that tuple. Without loss of generality assume for tuple $(q1, q2, ..., qbD)$ the first D strings received are $x_{q1}, ..., x_{qD}$ and $q1 < q2 < ... < qD$.
    - If this is the first round of the whole protocol, compute $s_{jq} = \mathsf{IExt}(x_{q1}, x_{q2}, ..., x_{qD})$. Otherwise compute $s_{jq} = \mathsf{SRIExt}(x_{q1}, x_{q2}, ..., x_{qD}, y_{q1})$.
    - If v_j successfully computes D_2 s_{jq}s, let y_j be the concatenation of these s_{jq}s, send "Complete" to all the other players in B and end.
    - If v_j receives $|B| - t - 2K$ "Complete" before he computes D_2 s_{jq}s, send "Complete" to all the other players in B, let y_j be any arbitrary string and abort.

Protocol 4.3.46. For an asynchronous one round network

Player Inputs: There are two sets of players, A and B. Every player $u_i \in A$ has private random bits $z_i \in \{0,1\}^k$.
Player Outputs: For every honest player $u_j \in B$, u_j ends up with private random bits $z_j \in \{0,1\}^m$, while every honest player $u_i \in A$ still has $0.9k$ private random bits left.

Sub-Routines and Parameters:

- Let Raz be the strong 2-sources extractor in Theorem 3.5.7.
- $|A| \geq 10t$.

Round 1 :

1. Every player $u_i \in A$ takes 0.1 fraction of his private random bits z_i, let the fraction be y_i and sends y_i to all the players in B.
2. For every player $u_j \in B$, u_j waits to receive $7t$ y_is. Let s_j be the concatenation of all the y_is. Compute $z_j = \mathsf{Raz}(s_j, x_j)$.

The asynchronous network extractor is described as Protocol 4.3.47.

Protocol 4.3.47. For an asynchronous network

Player Inputs: Every player u_i has a weak random string $x_i \in \{0,1\}^n$.
Player Outputs: For some players u_j, u_j ends up with private random bits $z_j \in \{0,1\}^m$

Sub-Routines and Parameters:

- Protocol 4.3.45, Protocol 4.3.46 and Protocol 4.3.34.
- BasicExt is the extractor in Theorem 3.5.11.
- SRIExt is the Somewhere Random vs Independent Source extractor in Theorem 3.5.10.
- b is the constant promised to exist by Lemma 4.3.39.
- R is given as the number of rounds of the protocol.

Divide p players into R disjoint sets $A_1, ..., A_R$. $|A_1| = |A_2| = ... = |A_{R-1}| = 3bt$.

OP :A_R is the set of remaining players.

NOP :$|A_R| = min\{p - 3b(R-1)t, 10t\}$. A_{R+1} is the set of remaining players. It's possible that $A_{R+1} = \Phi$.

Round l, $l = 1, ..., R-1$: Run Protocol 4.3.45 with $A = A_l$, $B = A_{l+1}$. In round 1, players in A don't have the optional somewhere random string y_i. In subsequent rounds, players in A have the somewhere random string y_i obtained by the end of the previous round.

OP, Round R, the last Round : Run Protocol 4.3.34 on A_R with SRIExt as the extractor.

NOP, Round $R-1$: At the end of round $R-1$, player u_j in A_R obtains private random bits by computing $z_j = \mathsf{BasicExt}(x_j, y_j)$. Here x_j is u_j's weak random string, y_j is the somewhere random string obtained by the end of the previous round.

NOP, Last Round : If $A_{R+1} \neq \Phi$, run Protocol 4.3.46 with $A = A_R$, $B = A_{R+1}$.

Remark 4.3.48. In the above protocol OP stands for the step when we use the optimal seeded extractor as in Definition 4.3.41, NOP stands for the step when we use the non-optimal seeded extractor as in Theorem 4.3.43.

4.3.3 Proof of the Theorems

We prove the following theorem about asynchronous extractor:

Theorem 4.3.49. *Assume we have explicit construction of the optimal seeded extractor in Definition 4.3.41. There exist constants $c_1, c_2 > 0$ such that if $k > \log p$ and $n \leq \mathrm{poly}(p)$ then as long as $t > p^{0.1}$, for sufficiently large p, there exists a $(t, p/c_1 - c_2(\frac{\log \mathsf{C}}{\log\log k} + 1)t, \epsilon + 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most $\frac{1.1 \log \mathsf{C}}{\log\log k} + 2$ rounds in the full-information model. Here ϵ is the error of the extractor in Assumption 4.3.16.*

Remark 4.3.50. The parameter $p^{0.1}$ can be replaced by p^{δ} for any constant $\delta > 0$ and the constant 1.1 can be replaced by $1 + \alpha$ for any constant $\alpha > 0$.

To prove this theorem we need the following lemma, which is similar to Lemma 4.3.27 for the synchronous extractor.

Lemma 4.3.51. *Let C_l be the number of independent (n, k) sources needed for* IExt *or* SRIExt *in round l of the asynchronous extractor. Define Bad_l to be the number of honest players in A_l that don't have a valid somewhere random string for $l \geq 2$ and let $Bad_1 = 0$. Assume we use the optimal seeded extractor in Definition 4.3.41 in the protocol. There exist a constant $c_0 > 0$ such that the following claims hold.*

1. *$\forall l, Bad_l \leq 2t$ and $\mathsf{C}_l = o(\log n)$*
2. *If $\mathsf{C}_l > \log k$, then $\mathsf{C}_{l+1} \leq \frac{c_0}{\log k}\mathsf{C}_l$*
3. *If $\mathsf{C}_l \leq \log k$, then C_{l+1} is a constant $C \leq c_0$*

Proof. Similar as in Lemma 4.3.27, we prove 1 by induction. For $l = 1$, the number of independent (n, k) sources needed for IExt is $\mathsf{C}_1 = O(\frac{\log n}{\log k}) = o(\log n)$ by Theorem 3.5.9. Also $Bad_1 = 0 < 2t$.

Let b be the constant promised to exist by Lemma 4.3.39. Now assume for round l, $\mathsf{C}_l = o(\log n)$ and $Bad_l \leq 2t$. Note $b\mathsf{C}_l = o(\log n) = o(\log t)$ as $n = O(\mathrm{poly}(p))$ and $t > p^{0.1}$. Let U_l denote the subset of honest players in A_l that have a valid somewhere random source for $l \geq 2$, and the subset of honest players in A_l for $l = 1$, then $|U_l| \geq 3bt - t - Bad_l \geq 3(b-1)t$. Thus $\frac{|U_l|}{|A_l|} \geq \frac{b-1}{b}$, which means the honest players consist of a large fraction of the players.

Consider Protocol 4.3.45 for round l. An AND-disperser $G = (N, M, bD, \alpha, \beta)$ is constructed with $M = |A_l| = 3bt, D = \mathsf{C}_l, \alpha = \frac{b-1}{b}$. By Lemma 4.3.39, $N \leq Md^{bD} = Md^{b\mathsf{C}_l}$ for some constant

d, and there exists a subset $V \subset [N]$ with $|V| = \beta N > \mu^{bD} N$ s.t. $\Gamma(V) \subset U_l$. V is the set of "good" tuples that consist of all honest players (note $|V| > 0$ as $D = o(\log N)$).

Let $W = \{v \in [N], |\Gamma(v) \cap U_l| < D\}$ and $\gamma = |W|/N$, then $\beta > \mu^{bD} = 3\epsilon_0 > 3\gamma$. W is the set of "bad" tuples that consist of less tha $D = \mathsf{C}_l$ honest players.

Now consider the extractor $\mathsf{Ext} : [N_1] \times [D_1] \to [M_1]$. As $M_1 = N$ and we identify the vertices in $[M_1]$ and $[N]$, we can equivalently view V and W as subsets of $[M_1]$. Let $S_1 = \{v \in [N_1], \frac{|\Gamma(v) \cap V|}{D_1} \leq \beta - \epsilon_0\}$, $S_2 = \{v \in [N_1], \frac{|\Gamma(v) \cap W|}{D_1} \geq \gamma + \epsilon_0\}$ and $S_0 = [N_1]/(S_1 \cup S_2)$. S_0 is the set of players in B that have roughly the right fraction of "good" and "bad" tuples in the neighbors. As Ext is a (K, ϵ_0) extractor, by Proposition 4.3.44 $|S_1| \leq K$ and $|S_2| \leq K$. Therefore $|S_0| \geq N_1 - 2K = |B| - 2K$.

For each honest player $u_j \in S_0$, $|\Gamma(u_j) \cap V| > (\beta_1 - \epsilon_0)D_1$ and $|\Gamma(u_j) \cap W| < (\gamma + \epsilon_0)D_1$. Thus u_j has at least $(1 - \gamma - \epsilon_0)D_1 + 1 > (1 - 2\epsilon_0)D_1 + 1 = D_2$ neighbors in $[M_1]/W$. Each of these neighbors corresponds to a tuple of size bD that consists of at least D honest players. For each of these tuples, u_j will eventually receive D strings. Thus u_j will receive D strings from D_2 tuples eventually. Also, as $D_2 = (1 - 2\epsilon_0)D_1 + 1 > (1 - (\beta - \epsilon_0))D_1 + 1$, at least one of these D_2 tuples must be in V and thus consists of all honest players. The strings sent from this tuple and received by u_j are all from honest players. Therefore if u_j finishes computing the D_2 s_{jq}s, then at least one of them is close to uniform and thus u_j obtains a valid somewhere random string y_j. We have at least $|S_0| - t = |B| - t - 2K$ such honest players, therefore eventually every player u_j in B will finish computing y_j or receive $|B| - t - 2K$ "complete" and abort. Thus the protocol for any round will eventually end.

Now if no player aborts then all honest players obtain valid somewhere random strings. Otherwise consider the first honest player who aborts, he aborts because he receives $|B| - t - 2K$ "complete". Let the number of faulty players in B be t_B, then at least $|B| - t - 2K - t_B$ "complete" are from honest players that don't abort. Thus at least $|B| - t - 2K - t_B$ honest players obtain valid somewhere random strings. As $K = t/2^{bD} \leq t/2$, $Bad_{l+1} \leq t + 2K \leq 2t$. Therefore $Bad_l \leq 2t$ holds for all l.

Now consider the number of independent (n, k) sources needed for a player in round $l + 1$. The somewhere random string y_j obtained by player $u_j \in A_{l+1}$ at the end of round l is of size $D_2 \times k$, where $D_2 = (1 - 2\epsilon_0)D_1 \leq D_1$. By Lemma 4.3.42, $D_1 = O(\frac{M_1}{K\epsilon_0^2} \text{polylog} N_1) = 2^{O(D)} \text{polylog}(t)$. Therefore by Theorem 3.5.10 the number of independent (n, k) sources needed in round $l + 1$ is

$$\mathsf{C}_{l+1} = O(\frac{\log D_2}{\log k}) = O(\frac{D}{\log k} + \frac{\log \log t}{\log k}) \leq c_1(\frac{\mathsf{C}_l}{\log k} + 1)$$

for some constant $c_1 > 0$ as $k > \log p > \log t$.

Now it's clear that as long as $\mathsf{C}_l = \omega(1)$, $\mathsf{C}_{l+1} < \mathsf{C}_l$. As $\mathsf{C}_1 = o(\log n)$ we have $\mathsf{C}_l = o(\log n)$ for all l. Thus claim 1 holds.

Let $c_0 = 2c_1$. If $\mathsf{C}_l \leq \log k$, then $\mathsf{C}_{l+1} \leq c_1(\frac{\mathsf{C}_l}{\log k} + 1) \leq c_0$ is a constant. Therefore claim 3 holds.

If $\mathsf{C}_l > \log k$, then $\frac{\mathsf{C}_l}{\log k} > 1$ and thus

$$\mathsf{C}_{l+1} \leq c_1(\frac{\mathsf{C}_l}{\log k} + 1) \leq 2c_1(\frac{\mathsf{C}_l}{\log k}) \leq \frac{c_0}{\log k}\mathsf{C}_l.$$

Thus claim 2 holds.

□

Proof of Theorem 4.3.49. Run Protocol 4.3.47 with round number R to be chosen later. Let C_l be the number of independent (n,k) sources needed in round l. By Lemma 4.3.51, there exists a constant $c_0 > 0$ such that if $\mathsf{C}_l \geq \log k$, then $\mathsf{C}_{l+1} \leq \frac{c_0}{\log k}\mathsf{C}_l$. Consider the first round l_0 when $\mathsf{C}_{l_0} \leq \log k$. Let $R = l_0$, by Lemma 4.3.51 C_{R+1} will be a constant $C \leq c_0$. In round $R+1$ we run Protocol 4.3.34 on A_{R+1} where each $u_{i1} \in A_{R+1}$ waits to receive $C-1$ strings from the other players and compute $z_{i1} = \mathsf{SRIExt}(x_{i1}, x_{i2}, ..., x_{iC}, y_{i1})$. By the same analysis in Theorem 4.3.35, at least $|A_R|/C - Bad_R - t \geq |A_R|/c_0 - 3t$ honest players end up with private random bits.

By the recursion $\mathsf{C}_{l+1} \leq \frac{c_0}{\log k}\mathsf{C}_l$ we get

$$\mathsf{C}_{l_0} \leq \left(\frac{c_0}{\log k}\right)^{l_0-1} \mathsf{C}_1 = \left(\frac{c_0}{\log k}\right)^{l_0-1} \mathsf{C}$$

To ensure $\mathsf{C}_{l_0} \leq \log k$ it suffices to have

$$\left(\frac{c_0}{\log k}\right)^{l_0-1} \mathsf{C} \leq \log k.$$

we get

$$l_0 \geq \frac{\log \mathsf{C} - \log\log k}{\log\log k - \log c_0} + 1 = \frac{\log \mathsf{C} - \log c_0}{\log\log k - \log c_0}$$

Thus it suffices to take

$$R = \left\lceil \frac{\log \mathsf{C} - \log c_0}{\log\log k - \log c_0} \right\rceil \leq (1+o(1))\frac{\log \mathsf{C}}{\log\log k} + 1.$$

Now $|A_{R+1}| = p - 3bRt$, therefore the number of honest players that get private random bits is at least $|A_{R+1}|/c_0 - 3t = p/c_0 - 3(bR/c_0 + 1)t$. In each round when we apply the extractor, the error will increase by $2^{-k^{\Omega(1)}}$, thus the total error is at most $\epsilon + R2^{-k^{\Omega(1)}}$.

Note that $R \leq (1+o(1))\frac{\log \mathsf{C}}{\log\log k} + 1$. Thus for p large enough $R < \frac{1.1 \log \mathsf{C}}{\log\log k} + 1$. Together with the fact $\mathsf{C} = o(\log n)$ and $k > \log p = \Omega(\log n)$ this implies that the error $R2^{-k^{\Omega(1)}} = 2^{-k^{\Omega(1)}}$.

Choose $c_1 = c_0$, $c_2 = 3(1.1b/c_0 + 1)$, we have that Protocol 4.3.47 is a $(t, p/c_1 - c_2(\frac{\log \mathsf{C}}{\log\log k} + 1)t, \epsilon + 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most $\frac{1.1 \log \mathsf{C}}{\log\log k} + 2$ rounds in the full-information model. ■

If we use the extractor in Theorem 4.3.43, then we get the following theorem:

Theorem 4.3.52. *There exists a constant $c_1 > 0$ such that if $\frac{\log\log\log n}{\log\log k} = o(1)$ and $n = \mathrm{poly}(p)$, then as long as $t > p^{0.1}$, for sufficiently large p there exists a $(t, p - c_1(\frac{\log \mathsf{C}}{\log\log k} + 1)t, \epsilon + 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most $\frac{1.1 \log \mathsf{C}}{\log\log k} + 2$ rounds in the full-information model. Here ϵ is the error of the extractor in Assumption 4.3.16.*

Remark 4.3.53. The parameter $p^{0.1}$ can be replaced by p^{δ} for any constant $\delta > 0$ and the constant 1.1 can be replaced by $1 + \alpha$ for any constant $\alpha > 0$.

Proof. [Proof Sketch] Run Protocol 4.3.47 with round number R to be chosen later. Now basically repeat the proof of Lemma 4.3.51 and Theorem 4.3.49. Let Bad_l be the number of honest players in A_l that don't have a valid independent somewhere random source for $l \geq 2$, and define $Bad_1 = 0$. Let C_l be the number of independent (n,k) sources needed for a player in round l. Again by induction we can show that $\forall l, Bad_l \leq 2t$. What is different now is the relation between C_{l+1} and C_l.

Consider the (K, ϵ_0) extractor graph $\mathsf{Ext} : [N_1] \times [D_1] \to [M_1]$ constructed in the protocol. Similar as in Lemma 4.3.42, we have

$$D_1 = max\{\frac{M_1}{K\epsilon_0^2}, 2^{O((\log\log N_1)^3 \log(1/\epsilon_0))}\}.$$

Note $\frac{M_1}{K\epsilon_0^2} = 2^{O(D)}$ and $O((\log\log N_1)^3 \log(1/\epsilon_0)) = O(D(\log\log t)^3)$. Thus

$$D_1 = 2^{O(D(\log\log t)^3)} \leq 2^{O(\mathsf{C}_l(\log\log p)^3)} = 2^{O(\mathsf{C}_l(\log\log n)^3)}.$$

The last equality follows because $n = \mathrm{poly}(p)$. Thus

$$\mathsf{C}_{l+1} = O(\frac{\log D_1}{\log k}) \leq O(\frac{\mathsf{C}_l(\log\log n)^3}{\log k}) \leq \frac{c_0(\log\log n)^3}{\log k}\mathsf{C}_l$$

for some constant $c_0 > 0$.

Therefore

$$\mathsf{C}_l \leq \frac{c_0^{l-1}(\log\log n)^{3(l-1)}}{\log^{l-1} k}\mathsf{C}.$$

If $\frac{\log\log\log n}{\log\log k} = o(1)$, then $\log k > c_0(\log\log n)^3$ and C_l will eventually decrease to 1. Let l_0 be the first round where $\mathsf{C}_{l_0} \leq 1$ and $R = l_0 - 1$, then by the end of round R all but Bad_{R+1} of the honest players in A_{R+1} obtain private random bits by computing $z_j = \mathsf{BasicExt}(x_j, y_j)$.

Let

$$\frac{c_0^{l_0-1}(\log\log n)^{3(l_0-1)}}{\log^{l_0-1} k}\mathsf{C} \leq 1$$

We get

$$l_0 \geq \frac{\log \mathsf{C}}{\log\log k - 3\log\log\log n - \log c_0} + 1$$

Thus it suffices to take

$$R = \left\lceil \frac{\log \mathsf{C}}{\log\log k - 3\log\log\log n - \log c_0} \right\rceil \leq (1+o(1))\frac{\log \mathsf{C}}{\log\log k} + 1.$$

The equality follows because $\frac{\log\log\log n}{\log\log k} = o(1)$.

If $A_{R+2} \neq \phi$, then in round $R+1$, we run Protocol 4.3.46 with $A = A_{R+1}$ and $B = A_{R+2}$. As $|A_{R+1}| \geq 10t$ and $Bad_{R+1} \leq 2t$, every player in A_{R+2} will eventually receive $|A_{R+1}| - Bad_{R+1} - t \geq 7t$ strings. Among these strings at most $Bad_{R+1} + t \leq 3t$ are not ϵ' close to uniform and independent of each other. Thus the concatenated string s_j is ϵ' close to have min entropy rate at least $\frac{4}{7}$ and independent of x_j. Therefore by Theorem 3.5.7 $z_j = \mathsf{Raz}(s_j, x_j)$ is $2^{-k^{\Omega(1)}}$ close to uniform and independent of the transcript so far.

Now there are at most $3bRt$ honest players in the first R rounds. In round $R+1$ there can be at most $Bad_{R+1} \leq 2t$ honest players that don't get private random bits. Therefore the number of honest players that get private random bits is at least $p - t - 3bRt - 2t = p - (3bR+3)t$. In each round when we apply the extractor, the error increases by $2^{-k^{\Omega(1)}}$, thus the total error is at most $\epsilon + R2^{-k^{\Omega(1)}}$.

Note that $R \leq (1+o(1))\frac{\log \mathsf{C}}{\log\log k}+1$. Thus for p large enough $R < \frac{1.1\log \mathsf{C}}{\log\log k}+1$. Together with the fact $\mathsf{C} = o(\log n)$ and $\log\log\log n = o(\log\log k)$ this implies that the error $R2^{-k^{\Omega(1)}} = 2^{-k^{\Omega(1)}}$.

Choose $c_1 = 3(1.1b+1)$, we have Protocol 4.3.47 is a $(t, p - c_1(\frac{\log \mathsf{C}}{\log\log k}+1)t, \epsilon + 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most $\frac{1.1\log \mathsf{C}}{\log\log k}+2$ rounds in the full-information model.

■

Using the extractor in Theorem 3.5.9 where $\mathsf{C} = O(\frac{\log n}{\log k})$, the above theorem gives the following theorem about (n,k) sources, and corollaries which tolerate a linear fraction of faulty players.

Theorem 4.3.54. *There exists a constant $c > 0$ such that if $\frac{\log\log\log n}{\log\log k} = o(1)$ and $n = \mathrm{poly}(p)$, then as long as $t > p^{0.1}$, for sufficiently large p there exists a $(t, p - c\frac{\log\log n}{\log\log k}t, 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in $O(\log\log n/\log\log k)$ rounds in the full-information model.*

Corollary 4.3.55. *There exists a constant $c > 0$ such that for every constant $\delta > 0$ and p large enough, there is a $(t < \frac{p}{c}, p - ct, 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most 2 rounds for $k = n^{\delta}$ in the full-information model.*

Corollary 4.3.56. *There exists a constant $c > 0$ such that if $n = \mathrm{poly}(p)$, then for every constant $\delta > 0$ and p large enough, there is a $(p^{0.1} < t < \frac{\delta p}{c}, p - ct/\delta, 2^{-k^{\Omega(1)}})$ asynchronous extractor that runs in at most $1/\delta + 1$ rounds for $k = 2^{\log^{\delta} n}$ in the full-information model.*

Remark 4.3.57. The asynchronous network extractor tolerates a linear fraction of faulty players and guarantees a linear fraction of honest players end up with private random bits, even for min-entropy roughly as small as $k = 2^{\log^{\delta} n}$.

In the case $k = n^{\delta}$, we don't need $t > p^{0.1}$ and we can deal with $n \geq \mathrm{poly}(p)$, since C is a constant. Moreover if $t = \Theta(p)$, then the protocol runs in one round.

4.4 Applications in Distributed Computing

In this section we use our network extractors to get new protocols for Byzantine agreement, leader election and collective coin flipping using weak random sources.

4.4.1 Collective Coin-Flipping and Leader Election

We use the following theorem as a black box.

Theorem 4.4.1 ([RZ01, Fei99]). *For every $\beta < 1/2$, there exists a polynomial time computable $\log^* p + O(1)$ round protocol for leader election tolerating $t \leq \beta p$ faulty players, as long as each player has $O(\log p)$ truly random bits, in the full-information model with a broadcast channel.*

Given this theorem, the obvious protocol in the case that each player only has access to a weak random source is to first run a network extractor and then run the protocol for leader election assuming that each player has access to truly random bits. We can do slightly better than this by observing that our network extractor for low entropy sources (Theorem 4.3.3) actually separates the players into two sets, and guarantees that at most roughly t of the players in a set of size roughly $p - t$ don't have access to private randomness.

Theorem 4.4.2 (Leader Election for Low Entropy). *Let $\alpha, \gamma > 0$ be any constants. There exists a constant $\beta > 0$ such that if each player has access to a (n, n^{γ}) source, with $\gamma\beta \log n > \log\log p$ and p is large enough, there exists a polynomial time computable synchronous $\log^* p + O(1)$ round protocol for Leader Election tolerating $(1/3 - \alpha)p$ faulty players in the full information model.*

Proof Sketch. Let $t = \delta p$ and let $1/3 > \gamma > \delta$ be a constant very close to δ. We start by running Protocol 4.3.10 on the players. This leaves us with a set of players of size $1 - \gamma p$, of which at most γp players have access to bits which are $p 2^{-k^{\Omega(1)}}$ close to being truly random. Since we can choose β in the theorem, we can make this error an arbitrarily small constant.

Since $\gamma < 1/3$, this set of players has a $\frac{\gamma}{1-\gamma} < 1/2$ fraction of faulty players. The rest of the players have randomness that is close to being private and uniform.

We then run the protocol promised by Theorem 4.4.1 on this set to elect a leader. ■

In the case that we have access to sources of randomness with min-entropy rate greater than 1/2 or access to block sources with 2 blocks each, we can use our much better network extractors for these situations to get results that match the best results for the case that each player has access to truly random bits.

When the min-entropy is even smaller, we can use the network extractor from Theorem 4.3.4 and its corollaries (Corollary 4.3.29, Corollary 4.3.30) to get the following theorems.

Theorem 4.4.3. *There exists a constant $c > 0$ such that for any constant $\alpha > 0$ and p large enough, there exists a polynomial time computable $\log^* p + O(1)$ rounds protocol for Leader Election tolerating $t \leq (1/4 - \alpha)p$ faulty players in the full information model, assuming each player has a weak source with min-entropy $k \geq 2^{c\sqrt{\log n}}$.*

Theorem 4.4.4. *For all constants $\alpha, \delta > 0$, there exists a constant $c > 0$ such that for sufficiently large p there exists a polynomial time computable $\log^* p + O(1)$ rounds protocol for Leader Election tolerating $t \leq (1/(2 + 1/\delta) - \alpha)p$ faulty players in the full information model, assuming each player has a weak source with min-entropy $k \geq 2^{c \log^{\delta} n}$.*

4.4.2 Byzantine Agreement

First we state the best protocols that are available for the case of Byzantine agreement when each player has access to truly random bits. For synchronous networks, the following theorem is available:

Theorem 4.4.5 ([GPV06])**.** *For every $\beta < 1/3 - \epsilon$, there exists a $O(\frac{\log p}{\epsilon^2})$ round protocol for Byzantine agreement in a synchronous network tolerating βp Byzantine faults in the full information model.*

In the case of asynchronous networks, we have the following theorem:

Theorem 4.4.6 ([KKK$^+$08])**.** *For any constants $\epsilon, c > 0$ there exists an expected* $\mathrm{polylog}(p)$ *round protocol for Byzantine agreement in an asynchronous full-information network tolerating $t < \frac{p}{6+\epsilon}$ faulty players with success probability $1 - 1/\ln^c n$.*

Now we discuss how to achieve Byzantine agreement when each player only has access to weak sources. We observe that for Byzantine agreement, it suffices that more than $2p/3$ of the players achieve consensus. Once we have a protocol that guarantees this, we can easily guarantee that all non-faulty players share the consensus, simply by taking a majority vote among the first $\lceil \frac{2p}{3} \rceil$ votes received.

Theorem 4.4.7 (Synchronous Byzantine Agreement for Low Entropy)**.** *Let $\alpha, \gamma > 0$ be any constants. There exists a constant $\beta > 0$ such that if each player has access to a (n, n^{γ}) source, with $\gamma\beta \log n > \log\log p$ and p is large enough, there exists a polynomial time computable synchronous*

$O(\log p)$ *expected round protocol for Byzantine Agreement tolerating* $(1/4-\alpha)p$ *faulty players in the full information model.*

Proof Sketch. Let $t=\delta p$ and let $1/4>\gamma>\delta$ be a constant very close to δ. We start by running Protocol 4.3.10 on the players. This leaves us with a set of players of size $(1-\gamma)p$, of which at most γp players don't have access to bits which are $p2^{-k^{\Omega(1)}}$ close to being truly random. Since we can choose β in the theorem, we can make this error an arbitrarily small constant.

Since $\gamma<1/4$, this set of players has a $\frac{\gamma}{1-\gamma}<1/3$ fraction of faulty players. The rest have randomness that is close to being private and uniform.

We then run the protocol promised by Theorem 4.4.5 on this set. This guarantees that we achieve consensus on this set. Finally, we have one more round where every player in this special set transmits their agreed value to the rest of the players. Everybody takes a majority vote to decide on their final value. Since at most 1/3 of the players in the set transmit a value that is not the consensus value, we terminate with a consensus for every non-faulty player. ■

As before, in the case that the players have access to sources with min-entropy rate greater than half, or block sources with two blocks, we use our network extractors to obtain protocols that are as good as the best protocols when the players have access to truly random bits.

When the min-entropy is even smaller, we can use the network extractor from Theorem 4.3.4 and its corollaries (Corollary 4.3.29, Corollary 4.3.30) to get the following theorems.

Theorem 4.4.8. *There exists a constant* $c>0$ *such that for any constant* $\alpha>0$ *and* p *large enough, there exists an expected* $O(\log p)$ *rounds Byzantine Agreement protocol tolerating* $t\le(1/5-\alpha)p$ *faulty players in the synchronous full information model, assuming each player has a weak source with min-entropy* $k\ge 2^{c\sqrt{\log n}}$.

Theorem 4.4.9. *For all constants* $\alpha,\delta>0$, *there exists a constant* $c>0$ *such that for* p *large enough there exists an expected* $O(\log p)$ *rounds Byzantine Agreement protocol tolerating* $t\le(1/(3+1/\delta)-\alpha)p$ *faulty players in the synchronous full information model, assuming each player has a weak source with min-entropy* $k\ge 2^{c\log^{\delta} n}$.

For asynchronous Byzantine Agreement, we have the following theorems:

Theorem 4.4.10 (Asynchronous Byzantine Agreement for High Entropy)**.** *For any constant* $\epsilon>0$ *there exists an expected* $\operatorname{polylog}(p)$ *rounds Byzantine Agreement protocol tolerating* $t<\frac{p}{8+\epsilon}$ *faulty players in the asynchronous full information model, assuming each player has a weak source with min-entropy* $k\ge(1/2+\gamma)n$ *for some constant* $\gamma>0$.

Proof Sketch. First run Protocol 4.3.32, then apply the protocol in Theorem 4.4.6 on set B. Note $|B| = p - 2t - 1 \geq (6 + \epsilon)t$ while all the honest players in B obtain private random bits that are $2^{-k^{\Omega(1)}}$ to uniform by Theorem 4.3.33. There can be at most t faulty players in B. Thus BA can be achieved on B in expected $\text{polylog}(p)$ rounds. We then have one more round where each player in B sends his agreed value to all players in A, and every player in A waits to receive $2t + 1$ values and takes the majority vote as his value. Note as B has at least $2t + 1$ honest players every player in A will eventually receive $2t + 1$ values, of which at most t are faulty. Thus by taking majority vote every player in A will also agree to the correct value.

■

For polynomially small min-entropy, we have the following theorem:

Theorem 4.4.11. *There exists a constant $0 < \alpha < 1$ such that for every constant $\delta > 0$ and p large enough, there exists an expected* $\text{polylog}(p)$ *rounds Byzantine Agreement protocol tolerating $t \leq \alpha p$ faulty players in the asynchronous full information model, assuming each player has a weak source with min-entropy $k \geq n^{\delta}$.*

proof sketch. First run Protocol 4.3.47. By Corollary 4.3.55 there is a constant $c > 0$ such that at least $p - ct$ honest players end up with private random bits. Take $\alpha = 1/7c$, then as long as $t \leq \alpha p$ we have $p - ct \geq 6p/7 > 5p/6$. Thus we can run the protocol in Theorem 4.4.6 to achieve BA in $p - ct$ honest players in expected $\text{polylog}(p)$ rounds. Now we do the same thing as in Theorem 4.4.10, use one more round to achieve BA in all the honest players.

■

Similarly we have the following theorem:

Theorem 4.4.12. *There exists a constant $0 < \alpha < 1$ such that for every constant $\beta > 0$ and p large enough, there exists an expected* $\text{polylog}(p)$ *rounds Byzantine Agreement protocol tolerating $t \leq \alpha\beta p$ faulty players in the asynchronous full information model, assuming each player has a weak source with min-entropy $k \geq 2^{\log^{\beta} n}$.*

Chapter 5

Cryptography with General Weak Random Sources

In this chapter we study the problem of using general weak random sources in cryptography. This kind of questions have been considered by quite a few researchers [MW97b, DS02, DO03, DOPS04, CPS07]. Dodis et al. [DS02, DOPS04] showed how to make interactive proofs sound with respect to weak sources, and showed how to build secure signatures under some strong assumptions. Canetti et al. [CPS07] showed how to obtain universally composable (UC) security, under the assumption that there exists a common reference string (CRS), which is a samplable high min-entropy source.

The negative results have been more impressive. Dodis et al. [DOPS04] showed that almost all of the classic cryptographic tasks, including encryption, bit commitment, secret sharing, and secure two-party computation (for nontrivial functions), are impossible even with an (n, 0.9n)-source. Given these negative results, we seek to provide some positive results on the problem of secure multi-party computation where each party only has access to a general weak random source. The point here is that we now have more than one general weak random source, so we can indeed do something non-trivial.

As briefly described in the introduction, we attack this problem by building network extractor in the computational setting. In this setting we assume that the adversary is computationally bounded, i.e., it is a polynomial time Turing machine or a polynomial sized circuit. Thus, we also only require that the output distribution of the honest processors to be indistinguishable from being private and uniform. Below we give the formal definition of a *computational* network extractors.

Here we assume that all the processors involved (honest and faulty) are computationally bounded (i.e., run in time poly(n), where n is the length of the weak random sources), and the outputs need only be computationally (rather than statistically) indistinguishable from uniform. We restrict our attention to the synchronous setting.

Definition 5.0.13 (Computational Network Extractor)**.** A protocol for p processors is a (t, g) *computational network extractor* for min-entropy k if for any min-entropy k independent sources

$X_1, \ldots, X_p$ over $\{0,1\}^n$ and any choice of t faulty processors, with probability $1 - \mathrm{negl}(n)$, after running the protocol there are g honest processors $\mathcal{G} = \{i_1, \ldots, i_g\}$ such that

$$\{(B, (X_i)_{i \notin \mathcal{G}}, (Z_i)_{i \in \mathcal{G}}\}_{n \in \mathbb{N}} \approx \{B, (X_i)_{i \notin \mathcal{G}}, U_{gm}\}_{n \in \mathbb{N}}$$

where U_{gm} is the uniform distribution on gm bits, independent of B and $(X_i)_{i \notin \mathcal{G}}$, and where $\approx$ denotes computational indistinguishability.

Like most cryptographic constructions, our constructions of computational network extractors rely on some computational assumptions. However, in this case, we show that we can achieve better results than network extractors in the information theoretic setting–most, or even *all*, honest processors get private randomness, even for weak random sources with low min-entropy.

Our first result shows that if trapdoor permutations exist, then there exists a computational network extractor for sources with min-entropy $k = n^{\Omega(1)}$, in which almost every non-faulty processor ends up with a (computationally) private random string.

Theorem 5.0.14. *Assume that trapdoor permutations exist. Then for every $\alpha, \beta, \gamma, \delta > 0$ there exists a constant $0 < c < 1$ (that depends only on β) such that for every $n^\delta \leq p \leq k^c$ there exists a $(t = \gamma p, p - (1+\alpha)t)$ computational network extractor for min-entropy $k \geq n^\beta$ in the full information model.*

Next, we show that under a stronger assumption, we can actually construct computational network extractors where all of the honest processors end up with private random bits. The assumption is that there exist one-way permutations that are (very) hard to invert, even when the input is sampled from a weak source.

Definition 5.0.15 (One-Way Functions for Weak Sources)**.** We call a family of polynomial time computable permutations $f : \{0,1\}^n \to \{0,1\}^n$ **one-way for k-sources** if for every (n, k) source X, and every circuit $\mathcal{A}$ of size $2^{O(\log^2 n)}$, $\Pr[\mathcal{A}(f(X)) = X]$ is negligible.

We note this is equivalent to saying that f is exponentially hard to invert under the uniform distribution.

Proposition 5.0.16. *If $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}$ is one way for k-sources, then for every circuit $\mathcal{A}$ of size $2^{O(\log^2 n)}$, $\Pr_{x \leftarrow U}[\mathcal{A}(f_n(x)) = x] = \mathrm{negl}(n) 2^{-(n-k)}$, and vice versa.*

About our assumption

- Ideally, we would like to achieve our goals assuming that it is hard to invert these permutations with polynomial sized circuits. This is actually enough for achieving error $1/\text{poly}(n)$. For our cryptographic applications we need the error to be negligible, thus we need the stronger definition above.

- We actually do not need the function f to be a permutation. All we need is that the function f is not "too lossy." More formally, for any constant $\delta > 0$ we assume the existence of a family of functions $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ and a constant $\gamma = \gamma(\delta)$ such that for every $(n, \delta n)$-source X, $f(X)$ is a $(n, \gamma n)$-source.

A candidate function is suggested by Goldreich in [Gol09]. We note that the assumption that there exist one-way functions w.r.t. weak random sources has been used before. For example, Canetti [Can97] conjectured that the discrete-log function is one-way w.r.t. any weak source with sufficient min-entropy. Assumptions of a similar flavor were also used in several other crypto results, e.g., [Wee05, DP08, Pie09].

Under this assumption, we obtain the following results.

Theorem 5.0.17 (Network Extractors for Linear Min-Entropy)**.** *Fix a constant $\delta > 0$, and suppose that there exists a family of one-way permutations for $(n, 0.3\delta n)$-sources. Then, there is a network extractor protocol where each player takes as input an independent $(n, \delta n)$-source, and as long as there are at least 2 honest players, all the honest players end up with a string that is computationally indistinguishable from being uniform and private.*

Theorem 5.0.18 (Network Extractors for Polynomial Min-Entropy)**.** *Fix a constant $\delta > 0$, and suppose that there exists a family of one-way permutations for $(n, 0.3n^\delta)$-sources. Then, there exists a constant $u = u(\delta)$ such that there is a network extractor protocol where each player takes as input an independent (n, n^δ)-source, and as long as there are at least u honest players, all the honest players end up with a string that is computationally indistinguishable from being uniform and private.*

Our network extractor constructions establish that as long as such one-way permutations exist, weak sources are the same as true randomness for the purpose of running cryptographic protocols, as formalized below.

Corollary 5.0.19. *Fix a constant $\delta > 0$. Assume that there exists a family of one-way permutations for $(n, 0.3\delta n)$-sources, and assume that there exists a family of enhanced trapdoor permutations. Then any functionality can be computed securely even if each party has only access to an (independent) $(n, \delta n)$-source, as long as there are at least two honest parties.*

Corollary 5.0.20. *Fix a constant $\delta > 0$. Assume that there exists a family of one-way permutations for $(n, 0.3n^{\delta})$-sources, and assume that there exists a family of enhanced trapdoor permutations. Then there exists a constant $u = u(\delta)$ such that any functionality can be computed securely even if each party has only access to an (independent) (n, n^{δ})-source, as long as there are at least u honest parties.*

5.1 The Constructions

In this section we describe our constructions of computational network extractors. We first describe the constructions under standard computational assumptions.

5.1.1 Computational Network Extractors under Standard Assumptions

5.1.1.1 High Level Ideas

We now focus on the case where each player has an independent (n, k) source, for $k = n^{\beta}$. The information theoretic protocols for this setting of parameters start with a large set of players (more than t players) revealing their sources. This immediately results with a significant loss in the number of players that end up with private randomness. Namely, the guarantee is that only $p - 2t$ players end up with private randomness. Thus, this network extractor is meaningful only in the case of honest majority.

In the computational setting, we construct a protocol in which only a *small* set of honest players (much smaller than t) reveal their sources. At first this seems to be useless, since it may be the case that all the players who reveal their sources are malicious. However, we guarantee that some small subset of the players who reveal their sources are indeed honest. More specifically, the players reveal their sources in some pre-specified order until the point where "enough" honest players (though much less than t) revealed their sources. At first this seems impossible to do since we do not know who the honest players are. Nonetheless, we achieve this using computational assumptions.

We construct a protocol that proceeds in d rounds, where in the j'th round all the players in the j'th set (which is of size significantly smaller than t) announce their sources to all the players.

The intuition is that if, in some round j, a sufficient number of honest players announce their sources then we are in good shape, since the rest of the players can use the announced strings to generate a somewhere random matrix: each row in the matrix corresponds to the output of a C-source extractor applied to C of the announced strings. On the other hand, if almost all of the players in the set are dishonest, we didn't lose much by having them announce their private sources, and it seems like we made some kind of progress since the fraction of honest players among the remaining players has gone up.

However, instructing all the players to announce their sources in the appropriate round is obviously not a good idea, since then all of the honest players will completely reveal their sources to the adversary. Yet, consider the *first* round j in which a significant number of honest players announce their sources. At the end of this round, only a small set of honest players have announced their sources, and every honest player who hasn't announced her source is in possession of a private random string! We use this fact to our advantage.

We change the above protocol by using computational assumptions to ensure that after this "good" round the players do not reveal any information about their source (to a computationally bounded adversary). To this end, instead of instructing each player to announce her source in the appropriate round, we instruct each player to announce a *function* f of her source. On the one hand, this function f should maintain the entropy of the source (i.e., should be injective). On the other hand, f should hide all information about the source. These two requirements, of being injective and hiding, can be achieved simultaneously only in the computational setting, under cryptographic assumptions. Moreover, in order to hide all information about the source, the function f needs to be randomized.

Note that at the end of the "good" round, many of the players have a random string. So, one could try using part of this (supposedly) random string as randomness for the function. Unfortunately, this will not work since these random strings depend on the sources, and for the function f to be hiding the random string should be independent of the sources. We overcome this obstacle as follows.

Recall that in every round (in particular the "good" round), a set of strings are announced, and each player uses the announced strings to try to extract a random (uniformly distributed) string from her private source. Each player first saves a chunk of her (supposedly) random string as private randomness, where at the end of the protocol all these d chunks (one chunk per each round) will be xored and will consist the player's output. Then, all the players use a small fresh chunk of these (supposedly) random strings to generate for each player P_i a private random string *independent* of the source x_i. This is done as follows: First, each player stretches her small chunk of randomness using a pseudo-random generator (which is known to exist assuming the existence of one-way functions). Then, for each player P_i, all players use a portion of the chunk to run a coin flipping protocol, in which only player P_i receives a random string r_i. Now each player P_i has a (supposedly) private random string r_i, which is independent of her source x_i, and will continue to the next round of the protocol, while using $f(x_i, r_i)$ as her private source.

However, this is still not good enough, since for the proof to work we need to ensure that each random string r_i is independent of *all* the sources simultaneously. To this end, we instruct all the players to use a small fresh chunk of their random strings to elect a small set of leaders. Each of these leaders will no longer use their private source, and will use the all-zero string instead. Now, if there is at least one honest leader, then each r_i is independent of *all* the remaining sources simultaneously. Finally, in order to ensure that with high probability, at least one of the leaders is indeed honest, we need to assume that the number of players is large enough (this is where we use the assumption that the number of players is polynomially related to n).

This is the high-level idea of the proof of Theorem **??**. Note that in the above protocol (which we will refer to as the initial protocol) several players don't end up with private randomness. In particular, the players that announced their source before the "good" round may not get private randomness since, at the point of announcement, the (randomized) function of their source may actually reveal significant information about their source. This is the case since the string used as randomness in the function may actually not be random at all (as randomness is not available yet). Similarly, the leaders who were elected before the "good" round may not get private randomness. However, the number of rounds d is small (in particular, is $o(p)$), the number of players who reveal their source in each round is small (in particular, is $o(p)$), and the number of leaders elected in each round is very small. Thus, we conclude that the number of honest player that *do not* end up with private randomness is small, and is $o(p)$ according to our setting of parameters (which is significantly smaller than the number of dishonest players t, which is assumed to be a constant fraction of the total number of players).

There is another technicality that we overlooked. The initial protocol as described above, needs a broadcast channel (or alternatively, a public-key infrastructure) to execute the coin-flipping and leader election protocols. Intuitively, a broadcast channel is needed to agree on the output.

Looking closely at the protocol, we notice that we do not actually need a broadcast channel for our coin flipping protocols, since in each of these protocols only a *single* player receives an output. So, to run these coin-flipping protocols all we need is to instruct all the players to send this player a non-malleable commitment to a random string, and then reveal the random string. This player will then use the xor all these random strings as her private output.

On the other hand, the leader election protocol seems to really need a broadcast channel. To eliminate this need, we change the protocol yet again. Instead of running a single leader election protocol per round, in which all players need to agree on who the leaders are, we run p protocols in each round (one protocol per player), where in the i'th protocol only player P_i receives an output. If the output is 1 then player P_i thinks of herself as elected as leader, and if the output is 0 then she thinks of herself as a non-leader. Each of these protocols will output 1 only with small probability – if we want to elect ℓ leaders per round, then each of these protocols will output 1 with probability ℓ/p. The players will use a fresh chunk of their (supposedly) random string for each of these protocols.

5.1.1.2 The Protocol

Now we give the protocol for the computational network extractor.

Protocol 5.1.1. For a synchronous computational network

Player Inputs: Each player P_i has an (n,k) source $x_i \in \{0,1\}^n$, with $k = n^\epsilon$.
Player Outputs: Each player P_i outputs a (private random) string $s_i \in \{0,1\}^{n^\tau}$, where $\tau \triangleq \epsilon/4$.

Sub-Routines and Parameters:

1. IExt as in Theorem 3.5.9, and $\mathsf{BasicExt}$ as in Theorem 3.5.11.

2. Function $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ such that for every $r \in \{0,1\}^n$ the function $F(\cdot, r)$ is injective, and for every $x, y \in \{0,1\}^n$, $F(x,U) \approx F(y,U)$, where $U \in_R \{0,1\}^n$. For example, take $F(x,u) = x \oplus u$.

Let $d \triangleq p^{1/3}$ and $\delta \triangleq p^{2/3}$.[a]
The protocol consists of d rounds. Round $j \in [d]$ proceeds as follows:

1. Players $P_{(j-1)\delta+1}, \ldots, P_{j\delta}$ broadcast their j'th round sources (where the 1'st round sources are the input sources). Denote these δ strings by $y_1, \ldots, y_\delta$. We think of these strings as (n, k_0) sources, where $k_0 = n^{\epsilon/2}$.[b]

2. Let $\mathsf{IExt} : (\{0,1\}^n)^u \to \{0,1\}^{k_0}$ be an extractor as in Theorem 3.5.9.

   For each $i_1, \ldots, i_u \in [\delta]$ compute $m_{i_1,\ldots,i_u} \triangleq \mathsf{IExt}(y_{i_1}, \ldots, y_{i_u})$, and let M be the matrix whose $(i_1, \ldots, i_u)$-row is $m_{i_1,\ldots,i_u}$. Note that M is a (δ^u, k_0)-matrix.

3. Let $\mathsf{BasicExt} : \{0,1\}^n \times \{0,1\}^{\delta^u \cdot k_0} \to \{0,1\}^m$ be an extractor as in Theorem 3.5.11,[c] where $m = k_0 - k_0^{\Omega(1)}$. Each player P_i does the following:

   (a) Compute $z_i^j = \mathsf{BasicExt}(x_i^j, M)$ where x_i^j is the j'th round source of player P_i.

   (b) Parse $z_i^j = (s_i^j, w_i^j, v_i^j, \cdot)$, where $s_i^j, w_i^j, v_i^j \in \{0,1\}^{n^\tau}$.

   (c) Save s_i^j as private randomness.

4. Each player P_i partitions her string w_i^j into p disjoint parts,[d] and uses the k'th part to run a secure multi-party coin-tossing protocol in which only player P_k gets a (private) n-bit output. For each player P_i, we denote its output by $r_i^j \in \{0,1\}^n$.

5. Run a secure multi-party protocol for electing $\ell \triangleq p^{1/6}$ random leaders in $\{j\delta+1, \ldots, p\}$, where the randomness of player P_i is v_i^j. Denote this set of leaders by L^j.[e]

6. Start round $j+1$, where each player $P_i \notin L^j$ uses $x_i^{j+1} \triangleq F(x_i^j, r_i^j)$ as his input source, and each player $P_i \in L^j$ uses the zero string as his input source.

Once the d rounds terminate, each player P_i outputs $s_i = \bigoplus_{j=1}^d s_i^j$.

[a] We assume for the sake of simplicity that $p^{1/3}$ and $p^{2/3}$ are integers.
[b] We fix the parameters so that, with high probability, the j'th round source has min-entropy k_0 (given the transcript in all previous rounds).
[c] We fix the parameters so that $\delta^u \le k_0^\gamma$ for some $\gamma \in (0, 1/2)$, as required in Theorem 3.5.11.
[d] Note that w_i^j can indeed be partitioned to p disjoint parts, each of size $n^{O(1)}$, since $p \le n^{\epsilon/4u} << n^\tau$.
[e] If in any of these protocols there was an abort, restart without the aborting (malicious) player.

5.1.1.3 The Analysis

Here we presenet the proof of Theorem 5.0.14. We show that Protocol 5.1.1 is a computational network extractor with the desired parameters.

Throughout the proof we use the following notation. For any $j \in [d]$, let

- $B^j = \{B_i^j\}_{i \in [p]}$, where B_i^j is the concatenation of all the strings broadcasted by player P_i in all rounds $\leq j$. Thus, B^j consists of all the strings broadcasted by all the players in all rounds $\leq j$.
- $S^j = \{S_i^t\}_{i \in [p], t \leq j}, W^j = \{W_i^t\}_{i \in [p], t \leq j}$, and $V^j = \{V_i^t\}_{i \in [p], t \leq j}$, where S_i^t, W_i^t, V_i^t were computed by player P_i in Step 3(b) of round t.

Fix a parameter

$$h \triangleq \delta^{3/4} = p^{1/2}.$$

Let $j \in [d]$ be the *first* round such that the set of players $\{P_{(j-1)\delta+1}, \ldots, P_{j\delta}\}$ (who reveal their j'th round source in the beginning of round j) contains at least h honest players. The existence of such j follows from the fact that there are d rounds, a total of $p - t$ honest players, and $h \leq \frac{p-t}{d}$ (for large enough n). Moreover, by definition of j,

$$h(j-1) + (d - (j-1))\delta \geq p - t = (1 - \beta)p.$$

Applying basic algebraic manipulations, we conclude that for every constant $\gamma > \beta$ (and for every large enough n),

$$j - 1 \leq \frac{\beta p}{\delta - h} \leq \gamma d. \tag{5.1}$$

Throughout the proof, we think of j as fixed. By definition of j, there are at least u honest players in

$$\{P_{(j-1)\delta+1}, \ldots, P_{j\delta}\} \setminus (L^1 \cup \ldots \cup L^{j-1})$$

(where $u = O(\frac{\log n}{\log k_0})$ is defined as in Theorem 3.5.9). This follows from the fact that for every constant $\gamma \in (\beta, 1)$ (and for every large enough n),

$$|L^1 \cup \ldots \cup L^{j-1}| \leq (j-1)\ell \leq \gamma d\ell = \gamma p^{1/3} p^{1/6} = \gamma p^{1/2} \leq h - u,$$

where the first inequality follows from the union bound, and the second inequality follows from Equation (5.1).

Let

$$\mathcal{G} \triangleq \{i > j\delta : \ P_i \text{ is honest } \wedge \ P_i \notin L^1 \cup \ldots \cup L^{j-1}\}.$$

Note that

$$|\mathcal{G}| \geq p - t - (j-1)h - \delta - (j-1)\ell \geq (1-\beta)p - (j-1)(h+\ell) - \delta \geq (1-\beta)p - \gamma d(h+\ell) - \delta.$$

By our choice of parameters, for every constant $\gamma > \beta$ (and for every large enough n),

$$|\mathcal{G}| \geq (1-\gamma)p.$$

Therefore, what remains to show is that

$$\big(B, (S_i)_{i\in\mathcal{G}}\big) \approx \big(B, (U_i)_{i\in\mathcal{G}}\big), \tag{5.2}$$

where $(U_i)_{i\in\mathcal{G}}$ are independent random variables uniformly distributed in $\{0,1\}^{n^\tau}$.

Assume for the sake of contradiction that there exists a non-uniform PPT adversary $\mathcal{A}_1$ and a polynomial q such that for infinitely many n's,

$$\big|\Pr[\mathcal{A}_1\big(B, (S_i)_{i\in\mathcal{G}}\big) = 1] - \Pr[\mathcal{A}_1\big(B, (U_i)_{i\in\mathcal{G}}\big) = 1]\big| \geq \frac{1}{q(n)}.$$

Recall that for every honest player $P_i \in L^j$, the source X_i^{j+1} is the all zero string. Therefore, there exists a non-uniform PPT adversary $\mathcal{A}_2$, that has all the sources of the dishonest players hardwired into it, such that for infinitely many n's,

$$\begin{aligned}\Big|\Pr\Big[\mathcal{A}_2\Big(B^j, \big(\oplus_{t=1}^{j} S_i^t\big)_{i\in\mathcal{G}}, \{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}\Big) = 1\Big] - \\ \Pr\Big[\mathcal{A}_2\Big(B^j, \big(U_i\big)_{i\in\mathcal{G}}, \{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}\Big) = 1\Big]\Big| \geq \frac{1}{q(n)}.\end{aligned}$$

Denote by $\mathcal{T} \subseteq [p]$ the set of corrupted players. We say that a tuple

$$(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}}) \leftarrow (B^{j-1}, S^{j-1}, W^{j-1}, V^j, Y_1, \ldots, Y_\delta, \{R_i^j\}_{i\in\mathcal{T}})$$

is GOOD if the following three properties are satisfied (where $y_1, \ldots, y_\delta$ are the strings broadcasted in the beginning of the j'th round, and r_i^j is the string that player P_i gets from the coin-tossing protocol in step 4 of round j).

1. There are infinitely many n's for which,

$$\begin{aligned}&\Big|\Pr\Big[\mathcal{A}_2\Big(B^j, \big(\oplus_{t=1}^{j} S_i^t\big)_{i\in\mathcal{G}}, \{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}\Big) = 1\Big] - \\ &\Pr\Big[\mathcal{A}_2\Big(B^j, (U_i)_{i\in\mathcal{G}}, \{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}\Big) = 1\Big]\Big| \geq \frac{1}{2q(n)},\end{aligned}$$

   where all the random variables are conditioned on $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}})$.

2.
$$|((S_i^j)_{i\in\mathcal{G}}, (\tilde{W}_i^j)_{i\in\mathcal{G}}) - ((U_i^1)_{i\in\mathcal{G}}, (U_i^2)_{i\in\mathcal{G}})| = \text{negl}(n) \tag{5.3}$$

   where each $\tilde{W}_i^j$ is a truncated form of W_i^j, while omitting the parts that are used to generate $\{r_i^j\}_{i\in\mathcal{T}}$. The random variables $\left(S_i^j, \tilde{W}_i^j\right)_{i\in\mathcal{G}}$ are conditioned on

$$(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1}, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}}),$$

   and the random variables $(U_i^1)_{i\in\mathcal{G}}, (U_i^2)_{i\in\mathcal{G}}$ are independent; each U_i^1 is uniformly distributed in $\{0,1\}^{n^\tau}$, and each U_i^2 is uniformly distributed in $\{0,1\}^{n^\tau\left(1-\frac{|\mathcal{T}|}{p}\right)}$.

3. There is at least one (honest) player in $\mathcal{G} \cap L^j$.

Claim 5.1.2. *There exists a* GOOD *tuple.*

Proof. A standard probabilistic argument shows that the probability that a random tuple

$$(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}}) \leftarrow (B^{j-1}, S^{j-1}, W^{j-1}, V^j, Y_1, \ldots, Y_\delta, \{R_i^j\}_{i\in\mathcal{T}})$$

satisfies the first property with probability at least $\frac{1}{2p(n)}$. We next show that the second and third properties each hold with probability $1 - \text{negl}(n)$. The fact that the third property holds with probability $1 - \text{negl}(n)$ follows from the fact that $|L^j| = p^{1/6}$, the number of honest players in $\{j\delta + 1, \ldots, p\}$ is $O(p)$, the total number of players p is polynomially related to n, and from the fact that for a random $v^j \leftarrow V^j$, the set L^j is random in $\{j\delta + 1, \ldots, p\}$. It remains to prove that the second property also holds with probability $1 - \text{negl}(n)$. This part is quite involved. We start with introducing the following notation.

For a tuple $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1}) \leftarrow (B^{j-1}, S^{j-1}, W^{j-1}, V^{j-1})$ and for each $i \in \mathcal{G}$, let

$$X_i' \triangleq X_i^j|_{b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1}}.$$

Similarly, for each $i \in [\delta]$ let

$$Y'_i \triangleq Y_i|_{b^{j-1},s^{j-1},w^{j-1},v^{j-1}}.$$

We first prove the following claim.

Claim 5.1.3. *With probability* $1-\mathrm{negl}(n)$ *(over* $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$*) it holds that every random variable in* $(X'_i)_{i\in\mathcal{G}}$ *and every at least* u *random variables in* $\{Y'_1, \dots, Y'_\delta\}$ *have min-entropy* $k_0 = n^{\epsilon/2}$.

Proof. Recall that (before fixing $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$), all the random variable in $(X^j_i)_{i\in\mathcal{G}}$ and at least u random variables in $\{Y_1, \dots, Y_\delta\}$ have min-entropy $k = n^\epsilon$. Thus (using the union bound) it suffices to prove the following:

1. For every $i \in \mathcal{G}$, with probability $1 - \mathrm{negl}(n)$ (over $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$) it holds that X'_i has min-entropy $k_0 = n^{\epsilon/2}$.

2. For every $i \in \{1, \dots, \delta\}$ such that Y_i has min-entropy k, with probability $1 - \mathrm{negl}(n)$ (over $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$) it holds that Y'_i has min-entropy $k_0 = n^{\epsilon/2}$.

We prove the first item (the second item follows exactly in the same manner). Fix any $i \in \mathcal{G}$. Denote by BAD the set of all the tuples $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$ such that the min-entropy of X'_i (conditioned on $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$) is smaller than k_0. Assume for the sake of contradiction that there exists a polynomial q and infinitely many n's such that

$$\Pr[(B^{j-1}, S^{j-1}, W^{j-1}, V^{j-1}) \in BAD] > \frac{1}{q(n)}.$$

This implies that for infinitely many n's there exists a fixing of the sources of all the other players, $(X_\ell)_{\ell\neq i} = (x_\ell)_{\ell\neq i}$, such that

$$\Pr[(B^{j-1}, S^{j-1}, W^{j-1}, V^{j-1}) \in BAD] > \frac{1}{q(n)},$$

where the above probability is conditioned on $(x_\ell)_{\ell\neq i}$.

Recall that for every $t \in [d]$ it holds that $|s^t_i| = |w^t_i| = |v^t_i| = n^\tau$. Therefore, after fixing $(x_\ell)_{\ell\neq i}$, the support of $(B^{j-1}, S^{j-1}, W^{j-1}, V^{j-1})$ is of size at most 2^{3dn^τ}. Thus, by applying the union bound, this implies that for infinitely many n's there exists $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1}) \in BAD$ such that

$$\Pr[(B^{j-1}, S^{j-1}, W^{j-1}, V^{j-1}) = (b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})] \geq \frac{2^{-3dn^{\tau}}}{q(n)}.$$

For each of the above n's, fix $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1}) \in BAD$ as above. By definition of BAD, there exists x such that

$$\Pr[X'_i = x] \geq 2^{-k_0},$$

where the probability is conditioned on $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$ and on $(x_\ell)_{\ell \neq i}$, as above.

Thus,

$$\Pr[X_i^j = x] \geq \Pr[X'_i = x | (B^{j-1}, S^{j-1}, W^{j-1}, V^{j-1}) = (b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})] \cdot \tag{5.4}$$

$$\Pr[(B^{j-1}, S^{j-1}, W^{j-1}, V^{j-1}) = (b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})] \geq \tag{5.5}$$

$$2^{-k_0} \cdot \frac{2^{-3dn^{\tau}}}{q(n)} > 2^{-k}, \tag{5.6}$$

where the latter inequality follows from our parameter settings $\tau = \epsilon/4$ and $p \leq n^{\frac{\epsilon}{4u}}$. This contradicts the fact that the min-entropy of X_i^j is k, which in turn follows from the fact that for every $r \in \{0,1\}^{n^{\tau}}$ the function $F(\cdot, r)$ is injective. $\square$

Next, we claim that the random variables $(X'_i)_{i \in \mathcal{G}}$ are all independent, and are independent of $(Y'_i)_{i=1}^{\delta}$. This can be seen by induction on j. In particular, this implies that the random variables $(X'_i)_{i \in \mathcal{G}}$ are independent of the matrix

$$M' \triangleq M|_{B^{j-1}=b^{j-1}, S^{j-1}=s^{j-1}, W^{j-1}=w^{j-1}, V^{j-1}=v^{j-1}},$$

computed in Step 2 of round j. Thus, using Theorem 3.5.9 we conclude that with probability $1 - \text{negl}(n)$ (over $(b^{j-1}, s^{j-1}, w^{j-1}, v^{j-1})$) it holds that M' is $2^{-n^{\Omega(1)}}$-close to a SR-matrix. Let

$$Z'_i \triangleq \mathsf{BasicExt}(X'_i, M'),$$

as computed by P_i in Step $3(a)$ of round j. Then, with probability $1 - \text{negl}(n)$, for every $i \in \mathcal{G}$, the random variable Z'_i is $2^{-n^{\Omega(1)}}$-close to a uniform string. Moreover, the fact that $\mathsf{BasicExt}$ is a *strong* extractor implies that, with probability $1 - \text{negl}(n)$, for every $i \in \mathcal{G}$,

$$|(Y'_1, \ldots, Y'_\delta, Z'_i) - ((Y'_1, \ldots, Y'_\delta, U_i)| < 2^{-n^{\Omega(1)}}.$$

This, together with the fact that the random variables $(Z'_i)_{i\in\mathcal{G}}$ are all independent conditioned on $(Y'_1,\ldots,Y'_\delta)$, implies that with probability $1-\mathrm{negl}(n)$,

$$|(Y'_1,\ldots,Y'_\delta,(Z'_i)_{i\in\mathcal{G}})-(Y'_1,\ldots,Y'_\delta,(U_i)_{i\in\mathcal{G}})|<|\mathcal{G}|2^{-n^{\Omega(1)}}<p2^{-n^{\Omega(1)}}=2^{-n^{\Omega(1)}}.$$

where the first inequality follows from a standard hybrid argument. Therefore, with probability $1-\mathrm{negl}(n)$ over

$$(b^{j-1},s^{j-1},w^{j-1},v^{j-1},y_1,\ldots,y_\delta)\leftarrow(B^{j-1},S^{j-1},W^{j-1},V^{j-1},Y_1,\ldots,Y_\delta),$$

it holds that

$$|(Z_i^j|_{b^{j-1},s^{j-1},w^{j-1},v^{j-1},y_1,\ldots,y_\delta})_{i\in\mathcal{G}}-(U_i)_{i\in\mathcal{G}}|=2^{-n^{\Omega(1)}}. \tag{5.7}$$

Recall that

$$Z_i^j=(S_i^j,W_i^j,V_i^j,\cdot).$$

Therefore, Equation (5.7) implies in particular that, with probability $1-\mathrm{negl}(n)$ over

$$(b^{j-1},s^{j-1},w^{j-1},v^{j-1},y_1,\ldots,y_\delta)\leftarrow(B^{j-1},S^{j-1},W^{j-1},V^{j-1},Y_1,\ldots,Y_\delta),$$

it holds that

$$|((S_i^j)_{i\in\mathcal{G}},(W_i^j)_{i\in\mathcal{G}},(V_i^j)_{i\in\mathcal{G}})-((U_i^1)_{i\in\mathcal{G}},(U_i^2)_{i\in\mathcal{G}},(U_i^3)_{i\in\mathcal{G}})|=2^{-n^{\Omega(1)}}.$$

where all the random variables are conditioned on $(b^{j-1},s^{j-1},w^{j-1},v^{j-1},y_1,\ldots,y_\delta)$. This implies that, with probability $1-\mathrm{negl}(n)$ over

$$(b^{j-1},s^{j-1},w^{j-1},v^{j},y_1,\ldots,y_\delta)\leftarrow(B^{j-1},S^{j-1},W^{j-1},V^{j},Y_1,\ldots,Y_\delta),$$

it holds that

$$|((S_i^j)_{i\in\mathcal{G}},(W_i^j)_{i\in\mathcal{G}})-((U_i^1)_{i\in\mathcal{G}},(U_i^2)_{i\in\mathcal{G}})|=2^{-n^{\Omega(1)}},$$

where all the random variables are conditioned on $(b^{j-1},s^{j-1},w^{j-1},v^{j},y_1,\ldots,y_\delta)$. Finally, this in turn implies that, with probability $1-\mathrm{negl}(n)$ over

$$(b^{j-1},s^{j-1},w^{j-1},v^{j-1},y_1,\ldots,y_\delta,\{r_i^j\}_{i\in\mathcal{T}})\leftarrow(B^{j-1},S^{j-1},W^{j-1},V^{j-1},Y_1,\ldots,Y_\delta,\{R_i^j\}_{i\in\mathcal{T}}),$$

it holds that

$$|((S_i^j)_{i\in\mathcal{G}},(\tilde{W}_i^j)_{i\in\mathcal{G}})-((U_i^1)_{i\in\mathcal{G}},(U_i^2)_{i\in\mathcal{G}})|=2^{-n^{\Omega(1)}},$$

where all the random variables are conditioned on $(b^{j-1},s^{j-1},w^{j-1},v^{j},y_1,\ldots,y_\delta,\{r_i^j\}_{i\in\mathcal{T}})$, as desired.

□

Fix a GOOD tuple $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}})$. Parse

$$B^j = (B^{j-1}, Y_1, \ldots, Y_\delta, VIEW_1, VIEW_2),$$

where $VIEW_1$ consists of the transcripts of all the secure coin-flipping protocols, and $VIEW_2$ is the transcript of the leader electing protocol.

The first property of a GOOD tuple implies that there exists a non-uniform PPT adversary $\mathcal{A}_3$ (that has the values $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}})$ hardwired into it) such that for infinitely many n's,

$$\begin{aligned}\Big| \Pr\Big[\mathcal{A}_3\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, VIEW_1, VIEW_2, \big(\oplus_{t=1}^{j} S_i^t\big)_{i\in\mathcal{G}}\Big) = 1\Big] - \\ \Pr\Big[\mathcal{A}_3\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, VIEW_1, VIEW_2, (U_i)_{i\in\mathcal{G}}\Big) = 1\Big]\Big| \geq \frac{1}{2q(n)},\end{aligned}$$

where all the random variables are conditioned on $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}})$.

The fact that s^{j-1} is fixed implies that there exists a non-uniform PPT adversary $\mathcal{A}_4$ such that for infinitely many n's,

$$\begin{aligned}\Big| \Pr\Big[\mathcal{A}_4\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, VIEW_1, VIEW_2, \big(S_i^j\big)_{i\in\mathcal{G}}\Big) = 1\Big] - \\ \Pr\Big[\mathcal{A}_4\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, VIEW_1, VIEW_2, (U_i)_{i\in\mathcal{G}}\Big) = 1\Big]\Big| \geq \frac{1}{2q(n)},\end{aligned}$$

where all the random variables are conditioned on $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}})$.

The fact that the coin tossing protocols are *secure* implies that $VIEW_1$ can be simulated. Namely, there exists a non-uniform PPT adversary $\mathcal{A}_5$ such that for infinitely many n's,

$$\begin{aligned}\Big| \Pr\Big[\mathcal{A}_5\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, \big(R_i^j\big)_{i\in\mathcal{T}}, VIEW_2, \big(S_i^j\big)_{i\in\mathcal{G}}\Big) = 1\Big] - \\ \Pr\Big[\mathcal{A}_5\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, \big(R_i^j\big)_{i\in\mathcal{T}}, VIEW_2, (U_i)_{i\in\mathcal{G}}\Big) = 1\Big]\Big| \geq \frac{1}{2q(n)} - \mathrm{negl}(n),\end{aligned}$$

where all the random variables are conditioned on $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\in\mathcal{T}})$.

The fact that the random variables $\big(R_i^j\big)_{i\in\mathcal{T}}$ are fixed to $\big(r_i^j\big)_{i\in\mathcal{T}}$ implies that there exists a non-uniform PPT adversary $\mathcal{A}_6$ such that for infinitely many n's,

$$\begin{aligned}\Big| \Pr\Big[\mathcal{A}_6\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, VIEW_2, \big(S_i^j\big)_{i\in\mathcal{G}}\Big) = 1\Big] - \\ \Pr\Big[\mathcal{A}_6\Big(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, VIEW_2, (U_i)_{i\in\mathcal{G}}\Big) = 1\Big]\Big| \geq \frac{1}{2q(n)} - \mathrm{negl}(n),\end{aligned}$$

where all the random variables are conditioned on $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i \in \mathcal{T}})$.

The fact that the leader election protocol is secure implies that there exists a non-uniform PPT adversary $\mathcal{A}_7$ such that for infinitely many n's,

$$\begin{aligned}&\Big|\Pr\left[\mathcal{A}_7\left(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, L^j, (S_i^j)_{i\in\mathcal{G}}\right) = 1\right] - \\ &\Pr\left[\mathcal{A}_7\left(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, L^j, (U_i)_{i\in\mathcal{G}}\right) = 1\right]\Big| \geq \frac{1}{2q(n)} - \text{negl}(n).\end{aligned}$$

This, together with the fact that the random variable V^j is fixed to v^j implies that there exists a non-uniform PPT adversary $\mathcal{A}_8$ such that for infinitely many n's,

$$\begin{aligned}&\Big|\Pr\left[\mathcal{A}_8\left(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, (S_i^j)_{i\in\mathcal{G}}\right) = 1\right] - \\ &\Pr\left[\mathcal{A}_8\left(\{F(X_i^j, R_i^j)\}_{i\in\mathcal{G}\setminus L^j}, (U_i)_{i\in\mathcal{G}}\right) = 1\right]\Big| \geq \frac{1}{2q(n)} - \text{negl}(n).\end{aligned}$$

Fix any $\alpha \in \mathcal{G} \cap L^j$ (the existence of such α follows from the fourth property of GOOD). In the remaining of the proof we contradict the above equation by proving that

$$\{(S_i^j)_{i\in\mathcal{G}}, (F(X_i^j, R_i^j))_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}} \approx \{(U_i)_{i\in\mathcal{G}}, (F(X_i^j, R_i^j))_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}}.$$

To this end we prove the following three equations (and use a hybrid argument):

$$\{(S_i^j)_{i\in\mathcal{G}}, (F(X_i^j, R_i^j))_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}} \approx \{(S_i^j)_{i\in\mathcal{G}}, (F(0^n, U_i^2)_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}}, \tag{5.8}$$

$$\{(S_i^j)_{i\in\mathcal{G}}, (F(0^n, U_i^2)_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}} \approx \{(U_i^1)_{i\in\mathcal{G}}, (F(0^n, U_i^2)_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}}, \tag{5.9}$$

and

$$\{(U_i^1)_{i\in\mathcal{G}}, (F(0^n, U_i^2)_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}} \approx \{(U_i)_{i\in\mathcal{G}}, (F(X_i^j, R_i^j))_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}}. \tag{5.10}$$

Equation (5.9) follows immediately from the second property of GOOD. Assume for the sake of contradiction that Equation (5.8) does not hold. This (contradiction) assumption, together with the second property of GOOD, implies that there exists $(s_i^j)_{i\in\mathcal{G}}$ such that

$$|(\tilde{W}_i^j)_{i\in\mathcal{G}} - (U_i)_{i\in\mathcal{G}}| = \text{negl}(n), \tag{5.11}$$

and

$$\{(F(X_i^j, R_i^j))_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}} \not\approx \{(U_i)_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}}, \tag{5.12}$$

where all the random variables are conditioned on $(S_i^j)_{i\in\mathcal{G}} = (s_i^j)_{i\in\mathcal{G}}$ (and on the GOOD tuple $(b^{j-1}, s^{j-1}, w^{j-1}, v^j, y_1, \ldots, y_\delta, \{r_i^j\}_{i\imath\mathcal{T}})$). Assume for the sake of simplicity of the analysis that after the above fixing of $(S_i^j)_{i\in\mathcal{G}} = (s_i^j)_{i\in\mathcal{G}}$, all the random variables $(\tilde{W}_i^j)_{i\imath\mathcal{G}}$ are independent and uniformly distributed (rather than being statistically close to such, as implied by Equation (5.11)). In particular, the part of W_α^j that is used to generate $(R_i^j)_{i\in\mathcal{G}}$ is uniformly distributed and is independent of $(X_i^j)_{i\in\mathcal{G}\setminus\{\alpha\}}$. Thus $(R_i^j)_{i\in\mathcal{G}}$ is uniformly distributed conditioned on $(X_i^j)_{i\in\mathcal{G}\setminus\{\alpha\}}$.[1] This, together with the semantic security of F (i.e., the property that for every $x, y \in \{0,1\}^n$ it holds that $F(x,U) \approx F(y,U)$), implies that

$$\{(F(X_i^j, R_i^j))_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}} \approx \{(F(0^n, U_i)_{i\in\mathcal{G}\setminus\{\alpha\}}\}_{n\in\mathbb{N}},$$

contradicting Equation (5.12). The proof that Equation (5.10) holds is very similar, and is therefore omitted. □

5.1.2 Computational Network Extractors under Stronger Assumptions

In this section we show that we can construct better computational network extractors under the assumption that there exist one-way permutations for weak random sources. Indeed, we will show that in this case we can guarantee that all honest processors end up with private random bits. We construct two network extractor protocols. One for the case where each player has an independent source with linear entropy δn, and one for the case where each player has an independent source with polynomially-small entropy n^δ.

5.1.2.1 Computational Network Extractors for Linear Min-Entropy

The construction of such a computational network extractor relies on a two-source extractor constructed based on the assumption that there exist one-way permutations for weak random sources. The construction and the analysis of the two source extractor will be presented in the next chapter, while here we focus on the construction of the computational network extractor. To this end, we state the theorem of the two source extractor as below.

Theorem 5.1.4. *Fix a constant $\alpha > 0$ and parameters $t = \frac{4}{\alpha}$ and $k \geq n^{\Omega(1)}$. Assume that there exists a polynomial time computable permutation $f : \{0,1\}^n \to \{0,1\}^n$ such that for any $(n, 0.3k)$-source Y, any circuit of size $\mathrm{poly}(n^{\log n})$ can invert $f(Y)$ with only negligible probability. Then there*

[1]Note that the strings $(R_i^j)_{i\in\mathcal{G}}$ are uniformly distributed, even though the dishonest players may have tried to skew the output by aborting. This is the case since for any $i \in \mathcal{G}$, in the i'th coin flipping protocol only player P_i gets an output R_i^j, and thus an abort occurs independently of the value of R_i^j.

exists a polynomial time computable function $\mathsf{TExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ *such that for any* $(n, \alpha n)$*-source* X*, any* (n,k)*-source* Y *that is independent of* X*, and any deterministic function* h *(not necessarily computable in polynomial time) on* $\{0,1\}^n$

$$(\mathsf{TExt}(X,Y), X, h(X), f^{(t+1)}(Y)) \approx (U_m, X, h(X), f^{(t+1)}(Y)).$$

We first present a protocol where all the honest players *except one* end up with private randomness. Note that if we knew of one player j that is honest, then the protocol would be very simple: Player j will simply reveal his source, and all other players would apply the 2-source extractor TExt (presented above) to this source and to their own source. The fact TExt is a strong extractor immediately implies that all the honest players, except for player j, would end up with private randomness. However, since we don't know of any player that is honest, it is tempting to try the following approach.

Naive Network Extractor Protocol for Linear Min-Entropy

1. The protocol proceeds in p rounds, where p is the number of players. In round i:
   (a) Player i sends x_i to all other players.
   (b) Each player ℓ computes $r^i_\ell = \mathsf{TExt}(x_i, x_\ell)$.
2. Each player i outputs the bitwise xor $r^1_i \oplus \cdots \oplus r^p_i$.

Let j denote the first honest player. Then, for every player i different than j, r^j_i is uniform. Despite this, the output of player i may not be random, and may even be a fixed constant. The problem is that the random variables $r^1_i, \ldots, r^p_i$ are not independent, and a malicious adversary can actually cause the output to be a fixed constant.

The idea is to get around this problem by using computational assumptions. To this end, each player ℓ, instead of using the same source X_ℓ in each round, will use the source $f^i(X_\ell)$ in round ℓ. However, for this approach to work we need the guarantee that

$$(\mathsf{TExt}(X_i, X_\ell), X_\ell, f(X_i)) \approx (\mathsf{Uniform}, X_\ell, f(X_i)).$$

Our extractor TExt does not satisfy this, but luckily our extractor does satisfty the following (similar) guarantee:

$$(\mathsf{TExt}(X_i, X_j), X_j, f^{t+1}(X_i)) \approx (\mathsf{Uniform}, X_j, f^{t+1}(X_i)),$$

where t is a constant that depends on δ.

So, instead we consider the following network extractor protocol, which has the guarantee that all the honest players, except for the first one, end up with private random-looking strings.

Lossy Network Extractor Protocol for Linear Min-Entropy

1. Let $g = f^{t+1}$. The protocol proceeds in p rounds, where p is the number of players. In round i:
   (a) Player i sends $g^i(x_i)$ to all other players.
   (b) Each player ℓ computes $r_\ell^i = \mathsf{TExt}(g^i(x_i), g^i(x_\ell))$.
2. Each player i outputs the bitwise xor $r_i^1 \oplus \cdots \oplus r_i^p$.

Now we can prove that all the honest players, except player j (who is the first honest player), end up with private random-looking strings. The analysis proceeds in three steps.

1. We first fix all sources sent before round j, and we fix $\{r_\ell^i\}_{\ell\in[p],i<j}$, which were all computed before round j. We claim that even conditioned on all these fixings, the sources are still independent, and with high probability they all have "enough" entropy left.
2. Next, we claim that the strings $\{r_\ell^j\}$ of all the honest players $\ell \neq j$, are independent and uniformly distributed.
3. Finally, we claim that the rest of the r_ℓ^i for $i > j$ are (computationally) independent of $\{r_\ell^j\}$, which implies that the output of all the honest players, except player j, are computationally indistinguishable from random. For this we use the fact that for any two independent variables Y_i and Y_j with "sufficient" entropy

$$(\mathsf{TExt}(Y_i, Y_j), Y_j, g(Y_i)) \approx (\mathsf{Uniform}, Y_j, g(Y_i)). \tag{5.13}$$

Note that in the protocol above, player j, who is the first honest player, does not necessarily end with private randomness. To fix this, we add another phase to the protocol. So, the protocol consists of two phases. In the first phase, the players run the (lossy) protocol presented above. In the second phase, the idea is that all the honest players use their (supposedly) random string, generated in the first phase, to run a coin flipping protocol and generate a public random-looking seed V. Recall that we assumed that there are at least two honest players, therefore there is at least one honest player besides player j. Thus, V is indeed random-looking. Finally, each player i will extract randomness from his own source X_i using the seed V.

This approach would indeed work if there were at least *three* honest players, since in that case we could argue that V is random-looking and is *independent* of each of the sources X_i. Thus, we could use it to extract (private) randomness from each source.

However, if there are only *two* honest players then this approach does not seem to work, since in this case we cannot argue that V is independent of all the sources. Indeed if there is a single honest player ℓ besides player j (who is the first honest player) then it may be the case that V depends on the source of player ℓ. This is the case since player ℓ maybe the only player who used "good" randomness for the coin-flipping protocol. As before, we get around this dependence by using reconstructive properties of extractors.

Our Final Network Extractor Protocol for Linear Min-Entropy. We first note that Theorem 3.1.12 immediately implies the following corollary:

Corollary 5.1.5. *If f is a one way function for $0.3\alpha n$-sources, then there is an efficiently computable function* LExt *as in Theorem 3.1.12, with parameters $\epsilon = \frac{1}{\text{poly}(n^{\log n})}$ and $d = O(\log(n/\epsilon)) =$* $\text{polylog}(n)$*, such that for any $(n, 0.9\alpha n)$ source X,*

$$(\mathsf{LExt}(X, U_d), f(X), U_d) \approx (U_m, f(X), U_d).$$

- **Parameters.**
  - Constant $\alpha > 0$, where αn is the min-entropy of each of the input sources X_i.
  - Parameters d, ϵ as in Corollary 5.1.5 and $m = \text{polylog}(n)$. Note $d = \text{polylog}(n)$ and $\epsilon = \text{negl}(n)$.
  - $t = \lceil \frac{4}{0.9\alpha} \rceil$.
- **Ingredients.**
  - The 2-source extractor $\mathsf{TExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ from Theorem 6.1.2.
  - $\mathsf{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ a $(0.9\alpha n, \epsilon)$-extractor as in Theorem 3.1.12.
  - $f : \{0,1\}^n \to \{0,1\}^n$ a one way permutation for $0.3\alpha n$-sources. Let $g = f^{(t+1)}$.
- **The protocol.** The protocol proceeds in two phases.

- **Phase 1.** The first phase of the protocol proceeds in p rounds (where p is the number of players). In round $j \in [p]$ the players do the following.
  1. The j'th player sends $g^{(2j)}(X_j)$ to all other players, where $g = f^{(t+1)}$ is described above.
  2. Each player i computes $R_i^j = \mathsf{TExt}(g^{(2j)}(X_j), g^{(2j)}(X_i))$. Note $|R_i^j| = m = \text{polylog}(n)$.

  At the end of the p'th round each player i computes $R_i = \oplus_{j=1}^p R_i^j$.

  We show that at the end of this phase all the honest players, except for the first one, end up with private randomness. In order to ensure that *all* the honest players, *including the first one*, end up with private randomness, we proceed to the second phase.
- **Phase 2.** Each player i partitions R_i into two equal parts $R_i = (V_i, W_i)$. All players engage in a secure multiparty computation to compute $V = \oplus_{i=1}^p V_i$[2], where each player i uses W_i as its internal randomness.

  Finally, each player i outputs $Z_i = \mathsf{LExt}(g^{(2i-1)}(X_i), V)$.

We have the following theorem.

Theorem 5.1.6. *For any $p \geq 2$, any $t \leq p-2$, and any $n \geq p^{1+\gamma}$ (for some constant $\gamma > 0$), the protocol described above is a $(t, p-t)$-computational network extractor protocol.*

Proof. Fix any p and any t such that $p-t \geq 2$. Let $\mathcal{G} \subseteq [p]$ denote the set of all honest players. Let j denote the GOOD round, where the first honest player sends its source; i.e., $\mathcal{G} \subseteq \{j, \ldots, p\}$. We assume without loss of generality that the PPT adversary (who controls all the malicious players) is deterministic. Thus, it suffices to prove that

$$\{B, (Z_i)_{i \in \mathcal{G}}\} \approx \{B, U_{gm}\}.$$

The proof proceeds in two parts. In the first part we prove that all the R_i's of all the honest players, except player j, appear to be independent and uniformly distributed, even conditioned on the entire transcript of phase 1. Actually ,we prove the following stronger statement:

$$\{X_j, (g^{(2j+1)}(X_i))_{i \in \mathcal{G} \setminus \{j\}}, (R_i)_{i \in \mathcal{G} \setminus \{j\}}\} \approx \{X_j, (g^{(2j+1)}(X_i))_{i \in \mathcal{G} \setminus \{j\}}, U_{(g-1)m}\} \tag{5.14}$$

[2]Here, if the adversary aborts the protocol, we simply discard the aborting party and restart the secure computation with fresh V_i's

Note that this statement is indeed stronger since for every $i \in \mathcal{G} \setminus \{j\}$ it holds that $i > j$, which implies that $2i > 2j + 1$, and therefore it is easy to compute $g^{(2i)}(X_i)$ from $g^{(2j+1)}(X_i)$.

In the second part we use Equation (5.14) to prove that indeed

$$\{(B, (Z_i)_{i \in \mathcal{G}}\} \approx \{(B, U_{gm}\}.$$

Part 1. Assume for the sake of contradiction that there exists a non-uniform PPT adversary $\mathcal{A}$ and a polynomial q such that for infinitely many n's

$$\begin{aligned}\big| \Pr[\mathcal{A}(X_j, (g^{(2j+1)}(X_i))_{i \in \mathcal{G}\setminus\{j\}}, (R_i)_{i \in \mathcal{G}\setminus\{j\}}) = 1] - \\ \Pr[\mathcal{A}(X_j, (g^{(2j+1)}(X_i))_{i \in \mathcal{G}\setminus\{j\}}, U_{(g-1)m}) = 1] \big| \geq \frac{1}{q(n)}.\end{aligned}$$

Let B_{j-1} denote the transcript until round $j-1$. Fix any tuple

$$\{b_{j-1}, (r_i^1, \dots, r_i^{j-1})_{i \in \mathcal{G}}\} \leftarrow \{B_{j-1}, (R_i^1, \dots, R_i^{j-1})_{i \in \mathcal{G}}\}.$$

Denote any random variable Y, conditioned on the above tuple, by Y'. We say that the tuple above is BAD if the following properties are satisfied.

1. There are infinitely many n's for which

$$\begin{aligned}\big| \Pr[\mathcal{A}(X'_j, (g^{(2j+1)}(X'_i))_{i \in \mathcal{G}\setminus\{j\}}, (R'_i)_{i \in \mathcal{G}\setminus\{j\}}) = 1] - \\ \Pr[\mathcal{A}(X'_j, (g^{(2j+1)}(X'_i))_{i \in \mathcal{G}\setminus\{j\}}, U_{(g-1)m}) = 1] \big| \geq \frac{1}{2q(n)}.\end{aligned}$$

2. For every $i \in \mathcal{G}$, the random variable X'_i has min-entropy $0.9\alpha n$ and $\{X'_i\}_i \in \mathcal{G}$ are independent random variables.

Claim 5.1.7.

$$\Pr[\{b_{j-1}, (r_i^1, \dots, r_i^{j-1})_{i \in \mathcal{G}}\} \textit{ is } \text{BAD}] \geq \frac{1}{3q(n)}.$$

Proof of Claim 5.1.7. A standard argument shows that a random tuple

$$\{b_{j-1}, (r_i^1, \ldots, r_i^{j-1})_{i \in \mathcal{G}}\} \leftarrow \{B_{j-1}, (R_i^1, \ldots, R_i^{j-1})_{i \in \mathcal{G}}\}$$

satisfies the first property with probability at least $\frac{1}{2q(n)}$.

The fact that the random variables $\{X_i'\}_{i \in \mathcal{G}}$ remain independent can be seen by induction on the number of rounds. Moreover, since all the R_i^j's are of size only $\text{polylog}(n)$ and $n \geq p^{1+\gamma}$, by Lemma 3.5.14 with probability $1 - \text{negl}(n)$ over the fixings, every X_i' has min-entropy at least $0.9\alpha n$ (take $\epsilon = 2^{-\log^2 n}$ in that lemma).

Thus the probability that a random tuple is BAD is at least $\frac{1}{2q(n)} - \text{negl}(n) \geq \frac{1}{3q(n)}$ □

Fix any BAD tuple

$$\{b_{j-1}, (r_i^1, \ldots, r_i^{j-1})_{i \in \mathcal{G}}\} \leftarrow \{B_{j-1}, (R_i^1, \ldots, R_i^{j-1})_{i \in \mathcal{G}}\}.$$

Note that $R_i = \oplus_{j=1}^p R_i^j$. Since $(R_i^1, \ldots, R_i^{j-1})_{i \in \mathcal{G}}$ are fixed, and $(R_i^{j+1}, \ldots, R_i^p)_{i \in \mathcal{G}}$ can be computed in polynomial time from $(g^{(2j+1)}(X_i'))_{i \in \mathcal{G} \setminus \{j\}}$, there exists another non-uniform PPT adversary $\mathcal{A}_1$ such that

$$\begin{aligned}\big|\, &\Pr[\mathcal{A}_1(X_j', (g^{(2j+1)}(X_i'))_{i \in \mathcal{G} \setminus \{j\}}, (R_i'^j)_{i \in \mathcal{G} \setminus \{j\}}) = 1] - \\ &\Pr[\mathcal{A}_1(X_j', (g^{(2j+1)}(X_i'))_{i \in \mathcal{G} \setminus \{j\}}, U_{(g-1)m}) = 1]\big| \geq \frac{1}{2q(n)}.\end{aligned}$$

A standard hybrid argument shows that there exists $\ell \in \mathcal{G} \setminus \{j\}$ (note there are at least 2 honest players) s.t.

$$\begin{aligned}\big|\, &\Pr[\mathcal{A}_1(X_j', (g^{(2j+1)}(X_i'))_{i \in \mathcal{G} \setminus \{j\}}, (R_i'^j)_{i \in \mathcal{G} \setminus \{j,\ell\}}, R_\ell'^j) = 1] - \\ &\Pr[\mathcal{A}_1(X_j', (g^{(2j+1)}(X_i'))_{i \in \mathcal{G} \setminus \{j\}}, (R_i'^j)_{i \in \mathcal{G} \setminus \{j,\ell\}}, U_m) = 1]\big| \geq \frac{1}{2p \cdot q(n)}.\end{aligned}$$

Since the random variables $\{X_i'\}_{i \in \mathcal{G}}$ remain independent, there is a fixing of $(X_i')_{i \in \mathcal{G} \setminus \{j,\ell\}}$ that preserves the distinguishing probability. Note after this fixing, $(R_i'^j)_{i \in \mathcal{G} \setminus \{j,\ell\}}$ is a deterministic function of X_j'. Thus, there exists another non-uniform PPT adversary $\mathcal{B}$, that has all the fixings hardwired into it, s.t.

$$\left| \Pr[\mathcal{B}(X'_j, g^{(2j+1)}(X'_\ell), R'^j_\ell) = 1] - \Pr[\mathcal{B}(X'_j, g^{(2j+1)}(X'_\ell), U_m) = 1] \right| \geq \frac{1}{2p \cdot q(n)}. \tag{5.15}$$

Note X'_j and X'_ℓ are independent, both have min-entropy at least $0.9\alpha n$ and X'_j is a deterministic function of $g^{(2j)}(X'_j)$. Moreover, recall that

$$R'^j_\ell = \mathsf{TExt}(g^{(2j)}(X'_j), g^{(2j)}(X'_\ell)).$$

Thus, according to Theorem 6.1.2 (more precisely, Equation (6.1), where $h = g^{(-2j)}$),

$$(R'^j_\ell, X'_j, f^{(t+1)}(g^{(2j)}(X'_\ell))) \approx (U_m, X'_j, f^{(t+1)}(g^{(2j)}(X'_\ell))).$$

Note $f^{(t+1)}(g^{(2j)}(X'_\ell)) = g^{(2j+1)}(X'_\ell))$, thus

$$(R'^j_\ell, X'_j, g^{(2j+1)}(X'_\ell)) \approx (U_m, X'_j, g^{(2j+1)}(X'_\ell)),$$

which contradicts Equation (5.15). Therefore, Equation (5.14) holds. Namely,

$$\{X_j, (g^{(2j+1)}(X_i))_{i \in \mathcal{G} \setminus \{j\}}, (R_i)_{i \in \mathcal{G} \setminus \{j\}}\} \approx \{X_j, (g^{(2j+1)}(X_i))_{i \in \mathcal{G} \setminus \{j\}}, U_{(g-1)m}\}$$

Part 2. We now use Equation (5.14) to prove our final statement

$$\{B, (Z_i)_{i \in \mathcal{G}}\} \approx \{B, U_{gm}\}.$$

We parse $B = (B_1, B_2)$ where B_1 denotes the transcript of Phase 1, and B_2 denotes the transcript of Phase 2. Thus, we need to prove that

$$\{B_1, B_2, (Z_i)_{i \in \mathcal{G}}\} \approx \{B_1, B_2, U_{gm}\}.$$

Recall that Phase 2 consists only of a secure multiparty computation of $V = \oplus V_i$. By the definition of secure multiparty computation, all the transcript of the second phase can be simulated given V, given all the sources of the malicious players $(X_i)_{i \notin \mathcal{G}}$, and given R_j. The reason we need to give also R_j is that during this secure computation player P_j (who is the first honest player), may not have private randomness, and therefore we think of this player as being malicious. Thus, it suffices to prove that

$$\{B_1, R_j, V, (Z_i)_{i \in \mathcal{G}}\} \approx \{B_1, R_j, V, U_{gm}\}. \tag{5.16}$$

We first notice that for every $i \in \mathcal{G} - \{j\}$,

$$2i - 1 \geq 2j + 1.$$

This follows from our assumption that j is the first honest player, and thus for every $i \in \mathcal{G} \setminus \{j\}$ it holds that $i \geq j + 1$.

This, together with Equation (5.14), implies that

$$\{X_j, (g^{(2i-1)}(X_i))_{i \in \mathcal{G}\setminus\{j\}}, (R_i)_{i \in \mathcal{G}\setminus\{j\}}\} \approx \{X_j, (g^{(2i-1)}(X_i))_{i \in \mathcal{G}\setminus\{j\}}, U_{(g-1)m}\},$$

which in turn implies that

$$\{R_j, (g^{(2i-1)}(X_i))_{i \in \mathcal{G}}, V\} \approx \{R_j, (g^{(2i-1)}(X_i))_{i \in \mathcal{G}}, U\}. \tag{5.17}$$

Remark 5.1.8. Here we would like U to be uniform, but the complete proof for this is somewhat involved: the adversary can choose to abort the protocol in the secure computation. What is true is that the indistinguishability holds with U being chosen adversarially from a set of t uniformly sampled strings. Since the number of aborts is at most the number of dishonest players (which is bounded by $\text{poly}(n)$), any adversary that can distinguish the two sides with a truly uniform U can also succeed in the case that U is distributed as we described.

Next, notice that it is easy to simulate the transcript B_1 given

$$\left((X_i)_{i \notin \mathcal{G}}, (g^{(2i)}(X_i))_{i \in \mathcal{G}}\right).$$

Therefore, to prove Equation (5.16) it suffices to prove that

$$\{R_j, (g^{(2i)}(X_i))_{i \in \mathcal{G}}, V, (\mathsf{LExt}(g^{(2i-1)}(X_i), V))_{i \in \mathcal{G}}\} \approx \{R_j, (g^{(2i)}(X_i))_{i \in \mathcal{G}}, V, U_{gm}\}.$$

This is immediately implied from Equation (5.17), from Corollary 5.1.5, and from the fact that all the sources $\{X_i\}_{i \in \mathcal{G}}$ are independent. □

5.1.2.2 Network Extractor for Polynomially Small Min-Entropy

In this section we give a computational network extractor protocol where each player has an independent (n, k) source with $k = n^\alpha$ for any constant $0 < \alpha < 1$. Our protocol works as long as there are a constant number of honest players. We assume the existence of one way permutations for $0.3n^\alpha$-sources.

Corollary 5.1.9. *If f is a one way permutation for $0.3n^{\alpha}$-sources, then there is an efficiently computable function* LExt *as in Theorem 3.1.12, with parameters $\epsilon = \frac{1}{\mathrm{poly}(n^{\log n})}$ and $d = O(\log(n/\epsilon)) =$ $\mathrm{polylog}(n)$, such that for any $(n, 0.9n^{\alpha})$ source X,*

$$(\mathsf{LExt}(X, U_d), f(X), U_d) \approx (U_m, f(X), U_d).$$

Protocol 5.1.10. For a synchronous computational network

Player Inputs: Each player P_i has a string x_i sampled from an independent (n, n^α) source X_i.
Player Outputs: Each player P_i outputs a (private random) string w_i.

Sub-Routines and Parameters:

1. IExt as in Theorem 3.5.9, and $\mathsf{BasicExt}$ as in Theorem 3.5.11.
2. $f : \{0,1\}^n \to \{0,1\}^n$ a one way permutation for $0.3n^\alpha$-sources.
3. The seeded extractor LExt as in Corollary 5.1.9.

The protocol proceeds in two phases.

- **Phase 1.** The first phase of the protocol proceeds in p rounds (where p is the number of players). In round $j \in [p]$ the players do the following.
    1. Player P_j sends $f^{(j+1)}(x_j)$ to all other players. Denote all the j strings broadcasted[a] so far by $y_1, \ldots, y_j$. The following steps apply to the remaining players (players P_i with $i > j$).
    2. Let u be the number of independent sources IExt takes. For each $i_1, \ldots, i_u \in [j]$, each player P_i computes $m_{i_1,\ldots,i_u} \triangleq \mathsf{IExt}(y_{i_1}, \ldots, y_{i_u})$. Let M_j be the matrix whose $(i_1, \ldots, i_u)$-row is $m_{i_1,\ldots,i_u}$. Note that M_j is a (j^u, k)-matrix.
    3. Each player P_i computes $e_i^j = \mathsf{BasicExt}(f^{(j)}(x_i), M_j)$ and truncates the output so that $|e_i^j| = \log^3 n$. Parse $e_i^j = (s_i^j, r_i^j)$.
    4. All players $P_i, i > j$ engage in a secure multi-party computation to compute $r^j = \oplus r_i^j$, where each player P_i uses s_i^j as its internal randomness.
    5. Each player P_i computes $z_i^j = \mathsf{LExt}(f^{(j)}(x_i), r^j)$ and truncates the output so that $|z_i^j| = O(\log^2 n)$.

    At the end of the p'th round, each player $P_i, i \in [p]$ computes $z_i = \oplus_{j=1}^p z_i^j$.

- **Phase 2.**
    1. Each player P_i parses $z_i = (s_i, r_i)$. All the players $\{P_i\}_{i \in [p]}$ engage in a secure multi-party computation to compute $r = \oplus r_i$, where each player P_i uses s_i as its internal randomness.
    2. Each player P_i computes $w_i = \mathsf{LExt}(f^{(i)}(x_i), r)$, and outputs w_i as its final output.

[a]For the sake of simplicity, think of the network as having broadcast channels, although our protocol also works in the case of point to point channels.

Theorem 5.1.11. *Let $k = n^{\alpha}$ for some constant $0 < \alpha < 1$. Let $u = O(\frac{1}{\alpha})$ be the number of independent (n, k) sources* IExt *needs. There exists a constant $0 < \gamma < 1$ such that for any p s.t. $u + 2 \leq p \leq k^{\gamma/u}$ and any $t \leq p - u - 2$, Protocol 5.1.10 is a $(t, p - t)$ computational network extractor.*

To prove the theorem, we first prove the following lemma (in the lemma and the analysis we use capital letters to denote the corresponding strings viewed as random variables):

Lemma 5.1.12. *Let $\ell \in [p]$ be the smallest element such that the set $\{P_1, \ldots, P_\ell\}$ contains at least u honest players: $P_{h_1}, \cdots, P_{h_u}$. Denote the remaining honest players by $P_{g_1}, \cdots, P_{g_v}$. Let e denote the concatenation of all e_i^j's, and b denote the concatenation of the broadcasted sources of all faulty players. Then*

$$(\{Z_{g_i}\}_{i \in [v]}, \{X_{h_i}\}_{i \in [u]}, \{f^{(\ell+1)}(X_{g_i})\}_{i \in [v]}, E, B) \approx (\{U_{g_i}\}_{i \in [v]}, \{X_{h_i}\}_{i \in [u]}, \{f^{(\ell+1)}(X_{g_i})\}_{i \in [v]}, E, B).$$

In other words, at the end of phase 1, the outputs of all the honest players, except the fist u honest players, are indistinguishable from being independent and uniform, even given $X_{h_1}, \cdots, X_{h_u}$, $f^{(\ell+1)}(X_{g_1}), \cdots, f^{(\ell+1)}(X_{g_v})$, all the E_i^j's, and all the sources broadcasted by the faulty players.

Remark 5.1.13. This statement is stronger than the statement that $Z_{g_1}, \cdots, Z_{g_v}$ are indistinguishable from being independent and uniform given all the transcript of Phase 1, because the transcript can be computed in polynomial time from $(X_{h_1}, \cdots, X_{h_u}, f^{(\ell+1)}(X_{g_1}), \cdots, f^{(\ell+1)}(X_{g_v}), E, B)$ (Note that the players $P_{g_1}, \cdots, P_{g_v}$ broadcast their sources after round ℓ, thus $g_i > \ell$. So the broadcasted sources $\{f^{(g_i+1)}(X_{g_i})\}$ can be computed efficiently from $\{f^{(\ell+1)}(X_{g_i})\}$).

Outline of the Proof. We first give an informal outline of the proof, since the proof involves a lot of notations.

We are going to fix the "good" round $\ell = h_u$, where u honest players have broadcasted their sources. We then argue that in this round, the output E_i^{ℓ} of all honest players that haven't broadcasted their sources are statistically close to uniform, independent of the transcript so far and independent of each other. To do this, note the sources broadcasted by honest players are independent, and each has min-entropy n^{α}. Thus when we apply IExt to the sources from u honest players, the output will be close to uniform, and therefore, the matrix M_ℓ in round ℓ is close to a somewhere random source. The hope is that when we apply BasicExt to M_ℓ and a remaining honest player's source, the output will be close to uniform and independent of the transcript so far by Theorem 3.5.11.

However, the transcript contains information(specifically, e_i^j's) about the remaining honest players' sources. Thus we'll have to first fix the transcript, and argue that conditioned on a TYPICAL fixing, a remaining honest player's source and M_ℓ still satisfy the conditions in Theorem 3.5.11. To do this, we first fix $(X_{h_1}, \ldots, X_{h_{u-1}})$. Note that conditioned on this fixing, $\mathsf{IExt}(X_{h_1}, \ldots, X_{h_u})$ is a deterministic function of X_{h_u}. We then show by induction on round $j < \ell$ that conditioned on any fixing of the transcript, X_{h_u} and the remaining honest players' sources are still independent. Moreover, since the size of e_i^j's are small, by Lemma 3.5.14 and Lemma 2.3.14 conditioned a typical fixing of the transcript so far, $\mathsf{IExt}(X_{h_1}, \ldots, X_{h_u})$ is close to a $(k, k - k^\beta)$ source, and any remaining honest player's source is close to an $(n, k - k^\beta)$ source, where β is the constant in Theorem 3.5.11. Thus M_ℓ is close to a $(\ell^u \times k)(k - k^\beta)$-source and is independent of any remaining honest player's source. Now note $\ell < p$, thus as long as p is small, by Theorem 3.5.11 the output E_i^ℓ of all honest players that haven't broadcasted their sources are statistically close to uniform, independent of the transcript so far and independent of each other.

Next, we argue that $Z_{g_i}^\ell$ is indistinguishable from being uniform and independent of the transcript so far, and the subsequent computations do not reveal any information about it to a computationally bounded adversary. Thus the final output of any P_{g_i} is indistinguishable from being uniform and private.

To do this, consider a particular honest player P_{g_1}. The fact that there are at least $u + 2$ honest players implies there are at least 2 honest players in $\{g_i\}$. Pick another honest player P_{g_2}. Assume for the sake of contradiction that there exists an adversary that distinguishes $Z_{g_1}^\ell$ from uniform, given the transcript and the subsequent computations. Then there is a fixing of all the transcript so far(including $E_{g_1}^\ell$) and all the sources $\{X_{g_i}\}_{i \neq \{1,2\}}$ such that conditioned on the fixing, the adversary still distinguishes $Z_{g_1}^\ell$ from uniform. Note that now all subsequent transcript can be computed in polynomial time from X_{g_2} and $f^{(\ell+1)}(X_{g_1})$. Thus there exists another adversary that distinguishes $Z_{g_1}^\ell$ from uniform given X_{g_2} and $f^{(\ell+1)}(X_{g_1})$. Recall $Z_{g_1}^\ell = \mathsf{LExt}(f^{(\ell)}(X_{g_1}), R^\ell)$ and now R^ℓ is a deterministic function of X_{g_2}. Thus Lemma 3.1.13 implies there is another adversary that distinguishes $Z_{g_1}^\ell$ from uniform given R^ℓ and $f^{(\ell+1)}(X_{g_1})$. Since $E_{g_1}^\ell$ is statistically close to uniform, the property of the secure multiparty computation guarantees that R^ℓ is indistinguishable from being uniform and independent of X_{g_1}. Thus the existence of the above adversary contradicts Theorem 3.1.12.

Proof of Lemma 5.1.12. We assume without loss of generality that the PPT adversary (who controls all the malicious players) is deterministic. Thus, it suffice to prove

$$(\{Z_{g_i}\}_{i\in[v]}, \{X_{h_i}\}_{i\in[u]}, \{f^{(\ell+1)}(X_{g_i})\}_{i\in[v]}, E) \approx (\{U_{g_i}\}_{i\in[v]}, \{X_{h_i}\}_{i\in[u]}, \{f^{(\ell+1)}(X_{g_i})\}_{i\in[v]}, E) \quad (5.18)$$

Note that after round ℓ, there are u honest players who have already broadcasted their sources. The fact that f is deterministic and injective implies that $Y_{h_1}, \cdots, Y_{h_u}$ are all independent and each has min-entropy $k = n^{\alpha}$. Thus by Theorem 3.5.9, $M_{h_1,\cdots,h_u} = \mathsf{IExt}(Y_{h_1}, \cdots, Y_{h_u})$ is $2^{-k^{\Omega(1)}}$-close to being uniform. We now introduce some notations:

- $E^j = \{E_i^q\}_{i\in[p],q\le j}$, $Z^j = \{Z_i^q\}_{i\in[p],q\le j}$, where E_i^q is computed by player P_i in step 3 of round q in phase 1, and Z_i^q is computed by player P_i in step 5 of round q in phase 1. Thus, E^j consists of all E_i^q's computed by all players in all rounds $\le j$, Z^j consists of all Z_i^q's of all players in all rounds $\le j$.

Now fix

$$(x_{h_1}, \ldots, x_{h_{u-1}}) \leftarrow (X_{h_1}, \ldots, X_{h_{u-1}})$$

$$(e^{\ell-1}, z^{\ell-1}) \leftarrow (E^{\ell-1}, Z^{\ell-1})$$

For any random variable Z, we denote by Z' the random variable Z conditioned on these fixings. Let TYPICAL denote the event that conditioned on these fixings, the following properties are satisfied:

- $X'_{h_u}, X'_{g_1}, \cdots, X'_{g_v}$ are independent random variables.
- $M'_{h_1,\cdots,h_u}$ is $2^{-k^{\Omega(1)}}$-close to having min-entropy $k - k^{\beta}$.
- $\forall i \in [v]$, X'_{g_i} has min-entropy $k - k^{\beta}$.

Here β is the parameter in Theorem 3.5.11. We have

Claim 5.1.14.

$$\Pr[\text{TYPICAL}] = 1 - \text{negl}(n).$$

The proof of this claim is by induction on $j < \ell$ and is very similar to the proofs of Claim 6.1.4 and Claim 6.1.5, therefore we omit the details here. The only difference is that initially $M_{h_1,\cdots,h_u}$ is only $2^{-k^{\Omega(1)}}$-close to being uniform(having min-entropy k). Thus when dealing with it we need to use Lemma 2.3.14 instead of Lemma 3.5.14.

Now, further fix

$$(x'_{h_u} \leftarrow X'_{h_u})$$

For any random variable Z', we let $Z'' = Z'|(X'_{h_u} = x'_{h_u})$. Let TYPICAL2 denote the event that conditioned on all the above fixings, the following properties are satisfied:

- $\forall i \in [v]$, $(E^{\ell}_{g_i})''$ is $2^{-k^{\Omega(1)}}$-close to being uniform, and is a deterministic function of X''_{g_i}.
- $X''_{g_1}, \cdots, X''_{g_v}$ are independent random variables.

Claim 5.1.15. *If* TYPICAL *holds, then*

$$\Pr[\text{TYPICAL2}] = 1 - \text{negl}(n).$$

As before, the proof of this claim is very similar to the proofs of Claims 6.1.4 and 6.1.5, and therefore we omit the details here. One thing that needs to be noted is that M_ℓ is a $\ell^u \times k$ matrix. Thus as long as $p < k^{\gamma/u}$, where γ is the constant in Theorem 3.5.11, the claim follows from Theorem 3.5.11.

Now, assume for the sake of contradiction that Equation (5.18) does not hold. Namely, assume that there exists a polynomial time non-uniform adversary $\mathcal{A}_1$ and there exists a polynomial q such that for infinitely many n's,

$$\begin{aligned}&|\Pr[\mathcal{A}_1(\{Z_{g_j}\}_{j\in[v]}, \{X_{h_j}\}_{j\in[u]}, \{f^{(\ell+1)}(X_{g_j})\}_{j\in[v]}, E) = 1] - \\ &\Pr[\mathcal{A}_1(\{U_{g_j}\}_{j\in[v]}, \{X_{h_j}\}_{j\in[u]}, \{f^{(\ell+1)}(X_{g_j})\}_{j\in[v]}, E) = 1]| > \frac{1}{q(n)}\end{aligned}$$

A standard hybrid argument implies that there exists $i \in [v]$ such that for infinitely many n's,

$$\begin{aligned}&|\Pr[\mathcal{A}_1(Z_{g_i}, \{Z_{g_j}\}_{j\neq i, j\in[v]}, \{X_{h_j}\}_{j\in[u]}, \{f^{(\ell+1)}(X_{g_j})\}_{j\in[v]}, E) = 1] - \\ &\Pr[\mathcal{A}_1(U, \{Z_{g_j}\}_{j\neq i, j\in[v]}, \{X_{h_j}\}_{j\in[u]}, \{f^{(\ell+1)}(X_{g_j})\}_{j\in[v]}, E) = 1]| > \frac{1}{p \cdot q(n)}\end{aligned}$$

We assume without loss of generality that $i = 1$. Recall that there are at least $u+2$ honest players. Thus, there are at least 2 honest players in $\{g_j\}_{j\in[v]}$. Pick another honest player P_{g_2}. We say that a tuple

$$(e^{\ell-1}, z^{\ell-1}, e^{\ell}_{g_1}, \{x_{h_j}\}_{j\in[u]}, \{x_{g_j}\}_{j\in[v],j\neq\{1,2\}}) \leftarrow (E^{\ell-1}, Z^{\ell-1}, E^{\ell}_{g_1}, \{X_{h_j}\}_{j\in[u]}, \{X_{g_j}\}_{j\in[v],j\neq\{1,2\}})$$

is BAD if conditioned on the fixing of this tuple, the following properties are satisfied:

1. There exists a non-uniform polynomial time adversary $\mathcal{A}_2$ such that for infinitely many n's,

$$\left|\Pr[\mathcal{A}_2(Z^{\ell}_{g_1}, R^{\ell}, X_{g_2}, f^{(\ell+1)}(X_{g_1})) = 1] - \Pr[\mathcal{A}_2(U, R^{\ell}, X_{g_2}, f^{(\ell+1)}(X_{g_1})) = 1]\right| \geq \frac{1}{2p\cdot q(n)},$$

2. X_{g_1} and X_{g_2} are independent. $E^{\ell}_{g_2}$ is a deterministic function of X_{g_2} and is $2^{-k^{\Omega(1)}}$-close to being uniform. X_{g_1} has min-entropy $k - o(k)$.

Claim 5.1.16. *There exists a BAD tuple.*

Again, the proof of this claim is rather standard and is very similar to the proof of Claim 5.1.7, we thus omit the details here.

Now fix a BAD tuple $(e^{\ell-1}, z^{\ell-1}, e^{\ell}_{g_1}, \{x_{h_j}\}_{j\in[u]}, \{x_{g_j}\}_{j\in[v],j\neq\{1,2\}})$. Then all R^{ℓ}_j's from honest players except P_{g_2} are fixed. The R^{ℓ}_j's from faulty players are a deterministic function of the transcript so far, thus are also fixed. Note $R^{\ell} = \bigoplus R^{\ell}_j$ and $R^{\ell}_{g_2}$ is a deterministic function of X_{g_2}. Thus R^{ℓ} is now a deterministic function of X_{g_2}. Note X_{g_1} and X_{g_2} are independent conditioned on the fixing, thus R^l is independent of X_{g_1}. Moreover, since $E^{\ell}_{g_2}$ is $2^{-k^{\Omega(1)}}$-close to being uniform, the property of the secure multiparty computation protocol guarantees that the R^{ℓ}_j's from faulty players are indistinguishable from being independent of $R^{\ell}_{g_2}$. Thus $R^{\ell} = \bigoplus R^{\ell}_i$ is indistinguishable from being uniform. Note $Z^{\ell}_{g_1} = \mathsf{LExt}(f^{(\ell)}(X_{g_1}), R^{\ell})$, thus

$$(Z^{\ell}_{g_1}, R^{\ell}, f^{(\ell+1)}(X_{g_1})) \approx (\mathsf{LExt}(f^{(\ell)}(X_{g_1}), U_d), U_d, f^{(\ell+1)}(X_{g_1})), \tag{5.19}$$

On the other hand, note that when we fix the BAD tuple, X_{g_1} and X_{g_2} are independent, and R^{ℓ} is a deterministic function of X_{g_2}. Thus by the first property of the BAD tuple and

Lemma 3.1.13, there exists another non-uniform adversary $\mathcal{A}_3$ that runs in time $2^{|R^\ell|} n Time(\mathcal{A}_2) = \text{poly}(n, \frac{1}{\epsilon}) Time(\mathcal{A}_2) = \text{poly}(n^{\log n})$ such that

$$\left| \Pr[\mathcal{A}_3(Z^\ell_{g_1}, R^\ell, f^{(\ell+1)}(X_{g_1})) = 1] - \Pr[\mathcal{A}_3(U, R^\ell, f^{(\ell+1)}(X_{g_1})) = 1] \right| \geq \frac{1}{2p \cdot q(n)}.$$

Note $R_l \approx U_d$, combined with Equation 5.19 we get

$$\left| \Pr[\mathcal{A}_3(\mathsf{LExt}(f^{(\ell)}(X_{g_1}), U_d), U_d, f^{(\ell+1)}(X_{g_1})) = 1] - \Pr[\mathcal{A}_3(U, U_d, f^{(\ell+1)}(X_{g_1})) = 1] \right| > \frac{1}{3p \cdot q(n)}.$$

Note $f^{(\ell)}(X_{g_1})$ has min-entropy $k - o(k) > 0.9k = 0.9n^\alpha$ conditioned on all the fixings and f is a one way permutation for $0.3n^\alpha$-sources. Note $\mathcal{A}_3$ that runs in time $\text{poly}(n^{\log n})$. Thus this contradicts Theorem 3.1.12. □

Once we have the lemma, it's fairly easy to prove the main theorem. We first prove that if $\{Z_{g_i}\}$'s are really $\{U_{g_i}\}$'s, then all W_i's of honest players are indistinguishable from being uniform and private. Then since $\{Z_{g_i}\}$'s are indistinguishable from $\{U_{g_i}\}$'s, the theorem follows.

To prove the statement above, consider any particular honest player P_j. Assume there exists a PPT adversary that distinguishes W_j and uniform given the transcript in Phase 1 and Phase 2. We first fix all honest players' sources except P_j. There is a fixing of the sources such that the adversary still distinguishes W_j and uniform given the transcript. Note after this fixing all transcript in Phase 1 and Phase 2 except $\{U_{g_i}\}$'s are a deterministic function of X_j. Now we further fix all the transcript and $\{U_{g_i}\}$'s except $f^{(j+1)}(X_j)$ and U_{g_1}. Again there is a fixing such that the adversary still distinguishes W_j and uniform. Now the adversary is only given $f^{(j+1)}(X_j)$ and U_{g_1}. Recall $W_j = \mathsf{LExt}(f^{(j)}(X_j), R)$ and note now R is a deterministic function of U_{g_1}. Thus Lemma 3.1.13 implies there exists another adversary that distinguishes W_j and uniform given R and $f^{(j+1)}(X_j)$. The property of the secure multiparty computation guarantees that R is indistinguishable from being uniform and independent of X_j. Note all E^j_i's and Z^j_i's are small thus conditioned on all the fixings mentioned above $f^{(j)}(X_j)$ still has min-entropy $> 0.9k$. Thus the existence of the above adversary contradicts Theorem 3.1.12.

Proof of Theorem 5.1.11. Again, we assume without loss of generality that the PPT adversary (who controls all the malicious players) is deterministic. At the end of Phase 1, we have

$$(\{Z_{g_i}\}_{i\in[v]}, \{X_{h_i}\}_{i\in[u]}, \{f^{(\ell+1)}(X_{g_i})\}_{i\in[v]}, E) \approx (\{U_{g_i}\}_{i\in[v]}, \{X_{h_i}\}_{i\in[u]}, \{f^{(\ell+1)}(X_{g_i})\}_{i\in[v]}, E)$$

Let the set of all honest players be $\mathcal{G}$, i.e., $\mathcal{G} = \{h_i\} \cup \{g_i\}$. Note that $(\{Z_{h_i}\}_{i\in[u]}, \{f^{(h_i)}(X_{h_i})\}_{i\in[u]}, \{f^{(g_i)}(X_{g_i})\}_{i\in[v]})$ can be computed in polynomial time from $(\{X_{h_i}\}_{i\in[u]}, \{f^{(\ell+1)}(X_{g_i})\}_{i\in[v]}, E)$ (keep in mind that $g_i \geq l+1$). Thus we have

$$(\{Z_{g_i}\}_{i\in[v]}, \{Z_{h_i}\}_{i\in[u]}, \{f^{(i)}(X_i)\}_{i\in\mathcal{G}}, E) \approx (\{U_{g_i}\}_{i\in[v]}, \{Z_{h_i}\}_{i\in[u]}, \{f^{(i)}(X_i)\}_{i\in\mathcal{G}}, E) \tag{5.20}$$

Note that the transcript in Phase 1 can be computed in polynomial time from $(E, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}})$, and the transcript in Phase 2 can be computed in polynomial time from $(\{Z_i\}_{i\in[p]})$. Moreover, $(\{Z_i\}_{i\notin\mathcal{G}})$ can be computed in polynomial time from the transcript in Phase 1. Thus to prove the theorem it suffices to prove

$$\begin{aligned}&(\{W_i\}_{i\in\mathcal{G}}, \{Z_i\}_{i\in\mathcal{G}}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) \approx \\ &(\{U_i\}_{i\in\mathcal{G}}, \{Z_i\}_{i\in\mathcal{G}}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E).\end{aligned}$$

Now if we run Phase 2 with the two distributions on both sides of Equation 5.20, and let $\bar{w}_i$ denote the output of player P_i when we run the protocol with the right hand side distribution, we get

$$\begin{aligned}&(\{W_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{Z_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) \approx \\ &(\{\bar{W}_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E).\end{aligned} \tag{5.21}$$

We'll first prove

$$\begin{aligned}&(\{\bar{W}_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) \approx \\ &(\{U_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E).\end{aligned}$$

Assume for the sake of contradiction that there exists a non-uniform polynomial time adversary $\mathcal{A}_1$ and a polynomial q such that for infinitely many n's,

$$|\Pr[\mathcal{A}_1(\{\bar{W}_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) = 1] - \Pr[\mathcal{A}_1(U_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) = 1]| > \frac{1}{q(n)}$$

A standard hybrid argument implies that there exists $j \in \mathcal{G}$ such that

$$|\Pr[\mathcal{A}_1(\bar{W}_j, \{\bar{W}_i\}_{i\in\mathcal{G}, i\neq j}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) = 1] - \Pr[\mathcal{A}_1(U, \{\bar{W}_i\}_{i\in\mathcal{G}, i\neq j}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) = 1]| > \frac{1}{p\cdot q(n)}$$

for infinitely many n's.

We say that a tuple

$$(e, \{z_{h_i}\}_{i\in[u]}, \{x_i\}_{i\neq j, i\in\mathcal{G}}) \leftarrow (E, \{Z_{h_i}\}_{i\in[u]}, \{X_i\}_{i\neq j, i\in\mathcal{G}})$$

is BAD if conditioned on the fixing of this tuple, the following properties are satisfied:

1. There exits a non-uniform polynomial time adversary $\mathcal{A}_2$ such that for infinitely many n's

$$\left|\Pr[\mathcal{A}_2(\bar{W}_j, \{U_{g_i}\}_{i\in[v]}, f^{(j+1)}(X_j)) = 1] - \Pr[\mathcal{A}_2(U, \{U_{g_i}\}_{i\in[v]}, f^{(j+1)}(X_j)) = 1]\right| \geq \frac{1}{2p\cdot q(n)},$$

2. X_j has min-entropy $k - o(k)$.

Claim 5.1.17. *There exists a BAD tuple.*

Proof of Claim 5.1.17. A standard probabilistic argument shows that a random tuple

$$(e, \{z_{h_i}\}_{i\in[u]}, \{x_i\}_{i\neq j, i\in\mathcal{G}}) \leftarrow (E, \{Z_{h_i}\}_{i\in[u]}, \{X_i\}_{i\neq j, i\in\mathcal{G}})$$

satisfies

$$|\Pr[\mathcal{A}_1(\bar{W}_j, \{\bar{W}_i\}_{i\in\mathcal{G}, i\neq j}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) = 1] - \Pr[\mathcal{A}_1(U, \{\bar{W}_i\}_{i\in\mathcal{G}, i\neq j}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) = 1]| \geq \frac{1}{2p\cdot q(n)}$$

with probability at least $\frac{1}{2p\cdot q(n)}$.

Note that once $(E, \{Z_{h_i}\}_{i\in[u]}, \{X_i\}_{i\neq j, i\in\mathcal{G}})$ are fixed, $(\{\bar{W}_i\}_{i\in\mathcal{G}, i\neq j}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}, i\neq j})$ can be computed in polynomial time from $\{U_{g_i}\}_{i\in[v]}$. Thus there exists a non-uniform adversary PPT $\mathcal{A}_2$ that has the fixings hardwired into it such that

$$\left|\Pr[\mathcal{A}_2(\bar{W}_j, \{U_{g_i}\}_{i\in[v]}, f^{(j+1)}(X_j)) = 1] - \Pr[\mathcal{A}_2(U, \{U_{g_i}\}_{i\in[v]}, f^{(j+1)}(X_j)) = 1]\right| \geq \frac{1}{2p\cdot q(n)}$$

Furthermore, since $\{X_i\}_{i\in\mathcal{G}}$ are independent, it's easy to show by induction on round $j' \in [p]$ that the only fixings that can cause X_j to lose entropy are $E_j^{j'}$'s and Z_j, and these are a deterministic function of X_j(conditioned on the fixings). The total length of these strings is at most $O(p\log^3 n) = o(k)$ since $k = n^\alpha$ and $p \leq k^{\gamma/u}$. Thus by Lemma 3.5.14 with probability $1 - \text{negl}(n)$ over the fixings of $(E, \{Z_{h_i}\}_{i\in[u]}, \{X_i\}_{i\neq j, i\in\mathcal{G}})$, X_j has min-entropy $k - o(k)$. The claim thus follows. □

Now we further fix $\{U_{g_i}\}_{i\in[v]}$ except U_{g_1}. There is a fixing of $\{U_{g_i}\}_{i\in[v], i\neq 1}$ that preserves this probability. Thus there exists a non-uniform PPT adversary $\mathcal{A}_3$ that has the fixings hardwired into it such that conditioned on the fixings,

$$\left|\Pr[\mathcal{A}_3(\bar{W}_j, U_{g_1}, f^{(j+1)}(X_j)) = 1] - \Pr[\mathcal{A}_3(U, U_{g_1}, f^{(j+1)}(X_j)) = 1]\right| \geq \frac{1}{2p\cdot q(n)}$$

for infinitely many n's.

Moreover, after all these fixings R is a deterministic function of U_{g_1}. Note that $\bar{W}_j = \mathsf{LExt}(f^{(j)}(X_j), R)$ and U_{g_1} is independent of X_j. Thus by Lemma 3.1.13 there exists a non-uniform adversary $\mathcal{A}_4$ that runs in time $2^{|R|}nTime(A_3) = \text{poly}(n, \frac{1}{\epsilon})Time(A_3) = \text{poly}(n^{\log n})$ such that

$$\left|\Pr[\mathcal{A}_4(\bar{W}_j, R, f^{(j+1)}(X_j)) = 1] - \Pr[\mathcal{A}_4(U, R, f^{(j+1)}(X_j)) = 1]\right| \geq \frac{1}{2p\cdot q(n)}.$$

Note U_{g_1} is uniform and independent of all the other random variables, thus the property of the secure multiparty computation protocol guarantees that $R = \bigoplus R_i$ is indistinguishable from being uniform and independent of X_j. Further note conditioned on all the fixings above, $f^{(j)}(X_j)$ has min-entropy $k - o(k) > 0.9k$. Note f is a one way permutation for $0.3k$-sources and $\mathcal{A}_4$ runs in time $\mathrm{poly}(n^{\log n})$. Thus this contradicts Theorem 3.1.12.

Therefore, we must have

$$(\{\bar{W}_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) \approx$$
$$(\{U_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E).$$

From Equation 5.20 we get

$$(\{U_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) \approx$$
$$(\{U_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{Z_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E).$$

Therefore

$$(\{\bar{W}_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{U_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) \approx$$
$$(\{U_i\}_{i\in\mathcal{G}}, \{Z_{h_i}\}_{i\in[u]}, \{Z_{g_i}\}_{i\in[v]}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E).$$

Together with Equation 5.21 this implies

$$(\{W_i\}_{i\in\mathcal{G}}, \{Z_i\}_{i\in\mathcal{G}}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E) \approx$$
$$(\{U_i\}_{i\in\mathcal{G}}, \{Z_i\}_{i\in\mathcal{G}}, \{f^{(i+1)}(X_i)\}_{i\in\mathcal{G}}, E).$$

as desired. □

Chapter 6

Improved Constructions of Extractors

In this chapter we give results that improve previous constructions of extractors in various ways. As described in the introduction, extractors are algorithms that take as input different kinds of weak random sources and output a distribution that is close to uniform. As illustrated in the previous chapters, extractors play an important role in applications that involve weak random sources. Extractors are also combinatorial objects that are interesting in their own rights.

6.1 A Two-Source Extractor under Computational Assumptions

In this section, we show that we can build a better two-source extractor if we assume a computational assumption. The assumption is the one that has already been used in Chapter 5– there exist one way permutations for weak random sources. Informally, we have the following theorem.

Theorem 6.1.1 (2-Source Extractor). *Suppose that for every $\delta > 0$, there exists a family of one-way permutations for $(n, 0.3\delta n)$-sources. Then there is a polynomial time computable 2-source extractor for 2 independent $(n, \delta n)$-sources that extracts $n^{\Omega(1)}$ bits that are computationally indistinguishable from uniform.*

Note that the first $O(\log n)$ bits of the output of our extractor are actually guaranteed to be statistically indistinguishable from uniform, since every statistical test on such a small number of bits can be efficiently simulated. Thus, even though we make a computational assumption, our conclusion is of an information theoretic nature.

6.1.1 Overview of Our Construction

In this section we shall be slightly inaccurate, in order to easily convey what we consider to be the key ideas in our work. We first note that it is well known how to extract randomness from two independent sources, assuming one of them is a *block source.* A block source X is a source that can be partitioned into two parts $X = (X_1, X_2)$ in such a way that X_1 has entropy δn, and X_2 has entropy δn *even conditioned on any fixing of* $X_1 = x_1$. The entropy in such a source is spread out, and it is well known how to take advantage of such structure. For example, it is known how to extract randomness from a block source $X = (X_1, X_2)$ using an independent weak source Y, as long as the blocks X_1, X_2 and the weak source Y each have entropy δn [BRSW06, RZ08, BKS$^+$05].

Block sources are fairly general, in the sense that *every* weak source can be shown to be a convex combination of block sources — for every source X with linear entropy δn, if X is broken into a sufficiently large ($t = 100/\delta$) number of blocks $X = (X_1, X_2, \ldots, X_t)$, then X is a convex combination of sources, where each element in the convex combination has the structure that there is some index $j \in [t]$ for which (X_j, X) is a block source where each block has linear entropy. Intuitively, each block in the source has at most $\delta n/100$ bits, and so cannot contain all δn bits of entropy.

This fact alone is not enough to apply extractors for block sources, since the index j above is not known ahead of time. Still, we might be tempted to try the following approach:

Naive 2-Source Extractor for (X,Y)

1. Let BExt be an extractor for a block source and an independent weak source
2. Partition $x = (x_1, \ldots, x_t)$.
3. For every i, compute $r_i = \mathsf{BExt}(x_i, x, y)$.
4. Output the bitwise xor $r_1 \oplus \cdots \oplus r_t$.

Since it is no loss of generality to assume that there is some index j for which (X_j, X) is a block source, $R_j = \mathsf{BExt}(X_j, X, Y)$ must be uniform. Unfortunately, the reason this algorithm does not work is that the rest of the candidate random strings R_i are not independent of R_j, and so the output could be a fixed constant even though R_j is uniform.

Our actual construction is a variation of the above construction, where we use computational assumptions to enforce that R_j is independent (in some sense) from the other R_i's. More specifically, we use a one-way permutation for δn-sources to generate independence. This idea was implicit in the work of Goldreich-Levin [GL89] on finding hardcore predicates. There they showed that for any one-way function f, the triplet $(\langle X, R\rangle, R, f(X))$ is computationally indistinguishable from $(U, R, f(X))$, where U is a random bit, and X, R are both uniformly distributed in $\{0,1\}^n$. In other words, they showed that $\langle X, R \rangle$ looks random and *independent* of $(R, f(X))$, even though it may be uniquely determined by $(R, f(X))$. Their construction was an early example of a *reconstructive extractor*, a concept that was subsequently formalized and refined in a sequence of works [NW94, Tre01, TZ04, TUZ01, SU05, Uma05]. We now know of several different constructions of reconstructive extractors. We do not define this concept here, but what is important to know in our application is that every reconstructive extractor RExt must satisfy the property that if f is one-way with respect to a weak source X, then

$$(\mathsf{RExt}(X, R), R, f(X)) \approx (\mathsf{Uniform}, R, f(X)),$$

where $\approx$ denotes computational indistinguishability.

Given such an object, here is how we can use it to build a 2-Source extractor.

Our 2-Source Extractor

1. Let BExt be an extractor for a block source and an independent weak source and RExt be a reconstructive (seeded) extractor. Let f be a one-way permutation for weak sources.
2. Partition $x = (x_1, \ldots, x_t)$.
3. For every i, compute $z_i = \mathsf{BExt}(x_i, x, f^i(y))$.
4. Set $r_i = \mathsf{RExt}(f^i(y), z_i)$.
5. Output the bitwise xor $r_1 \oplus \cdots \oplus r_t$.

Here $f^i(y) = f(f(\cdots f(y) \cdots))$, where f is applied i times. The goal here is to break the dependence (at least in a computational sense) between the R_i's.

Outline of the analysis. To analyze this construction, we need to exploit a strong property of $\mathsf{BExt}(X_1, X_2, Y)$. It turns out that one can show that there is a random variable T on a few bits, such that for every fixing of (T, X_1),

- $\mathsf{BExt}(X_1, X_2, Y)$ is independent of Y.
- $\mathsf{BExt}(X_1, X_2, Y)$ is uniform.

In particular, since the output of BExt is only a few bits, this means that after fixing (X_1, T), we can fix the output of $\mathsf{BExt}(X_1, X_2, Y)$ and still be left with two independent sources X, Y with high entropy (here we assume the slightly inaccurate fact that fixing a binary string of length l can only reduce the entropy of another variable by l).

Recall that there is some index j for which (X_j, X) is a block source. In the first step of the analysis, we use the properties of BExt described above to fix $Z_1, \ldots, Z_{j-1}$ and $R_1, \ldots, R_{j-1}$. We claim that even after this fixing, (X_j, X) is a block source that is independent of the source Y with linear entropy. We do this by fixing each of the $Z_1, \ldots, Z_{j-1}$'s one by one. Each such fixing maintains the independence we want, yet does not reduce the entropy of the sources by much, since the Z_i's are short. Once all the Z_i's are fixed, the corresponding R_i's are deterministic functions of Y that output only a few bits, so we can fix them without reducing the entropy in Y by much. Care must be taken that all of these fixings do not ruin the entropy in X_j (in particular, fixing $X_1, \ldots, X_{j-1}$ should not ruin the entropy in X_j), but it turns out that this can be done.

We get that after all these fixings, Z_j must be uniform and independent of Y. Thus, by the properties of reconstructive extractors, the following two distributions are computationally indistinguishable:

$$(R_j, Z_j, f^{j+1}(Y)) \approx (\mathsf{Uniform}, Z_j, f^{j+1}(Y)).$$

In fact, we can actually prove the stronger statement that

$$(R_j, X, f^{j+1}(Y)) \approx (\mathsf{Uniform}, X, f^{j+1}(Y)).$$

Observe that information theoretically this is very far from true. In fact, R_j is a deterministic function of $(X, f^{j+1}(Y))$. Finally, since $(R_{j+1}, \ldots, R_t)$ are all efficiently computable from $(X, f^{j+1}(Y))$, we obtain

$$(X, f^{j+1}(Y), R_j) \approx (X, f^{j+1}(Y), \mathsf{Uniform}),$$

which implies that the output of our extractor is computationally indistinguishable from uniform.

In fact, our proof shows that the extractor is *strong* — the output looks uniform even if one of the inputs is known.

6.1.2 The Formal Analysis

We have the following theorem:

Theorem 6.1.2. *Fix a constant $\alpha > 0$ and parameters $t = \frac{4}{\alpha}$ and $k \geq n^{\Omega(1)}$. Assume that there exists a permutation $f : \{0,1\}^n \to \{0,1\}^n$ such that for any $(n, 0.3k)$-source Y, any non-uniform adversary that runs in time $\mathrm{poly}(n^{\log n})$ can invert $f(Y)$ with only negligible probability. Then $\mathsf{TExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ described above is a computational 2-source extractor such that for any $(n, \alpha n)$-source X, and any (n,k)-source Y that is independent of X,*

$$(\mathsf{TExt}(X,Y), X) \approx (U_m, X)$$

Remark.

- Rather than proving Theorem 6.1.2, we prove the following (stronger) statement:

  $$(\mathsf{TExt}(X,Y), X, h(X), f^{(t+1)}(Y)) \approx (U_m, X, h(X), f^{(t+1)}(Y)), \tag{6.1}$$

  where h is any deterministic function (not necessarily computable in polynomial time) on $\{0,1\}^n$. The reason is that we need this stronger variant for our network extractor protocol in Chapter 5.

- If we let $m = O(\log n)$, then the theorem implies that in particular,

  $$|\mathsf{TExt}(X,Y) - U_m| = \mathrm{negl}(n)$$

  .

To prove Equation (6.1) we first prove the following lemma(in the analysis we use capital letters to denote the corresponding strings viewed as random variables).

Lemma 6.1.3. *Divide X into $X = (X_1, \ldots, X_t)$ as in the construction of TExt, and let $Z = \mathsf{TExt}(X,Y)$. Suppose there exists $g \in [t]$ such that*

- *$X_1, \ldots, X_{g-1}$ are fixed.*
- *$H_\infty(X_g) \geq \frac{\alpha^2}{6}$.*
- *$H_\infty(X|X_g) \geq \frac{\alpha^2}{6}$.*

Then

$$(Z, X, h(X), f^{(t+1)}(Y)) \approx (U_m, X, h(X), f^{(t+1)}(Y))$$

Proof of Lemma 6.1.3. Let sr_i denote the string sr computed in $\mathsf{BExt}(x_i, x, f^{(i)}(Y))$. Fix

$$(sr_1, \ldots, sr_{g-1}) \leftarrow (SR_1, \ldots, SR_{g-1})$$

$$(r_1, \ldots, r_{g-1}) \leftarrow (R_1, \ldots, R_{g-1})$$

and

$$(z_1, \ldots, z_{g-1}) \leftarrow (Z_1, \ldots, Z_{g-1}).$$

For any random variable Z, we denote by Z' the random variable Z conditioned on these fixings.

Let TYPICAL denote the event that conditioned on these fixings, the following conditions are satisfied:

- X' and Y' are independent
- $H_\infty(Y') \geq k - \text{polylog}(n)$
- $H_\infty(X'_g) \geq \frac{\alpha^2}{8} n$
- With probability $1 - \text{negl}(n)$ over $(x'_g \leftarrow X'_g)$, $H_\infty(X' | X'_g = x'_g) \geq \frac{\alpha^2}{8} n$

Claim 6.1.4.

$$\Pr[\text{TYPICAL}] = 1 - \text{negl}(n).$$

Proof of Claim 6.1.4. Since $X_1, \ldots, X_{g-1}$ are fixed, $(SR_1, \ldots, SR_{g-1})$ is a deterministic function of Y. Thus, conditioning on $(sr_1, \ldots, sr_{g-1}) \leftarrow (SR_1, \ldots, SR_{g-1})$, X and Y are still independent. Moreover, since each sr_i has size cs, the total size of $(sr_1, \ldots, sr_{g-1})$ is bounded by $tcs = \text{polylog}(n)$. Thus, by Lemma 3.5.14, with probability $1 - \text{negl}(n)$ over these fixings, Y has min-entropy $k - \text{polylog}(n)$ (let $\epsilon = 2^{-\log^2 n}$ in the lemma).

Next, we further condition on $(r_1, \ldots, r_{g-1}) \leftarrow (R_1, \ldots, R_{g-1})$. Note that now $(R_1, \ldots, R_{g-1})$ is a deterministic function of X. Thus, conditioned on this fixing, X and Y are still independent. Moreover, since each r_i is of size $\text{polylog}(n)$, the total size of $(r_1, \ldots, r_{g-1})$ is bounded by $t|r_i| = \text{polylog}(n)$. Thus, by Lemma 3.5.14, with probability $1 - \text{negl}(n)$ over these fixings, X_g has min-entropy $H_\infty(X_g) - \text{polylog}(n) > \frac{\alpha^2}{8}n$ (let $\epsilon = 2^{-\log^2 n}$ in the lemma). Next, note that conditioned on any fixing of $x_g \in \mathsf{Supp}(X_g)$, we have $H_\infty(X) \geq \frac{\alpha^2}{6}n$, and with probability $1 - \text{negl}(n)$ over the further fixings of $(R_1, \ldots, R_{g-1})$, $H_\infty(X) > \frac{\alpha^2}{8}n$. Thus we have

$$\Pr_{X_g, R_1, \ldots, R_{g-1}}[H_\infty(X'|X'_g = x'_g) > \frac{\alpha^2}{8}n] \geq 1 - \epsilon_1,$$

where $\epsilon_1 = \text{negl}(n)$.

Now a standard averaging argument shows that, with probability at least $1 - \sqrt{\epsilon_1}$ over the fixings of $(R_1, \ldots, R_{g-1})$,

$$\Pr_{x'_g \leftarrow X'_g}[H_\infty(X'|X'_g = x'_g) > \frac{\alpha^2}{8}n] \geq 1 - \sqrt{\epsilon_1}.$$

Note $\epsilon_1 = \text{negl}(n)$, thus $\sqrt{\epsilon_1} = \text{negl}(n)$.

Finally, we further condition on $(z_1, \ldots, z_{g-1}) \leftarrow (Z_1, \ldots, Z_{g-1})$. Note that now $(Z_1, \ldots, Z_{g-1})$ is a deterministic function of Y. Thus, conditioned on this fixing, X and Y are still independent. Moreover, since each z_i is of size polylog(n), the total size of $(z_1, \ldots, z_{g-1})$ is bounded by $t|z_i| = \text{polylog}(n)$. Thus, by Lemma 3.5.14, with probability $1 - \text{negl}(n)$ over these fixings, Y has min-entropy $k - \text{polylog}(n) - \text{polylog}(n) = k - \text{polylog}(n)$ (let $\epsilon = 2^{-\log^2 n}$ in the lemma).

The probability that all of the above happen is at least $1 - \text{negl}(n) - \text{negl}(n) - \text{negl}(n) = 1 - \text{negl}(n)$. $\square$

Now fix

$$(x'_g, sr'_g) \leftarrow (X'_g, SR'_g).$$

For every random variable Z', denote by

$$Z'' = Z'|(X'_g = x'_g, SR'_g = sr'_g).$$

Let TYPICAL2 denote the event that conditioned on all the above fixings, the following holds:

- X'' and Y'' are independent, R''_g is a deterministic function of X''
- $H_\infty(Y'') \geq 0.9k$
- $(R''_g, Y'') \equiv (U_d, Y'')$

Claim 6.1.5. *If* TYPICAL *holds, then*

$$\Pr[\text{TYPICAL2}] = 1 - \text{negl}(n)$$

Proof of Claim 6.1.5. First note that when TYPICAL holds, X' and Y' are independent, and $H_\infty(X'_g) \geq \frac{\alpha^2}{8}n$. This means X'_g has min-entropy rate $\geq \frac{\alpha}{2}$. Therefore by Theorem 3.5.2 M'_g is $2^{-\Omega(n)}$-close to a $(c \times \ell)0.9\ell$-somewhere random source. Theorem 3.5.7 implies that there exists a somewhere-random source SR with c rows, each row of length s, s.t.

$$|(M'_g, SR'_g) - (M'_g, SR)| = \mathrm{negl}(n).$$

A standard averaging argument shows that with probability $1 - \mathrm{negl}(n)$ over the fixing of M'_g (and thus X'_g), we still have

$$|SR'_g - SR| = \mathrm{negl}(n).$$

Moreover, X'_g (and thus M'_g) is a deterministic function of X', thus conditioned on the fixing of X'_g (and thus M'_g), X' and Y' are still independent. Note once conditioned on M'_g, SR'_g is a deterministic function of Y', and is thus independent of X'. Also, with probability $1 - \mathrm{negl}(n)$ over the fixing of X'_g, $H_\infty(X') \geq \frac{\alpha^2}{8}n$. The probability that both these two events happen is $1 - \mathrm{negl}(n)$, and when this happens, Theorem 3.5.11 implies that

$$|(SR'_g, R_g) - (SR'_g, U_d)| < 2^{-n^{\Omega(1)}} + \mathrm{negl}(n) = \mathrm{negl}(n).$$

Since this happens with probability $1 - \mathrm{negl}(n)$, we actually have that

$$|(SR'_g, R_g) - (SR'_g, U_d)| = \mathrm{negl}(n).$$

Again, by a standard averaging argument, with probability $1 - \mathrm{negl}(n)$ over the fixing of SR'_g, we still have

$$|R_g - U_d| = \mathrm{negl}(n).$$

Note since SR'_g is a deterministic function of Y', conditioning on it still leaves X' and Y' independent. Moreover, since the size of sr_g is small, the same argument in the proof of Claim 6.1.4 implies that with probability $1 - \mathrm{negl}(n)$ over the fixings of SR'_g, $H_\infty(Y') \geq k - \mathrm{polylog}(n) - \mathrm{polylog}(n) > 0.9k$. Finally, conditioned on the fixing of SR'_g, R_g is a deterministic function of X', and is thus independent of Y'. Note $|R_g - U_d| = \mathrm{negl}(n)$, therefore

$$(R_g, Y'') \equiv (U_d, Y'')$$

The probability that all of the above are satisfied is $1 - \mathrm{negl}(n)$. $\square$

Next we prove the following claim.

Claim 6.1.6. *If both* TYPICAL *and* TYPICAL2 *hold, then*

$$(\oplus_{i=g}^{t} Z_i'', X'', h(X''), f^{t+1}(Y'')) \approx (U_m, X'', h(X''), f^{t+1}(Y''))$$

Proof of Claim 6.1.6. Assume for the sake of contradiction that there exists a non-uniform PPT adversary $\mathcal{A}_1$ and a polynomial q such that for infinitely many n's

$$\left|\Pr[\mathcal{A}_1(\oplus_{i=g}^{t} Z_i'', X'', h(X''), f^{t+1}(Y'')) = 1] - \Pr[\mathcal{A}_1(U_m, X'', h(X''), f^{t+1}(Y'')) = 1\right| \geq \frac{1}{q(n)}.$$

Since $Z_{g+1}'', \cdots, Z_t''$ and $f^{t+1}(Y'')$ can be computed from $(X'', f^{g+1}(Y''))$ in polynomial time, there exists another non-uniform PPT adversary $\mathcal{A}_2$ such that

$$\left|\Pr[\mathcal{A}_2(Z_g'', R_g'', X'', h(X''), f^{g+1}(Y'')) = 1] - \Pr[\mathcal{A}_2(U_m, R_g'', X'', h(X''), f^{g+1}(Y'')) = 1\right| \geq \frac{1}{q(n)}.$$

Recall that

$$Z_g'' = \mathsf{RExt}(f^{(g)}(Y''), R_g'')$$

and $f^{(g)}(Y'')$ is a deterministic function of $f^{(g+1)}(Y'')$(though not computable in polynomial time). Thus Z_g'' is a deterministic function of $f^{(g+1)}(Y'')$ and R_g''. Next note that R_g'' is a deterministic function of X'', and X'', Y'', U_m are independent. Thus Lemma 3.1.13 implies that there exists another non-uniform adversary $\mathcal{A}_3$ that runs in time $2^d \cdot n \cdot \mathrm{poly}(n) = \mathrm{poly}(n, \frac{1}{\epsilon}) \cdot \mathrm{poly}(n) = \mathrm{poly}(n, \frac{1}{\epsilon})$ such that

$$\left|\Pr[\mathcal{A}_3(Z_g'', R_g'', f^{g+1}(Y'')) = 1] - \Pr[\mathcal{A}_3(U_m, R_g'', f^{g+1}(Y'')) = 1]\right| \geq \frac{1}{q(n)}.$$

Note that the fact that TYPICAL2 holds implies that $(R_g'', Y'') \equiv (U_d, Y'')$. This, together with Proposition 2.1.2 implies that

$$\left|\Pr[\mathcal{A}_3(\mathsf{RExt}(f^{(g)}(Y''), U_d), U_d, f^{g+1}(Y'')) = 1] - \Pr[\mathcal{A}_3(U_m, U_d, f^{g+1}(Y'')) = 1]\right|$$
$$\geq \frac{1}{q(n)} - \mathrm{negl}(n) > \frac{1}{2q(n)},$$

Note Y'' has min-entropy $0.9k$ thus $f^{(g)}(Y'')$ also has min-entropy $0.9k$. Note f is one way for $0.3k$-sources and $\mathcal{A}_3$ runs in time $\text{poly}(n, \frac{1}{\epsilon}) = \text{poly}(n^{\log n})$, thus this contradicts Theorem 3.1.12. □

Now, since the event that both TYPICAL and TYPICAL2 hold happens with probability $1 - \text{negl}(n)$, by Lemma 2.4.4 we have

$$(\oplus_{i=g}^{t} Z_i, \{Z_i\}_{i\in[g-1]}, X, h(X), f^{t+1}(Y)) \approx (U_m, \{Z_i\}_{i\in[g-1]}, X, h(X), f^{t+1}(Y)).$$

Note that $Z = \oplus_{i=1}^{t} Z_i$, thus

$$(Z, X, h(X), f^{t+1}(Y)) \approx (U_m, X, h(X), f^{t+1}(Y)).$$

This proves the lemma. □

proof of Theorem 6.1.2. Since we divide X into $t = \frac{4}{\alpha}$ blocks, Lemma 2.3.6 says that X is $2^{-n^{\Omega(1)}}$-close to being a convex combination of $\{X^j\}_{j\in J}$ such that for every $j \in J$, X_j satisfies the conditions in Lemma 6.1.3. For every $j \in J$, let $Z^j = \mathsf{TExt}(X^j, Y)$, then

$$(Z^j, X^j, h(X^j), f^{(t+1)}(Y)) \approx (U_m, X^j, h(X^j), f^{(t+1)}(Y)).$$

Thus, by Lemma 2.4.5, the theorem holds. ■

The computational two source extractor described above outputs random bits that are computationally indistinguishable from being uniform, while assuming the existence of a one-way permutation f s.t. for any $(n, 0.3k)$ source X, any non-uniform adversary that runs in time $\text{poly}(n^{\log n})$ can only invert $f(X)$ with negligible probability. Note that the running time of the adversary is slightly super-polynomial. However, even if we only assume that any polynomial time adversary can only invert $f(X)$ with negligible error, we can still get a two-source extractor, but the error will only be polynomially small.

Theorem 6.1.7. *Let $\epsilon = \frac{1}{\text{poly}(n)}$ and $m = O(\log n)$ in the construction of* TExt. *Keep all the other parameters the same. Assume that there exists a permutation $f : \{0,1\}^n \to \{0,1\}^n$ such that for any $(n, 0.3k)$-source Y, any non-uniform adversary that runs in time* $\text{poly}(n)$ *can invert $f(Y)$ with only negligible probability. Then* TExt *is a 2-source extractor such that for any $(n, \alpha n)$-source X, and any (n, k)-source Y that is independent of X,*

$$|\mathsf{TExt}(X, Y) - U_m| < 3\epsilon.$$

Proof Sketch. The proof basically follows the same steps in the proof of Theorem 6.1.2. We first prove that for a source X that satisfies the conditions of Lemma 6.1.3, and $Z = \mathsf{TExt}(X, Y)$, we have

$$|Z - U_m| < 2.9\epsilon. \tag{6.2}$$

To this end, we prove that when both TYPICAL and TYPICAL2 hold, we have

$$|Z'' - U_m| < 2.5\epsilon. \tag{6.3}$$

Assume for the sake of contradiction that $|Z'' - U_m| \geq 2.5\epsilon$. Since $m = O(\log n)$, there exists a non-uniform PPT adversary $\mathcal{A}$(simply check all the 2^m strings) s.t.

$$\left|\Pr[\mathcal{A}(Z'') = 1] - \Pr[\mathcal{A}(U_m) = 1]\right| \geq 2.5\epsilon$$

Note $Z'' = \oplus_{i=1}^{t} Z_i''$. Since $Z_1, \ldots, Z_{g-1}$ are fixed, and $Z_{g+1}'', \cdots, Z_t''$ can be computed from $(X'', f^{g+1}(Y''))$ in polynomial time, there exists another non-uniform PPT adversary $\mathcal{A}_1$ such that

$$\left|\Pr[\mathcal{A}_1(Z_g'', R_g'', X'', f^{g+1}(Y'')) = 1] - \Pr[\mathcal{A}_1(U_m, R_g'', X'', f^{g+1}(Y'')) = 1]\right| \geq 2.5\epsilon$$

Recall that $Z_g'' = \mathsf{RExt}(f^{(g)}(Y''), R_g'')$, thus Lemma 3.1.13 implies that there exists another non-uniform adversary $\mathcal{A}_2$ that runs in time $2^d \cdot n \cdot \text{poly}(n) = \text{poly}(n, \frac{1}{\epsilon}) \cdot \text{poly}(n) = \text{poly}(n)$ such that

$$\left|\Pr[\mathcal{A}_2(Z_g'', R_g'', f^{g+1}(Y'')) = 1] - \Pr[\mathcal{A}_2(U_m, R_g'', f^{g+1}(Y'')) = 1\right| \geq 2.5\epsilon.$$

Note that the fact that TYPICAL2 holds implies that $(R_g'', Y'') \equiv (U_d, Y'')$. This, together with Proposition 2.1.2 implies that

$$\left|\Pr[\mathcal{A}_2(\mathsf{RExt}(f^{(g)}(Y''), U_d), U_d, f^{g+1}(Y'')) = 1] - \Pr[\mathcal{A}_2(U_m, U_d, f^{g+1}(Y'')) = 1\right|$$
$$\geq 2.5\epsilon - \text{negl}(n) > 2\epsilon,$$

Note $f^{(g)}(Y'')$ has min-entropy $0.9k$ since f is a permutation. Now a similar argument as in the proof of Theorem 3.1.12 implies that there exists another non-uniform adversary $\mathcal{A}_3$ that runs in time $\text{poly}(n, \frac{1}{\epsilon}) \cdot \text{poly}(n) = \text{poly}(n)$ and an $(n, 0.3k)$-source $\bar{Y}$ such that $\mathcal{A}_3$ inverts $f(\bar{Y})$ with probability at least $\epsilon/4$. This contradicts our assumption on f.

Thus Equation 6.3 does hold. Since the event that both TYPICAL and TYPICAL2 hold happens with probability $1 - \mathrm{negl}(n)$, Equation 6.2 holds. Now by Lemma 2.3.6, X is $2^{-n^{\Omega(1)}}$-close to being a convex combination of $\{X^j\}_{j \in J}$ such that for every $j \in J$, X_j satisfies the conditions in Lemma 6.1.3. Thus the theorem holds. ■

6.2 Three Source Extractors

In this section we give our improved constructions of three source extractors. The problem of constructing extractors for independent sources has a long history that dates back to more than twenty years ago [SV86, Vaz85, CG88], where it was shown that the inner product function is a two source extractor for two independent weak random sources with min-entropy rate $> 1/2$. After that there was no progress on this problem until recently, with the help of powerful theorems from additive combinatorics, a series of results appeared [BIW04, BKS+05, Raz05, Bou05, Rao06, BRSW06]. Perviously, the best explicit extractor for two independent (n, k) sources only achieves min-entropy $k = 0.499n$ [Bou05], the best known extractor for three independent sources achieves min-entropy $k = n^{0.9}$ [Rao06], and the best explicit extractor for independent (n, n^{α}) sources requires $O(1/\alpha)$ sources [Rao06, BRSW06]. Here we give the following theorem that improves previous three source extractors.

Theorem 6.2.1. *For every constant $0 < \delta < 1/2$, there exists a polynomial time computable function $\mathsf{THExt} : (\{0,1\}^n)^3 \to \{0,1\}^m$ such that if X, Y, Z are three independent (n, k) sources with $k = n^{1/2+\delta}$, then*

$$|\mathsf{THExt}(X, Y, Z) - U_m| < n^{-\Omega(\delta)}$$

with $m = \Omega(k)$.

The following table summarizes recent results on extractors for independent sources.

6.2.1 Overview of the Construction

Here we give a brief description of our constructions and the techniques used. For clarity and simplicity we shall be imprecise sometimes.

Our construction of the three source extractor mainly uses somewhere random sources and the extractor for such sources in [Rao06]. Informally, a somewhere random source is a matrix of random variables such that at least one of the rows is uniform. If we have two independent somewhere random sources $X = X_1 \circ \cdots \circ X_t$ and $Y = Y_1 \circ \cdots \circ Y_t$ with the same number of rows

Number of Sources	Min-Entropy	Output	Error	Ref
$O(\text{poly}(1/\delta))$	δn	$\Theta(n)$	$2^{-\Omega(n)}$	[BIW04]
3	δn, any constant δ	$\Theta(1)$	$O(1)$	[BKS+05]
3	One source: δn, any constant δ. Other sources may have $k \geq \text{polylog}(n)$.	$\Theta(1)$	$O(1)$	[Raz05]
2	One source: $(1/2+\delta)n$, any constant δ. Other source may have $k \geq \text{polylog}(n)$	$\Theta(k)$	$2^{-\Omega(k)}$	[Raz05]
2	$(1/2-\alpha_0)n$ for some small universal constant $\alpha_0 > 0$	$\Theta(n)$	$2^{-\Omega(n)}$	[Bou05]
$O(1/\delta)$	$k = n^{\delta}$	$\Theta(k)$	$k^{-\Omega(1)}$	[Rao06]
$O(1/\delta)$	$k = n^{\delta}$	$\Theta(k)$	$2^{-k^{\Omega(1)}}$	[BRSW06]
3	One source: δn, any constant δ. Other sources may have $k \geq \text{polylog}(n)$.	$\Theta(k)$	$2^{-k^{\Omega(1)}}$	[Rao06]
3	$k = n^{1-\alpha_0}$ for some small universal constant $\alpha_0 > 0$	$\Theta(k)$	$2^{-k^{\Omega(1)}}$	[Rao06]
3	$k = n^{1/2+\delta}$, any constant δ	$\Theta(k)$	$k^{-\Omega(1)}$	This work

Table 6.1: **Summary of Results on Extractors for Independent Sources.**

t, and there exists an i such that both X_i and Y_i are uniform, then we call X and Y independent *aligned* somewhere random sources. In [Rao06] it is shown that if we have two independent aligned somewhere random sources X, Y with t rows and each row has n bits, such that $t < n^{\gamma}$ for some arbitrary constant $0 < \gamma < 1$, then we can efficiently extract random bits from X and Y.

Now given three independent (n, k) sources X, Y, Z with $k = n^{1/2+\delta}$, our construction uses X to convert Y and Z into two somewhere random sources, such that with high probability over the fixing of X (and some additional random variables), they are independent aligned somewhere random sources, and the number of rows is significantly smaller than the length of each row. Then we will be done by using the extractor in [Rao06] described above.

To illustrate how we do this, first assume that we have a strong seeded extractor that only uses $\log n$ additional random bits and can extract almost all the entropy from an (n, k)-source with error $1/100$. A strong seeded extractor is a seeded extractor such that with high probability over the fixing of the seed, the output is still close to uniform. We now try this extractor on X, Y and Z with all $2^{\log n} = n$ possibilities of the seed and output $n^{\delta/2}$ bits. Thus we obtain three $n \times n^{\delta/2}$ matrices. Now we divide each matrix into $\sqrt{n}$ blocks with each block consisting of $\sqrt{n}$ rows. Therefore we get $X^1 \circ \cdots \circ X^t$, $Y^1 \circ \cdots \circ Y^t$ and $Z^1 \circ \cdots \circ Z^t$, where $t = \sqrt{n}$ and each X^i, Y^i, Z^i is a block. By a standard property of strong seeded extractors, with probability $1 - 1/10 = 9/10$

over the fixing of the seed, the output is 1/10-close to uniform. Therefore in each matrix, at least 9/10 fraction of the rows are close to uniform. We say a block is "good" if it contains at least one such row. Thus in each matrix at least 9/10 fraction of the blocks are good.

Now it's easy to see that there exists an i such that all X^i, Y^i and Z^i are good. In other words, in some sense the matrices are already "aligned". Next, for each i we compute an output R_i from (X^i, Y^i, X, Y) and an output R'_i from (X^i, Z^i, X, Z), with the property that if X^i, Y^i are good, then R_i is (close to) uniform, and if X^i, Z^i are good, then R'_i is (close to) uniform. We then concatenate $\{R_i\}$ to form a matrix SR_y and concatenate $\{R'_i\}$ to form a matrix SR_z. Since there exists an i such that all X^i, Y^i and Z^i are good, SR_y and SR_z are (close to) aligned somewhere random sources.

In the analysis below we consider a particular i such that all X^i, Y^i and Z^i are good (though we may not know what i is).

Now let's consider computing R_i from (X^i, Y^i, X, Y) (R'_i is computed the same way from (X^i, Z^i, X, Z)). Here we use a two-source extractor Raz in [Raz05]. This extractor is strong and it works as long as one of the source has min-entropy rate (the ratio between min-entropy and the length of the source) $> 1/2$, and even if the two sources have different lengths. We first apply Raz to Y^i and each row of X^i (note Y^i is treated as a whole) to obtain a matrix M. Note that if X^i, Y^i are good then they are both somewhere random, and thus Y^i has min-entropy at least $n^{\delta/2}$. Thus M is also a somewhere random source. Since Raz is a strong two-source extractor, we can fix X^i, and conditioned on this fixing M is still a somewhere random source. Moreover now M is a deterministic function of Y^i and is thus independent of X. Next note that the size of X^i is $\sqrt{n} \cdot n^{\delta/2} = n^{1/2+\delta/2}$ while the min-entropy of X is $n^{1/2+\delta}$. Thus with high probability over the fixings of X^i, X still has min-entropy at least $0.9n^{1/2+\delta}$. Therefore now we can apply a strong seeded extractor to X and each row of M and output $0.8n^{1/2+\delta}$ bits. Thus we obtain a $(\sqrt{n} \times 0.8n^{1/2+\delta})$ somewhere random source $\bar{X}^i$. Furthermore, since we applied a strong seeded extractor and now M is a deterministic function of Y^i, we can further fix Y^i and $\bar{X}^i$ is still somewhere random, meanwhile it is now a deterministic function of X.

Similarly, we can compute a somewhere random source $\bar{Y}^i$. Specifically, Since Raz is a strong two-source extractor, we can fix Y^i, and conditioned on this fixing M is still a somewhere random source. Moreover now M is a deterministic function of X^i and is thus independent of Y. Next note that the size of Y^i is $\sqrt{n} \cdot n^{\delta/2} = n^{1/2+\delta/2}$ while the min-entropy of Y is $n^{1/2+\delta}$. Thus with high probability over the fixings of Y^i, Y still has min-entropy at least $0.9n^{1/2+\delta}$. Therefore now we can apply a strong seeded extractor to Y and each row of M and output $0.8n^{1/2+\delta}$ bits. Thus we obtain a $(\sqrt{n} \times 0.8n^{1/2+\delta})$ somewhere random source $\bar{Y}^i$. Furthermore, since we applied a strong seeded extractor and now M is a deterministic function of X^i, we can further fix X^i and $\bar{Y}^i$ is still somewhere random, meanwhile it is now a deterministic function of X.

Therefore now after the fixings of (X^i, Y^i), we get two independent $(\sqrt{n} \times 0.8n^{1/2+\delta})$ somewhere random sources $(\bar{X}^i, \bar{Y}^i)$. It is easy to check that they are aligned. Note that the number of rows is significantly less than the length of each row, thus we can apply the extractor in [Rao06] to get a random string R_i with say $0.7n^{1/2+\delta}$ bits. Further notice that the extractor in [Rao06] is strong, thus we can fix X and R_i is still (close to) uniform. This means that we can fix X and SR_y is still somewhere random (recall that SR_y is the concatenation of $\{R_i\}$), moreover it is now a deterministic function of Y.

Similarly we can compute SR_z, and by the same argument we can fix X and SR_z is still somewhere random, moreover it is now a deterministic function of Z. Therefore now after we fix (X, Y^i, Z^i), we get two independent aligned $(\sqrt{n} \times 0.7n^{1/2+\delta})$ somewhere random sources, and again the extractor in [Rao06] can be used to obtain an output that is (close to) uniform.

The above argument works even if the seed length of the strong seeded extractor that we use on X, Y, Z (try all possibilities of the seed) is $(1+\alpha)\log n$ instead of $\log n$, as long as α can be an arbitrarily small constant. However, currently we don't have such extractors for min-entropy $k = n^{1/2+\delta}$. Fortunately, we have condensers with such short seed length. A (seeded) condenser is a generalization of a seeded extractor, such that the output is close to having high min-entropy instead of being uniform. In this paper we use the condenser built in [GUV09]. For any constant $\alpha > 0$ and any (n, k') source, this condenser uses $d = (1+1/\alpha) \cdot (\log n + \log k' + \log(1/\epsilon)) + O(1)$ additional random bits to convert the source roughly into a $((1+\alpha)k', k')$ source with error ϵ. Now we can take α to be a sufficiently large constant, say $10/\delta$, take k' to be small, say $n^{\delta/10}$ (note that an $(n, n^{1/2+\delta})$ source is also an $(n, n^{\delta/10})$ source), and take ϵ to be something like $n^{-\delta/10}$. This gives us a small seed length, such that $2^d = O(n^{1+\delta/3})$. Therefore the number of blocks is $O(n^{1/2+\delta/3})$, which is significantly less than $n^{1/2+\delta}$.

Now we can repeat the argument before. The condenser can also be shown to be strong, in the sense that with probability $1-2\sqrt{\epsilon}$ over the fixing of the seed, the output is $\sqrt{\epsilon}$-close to having min-entropy $k'-d$ (intuitively, this is because the seed length is d, thus conditioned on the seed the output can lose at most d bits of entropy). Now define a block to be "good" if it contains at least one "good" row that is $\sqrt{\epsilon}$-close to having min-entropy $k'-d$. Again we can show there is an i such that all X^i, Y^i and Z^i are good.

To finish the argument, we need to apply Raz to Y^i and each row of X^i. However now the good row in X^i is not uniform, in fact it may not even have min-entropy rate $> 1/2$. On the other hand, it does have a constant min-entropy rate. Therefore we now first apply the somewhere condenser from [BKS+05, Zuc07] to each row of X^i to boost the min-entropy rate to 0.9. The somewhere condenser outputs another constant number of rows for each row of X^i, and if the row of X^i is good, then one of the outputs is close to having min-entropy rate 0.9. Now we can apply Raz to Y^i and each output of the somewhere condenser, and proceed as before. Since this only increases the number of rows in $(\bar{X}^i, \bar{Y}^i)$ by a constant factor, it does not affect our analysis. Thus we obtain a three source extractor for min-entropy $k = n^{1/2+\delta}$.

6.2.2 The Construction

In this section we present our three source extractor. We have the following algorithm.

Algorithm 6.2.2 ($\mathsf{THExt}(x, y, z)$).

Input: x, y, z — a sample from three independent (n, k)-sources with $k = n^{1/2+\delta}$, for some arbitrary constant $0 < \delta < 1/2$.
Output: w — an m bit string.

Sub-Routines and Parameters:
Let Cond be a $(k_1 \to k_1+d, \epsilon_1)$ condenser from Theorem 3.5.12, such that $k_1 = n^{\delta/10}$, $\epsilon_1 = n^{-\delta/10}$ and $\alpha = 10/\delta$ where α is the parameter α in Theorem 3.5.12.
Let Zuc be a rate-$(0.09\delta \to 0.9, 2^{-\Omega(n)})$-somewhere-condenser form Theorem 3.5.3.
Let Raz be the strong 2-source extractor from Theorem 3.5.7.
Let Ext be the strong extractor from Theorem 3.5.13.
Let $\mathsf{2SRExt}$ be the extractor for two independent aligned SR-source from Theorem 3.5.8.

```
1. For every $s \in \{0,1\}^d$ compute $x_s = \mathsf{Cond}(x, s)$. Concatenate $\{x_s\}$ in the binary order of $s$ to form a matrix of $2^d$ rows. Divide the rows of the matrix sequentially into blocks $x^1, \cdots x^t$ with each block consisting of $\sqrt{n}$ rows. Do the same things to $y$ and $z$ and obtain blocks $y^1, \cdots, y^t$ and $z^1, \cdots, z^t$.

2. (Compute an SR-source from $x$ and $y$). For $i = 1$ to $t$ do the following.
    • For each row $x^i_j$ in block $x^i$ (there are $\sqrt{n}$ rows), apply $\mathsf{Zuc}$ to get a constant number of outputs $\{x^i_{j\ell}\}$.
    • For each $x^i_{j\ell}$ compute $v^i_{j\ell} = \mathsf{Raz}(x^i_{j\ell}, y_i)$ and output $m_2 = \Omega(k_1)$ bits.
    • For each $v^i_{j\ell}$ compute $\mathsf{Ext}(x, v^i_{j\ell})$, output $0.9n^{1/2+\delta}$ bits and concatenate these strings to form a matrix $\bar{x}^i$. Similarly for each $v^i_{j\ell}$ compute $\mathsf{Ext}(y, v^i_{j\ell})$, output $0.9n^{1/2+\delta}$ bits and concatenate these strings to form a matrix $\bar{y}^i$.
    • Compute $r_i = \mathsf{2SRExt}(\bar{x}^i, \bar{y}^i)$ and output $m_3 = 0.8n^{1/2+\delta}$ bits.

3. Concatenate $\{r_i, i = 1, \cdots, t\}$ to form a matrix $sr_y$.

4. Repeat step 2 and step 3 above for $x$ and $z$ to obtain a matrix $sr_z$.

5. Output $w = \mathsf{2SRExt}(sr_y, sr_z)$.
```

6.2.3 Analysis of the Extractor

In this section we analyze the three source extractor. Specifically, we prove the following theorem.

Theorem 6.2.3. *For any constant $0 < \delta < 1/2$, let X, Y, Z be three independent (n, k) sources with $k = n^{1/2+\delta}$. Then*

$$|\mathsf{THExt}(X, Y, Z) - U_m| < n^{-\Omega(\delta)}$$

with $m = \Omega(k)$.

Proof. Our goal is to show that SR_y and SR_z is (close to) a convex combination of independent aligned SR-sources with few rows. Then we'll be done by Theorem 3.5.8.

First note that $k > k_1$, thus an (n, k)-source is also an (n, k_1) source. Let S be the uniform distribution over d bits independent of (X, Y, Z). By Theorem 3.5.12 we have that $X_S = \mathsf{Cond}(X, S)$ is ϵ_1-close to being an $(m_1, k_1 + d)$ source with $m_1 \leq 2d + (1 + \alpha)k_1 < (2 + \alpha)k_1$, since $d = (1 + 1/\alpha) \cdot (\log n + \log k + \log(1/\epsilon)) + O(1) = O(\log n) = O(\log k_1)$.

Now by Lemma 3.5.15, with probability $1-2\sqrt{\epsilon_1}$ over the fixings of $S = s$, X_s is $\sqrt{\epsilon_1}$-close to being an $(m_1, k_1 - d)$ source. We say that a row X_s is good if it is $\sqrt{\epsilon_1}$-close to being an $(m_1, k_1 - d)$ source, and we say that a block X^i is good if it contains at least one good row. It's easy to see that the fraction of "bad" blocks in $\{X^i\}$ is at most $2\sqrt{\epsilon_1}$. Similarly this is also true for the blocks $\{Y^i\}$ and $\{Z^i\}$.

Now since $2\sqrt{\epsilon_1} < 1/3$, by the union bound there exists an i s.t. X^i, Y^i and Z^i are all good blocks. Without loss of generality assume that X^1, Y^1 and Z^1 are all good blocks. We are going to show that the first rows of SR_y and SR_z are close to uniform, thus SR_y and SR_z are aligned somewhere random sources.

We first show this for SR_y. Note that X^1 is a good block and Y^1 is also a good block. Therefore at least one row in X^1 is good. Without loss of generality assume that X_1^1 is a good row. Thus X_1^1 is $\sqrt{\epsilon_1}$-close to being an $(m_1, k_1 - d)$ source. Note that $k_1 - d > 0.99k_1$ since $d = O(\log k_1)$ and $m_1 < (2+\alpha)k_1$. Thus X_1^1 is close to having min-entropy rate $0.99/(2+\alpha) = 0.99/(2+10/\delta) = 0.99\delta/(10+2\delta) > 0.09\delta$ since $\delta < 1/2$.

Therefore by Theorem 3.5.3, $\mathsf{Zuc}(X_1^1)$ is $\sqrt{\epsilon_1}+2^{-\Omega(m_1)} = n^{-\Omega(\delta)}$-close to being a somewhere rate 0.9 source with $O(1)$ rows, and the length of each row is $\Omega(m_1) = \Omega(k_1)$.

We now have the following claim.

Claim 6.2.4. *With probability $1 - n^{-\Omega(\delta)}$ over the fixings of X^1 and Y^1, $\bar{X}^1$ is a deterministic function of X, $\bar{Y}^1$ is a deterministic function of Y, and they are $2^{-n^{\Omega(1)}}$-close to being two aligned $(O(\sqrt{n}) \times 0.9n^{1/2+\delta})$ SR-sources.*

proof of the claim. Note that $\mathsf{Zuc}(X_1^1)$ is $n^{-\Omega(\delta)}$-close to being a somewhere rate 0.9 source with $O(1)$ rows, and each row has length $\Omega(k_1)$. For simplicity, consider the case where $\mathsf{Zuc}(X_1^1)$ is close to an elementary somewhere rate 0.9 source (since $\mathsf{Zuc}(X_1^1)$ is $n^{-\Omega(\delta)}$-close to being a convex combination of such sources, this increases the error by at most $n^{-\Omega(\delta)}$). Without loss of generality assume that the first row X_{11}^1 is $n^{-\Omega(\delta)}$-close to having rate 0.9. Since Y^1 is a good block Y^1 is $\sqrt{\epsilon_1}$-close to having min-entropy at least $k_1 - d > 0.99k_1$, and Y^1 has length $m_1\sqrt{n} = \mathrm{poly}(k_1)$. Therefore by Theorem 3.5.7 we have

$$|(V_{11}^1, X_{11}^1) - (U_{m_2}, X_{11}^1)| < n^{-\Omega(\delta)} + 2^{-\Omega(k_1)} = n^{-\Omega(\delta)}$$

and

$$|(V_{11}^1, Y^1) - (U_{m_2}, Y^1)| < n^{-\Omega(\delta)} + 2^{-\Omega(k_1)} = n^{-\Omega(\delta)}.$$

Therefore with probability $1 - n^{-\Omega(\delta)}$ over the fixing of X_{11}^1, V_{11}^1 is $n^{-\Omega(\delta)}$-close to uniform. Since X_{11}^1 is a deterministic function of X^1, this also implies that with probability $1 - n^{-\Omega(\delta)}$ over the fixing of X^1, V_{11}^1 is $n^{-\Omega(\delta)}$-close to uniform. Note that after this fixing, V_{11}^1 is a deterministic function of Y^1, and is thus independent of X. Moreover, note that the length of X^1 is $m_1\sqrt{n} = O(k_1\sqrt{n}) = O(n^{1/2+\delta/10})$. Thus by Lemma 3.5.14, with probability $1 - 2^{-n^{\delta/10}}$ over the fixing of X^1, X has min-entropy at least $n^{1/2+\delta} - O(n^{1/2+\delta/10}) - n^{\delta/10} > 0.99n^{1/2+\delta}$.

Therefore, now by the strong extractor property of Ext from Theorem 3.5.13, with probability $1 - n^{-\Omega(\delta)}$ over the fixing of V_{11}^1, $\mathsf{Ext}(X, V_{11}^1)$ is $2^{-n^{\Omega(1)}}$-close to uniform. Since now V_{11}^1 is a deterministic function of Y^1, this also implies that with probability $1 - n^{-\Omega(\delta)}$ over the fixing of Y^1, $\mathsf{Ext}(X, V_{11}^1)$ is $2^{-n^{\Omega(1)}}$-close to uniform. Note also that after this fixing $\mathsf{Ext}(X, V_{11}^1)$ is a deterministic function of X. Therefore we have shown that with probability $1 - n^{-\Omega(\delta)}$ over the fixing of X^1 and Y^1, $\bar{X}^1$ is a deterministic function of X and is $2^{-n^{\Omega(1)}}$-close to an SR-source.

By symmetry we can also show that with probability $1 - n^{-\Omega(\delta)}$ over the fixing of X^1 and Y^1, $\bar{Y}^1$ is a deterministic function of Y and is $2^{-n^{\Omega(1)}}$-close to an SR-source.

Since both in $\bar{X}^1$ and $\bar{Y}^1$, the first row is close to uniform, they are close to being aligned SR-sources. ■

Now we have the following claim.

Claim 6.2.5. *With probability $1 - n^{-\Omega(\delta)}$ over the fixing of X, Y^1, Z^1, SR_y and SR_z are two independent aligned somewhere random sources.*

proof of the claim. Note that $\bar{X}^1$ and $\bar{Y}^1$ each has $\sqrt{n}$ rows, and each row has $0.9n^{1/2+\delta}$ bits. Thus by Claim 6.3.18 and Theorem 3.5.8, we have

$$|(R_1, \bar{X}^1) - (U_{m_3}, \bar{X}^1)| < n^{-\Omega(\delta)}.$$

This means that with probability $1 - n^{-\Omega(\delta)}$ over the fixing of $\bar{X}^1$, R_1 is $n^{-\Omega(\delta)}$-close to uniform. Since we have fixed X^1 and Y^1 before, now $\bar{X}^1$ is a deterministic function of X. Thus this also implies that with probability $1 - n^{-\Omega(\delta)}$ over the fixing of X, R_1 is $n^{-\Omega(\delta)}$-close to uniform. Moreover, now R_1 (and all the other R_i's) is a deterministic function of Y. Therefore with probability $1 - n^{-\Omega(\delta)}$ over the fixing of X, SR_y is $n^{-\Omega(\delta)}$-close to an SR-source. Moreover, after this fixing SR_y is a deterministic function of Y.

By the same argument, it follows that with probability $1 - n^{-\Omega(\delta)}$ over the fixings of X and Z^1, SR_z is $n^{-\Omega(\delta)}$-close to an SR-source, and SR_z is a deterministic function of Z. Thus SR_y and SR_z are independent.

Since both in SR_y and SR_z, the first row is close to uniform, they are close to being aligned independent SR-sources. ■

Now note that

$$2^d = O((nk_1/\epsilon_1)^{1+1/\alpha}) = O(n^{(1+\delta/5)(1+\delta/10)}) = O(n^{1+\delta/3}).$$

Thus SR_y and SR_z each has $2^d/\sqrt{n} = O(n^{1/2+\delta/3})$ rows, and each row has $0.8n^{1/2+\delta}$ bits. Therefore again by Theorem 3.5.8 we have

$$|\mathsf{THExt}(X, Y, Z) - U_m| < n^{-\Omega(\delta)} + 2^{-n^{\Omega(1)}} = n^{-\Omega(\delta)},$$

and $m = \Omega(k)$. ■

6.3 Affine Extractors and Dispersers

In this section we present our constructions of affine extractors and dispersers. In the case where the underlying field is $\mathbb{F} = \mathrm{GF}(2)$, it is well known how to construct extractors for affine sources with entropy rate greater than $1/2$. However the problem becomes much harder as the entropy rate drops to $1/2$ and below $1/2$. The best construction of affine extractors in this case is due to Bourgain [Bou07], who gave an extractor for affine sources with arbitrarily linear entropy that can output a constant fraction of the entropy with exponentially small error. Based on Bourgain's techniques, Yehudayoff [Yeh10] gave another construction in the same spirit that is slightly simpler to analyze. The construction of [Yeh10] also slightly improves Bourgain's result. Rao [Rao09] constructed extractors for affine sources with entropy as small as $\mathrm{polylog}(n)$, as long as the subspace of X has a basis of low-weight vectors.

The construction of [Bou07], and the slight modification of [Yeh10], are thus previously the only known constructions for general affine sources over $\mathrm{GF}(2)$ with arbitrarily linear entropy. However, both of these constructions make extensive use of inequality manipulations, and are quite complicated to analyze. Also, both of these constructions need to choose a polynomial very carefully, so that eventually the inequality manipulations would result in an estimate of exponential sums in finite fields. The polynomial chosen thus determines the performance of the extractor. From our point of view the choice of the polynomial is somewhat subtle and a bit unnatural. Thus one may ask the natural question of whether there exist other constructions of affine extractors for arbitrarily linear entropy sources.

Here we give a new construction of affine extractors that matches the results of [Bou07] and [Yeh10]. Our construction is conceptually much cleaner than those of [Bou07] and [Yeh10]. It mainly uses tools from previous constructions of extractors and produces the desired polynomial in a very natural way. Our construction gives new insights into the nature of affine extractors, and we believe that having two different constructions with the best known parameters more than double the chances of achieving even better constructions.

In the case of constructing dispersers for affine sources over $\mathrm{GF}(2)$, Barak et al. [BKS$^+$05] gave an affine disperser for sources with arbitrarily linear entropy that outputs a constant number of bits. Ben-Sasson and Kopparty [BSK09] constructed dispersers for affine sources with entropy $\Omega(n^{4/5})$. However, their construction only outputs 1 bit. In this paper, we construct dispersers for slightly sub-linear entropy affine sources that output $n^{\Omega(1)}$ bits.

We have the following theorems.

Theorem 6.3.1. *For every $\delta > 0$ there exists an efficient family of functions* $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ *such that $m = \Omega(n)$ and for every affine source X with entropy δn,* $|\mathsf{AExt}(X) - U_m| = 2^{-\Omega(n)}$.

Theorem 6.3.2. *There exists a constant $c > 1$ and an efficient family of functions* $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ *such that $m = n^{\Omega(1)}$ and for every affine source X with entropy $cn/\sqrt{\log \log n}$,* $|\mathsf{AExt}(X) - U_m| = 2^{-n^{\Omega(1)}}$.

Theorem 6.3.3. *There exists a constant $c > 1$ and an efficient family of functions* $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^m$ *such that $m = n^{\Omega(1)}$ and for every affine source X with entropy $cn/\sqrt{\log n}$,* $|\mathsf{Supp}(\mathsf{ADisp}(X))| = 2^m$.

6.3.1 Overview of Our Constructions

In some sense, our construction of affine extractors is similar in the spirit to our construction of 2-source extractors above. There we constructed an extractor for two independent weak random sources with linear entropy. The basic idea is that, when a source with linear entropy is divided into some constant number of blocks, it becomes (up to a convex combination and a small error) a somewhere block source. If we know the location of a good block (a block that contains high entropy but not all the entropy of the source), then it is fairly easy to extract random bits with the help of the other source. The problem is that we don't know where the good block is, thus we have to try all the possibilities and end up with a constant number of outputs such that one of them is close to uniform. These outputs can be correlated. The authors then modified this procedure by using computational assumptions to make the outputs "independent" of each other in some sense, and thus the xor of the outputs is close to uniform.

Here, we would like to do something similar. It is still true that when we divide an affine source with linear entropy into some constant number of blocks, it becomes an affine somewhere block source. However, unlike the two-source extractor, we now have two problems. First, we have only one affine source. Second, we do not rely on any computational assumptions. Thus we need some new techniques to deal with these problems.

6.3.1.1 Extractors for one affine block source

An affine block source is an affine source that consists of two blocks, such that the first block has some entropy, and the second block also has some entropy, even if conditioned on the fixing of the first block. We first construct an extractor for an affine block source.

To do this, we use special kinds of two-source extractors and seeded extractors: strong linear two-source (or seeded) extractors. A two-source (or seeded) extractor is strong if for most choices of one source (or the seed), the output is close to uniform. It is linear if for any fixing of one source (or the seed), the output is a linear function of the other source. There are known strong linear seeded and two-source extractors. For example, Trevisan's extractor [Tre01] is a strong linear seeded extractor, while the inner product function is a strong linear two-source extractor when the sum of the entropy rates of the two sources is greater than 1.

Given these extractors, our extractor for an affine block source is simple. Assume (X_1, X_2) is an affine block source with each block having entropy rate δ. We first use the condenser based on sum-product theorems [BKS+05, Zuc07] to convert X_1 into a somewhere rate-$(1-\delta/2)$ source, which is a matrix that has a constant number of rows such that one of the rows has entropy rate $1-\delta/2$. Next we apply the inner product function to each row and X_2. Although X_1 and X_2 are correlated, note that X_1 is a linear function of the source (X_1, X_2). Thus the structure of affine sources (Lemma 2.3.22) and the properties of strong linear two-source extractors guarantee that the output is close to a convex combination of affine somewhere random sources.

Note the affine somewhere random source has very few rows (a constant number), thus we can now use Rao's extractor for such sources [Rao09]. Rao's extractor uses the strong linear seeded extractor, and reduces the number of rows in the somewhere random source by a half each time. Thus by repeating a constant number of times we get an output R that is close to uniform.

In the real construction, we use R as a seed and apply the strong linear seeded extractor again to the source to get another output U that is close to uniform. The purpose is to make sure that the output is a linear function of the source (when conditioned on the fixings of some random variables), thus we can use the structure result in Lemma 2.3.22.

Again, the problem is that we don't know where the good block is. Thus we try all the possibilities and get a constant number of outputs $\{R_i, U_i\}$ such that one of $\{U_i\}$ is close to uniform.

6.3.1.2 Obtain one output

Similar to the two-source extractor construction, we need to find a way to make the outputs somewhat "independent" of each other, so that we can take the xor of them and get a string that is close to uniform. To do this, we use properties of polynomials over GF(2).

First of all, in the analysis we can fix the first good block X_g (though we don't know which block it is) and fix all random variables produced before this block. By restricting the size of the random variables produced (so that they don't steal much entropy), it is not hard to show that conditioned on all these fixings, with high probability U_g (the output of the affine block source extractor applied to (X_g, X)) is uniform. Thus in particular this means that conditioned on $(U_1, \cdots, U_{g-1})$, U_g is still uniform. The problem is that $U_{g+1}, \cdots, U_t$ (t is the number of blocks) could be correlated with U_g.

Our key observation is that, when we fix all the random variables produced before X_g and fix (X_g, R_g), U_g is a linear function L of the source X. Lemma 2.3.22 thus tells us that there exists an affine function L_g and an affine source B_g independent of U_g such that $X = L_g(U_g) + B_g$. Therefore, now conditioned on any fixing of $B_g = b$ the source X is an affine function (degree 1 polynomial) of U_g!

Since the subsequent computations are all functions of X, the random variables $U_{g+1}, \cdots, U_t$ are all functions of U_g. We note that any boolean function on $\{0,1\}^n$ can be expressed as a polynomial over GF(2), though the degree of the polynomial can be very high. On the other hand, if we can ensure that each bit of $U_{g+1}, \cdots, U_t$ is a low degree (say degree d) polynomial of the bits of U_g, then there is something natural to do–instead of just outputting U_i, we output a bit Z_i, which is a degree d_i polynomial of the bits of U_i. Now $Z_{g+1}, \cdots, Z_t$ are polynomials of degree $d \cdot d_{g+1}, \cdots, d \cdot d_t$ of the bits of U_g. As long as $d_g > max\{d \cdot d_{g+1}, \cdots, d \cdot d_t\}$, the xor of the Z_i's cannot be a constant. Thus we get a non-constant bit. In other words, we get a one-bit disperser. In fact, in the construction we choose $d_i > d_{i+1}$ for all i. Thus as long as each bit of $U_{g+1}, \cdots, U_t$ is a degree d polynomial of the bits of U_g, it suffices to have $d_g > d \cdot d_{g+1}$. Since we don't know the position of the good block, we take $d_i > d \cdot d_{i+1}$ for all i.

In the construction for linear entropy affine sources, the degree d is a constant. Thus we can take all the d_i's to be constants. The polynomial Z_i we take is simple too: just take d_i bits of U_i and compute the product. We show that each U_i has $\Omega(n)$ bits, thus we can take $\Omega(n)$ different blocks of d_i bits. Therefore instead of just outputting one bit, we can output $\Omega(n)$ bits. The only thing left now is to make sure each bit of $U_{g+1}, \cdots, U_t$ is a low degree polynomial of the bits of U_g.

6.3.1.3 Extractors that are low-degree polynomials

Let us examine how U_i is computed. First we convert a block X_i of X into a somewhere rate-$(1 - \delta/2)$ source Y_i, using the condenser based on sum-product theorems. In this step we need to apply the condenser a constant number of times, while each time the output is a degree 2 polynomial of the inputs. Next we apply the inner product function to each row of Y_i and X to obtain an affine somewhere random source SR_i. Again the output is a degree 2 polynomial of the inputs. We then use the extractor for such a source from [Rao09] to get a random seed R_i. Finally we use R_i and apply a strong linear seeded extractor to X to obtain U_i. In these two steps, we need to use a strong linear seeded extractor, which may not be a low-degree polynomial.

We note that in the above discussion some of the polynomials are over a finite field $\mathbb{F}_q$. However, in this paper all the finite fields $\mathbb{F}_q$ have size $q = 2^s$ for some integer s. Thus by mapping a string in $\{0, 1\}^s$ to an element in $\mathbb{F}_q$ using the standard binary expression, we see that whenever a function is a degree d polynomial over $\mathbb{F}_q$, each bit of the output is also a degree d polynomial (over GF(2)) of the input bits. Therefore all we need now is to construct a strong linear seeded extractor such that each bit of the output is a constant degree polynomial of the input bits.

The starting point is the well-known leftover hash lemma [ILL89], which roughly says that if R is uniformly distributed over a finite field $\mathbb{F}$ and X is a weak source over $\mathbb{F}$, then the last several bits of $R \cdot X$ (the operation is in $\mathbb{F}$) is a strong extractor. Note this is also a linear seeded extractor and the output is a degree 2-polynomial of the inputs. The only thing bad about this extractor is that it requires the seed to have as many bits as the source, which we cannot afford. Nevertheless, we use this extractor as an ingredient in our construction.

Our actual construction of the strong linear seeded extractor consists of three steps. First, we take a short seed and divide the source into many blocks with each block having the same number of bits as the seed. We apply the extractor from the leftover hash lemma to the seed and every block of the source, and concatenate the outputs. Thus we get a somewhere random source. Next, we take another short seed and apply the strong linear merger from [LRVW03] to the somewhere random source. We then get a short source with linear entropy. Finally, we take a short seed and apply the extractor from the leftover hash lemma to the short source, and we obtain bits that are close to uniform. It is easy to check that the extractor is a linear seeded extractor. It is also strong because in each step the extractor or merger is strong. It can also be easily checked that the output is a degree 4 polynomial of the inputs.

Once we use this extractor, it is fairly straight forward to check that each bit of U_i is a constant degree polynomial of the bits of X, and thus the bits of U_g.

6.3.1.4 Affine Extractors

Above we discussed the techniques in our construction of affine dispersers. Next we discuss how to get an affine extractor from the disperser.

Recall that in the above we take Z_g to be a degree $d_g > d \cdot d_{g+1}$ polynomial of the bits of U_g (the product of some d_g bits) and we argue that Z_g xored with $Z_{g+1}, \cdots, Z_t$ cannot be a constant. The key observation here is that this is not only true for $Z_{g+1}, \cdots, Z_t$, but also true for any polynomial of degree at most $d_g - 1$. In other words, let $d' = d_g - 1$ and let $P_{d'}$ stand for the class of all polynomials of degree at most d' of the bits of U_g, then the correlation between Z_g and $P_{d'}$ is at most $1 - 1/2^{d'}$. Therefore if we take several independent blocks of U_g, each having d_g bits, and take the xor of the products of the bits in each block, the correlation with $P_{d'}$ will decrease exponentially by the xor lemma from [VW08, BKS+10]. Since U_g has $\Omega(n)$ bits we can take $\Omega(n)$ blocks and the correlation will decrease to $2^{-\Omega(n)}$. Thus we get an extractor that outputs one bit with exponentially small error.

To output more bits, we divide the bits of U_g into $\Omega(n)$ blocks, each having d_g bits. We next take the generating matrix of a binary linear asymptotically good code, with the codeword length equaling the number of blocks. For each row of the generating matrix we associate one bit. The bit is computed by taking the xor of the products of the bits in the blocks that are indexed by the 1's in this row. By the properties of the asymptotically good linear code, the xor of any non-empty subset of these bits, will be the xor of the products of the bits from $\Omega(n)$ blocks. Thus it will be $2^{-\Omega(n)}$-close to uniform. In other words, these bits form a $2^{-\Omega(n)}$-biased space. Therefore we can take $\Omega(n)$ bits that are $2^{-\Omega(n)}$-close to uniform.

6.3.1.5 A word about the affine extractor for entropy rate 1/2

Our construction uses an affine extractor for entropy rate $1/2$ as a black box when we use Rao's extractor [Rao09]. For our application we need this extractor to be a constant degree polynomial and output a linear fraction of the entropy. In this paper we use a result in [Bou07]. A potential alternative is a result in [BSK09]. There the authors showed that given any non-trivial linear map $\pi : \mathbb{F}_{2^n} \to \mathbb{F}_2$, $\pi(X^7)$ is an extractor for any affine source X with entropy $2n/5 + 10 + d$, when X is viewed as an element in $\mathbb{F}_{2^n}$. A recent work of Haramaty and Shpilka [HS10] shows that the error of this extractor is $2^{-d^{\Omega(1)}}$. Combined with our techniques, this result already gives an extractor for affine sources with entropy rate $1/2$ that can output $n^{\Omega(1)}$ bits with error $2^{-n^{\Omega(1)}}$. Specifically, we interpret X^7 as a string in $\{0,1\}^n$, and we take the generating matrix of a binary linear asymptotically good code with codeword length n. For each codeword in the generating matrix we compute a bit Z_i which is the xor of the bits of X^7 that are indexed by 1's in this codeword. Thus the xor of any subset of $\{Z_i\}$ is a non-trivial linear map from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$, and thus is $2^{-n^{\Omega(1)}}$-close to uniform if X has entropy $n/2$. Therefore we can take $n^{\Omega(1)}$ bits from $\{Z_i\}$ with error $2^{-n^{\Omega(1)}}$. It is conjectured in [BSK09] that the error of the extractor should be $2^{-\Omega(n)}$. If this is true then we can output $\Omega(n)$ bits with error $2^{-\Omega(n)}$, and this can replace the extractor in [Bou07] completely.

6.3.2 The Constructions

Before we present the construction, we first give the constructions of several ingredients.

6.3.2.1 Low Degree Strong Linear Seeded Extractors

In this section we describe our construction of a strong linear seeded extractor. The extractor has the property that each bit of the output is a degree 4 polynomial of the bits of the input.

Theorem 6.3.4. *There exists a constant $0 < \beta < 1$ such that for every $0 < \delta < 1$ and any $1/\sqrt{n} < \alpha < 1$ there exists a polynomial time computable function $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ s.t.*

- *$d \leq \alpha n, m \geq \beta\delta\alpha n$.*
- *For any $(n, \delta n)$-affine source [1]X, let R be the uniform distribution on $\{0,1\}^d$ independent of X. Then $(\mathsf{LSExt}(X, R), R)$ is $2^{-\Omega(\delta\alpha^2 n)}$-close to uniform.*
- *Each bit of the output is a degree 4 polynomial of the bits of the two inputs, and for any fixing of r the output is a linear function of x.*

Proof. The construction of the extractor consists of 3 steps:

Step 1: We take $d_1 = \alpha n/3$ random bits R_1 and divide X into $3/\alpha$ blocks of equal length[2] $X = X_1 \circ \cdots \circ X_t$, where each X_i has $\alpha n/3$ bits. We now apply the function Hash as in Lemma 3.5.18 to every X_i and R_1 and concatenate the output to be $Y = Y_1 \circ \cdots \circ Y_t$, where $Y_i = \mathsf{Hash}(X_i, R_1)$. We let each Y_i output $l_1 = 0.9\delta\alpha n/3 = 0.3\delta\alpha n$ bits.

Claim 6.3.5. *With probability $1 - 2^{-\Omega(\delta\alpha n)}$ over R_1, Y is $2^{-\Omega(\delta\alpha n)}$-close to being a $(l_1, 3/\alpha)$-somewhere random source.*

Proof of the claim. By Lemma 2.3.23 there exist integers $k_1, \cdots, k_t$ such that for any fixing of the previous blocks, X_i has entropy k_i. Note $\sum_{i=1}^{t} k_i = \delta n$. Thus there exists $1 \leq i \leq t$ such

[1] Generally we don't need the source to be affine. However the analysis is simpler if the source is an affine source (mainly because in this case we don't need to go into convex combinations as in the case where the source is a general weak source).

[2] For simplicity, we assume that $3/\alpha$ is an integer, this does't affect the analysis.

that $k_i \geq \delta\alpha n/3$. Now by Lemma 3.5.18 we know that (Y_i, R_1) is $2^{-\Omega(\delta\alpha n)}$-close to (U, R_1). Therefore with probability $1 - 2^{-\Omega(\delta\alpha n)}$ over R_1, Y_i is $2^{-\Omega(\delta\alpha n)}$-close to uniform. Thus Y is $2^{-\Omega(\delta\alpha n)}$-close to being a $(l_1, 3/\alpha)$-somewhere random source. ■

Step 2: Let $\epsilon = 2^{-c\delta\alpha^2 n}$ for a constant $0 < c < 1$ to be chosen later. We take d_2 random bits R_2 and apply the merger as in Theorem 3.5.20 with parameter ϵ. Let $W = \mathsf{Merg}(Y, R_2)$.

Claim 6.3.6. *$d_2 \leq \alpha n/3$ and with probability $1 - 2^{-\Omega(\delta\alpha^2 n)}$ over R_2, W is $2^{-\Omega(\delta\alpha^2 n)}$-close to having min-entropy $0.4l_1$.*

Proof of the claim. By Theorem 3.5.20 the seed length $d_2 = O(c\delta\alpha^2 n/\alpha) = O(c\delta\alpha n)$. Note that $l_1 = 0.3\delta\alpha n$. Thus we can choose c s.t. $d_2 \leq \alpha n/3$ and the output of the merger has length $m = l_1/2 - O(d_2) \geq 0.4l_1$. Now by Theorem 3.5.20 we know that with probability $1 - 2^{-\Omega(\delta\alpha^2 n)}$ over R_2, W is $2^{-\Omega(\delta\alpha^2 n)}$-close to having min-entropy $0.4l_1$. ■

Step 3: Now W is a random variable over $l_1 < \alpha n/3$ bits. Take $d_3 = l_1$ random bits R_3 and apply the function Hash as in Lemma 3.5.18 to W and R_3 and output $m = 0.3l_1$ bits. The final output is $Z = \mathsf{Hash}(W, R_3)$.

The number of random bits used is $d = d_1 + d_2 + d_3 \leq \alpha n$. The number of bits of the output is $m = 0.3l_1 \geq \beta\delta\alpha n$ for some constant $0 < \beta < 1$. By Lemma 3.5.18 with probability $1 - 2^{-\Omega(\delta\alpha n)}$ over R_3, Z is $2^{-\Omega(\delta\alpha n)}$-close to uniform. Thus with probability $1 - 2^{-\Omega(\delta\alpha^2 n)}$ over R, Z is $2^{-\Omega(\delta\alpha^2 n)}$-close to uniform, which implies that $(\mathsf{LSExt}(X, R), R)$ is $2^{-\Omega(\delta\alpha^2 n)}$-close to uniform.

In each of the 3 steps the degree of the polynomial goes up by 1. Thus each bit of the output is a degree 4 polynomial of the bits of the two inputs. Now observe in each step for any fixing of $R_i = r$ the output is a linear function, thus for any fixing of $R = r$ the output is a linear function of x. ■

In the special case where α and δ are constants, we get the following corollary:

Corollary 6.3.7. *For all constants $0 < \delta, \alpha < 1$ there exists a polynomial time computable function $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ and a constant $0 < \beta < 1$ such that*

- *$d \leq \alpha n, m \geq \beta n$.*
- *For any $(n, \delta n)$-affine source X, let R be the uniform distribution on $\{0,1\}^d$ independent of X. Then $(\mathsf{LSExt}(X, R), R)$ is $2^{-\Omega(n)}$-close to uniform.*

- *Each bit of the output is a degree 4 polynomial of the bits of the two inputs, and for any fixing of r the output is a linear function of x.*

6.3.2.2 Extractors for Affine Somewhere Random Sources with Few Rows

In this section we describe our extractor for an affine somewhere random source with few rows. The construction is essentially the same as that in [Rao09], except that we use our low degree strong linear seeded extractor in the construction.

We need the following definition about the slice of a concatenation of strings.

Definition 6.3.8. [Rao09] Given ℓ strings of length n, $x = x_1, \cdots, x_\ell$, define $\mathsf{Slice}(x, w)$ to be the string $x' = x'_1, \cdots, x'_\ell$ such that for each i x'_i is the prefix of x_i of length w.

Algorithm 6.3.9 ($\mathsf{AffineCondense}(x)$)**.**

Input: x — a $t \times r$ matrix with $t < \sqrt[4]{r}$.
Output: y — a $\lceil t/2 \rceil \times m$ matrix with $m = \Omega(r/t^2)$.

Sub-Routines and Parameters:
Let $w = r/(10t)$.
Let $\mathsf{BAExt} : \{0,1\}^n \to \{0,1\}^d$ be the affine extractor from Theorem 3.5.21.
Let $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be the strong linear seeded extractor from Theorem 6.3.4.

1. Let z be the $\lceil t/2 \rceil \times 2w$ matrix obtained by concatenating pairs of rows in $\mathsf{Slice}(x, w)$, i.e., the i'th row z_i is $\mathsf{Slice}(x, w)_{2i-1} \circ \mathsf{Slice}(x, w)_{min\{2i,t\}}$.
2. Let s be the $\lceil t/2 \rceil \times d$ matrix whose i'th row is $\mathsf{BAExt}(z_i)$.
3. Let y be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{LSExt}(x, s_i)$.

Algorithm 6.3.10 (AffineSRExt(x)).

Input: x — a $t \times r$ matrix.
Output: z — an m bit string with $m = r/t^{O(\log t)}$.

Sub-Routines and Parameters:

```
1. If $x$ has one row, output $x$.
2. Else, set $y$ to be the output of AffineCondense($x$).
3. Set $x = y$ and go to the first step.
```

Lemma 6.3.11. *For any $t \times r$ affine somewhere random source X with $t < \sqrt[4]{r}$, $\mathsf{AffineCondense}(X)$ is $2^{-\Omega(r/t^4)}$-close to a convex combination of $\lceil t/2 \rceil \times m$ affine somewhere random sources, where $m = \Omega(r/t^2)$. Moreover, each bit of the output is a constant degree polynomial of the input bits.*

Proof. We essentially follow the proof in [Rao09], except that we use the specific strong linear seeded extractor LSExt.

Let $Z = \mathsf{Slice}(X, w)$ as in the algorithm. Note that $\mathsf{Slice}(X, w)$ is a linear function of X. Thus by Lemma 2.3.22, there exist independent affine sources A and B s.t. $X = A + B$, $H(A) = H(\mathsf{Slice}(A, w))$ and for every $b \in \mathsf{Supp}(B)$, $\mathsf{Slice}(b, w) = 0$. This implies that $Z = \mathsf{Slice}(X, w) = \mathsf{Slice}(A, w) + \mathsf{Slice}(B, w) = \mathsf{Slice}(A, w)$ is independent of B and $H(B) = H(X) - H(A) = H(X) - H(\mathsf{Slice}(X, w)) \geq r - wt$.

Note that Z is a linear function of X, thus conditioned on any fixing $Z = z$, $X|Z = z$ is an affine source. Moreover, conditioned on any fixing $Z = z$, Y is a linear function of $X|Z = z$ (because LSExt is a linear seeded extractor). Thus conditioned on any fixing $Z = z$, $Y|Z = z$ is affine. We next show that with high probability over $z \leftarrow_R Z$, $Y|Z = z$ is somewhere random.

Since X is somewhere random, there exists an index h s.t. Z_h is an affine source with entropy rate 1/2. Therefore by Theorem 3.5.21, s_h has $\Omega(w)$ bits and

$$|S_h - U_d| \leq 2^{-\Omega(w)}. \tag{6.4}$$

Note that S_h is a deterministic function of Z and is thus independent of B. Also s_h has $d = \Omega(w) = \Omega(r/t) = \Omega(1/t^2 \cdot tr)$ bits. Note that B has tr bits and $H(B) \geq r - wt \geq 0.9r$. Let

U_d be the uniform distribution over d bits independent of B. Then by Theorem 6.3.4 we have that $\mathsf{LSExt}(B, S_h)$ has $m = \Omega(1/t^3 \cdot tr) = \Omega(r/t^2)$ bits and

$$\Pr_{u \leftarrow_R U_d}[|\mathsf{LSExt}(B, u) - U_m| > \epsilon] < \epsilon$$

where $\epsilon = 2^{-\Omega(r/t^4)}$.

By Proposition 3.1.9 and equation 1 we thus have

$$\Pr_{s \leftarrow_R S_h}[|\mathsf{LSExt}(B, s) - U_m| > 0] < \epsilon + 2^{-\Omega(w)} = 2^{-\Omega(r/t^4)}.$$

For any $s \in \{0,1\}^d$, we have $\mathsf{LSExt}(X, s) = \mathsf{LSExt}(A + B, s) = \mathsf{LSExt}(A, s) + \mathsf{LSExt}(B, s)$. Note that S_h is a deterministic function of Z, Z is a deterministic function of A and $H(A) = H(Z)$. Thus A is also completely determined by Z. Therefore whenever $\mathsf{LSExt}(B, s)|Z = z$ is uniform, $\mathsf{LSExt}(X, s)|Z = z$ is also uniform.

Thus we get

$$\Pr_{z \leftarrow_R Z}[|\mathsf{LSExt}(X|Z = z, s_h) - U_m| > 0] \leq 2^{-\Omega(r/t^4)}.$$

This shows that Y is $2^{-\Omega(r/t^4)}$-close to being a convex combination of affine somewhere random sources. Now note that both BAExt and LSExt are constant degree polynomials, thus each bit of the output is a constant degree polynomial of the input bits. □

Repeating the condenser for $\log t$ times, we get the following theorem:

Theorem 6.3.12. *For every affine $t \times r$ somewhere random source X, $\mathsf{AffineSRExt}(X)$ outputs $m = r/t^{O(\log t)}$ bits that are $2^{-\Omega(r/t^{O(\log t)})}$-close to uniform. Moreover, each bit of the output is a degree $t^{O(1)}$ polynomial of the bits of the input.*

In the special case where t is a constant, we get the following corollary.

Corollary 6.3.13. *For every affine $t \times r$ somewhere random source X with $t = O(1)$, $\mathsf{AffineSRExt}(X)$ outputs $m = \Omega(r)$ bits that are $2^{-\Omega(r)}$-close to uniform. Moreover, each bit of the output is a constant degree polynomial of the bits of the input.*

6.3.2.3 Affine Dispersers for Linear Entropy Sources

Algorithm 6.3.14 ($\mathsf{ADisp}(x)$).

Input: x — an n bit string.
Output: z — an m bit string with $m = \Omega(n)$.

Sub-Routines and Parameters:
Let $\mathsf{Zuc} : \{0,1\}^n \to (\{0,1\}^{\Omega(n)})^{O(1)}$ be a rate-$(\delta/4 \to 1-\delta/4, 2^{-\Omega(n)})$-somewhere-condenser form Theorem 3.5.3.
Let $\mathsf{Had} : (\{0,1\}^n)^2 \to \{0,1\}^{\Omega(n)}$ be the two-source extractor from Theorem 3.5.22, set up to extract from two independent sources whose entropy rates sum up to more than $1 + \delta/4$.
Let $\mathsf{LSExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{m'}$ be the strong linear seeded extractor from Theorem 6.3.4.
Let $\mathsf{AffineSRExt}$ be the extractor for affine somewhere random sources from Theorem 6.3.12.

Divide x into $10/\delta$ blocks $x = x_1 \circ \cdots \circ x_t$ where $t = 10/\delta$ and each block has $\delta n/10$ bits.
For every $i, 1 \le i \le t$ do the following.

1. Let $y_{i1} \circ \cdots \circ y_{i\ell_1} = \mathsf{Zuc}(x_i)$, where y_{ij} is the j'th row of the matrix obtained by applying Zuc to x_i. Note $\ell_1 = O(1)$ and each y_{ij} has $\Omega(n)$ bits.
2. Divide x into ℓ_2 blocks of equal size $x = x'_1 \circ \cdots \circ x'_{\ell_2}$, with each block having the same number of bits as y_{ij}. Note $\ell_2 = O(1)$. Apply Had to every pair of x'_{j_2} and y_{ij_1}, and output $\delta^3 n/(3000\ell_1\ell_2)$ bits. Let sr_i be the matrix obtained by concatenating all the outputs $\mathsf{Had}(x'_{j_2}, y_{ij_1})$, i.e., each row of sr_i is $\mathsf{Had}(x'_{j_2}, y_{ij_1})$ for a pair (x'_{j_2}, y_{ij_1}).
3. Let $r_i = \mathsf{AffineSRExt}(sr_i)$.
4. Let $u_i = \mathsf{LSExt}(x, r_i)$, set up to output at most $\delta^3 n/(3000\ell_1\ell_2)$ bits.
5. Divide the bits of u_i into $s_i = \Omega(n)$ blocks of equal size, with each block having c_i number of bits, for some constant c_i to be chosen later. For every $j = 1, \cdots, s_i$, compute one bit v_{ij} by taking the product of all the bits in the j'th block, i.e., $v_{ij} = \Pi_{\ell=(j-1)c_i+1}^{jc_i} u_{i\ell}$.

Finally, output $\Omega(n)$ bits $\{z_j = \bigoplus_{i=1}^t v_{ij}\}$.

6.3.2.4 Affine Extractors for Linear Entropy Sources

Now we describe our construction of an affine extractor for linear entropy sources. First we need the following definition.

Definition 6.3.15. A linear binary code of length n and rank k is a linear subspace C with dimension k of the vector space $\mathbb{F}_2^n$. Let d be the distance of the code. C is asymptotically good if there exist constants $0 < \delta_1, \delta_2 < 1$ s.t. $k \geq \delta_1 n$ and $d \geq \delta_2 n$.

Note that every linear binary code has an associated generating matrix $G \in \mathbb{F}_2^{k \times n}$, and every codeword can be expressed as vG, for some vector $v \in \mathbb{F}_2^k$.

It is well known that we have explicit constructions of asymptotically good binary linear code. For example, the Justensen codes constructed in [Jus72].

The affine extractor is now described as follows.

Algorithm 6.3.16 ($\mathsf{AExt}(x)$)**.**

Input: x — an n bit string.
Output: z — an m bit string with $m = \Omega(n)$.

Sub-Routines and Parameters:
Let $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^{m_1}$ be the affine disperser in Algorithm 6.3.14, where $m_1 = \Omega(n)$. Let G be the generating matrix of an asymptotically good linear binary code with codeword length m_1. Thus G is an $\alpha m_1 \times m_1$ matrix for some constant $\alpha > 0$. Let G_i stand for the i'th row of the matrix.

1. Run Algorithm 6.3.14 and let the output be $Z = (Z_1, \cdots, Z_{m_1})$ where Z_i is the i'th bit.
2. For each codeword G_i, let $S_i = \{j \in [m_1] : G_{ij} = 1\}$ be the set of index s.t. the bit of the codeword G_i at that index is 1. Define
$$W_i = \bigoplus Z_{j, j \in S_i}$$
to be the bit associated with G_i, i.e., W_i is the XOR of the Z_j's whenever the j'th index of the codeword G_i is 1.
3. Take a constant $0 < \beta \leq \alpha$ to be chosen later. Output $W = (W_1, \cdots, W_{\beta m_1})$.

6.3.3 Analysis of the Affine Disperser

We have the following theorem about the disperser.

Theorem 6.3.17. *For every $\delta > 0$ there exists an efficient family of functions* $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^m$ *such that $m = \Omega(n)$ and for every affine source X with entropy δn,* $|\mathsf{Supp}(\mathsf{ADisp}(X))| = 2^m$.

Proof. We show that Algorithm 6.3.14 is an efficient family of such functions. First, by Lemma 2.3.23, when we divide X into $t = 10/\delta$ blocks of equal size, there exist positive integers $k_1, \cdots, k_t$ s.t. for any fixing of the previous blocks, $H(X_i) = k_i$ and $\sum_{i=1}^{t} k_i = \delta n$. Thus there must exist an i s.t. $k_i \geq \delta n/3t$. Let X_g be the first such block, i.e., g is the smallest i s.t. $k_i \geq \delta n/3t$.

Lemma 6.3.18. *Conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i \in \{1,\cdots,g-1\}}$, X is an affine source with $H(X_g) \geq \delta n/4t$ and $H(X) \geq 3\delta n/5$.*

Proof of the lemma. We first fix $(X_i = x_i)_{i \in \{1,\cdots,g-1\}}$. Note since X_i is a linear function of X, after this fixing X is still an affine source. Now by Lemma 2.3.23, after this fixing $H(X_g) \geq k_g \geq \delta n/3t$ and $H(X) \geq \sum_{i=g}^{t} k_i \geq \delta n - t \cdot \delta n/3t \geq 2\delta n/3$.

Note that conditioned on the fixing of $(X_i = x_i)_{i \in \{1,\cdots,g-1\}}$, SR_i is a linear function of X. Thus we can further fix $(SR_i = sr_i)_{i \in \{1,\cdots,g-1\}}$ and X is still an affine source. Note that SR_i has $\ell_1 \ell_2$ rows and each row has $\delta^3 n/(3000 \ell_1 \ell_2)$ bits, thus SR_i has a total number of $\delta^3 n/3000$ bits. Let $\overline{SR} = SR_1 \circ \cdots \circ SR_{g-1}$ and abuse notation to let $\overline{SR}(X)$ stand for the linear function of X that computes $\overline{SR}$. Note $\overline{SR}$ has at most $\delta^3 n/3000 \cdot t = \delta^2 n/300$ bits. By Lemma 2.3.22, there exist independent affine sources A and B s.t. $X = A + B$, $\overline{SR}(X) = \overline{SR}(A)$ and $H(\overline{SR}(A)) = H(A)$. Thus conditioned on any fixing of $\overline{SR}$,

$$\begin{aligned} H(X) \geq H(B) &\geq 2\delta n/3 - H(A) \\ &= 2\delta n/3 - H(\overline{SR}(X)) \geq 2\delta n/3 - \delta^2 n/300. \end{aligned}$$

Next note that X_g is a linear function of X. Thus $X_g(X) = X_g(A) + X_g(B)$. Therefore $H(X_g(B)) = H(X_g) - H(X_g(A)) \geq H(X_g) - H(A) \geq \delta n/3t - \delta^2 n/300$. Since $\overline{SR}(X) = \overline{SR}(A)$ we have that conditioned on any fixing of $\overline{SR}(X)$, $H(X_g) \geq H(X_g(B)) \geq \delta n/3t - \delta^2 n/300$.

Note that after the fixing of $(SR_i = sr_i)_{i \in \{1,\cdots,g-1\}}$, $(R_i)_{i \in \{1,\cdots,g-1\}}$ is also fixed, and now $(U_i)_{i \in \{1,\cdots,g-1\}}$ is a linear function of X. Thus by the same analysis we have that conditioned on any fixing of $(U_i)_{i \in \{1,\cdots,g-1\}}$, X is still an affine source. Moreover, $H(X) \geq 2\delta n/3 - \delta^2 n/300 - \delta^2 n/300 > 3\delta n/5$ and $H(X_g) \geq \delta n/3t - \delta^2 n/300 - \delta^2 n/300 > \delta n/4t$. ■

Now consider X and X_g conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i \in \{1,\cdots,g-1\}}$, we have the following lemma.

Lemma 6.3.19. *With probability $1-2^{-\Omega(n)}$ over the further fixings of $X_g = x_g$, R_g is $2^{-\Omega(n)}$-close to uniform.*

Proof of the lemma. First, note that $H(X_g) \geq \delta n/4t$, thus X_g has entropy rate at least $\delta/4$. Therefore by Theorem 3.5.3 $\mathsf{Zuc}(X_g)$ is $2^{-\Omega(n)}$-close to a somewhere-rate-$(1-\delta/4)$ source. Without loss of generality assume that Y_{g1} has rate $1-\delta/4$. Since X_g is a linear function of X, by Lemma 2.3.22 there exist independent affine sources A_g and B_g such that $X = A_g + B_g$, $X_g(X) = X_g(A_g)$ and $H(A_g) = H(X_g)$. Thus $H(B_g) = H(X) - H(A_g) = H(X) - H(X_g) \geq 3\delta n/5 - \delta n/10 = \delta n/2$. Note that when we divide X into ℓ_2 blocks $X = X'_1 \circ \cdots \circ X'_{\ell_2}$, each X'_j is a linear function of X. Thus $X'_j(X) = X'_j(A_g) + X'_j(B_g)$. Let $A_{gj} = X'_j(A_g)$ and $B_{gj} = X'_j(B_g)$. Since B_g is an affine source, by Lemma 2.3.23 the sum of all $H(B_{gj})$ is at least $H(B_g) \geq \delta n/2$. Thus at least one block must have entropy rate at least $\delta/2$. Let B_{gj} be such a block.

Note that Y_{g1} is a deterministic function of X_g, thus it is also a deterministic function of A_g and is independent of B_{gj}. Note Y_{g1} has rate $1-\delta/4$ and B_{gj} has rate $\delta/2$. Thus by Theorem 3.5.22 we have that with probability $1-2^{-\Omega(n)}$ over the fixings of Y_{g1} (and thus A_g and X_g), $\mathsf{Had}(B_{gj}, Y_{g1})$ is $2^{-\Omega(n)}$-close to uniform. Now note that for a fixed $Y_{g1} = y_{g1}$, the function Had is a linear function. Therefore

$$\mathsf{Had}(X'_j, y_{g1}) = \mathsf{Had}(A_{gj}, y_{g1}) + \mathsf{Had}(B_{gj}, y_{g1}).$$

Note that once A_g (equivalently, X_g) is fixed, $\mathsf{Had}(A_{gj}, y_{g1})$ is a fixed constant. Thus whenever a fixed A_g makes $\mathsf{Had}(B_{gj}, y_{g1})$ uniform, it also makes $\mathsf{Had}(X'_j, y_{g1})$ uniform. Therefore we have that with probability $1-2^{-\Omega(n)}$ over the fixings of A_g (and thus X_g), $\mathsf{Had}(X'_j, Y_{g1})$ is $2^{-\Omega(n)}$-close to uniform. When this happens, we have that SR_g is $2^{-\Omega(n)}$-close to an affine somewhere random source (it is affine since for a fixed $X_g = x_g$, SR_g is a linear function of X). Thus by Theorem 6.3.12 R_g is $2^{-\Omega(n)}$-close to uniform. ■

Now consider X conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i \in \{1,\cdots,g-1\}}$, and the event that the property in Lemma 6.3.19 is satisfied. We have the following lemma.

Lemma 6.3.20. *With probability $1-2^{-\Omega(n)}$ over the further fixings of SR_g (and thus R_g), U_g is uniform.*

Proof of the lemma. First note that X_g is a linear function of X. Thus conditioned on any fixing of $X_g = x_g$, X is still an affine source. Now after this fixing, SR_g is a linear function of X. Thus by Lemma 2.3.22, there exist independent affine sources A'_g and B'_g s.t. $X = A'_g + B'_g$,

$SR_g(X) = SR_g(A'_g)$ and $H(SR_g) = H(A'_g)$. Thus $H(B'_g) = H(X) - H(A'_g) = H(X) - H(SR_g) \geq 3\delta n/5 - \delta n/10 - \delta^3 n/3000 > \delta n/3$.

Next, note R_g is a deterministic function of SR_g and is $2^{-\Omega(n)}$-close to uniform by Lemma 6.3.19. Thus R_g is independent of B'_g. Note LSExt is a strong linear seeded extractor with error $2^{-\Omega(n)}$ by Theorem 6.3.4. Thus by Proposition 3.1.9 with probability $1 - 2^{-\Omega(n)}$ over the fixings of R_g (and thus SR_g), $\mathsf{LSExt}(B'_g, R_g)$ is uniform.

Finally note that for any fixing of SR_g (and thus R_g), $\mathsf{LSExt}(X, R_g)$ is a linear function of X. Thus by the same analysis as in Lemma 6.3.19 we have that with probability $1 - 2^{-\Omega(n)}$ over the fixings of SR_g (and thus R_g), U_g is uniform. ■

Now consider X conditioned on any fixing of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i \in \{1,\cdots,g-1\}}$, and the event that both properties in Lemma 6.3.19 and Lemma 6.3.20 hold. Note that even further conditioned on the fixings of $X_g = x_g$ and $SR_g = sr_g$, X is still an affine source. Now after this fixing U_g is a linear function of X. Thus by Lemma 2.3.22 there exist independent affine sources A''_g and B''_g s.t. $X = A''_g + B''_g$, $U_g(X) = U_g(A''_g)$ and $H(U_g) = H(A''_g)$. To prove the function ADisp is a disperser, it suffices to prove that for some $B''_g = b$, it is a disperser. We actually can prove that for any $B''_g = b$, ADisp is a disperser.

Lemma 6.3.21. *For any integer $m > 0$, let $Z' = (Z'_1, \cdots, Z'_m)$ where $Z'_j = \bigoplus_{i=g}^{t} V_{ij}$. Then conditioned on any fixing of $B''_g = b$, $|\mathsf{Supp}(Z')| = 2^m$.*

Proof of the lemma. We first show that Z'_1 can take both values in $\{0, 1\}$. To see this, notice that $Z'_1 = \bigoplus_{i=g}^{t} V_{i1}$ while $V_{g1} = \Pi_{\ell=1}^{c_g} U_{g\ell}$ is a polynomial of degree c_g over the bits of the uniform string U_g. Next, since now (conditioned on all the fixings) U_g is a linear function of X, by Lemma 2.3.22 there exists an affine function L_g s.t. $A''_g = L_g(U_g)$. Thus $X = A''_g + B''_g = L_g(U_g) + B''_g$. Note $U_g = U_g(X) = U_g(A''_g)$ is independent of B''_g. Now conditioned on any fixing of $B''_g = b$, $X = L_g(U_g) + b$ is an affine function (degree 1 polynomial) of U_g.

Given this observation, the following computations and the outputs $V_{(g+1)1}, \cdots, V_{t1}$ are all functions of U_g. If we can show that $\bigoplus_{i=g+1}^{t} V_{i1}$ is a polynomial of degree less than c_g over the bits of U_g, then we know that $Z'_1 = V_{g1} \oplus \bigoplus_{i=g+1}^{t} V_{i1}$ cannot be a constant. In fact, this is exactly how we choose the constants c_i's.

Let us see what conditions c_i's must satisfy. First, it's easy to see that we need to choose c_i's s.t. $c_i > c_{i+1}$ for every i. Next, we compute the degrees of each V_{i1} for $i > g$. First X is an affine function of U_g. Then by Theorem 3.5.3, each bit of $\mathsf{Zuc}(X_i)$ is a constant degree polynomial of the input bits. It's easy to see the function Had is a degree 2 polynomial. Thus each bit of SR_i is a degree 2 polynomial of the input bits. By Theorem 6.3.12 each bit of R_i is a constant degree polynomial of the input bits. By Theorem 6.3.4 each bit of U_i is a constant degree polynomial of the input bits. Thus we conclude that for every $i \geq g+1$, each bit of U_i is a constant $c(\delta)$-degree polynomial of the bits of U_g. Note each V_{i1} is a degree c_i polynomial of the bits of U_i. Therefore in order to make $\bigoplus_{i=g+1}^{t} V_{i1}$ a polynomial of degree less than c_g, it suffices to take

$$c_i > c(\delta)c_{i+1}$$

for every i. This ensures that Z_1' is not a constant.

Now we consider outputting $m > 1$ bits. By induction it suffices to prove that for any fixing of $(Z_1' = z_1, \cdots, Z_i' = z_i)$, Z_{i+1}' can take both values in $\{0,1\}$. For the sake of contradiction, suppose for some $(Z_1' = z_1, \cdots, Z_i' = z_i)$, Z_{i+1}' can only take the value z_{i+1}. Then we have

$$\begin{aligned} P_g =& (Z_1' + z_1 + 1)(Z_2' + z_2 + 1) \\ & \cdots (Z_i' + z_i + 1)(Z_{i+1}' + z_{i+1}) \equiv 0. \end{aligned}$$

However, note that $Z_j' = V_{gj} \oplus \bigoplus_{i=g+1}^{t} V_{ij}$ and V_{gj} is a monomial of degree c_g of the bits of U_g, while all the other V_{ij}'s are monomials of degree less than c_g of the bits of U_g. Also note that V_{gj}'s are monomials of different bits for different j's. Thus P_g is a polynomial of the bits of U_g and has one monomial of degree $(i+1)c_g$ (the highest degree monomial) with coefficient 1. Therefore P_g cannot always be 0. ■

Note that once $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i \in \{1,\cdots,g-1\}}$ are fixed, $(V_{ij})_{i \in \{1,\cdots,g-1\}}$ are also fixed. The theorem now follows immediately from Lemma 6.3.18, Lemma 6.3.19, Lemma 6.3.20 and Lemma 6.3.21. Since $t = O(1)$, all c_i's are constants. Thus the disperser outputs $\Omega(n)$ bits. ■

6.3.4 Analysis of the Affine Extractor

In this section we analyze our affine extractor. We first show how to extract 1 bit that is $2^{-\Omega(n)}$-close to uniform. For this we need the following definition.

Definition 6.3.22. For two functions $f, p : \{0,1\}^n \to \{0,1\}$, their correlation is defined as

$$\mathsf{Cor}(f,p) = \left| \Pr_x[f(x) = p(x)] - \Pr_x[f(x) \neq p(x)] \right|,$$

where the probability is over the uniform distribution. For a class C of functions, we denote by $\mathsf{Cor}(f, C)$ the maximum of $\mathsf{Cor}(f, p)$ over all functions $p \in C$.

Theorem 6.3.23. *[VW08, BKS+10] Let P_d stand for the class of all polynomials of degree at most d over* GF(2)*. Let $f : \{0,1\}^n \to \{0,1\}$ be a function such that $\mathsf{Cor}(f, P_d) \leq 1 - 1/2^d$ and $f^{\times m}$ be the XOR of the value of f on m independent inputs. Then*

$$\mathsf{Cor}(f^{\times m}, P_d) \leq \exp(-\Omega(m/(4^d \cdot d))).$$

Theorem 6.3.24. *Let $Z = (Z_1, \cdots, Z_m)$ be the output of the affine disperser in Algorithm 6.3.14, where $m = \Omega(n)$ and Z_i is the i'th bit. Take any integer $0 < s < m$ and take any subset $S \subseteq [m]$ with $|S| = s$. Let $W = \bigoplus Z_{i,i\in S}$. Then*

$$|W - U| = 2^{-\Omega(s)}.$$

Remark 6.3.25. Note that if we take $s = \Omega(n)$ then we get 1 bit that is $2^{-\Omega(n)}$-close to uniform.

Proof. Without loss of generality assume $S = \{1, \cdots, s\}$. As in the proof of Theorem 6.3.17, we have Lemma 6.3.18, Lemma 6.3.19 and Lemma 6.3.20. Now consider $(U, V)_g, \cdots, (U, V)_t$ conditioned on the fixings of $(X_i = x_i, SR_i = sr_i, R_i = r_i, U_i = u_i)_{i\in\{1,\cdots,g-1\}}$ and $X_g = x_g, SR_g = sr_g$. Let $W_g = \bigoplus_{i=1}^{s} Z_i'$, where $Z_i' = \bigoplus_{j=g}^{t} V_{ji}$. Note that

$$\begin{aligned} W_g = \bigoplus_{i=1}^{s} Z_i' &= \bigoplus_{i=1}^{s} (V_{gi} \oplus \bigoplus_{j=g+1}^{t} V_{ji}) \\ &= \bigoplus_{i=1}^{s} V_{gi} \oplus \bigoplus_{i=1}^{s} (\bigoplus_{j=g+1}^{t} V_{ji}). \end{aligned}$$

We know that U_g is uniform and each V_{gi} is a degree c_g monomial on c_g bits of U_g. We also know that $P_g = \bigoplus_{i=1}^{s}(\bigoplus_{j=g+1}^{t} V_{ji})$ is a degree at most $d = c_g - 1$ polynomial of the bits of U_g. Let P_d stand for the class of all polynomials of degree at most $d = c_g - 1$ over GF(2). Any polynomial in P_d can equal V_{gi} with probability at most $1 - 2^{-c_g} = 1 - 2^{-(d+1)}$, i.e.,

$$\mathsf{Cor}(V_{gi}, P_d) \leq 1 - 1/2^d.$$

Note that V_{gi}'s are the same functions (degree c_g monomial) on different and thus independent bits of U_g (since U_g is uniform). Thus by Theorem 6.3.23,

$$\mathsf{Cor}(\bigoplus_{i=1}^{s} V_{gi}, P_d) \leq 2^{-\Omega(s)}.$$

In particular, we have

$$\mathsf{Cor}(\bigoplus_{i=1}^{s} V_{gi}, \bigoplus_{i=1}^{s}(\bigoplus_{j=g+1}^{t} V_{ji})) \leq 2^{-\Omega(s)}.$$

Thus W_g is $2^{-\Omega(s)}$-close to uniform. Now by Lemma 6.3.18, Lemma 6.3.19 and Lemma 6.3.20, the error of $W = \bigoplus_{i=1}^{s} Z_i$ can go up by at most $2^{-\Omega(n)}$. Thus

$$|W - U| = 2^{-\Omega(n)} + 2^{-\Omega(s)} = 2^{-\Omega(s)}$$

since $s = O(n)$. ■

Now we can prove the main theorem about our affine extractor. We first restate the theorem.

Theorem 6.3.26. *For every $\delta > 0$ there exists an efficient family of functions $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ such that $m = \Omega(n)$ and for every affine source X with entropy δn, $|\mathsf{AExt}(X) - U_m| = 2^{-\Omega(n)}$.*

Proof. We show that Algorithm 6.3.16 is such a family of functions. First note that for any nonempty set $T \subseteq [\alpha m_1]$, the sum of the codewords $G_{i,i\in T}$ is also a codeword C_T. Let W_T be the bit associated with C_T, i.e., W_T is the XOR of the Z_j's whenever the j'th index of the codeword C_T is 1. Observe that

$$W_T = \bigoplus W_{i,i\in T}.$$

Since T is nonempty, the codeword C_T must have distance at least γm_1 from the codeword 0, for some constant $0 < \gamma < 1$. That is, C_T must have at least γm_1 1's. Thus by Theorem 6.3.24 $|W_T - U| = 2^{-\Omega(\gamma m_1)}$. This implies that for every nonempty subset T,

$$\left|\bigoplus W_{i,i\in T} - U\right| \leq 2^{-c_0 \gamma m_1}$$

for some constant $c_0 > 0$. In other words, the random variables $\{W_i\}$ form an ϵ-biased space for $\epsilon = 2^{-c_0 \gamma m_1}$. Thus by Lemma 3.5.24

$$|W - U| \leq 2^{\beta m_1/2} \cdot 2^{-c_0 \gamma m_1}.$$

Choose $0 < \beta \leq \alpha$ s.t. $\beta \leq c_0 \gamma$. Then

$$|W - U| \leq 2^{-c_0 \gamma m_1/2}.$$

Thus we have that $(W_1, \cdots, W_{\beta m_1})$ are $\Omega(n)$ bits that are $2^{-\Omega(n)}$-close to uniform. ■

6.3.5 Affine Extractors and Dispersers for Sub-Linear Entropy Sources

In this section we briefly show how we can modify the affine extractors and dispersers above to handle sources with slightly sub-linear entropy. The main observation is that in the construction of affine extractors for linear entropy, the polynomials that we use only have constant degrees. For an argument like the analysis of the extractor to hold, the degree of the polynomial can be close to $\log n$ by Theorem 6.3.23. For an argument like the analysis of the disperser to hold, the degree of the polynomial can be close to n (the degree cannot be larger than n since we can get at most n uniform random bits). We'll show that this will lead to an affine extractor for entropy $n/\sqrt{\log \log n}$ and an affine disperser for entropy $n/\sqrt{\log n}$.

Theorem 6.3.27. *There exists a constant $c > 1$ and an efficient family of functions* $\mathsf{ADisp} : \{0,1\}^n \to \{0,1\}^m$ *such that $m = n^{\Omega(1)}$ and for every affine source X with entropy $cn/\sqrt{\log n}$,* $|\mathsf{Supp}(\mathsf{ADisp}(X))| = 2^m$.

Proof Sketch. We essentially use the same algorithm as Algorithm 6.3.14, except the entropy rate δ now is sub-constant. We examine the analysis to see how small δ can be. We focus on the first good block X_g.

First we want to use the somewhere condenser from Theorem 3.5.3 to convert a rate-$\delta/4$ source into a somewhere rate-$(1-\delta/4)$ source. Note that now δ is sub-constant, so we do this in two steps. First, we repeatedly use Theorem 3.5.2 to convert the source into a somewhere rate-0.6 source. This will take $O(\log \frac{1}{\delta})$ times. Next, we repeatedly use Theorem 3.5.6 to convert the source into a somewhere rate-$(1-\delta/4)$ source. This will take $O(\frac{1}{\delta})$ times. Thus in step 1 of Algorithm 6.3.14 we get that $\ell_1 = 2^{O(\frac{1}{\delta})}$, and each Y_{gj} has $n/(2^{O(\frac{1}{\delta})})$ bits. The error is $2^{-n/2^{O(\frac{1}{\delta})}}$.

Now in step 2 of Algorithm 6.3.14, we get $\ell_2 = 2^{O(\frac{1}{\delta})}$. Thus the total number of rows in the matrix SR_g is $\ell_1\ell_2 = 2^{O(\frac{1}{\delta})}$, with each row having $\delta^3 n/(3000\ell_1\ell_2) = n/(2^{O(\frac{1}{\delta})})$ bits. By Theorem 3.5.22 the error is $2^{-n/2^{O(\frac{1}{\delta})}}$.

In step 3, we apply $\mathsf{AffineSRExt}$. By Theorem 6.3.12 we get each R_g has $n/(2^{O(\frac{1}{\delta^2})})$ bits, with error $2^{-n/2^{O(\frac{1}{\delta^2})}}$. By Theorem 6.3.4 after applying LSExt, U_g has $n/(2^{O(\frac{1}{\delta^2})})$ bits with error $2^{-n/2^{O(\frac{1}{\delta^2})}}$.

The last thing to verify is that the degrees of the polynomials produced in step 5 satisfy the requirements as in the analysis of Theorem 6.3.17. The analysis says that we need to have $c_i > c(\delta)c_{i+1}$ for all i. To see how this can be satisfied, we first estimate the quantity $c(\delta)$.

As in the analysis of Theorem 6.3.17, first X is an affine function of U_g. Now by Theorem 3.5.2 and Theorem 3.5.6, each bit of $\mathsf{Zuc}(X_i)$ is a degree $2^{O(\frac{1}{\delta})}$ polynomial of the input bits (since we repeat the condenser $O(\frac{1}{\delta})$ times). The function Had is a degree 2 polynomial. Thus each bit of SR_i is a degree 2 polynomial of the input bits. Now we apply $\mathsf{AffineSRExt}$, and by Theorem 6.3.12 each bit of the output is a degree $2^{O(\frac{1}{\delta})}$ polynomial of the input bits. Therefore each bit of R_i is a degree $2^{O(\frac{1}{\delta})} \cdot 2^{O(\frac{1}{\delta})} = 2^{O(\frac{1}{\delta})}$ polynomial of the bits of U_g. By Theorem 6.3.4 in LSExt each bit of U_i is a constant degree polynomial of the input bits. Thus we conclude that for every $i \geq g+1$, each bit of U_i is a degree $2^{O(\frac{1}{\delta})}$ polynomial of the bits of U_g.

Therefore we have

$$c(\delta) = 2^{O(\frac{1}{\delta})}.$$

Note that we need $c_i > c(\delta)c_{i+1}$ for every $i, 1 \leq i \leq 10/\delta$. Thus the c_i's are bounded by

$$c(\delta)^{10/\delta} = 2^{O(\frac{1}{\delta^2})}.$$

Since each U_i has $n/(2^{O(\frac{1}{\delta^2})})$ bits, it suffices to have

$$n/(2^{O(\frac{1}{\delta^2})}) > 2^{O(\frac{1}{\delta^2})}.$$

Thus by taking $\delta \geq c/\sqrt{\log n}$ for some constant $c > 1$ the disperser can output $n^{\Omega(1)}$ bits. ■

Similarly, we get an affine extractor for sub-linear entropy sources. However, unlike in the analysis of the affine disperser, the degree of the polynomial cannot be close to n, and can only be close to $\log n$ by Theorem 6.3.23. Thus we only get $\delta = c/\sqrt{\log\log n}$ for some constant $c > 1$.

Theorem 6.3.28. *There exists a constant $c > 1$ and an efficient family of functions* $\mathsf{AExt} : \{0,1\}^n \to \{0,1\}^m$ *such that* $m = n^{\Omega(1)}$ *and for every affine source* X *with entropy* $cn/\sqrt{\log\log n}$*,* $|\mathsf{AExt}(X) - U_m| = 2^{-n^{\Omega(1)}}$.

Proof Sketch. We essentially use the same algorithm as Algorithm 6.3.16, except the entropy rate δ now is sub-constant. We examine the analysis to see how small δ can be. We focus on the first good block X_g.

As in the analysis of the affine disperser above, we get that in step 4 of Algorithm 6.3.14, U_g has $n/(2^{O(\frac{1}{\delta^2})})$ bits, and the error is $2^{-n/2^{O(\frac{1}{\delta^2})}}$. We also get that

$$c(\delta) = 2^{O(\frac{1}{\delta})}$$

and thus the c_i's are bounded by

$$c(\delta)^{10/\delta} = 2^{O(\frac{1}{\delta^2})}.$$

For the xor lemma of Theorem 6.3.23 to give a non-trivial bound, it suffices to have

$$2^{O(\frac{1}{\delta^2})} \leq \alpha \log n$$

for some constant $0 < \alpha < 1$. This gives that $\delta \geq c/\sqrt{\log\log n}$ for some constant $c > 1$. Also, when this happens, the XOR lemma of Theorem 6.3.23 gives a correlation upper bound of $2^{-n^{\Omega(1)}}$. The error of U_g is $2^{-\Omega(n/\log n)}$. Thus by taking the generating matrix of a binary linear asymptotically good code and choosing $n^{\Omega(1)}$ rows, we see that the extractor outputs $n^{\Omega(1)}$ bits that are $2^{-n^{\Omega(1)}}$-close to uniform. ■

6.4 Non-malleable Extractors

In this section we give the first explicit construction of non-malleable extractors. We begin with the formal definition.

Definition 6.4.1. A function $\mathsf{nmExt} : [N] \times [D] \to [M]$ is a (k, ε)-non-malleable extractor if, for any source X with $H_\infty(X) \geq k$ and any function $\mathcal{A} : [D] \to [D]$ such that $\mathcal{A}(y) \neq y$ for all y, the following holds. When Y is chosen uniformly from $[D]$ and independent of X,

$$(\mathsf{nmExt}(X,Y), \mathsf{nmExt}(X,\mathcal{A}(Y)), Y) \approx_\varepsilon (U_{[M]}, \mathsf{nmExt}(X,\mathcal{A}(Y)), Y).$$

Note that this dramatically strengthens the definition of strong extractor. In a strong extractor, the output must be indistinguishable from uniform, even given the random seed. For a non-malleable extractor, a distinguisher is not only given a random seed, but also the output of the extractor with the given input and an arbitrarily correlated random seed. Note that $\mathsf{nmExt}(X, \mathcal{A}(Y))$ need not be close to uniform.

This kind of extractors is first proposed by Dodis and Wichs [DW09] to construct protocols for privacy amplification. They showed that non-malleable extractors exist with $k > 2m + 3\log(1/\varepsilon) + \log d + 9$ and $d > \log(n - k + 1) + 2\log(1/\varepsilon) + 7$, for $N = 2^n$, $M = 2^m$, and $D = 2^d$. However, they were not able to construct such non-malleable extractors.

Here we construct the first explicit non-malleable extractors.

6.4.1 The Construction of Non-malleable Extractors

We show that a specific near-Hadamard code that comes from the Paley graph works as a non-malleable extractor for min-entropy $k > n/2$. The Paley graph function is $\mathsf{nmExt}(x, y) = \chi(x - y)$, where x and y are viewed as elements in a finite field $\mathbb{F}$ of odd order q and χ is the quadratic character $\chi(x) = x^{(q-1)/2}$. (The output of χ is in $\{\pm 1\}$, which we convert to an element of $\{0, 1\}$.) The function $\mathsf{nmExt}(x, y) = \chi(x + y)$ works equally well. The proof involves estimating a nontrivial character sum.

We can output m bits by computing the discrete logarithm $\log_g(x + y) \mod M$. This extractor was originally introduced by Chor and Goldreich [CG88] in the context of two-source extractors. To make this efficient, we need M to divide $q - 1$. A widely-believed conjecture about primes in arithmetic progressions implies that such a q is not too large (see Conjecture 3.2.2). Our result is stated as follows.

Theorem 6.4.2. *For any constants $\alpha, \beta, \gamma > 0$ with $\beta + \gamma < \alpha/2$, there is an explicit $(k = (1/2 + \alpha)n, \varepsilon)$-non-malleable extractor $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ for $\varepsilon = 2^{-\gamma n}$ and any $m \leq \beta n$. It runs in polynomial time if Conjecture 3.2.2 holds or $m = O(\log n)$.*

Our basic extractor was introduced by Chor and Goldreich [CG88]. They showed that it was a two-source extractor for entropy rates bigger than $1/2$. Dodis and Oliveira [DO03] showed that it was strong. Neither result implies anything about non-malleability.

To output m bits, we set $M = 2^m$ and choose a prime power $q > M$. In our basic extractor, we require that $M|(q - 1)$. Later, we remove this assumption. Fix a generator g of $\mathbb{F}_q^\times$. We define

$\mathsf{nmExt} : \mathbb{F}_q^2 \to \mathbb{Z}_M$ by $\mathsf{nmExt}(x, y) = h(\log_g(x + y))$. Here $\log_g z$ is the discrete logarithm of z with respect to g, and $h : \mathbb{Z}_{q-1} \to \mathbb{Z}_M$ is given by $h(x) = x \mod M$.

In the special case $m = 1$, we only require that q is odd. In this case, $\mathsf{nmExt}(x, y)$ corresponds to the quadratic character of $x + y$, converted to $\{0, 1\}$ output. This is efficient to compute. Since there is no known efficient deterministic algorithm to find an n-bit prime, we may take $q = 3^\ell$, with $3^{\ell-1} < 2^n < 3^\ell$.

For general M, we use the Pohlig-Hellman algorithm to compute the discrete log mod M. This runs in polynomial time in the largest prime factor of M. Since in our case $M = 2^m$, this is polynomial time.

We still need a prime or prime power q such that $M|(q - 1)$. Unconditionally, we get a polynomial-time algorithm to output $m = c \log n$ bits for any $c > 0$. To output more bits efficiently, we rely on a widely believed conjecture. Under Conjecture 3.2.2, such a prime can be found efficiently by testing $M + 1, 2M + 1, 3M + 1, \ldots$ in succession.

Now we prove that nmExt is a non-malleable extractor.

Theorem 6.4.3. *The above function* $\mathsf{nmExt} : \mathbb{F}_q^2 \to \mathbb{Z}_M$ *is a* (k, ε)*-non-malleable extractor for* $\varepsilon = Mq^{1/4}2^{1-k/2}$.

Proof. The heart of our proof is a new character sum estimate, given in Theorem 6.4.8. We now show how to deduce Theorem 6.4.3 from the character sum estimate and Lemma 3.4.1. Let X be a distribution with $H_\infty(X) \geq k$, and let Y be uniform on $\mathbb{F}_q$. As is well-known, we may assume without loss of generality that X is uniform on a set of size 2^k. We set $G = \mathbb{Z}_M$, $(W, W') = (\mathsf{nmExt}(X, Y), \mathsf{nmExt}(X, \mathcal{A}(Y)))$, and we condition on $Y = y$.

Note that for ϕ a character of G, the function $\chi(z) = \phi(h(\log_g(z)))$ is a multiplicative character of $\mathbb{F}_q$. Therefore, Theorem 6.4.8 shows that $((W, W')|Y = y)$ satisfies the hypotheses of Lemma 3.4.1 for some α_y, where $\mathrm{E}_{y \leftarrow Y}[\alpha_y] \leq \alpha$ for $\alpha < q^{1/4}2^{1-k/2}$. Thus, by Lemma **??**, $((W, W')|Y = y)$ is $O(M\alpha_y)$-close to $((U, h(W'))|Y = y)$ for every y. Since this expression is linear in α_y, we conclude that (W, W', Y) is $O(M\alpha)$-close to $(U, h(W'), Y)$, as required. $\square$

Note that this theorem assumes that the seed is chosen uniformly from $\mathbb{F}_q$. However, this may not be the case, as we are given the input length n and min-entropy k, and we need to find a suitable $m < k/2 - n/4 - 2$ such that $2^m|(q - 1)$. Thus, it could be that q is not close to a power of 2. Instead, we may only assume that the seed has min-entropy at least $\log q - 1$. We can handle this, and in fact much lower min-entropy in the seed, as follows. First, we define a non-malleable extractor with a weakly-random seed.

Definition 6.4.4. A function $\mathsf{nmExt} : [N] \times [D] \to [M]$ is a (k, k', ε)-non-malleable extractor if, for any source X with $H_\infty(X) \geq k$, any seed Y with $H_\infty(Y) \geq k'$, and any function $\mathcal{A} : [D] \to [D]$ such that $\mathcal{A}(y) \neq y$ for all y, the following holds:

$$(\mathsf{nmExt}(X, Y), \mathsf{nmExt}(X, \mathcal{A}(Y)), Y) \approx_\varepsilon (U_{[M]}, \mathsf{nmExt}(X, \mathcal{A}(Y)), Y).$$

The following lemma shows that a non-malleable extractor with small error remains a non-malleable extractor even if the seed is weakly random.

Lemma 6.4.5. *A (k, ε)-non-malleable extractor $\mathsf{nmExt} : [N] \times [D] \to [M]$ is also a (k, k', ε')-non-malleable extractor with $\varepsilon' = (D/2^{k'})\varepsilon$.*

Proof. For $y \in [D]$, let $\varepsilon_y = \Delta((\mathsf{nmExt}(X, y), \mathsf{nmExt}(X, \mathcal{A}(y)), y), (U_{[M]}, \mathsf{nmExt}(X, \mathcal{A}(y)), y))$. Then for Y chosen uniformly from $[D]$,

$$\varepsilon \geq \Delta((\mathsf{nmExt}(X, Y), \mathsf{nmExt}(X, \mathcal{A}(Y)), Y), (U_{[M]}, \mathsf{nmExt}(X, \mathcal{A}(Y)), Y)) = \frac{1}{D} \sum_{y \in [D]} \varepsilon_y.$$

Thus, for Y' with $H_\infty(Y') \geq k'$, we get

$$\begin{aligned} &\Delta((\mathsf{nmExt}(X, Y'), \mathsf{nmExt}(X, \mathcal{A}(Y')), Y'), (U_{[M]}, \mathsf{nmExt}(X, \mathcal{A}(Y')), Y')) \\ &\qquad = \sum_{y \in [D]} \Pr[Y = y]\varepsilon_y \leq 2^{-k'} \sum_{y \in [D]} \varepsilon_y \leq (D/2^{k'})\varepsilon. \end{aligned}$$

□

It is now simple to analyze our non-malleable extractor as a function $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$. Here we work over $\mathbb{F}_q$, where q is the smallest prime (or prime power) congruent to 1 modulo $M = 2^m$. We let $d = \lfloor \log_2 q \rfloor$, which is $n + c \log n + O(1)$ under Conjecture 3.2.2. We could even let $d = n$ and the error would only grow by n^c.

Theorem 6.4.6. *Under Conjecture 3.2.2 with constant c, for any n, $k > n/2 + (c/2)\log n$, and $m < k/2 - n/4 - (c/4)\log n$, the above function $\mathsf{nmExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a polynomial-time computable, (k, ε)-non-malleable extractor for $\varepsilon = O(n^{c/4} 2^{m+n/4-k/2})$.*

Proof. Suppose that Conjecture 3.2.2 holds for the constant c. Then $q = O(n^c 2^n)$, and the seed has min-entropy $k' = d$. Applying Lemma 6.4.5, we obtain error

$$\varepsilon = (q/2^d) M q^{1/4} 2^{1-k/2} = O(n^{c/4} 2^{m+n/4-k/2}).$$

□

6.4.2 A Character Sum Estimate

We now prove the necessary character sum estimate. We prove a somewhat more general statement than is needed for the one-bit extractor, as the general statement is needed to output many bits. Throughout this section, we take $\mathbb{F} = \mathbb{F}_q$ to be a finite field with q elements. In addition, we suppose that $\chi : \mathbb{F}^\times \to \mathbb{C}^\times$ is a nontrivial character of order $d = q-1$, and we extend the domain of χ to $\mathbb{F}$ by taking $\chi(0) = 0$. Now we consider two arbitrary characters, where the first is nontrivial; without loss of generality we may take these to be $\chi_a(x) = (\chi(x))^a$ and $\chi_b(x) = (\chi(x))^b$, where $0 < a < q-1$ and $0 \leq b < q-1$. The following lemma is a consequence of Weil's resolution of the Riemann Hypothesis for curves over finite fields (see [Wei48]).

Lemma 6.4.7. *Suppose that $f \in \mathbb{F}[x]$ is a polynomial having m distinct roots which is not a dth power in $\mathbb{F}[x]$. Then*

$$\Big|\sum_{x\in\mathbb{F}} \chi(f(x))\Big| \leqslant (m-1)\sqrt{q}.$$

Proof. This is immediate from Theorem 2C′ of Schmidt [Sch76] (see page 43 of the latter source). □

Now we establish the main character sum estimate. Note that we need the assumption that $a \neq 0$: if $a = 0$ and $b = (q-1)/2$, we could take $\mathcal{A}(y) = 0$ and let $\mathcal{S}$ be the set of quadratic residues, and then one has no cancellation in the character sum.

Theorem 6.4.8. *Suppose that $\mathcal{S}$ is a non-empty subset of $\mathbb{F}$, and that $\mathcal{A} : \mathbb{F} \to \mathbb{F}$ is any function satisfying the property that $\mathcal{A}(y) \neq y$ for all $y \in \mathbb{F}$. Then one has*

$$\sum_{y\in\mathbb{F}}\Big|\sum_{s\in\mathcal{S}} \chi_a(s+y)\chi_b(s+\mathcal{A}(y))\Big| \leqslant 11^{1/4} q^{5/4} |\mathcal{S}|^{1/2}.$$

Proof. Write $\Theta = \sum_{y\in\mathbb{F}}\Big|\sum_{s\in\mathcal{S}} \chi_a(s+y)\chi_b(s+\mathcal{A}(y))\Big|$. We begin by applying Cauchy's inequality to obtain

$$\Theta^2 \leqslant q\sum_{y\in\mathbb{F}}\Big|\sum_{s\in\mathcal{S}} \chi_a(s+y)\chi_b(s+\mathcal{A}(y))\Big|^2 = q\sum_{s,t\in\mathcal{S}}\sum_{y\in\mathbb{F}} \psi_{s,t}(y),$$

in which we have written

$$\psi_{s,t}(y) = \chi_a(s+y)\chi_b(s+\mathcal{A}(y))\overline{\chi}_a(t+y)\overline{\chi}_b(t+\mathcal{A}(y)). \tag{6.5}$$

Applying Cauchy's inequality a second time, we deduce that

$$\Theta^4 \leqslant q^2|\mathcal{S}|^2 \sum_{s,t\in\mathcal{S}} \Big|\sum_{y\in\mathbb{F}} \psi_{s,t}(y)\Big|^2.$$

By positivity, the sum over s and t may be extended from $\mathcal{S}$ to the entire set $\mathbb{F}$, and thus we deduce that

$$\Theta^4 \leqslant q^2|\mathcal{S}|^2 \sum_{s,t\in\mathbb{F}} \sum_{y,z\in\mathbb{F}} \psi_{s,t}(y)\overline{\psi}_{s,t}(z). \tag{6.6}$$

On recalling the definition (6.5), we may expand the right hand side of (6.6) to obtain the bound

$$\Theta^4 \leqslant q^2|\mathcal{S}|^2 \sum_{y,z\in\mathbb{F}} |\nu(y,z)|^2, \tag{6.7}$$

where

$$\nu(y,z) = \sum_{s\in\mathbb{F}} \chi_a(s+y)\chi_b(s+\mathcal{A}(y))\overline{\chi}_a(s+z)\overline{\chi}_b(s+\mathcal{A}(z)).$$

Recall now the hypothesis that $y \neq \mathcal{A}(y)$. It follows that, considered as an element of $\mathbb{F}[x]$, the polynomial

$$h_{y,z}(x) = (x+y)^a(x+\mathcal{A}(y))^b(x+z)^{q-1-a}(x+\mathcal{A}(z))^{q-1-b}$$

can be a dth power only when $y = z$, or when $y = \mathcal{A}(z)$, $a = b$ and $z = \mathcal{A}(y)$. In order to confirm this assertion, observe first that when $y \neq z$ and $y \neq \mathcal{A}(z)$, then $h_{y,z}$ has a zero of multiplicity a at $-y$. Next, when $y = \mathcal{A}(z)$, one has $z \neq y$, and so when $a \neq b$ the polynomial $h_{y,z}$ has a zero of multiplicity $q-1+a-b$ at $-y$. Finally, when $y = \mathcal{A}(z)$ and $a = b$, then provided that $z \neq \mathcal{A}(y)$ one finds that $h_{y,z}$ has a zero of multiplicity $q-1-a$ at $-z$. In all of these situations it follows that $h_{y,z}$ has a zero of multiplicity not divisible by $d = q-1$. When $y \neq z$, and $(y,z) \neq (\mathcal{A}(z), \mathcal{A}(y))$, therefore, the polynomial $h_{y,z}(x)$ is not a dth power in $\mathbb{F}[x]$, and has at most 4 distinct roots. In such a situation, it therefore follows from Lemma 6.4.7 that

$$\nu(y,z) = \sum_{s\in\mathbb{F}} \chi(h_{y,z}(s))$$

is bounded in absolute value by $3\sqrt{q}$. Meanwhile, irrespective of the values of y and z, the expression $\nu(y,z)$ is trivially bounded in absolute value by q. Substituting these estimates into (6.7), we arrive at the upper bound

$$\begin{aligned}\Theta^4 &\leqslant q^2|\mathcal{S}|^2 \sum_{y\in\mathbb{F}} \Big(|\nu(y,y)|^2 + |\nu(y,\mathcal{A}(y))|^2 + \sum_{z\in\mathbb{F}\setminus\{y,\mathcal{A}(y)\}} |\nu(y,z)|^2\Big) \\ &\leqslant q^2|\mathcal{S}|^2 \sum_{y\in\mathbb{F}} (q^2 + q^2 + q(3\sqrt{q})^2) = 11q^5|\mathcal{S}|^2.\end{aligned}$$

We may thus conclude that $\Theta \leqslant 11^{1/4} q^{5/4} |\mathcal{S}|^{1/2}$. $\square$

A direct computation yields the following corollary.

Corollary 6.4.9. *Let α be a positive number with $\alpha \leqslant 1$. Then under the hypotheses of the statement of Theorem 6.4.8, one has*

$$\sum_{y \in \mathbb{F}} \Big| \sum_{s \in \mathcal{S}} \chi_a(s+y)\chi_b(s+\mathcal{A}(y)) \Big| < \alpha q |\mathcal{S}|$$

whenever $|\mathcal{S}| > \sqrt{11q}/\alpha^2$.

Chapter 7

Privacy Amplification with an Active Adversary

In this chapter we present our results on privacy amplification with an active adversary. As mentioned in the introduction, the basic setting of this problem is that two parties, Alice and Bob, share a weak random source W and they want to obtain private random bits through communications to each other. The communication is over a public channel where an adversary Eve is present. In this thesis, we consider the case where Eve is active and has unlimited computational resources. In other words, Eve can see whatever messages sent in this channel and he can modify the messages in an arbitrary way. The goal is to design a protocol such that despite the presence of Eve, with high probability Alice and Bob still end up with private random bits that are close to uniform.

Below we give the definition of a privacy amplification protocol (P_A, P_B), executed by two parties Alice and Bob sharing a secret $X \in \{0,1\}^n$, in the presence of an active, computationally unbounded adversary Eve, who might have some partial information E about X satisfying $H_\infty(X|E) \geqslant k$. Informally, this means that whenever a party (Alice or Bob) does not reject, the key R output by this party is random and statistically independent of Eve's view. Moreover, if both parties do not reject, they must output the same keys $R_A = R_B$ with overwhelming probability.

More formally, we assume that Eve is in full control of the communication channel between Alice and Bob, and can arbitrarily insert, delete, reorder or modify messages sent by Alice and Bob to each other. In particular, Eve's strategy P_E actually defines two correlated executions (P_A, P_E) and (P_E, P_B) between Alice and Eve, and Eve and Bob, called "left execution" and "right execution", respectively. We stress that the message scheduling for both of these executions is completely under Eve's control, and Eve might attempt to execute a run with one party for several rounds before resuming the execution with another party. However, Alice and Bob are assumed to have fresh, private and independent random tapes Y and W, respectively, which are not known to Eve (who, by virtue of being unbounded, can be assumed deterministic). At the end of the left execution $(P_A(X,Y), P_E(E))$, Alice outputs a key $R_A \in \{0,1\}^m \cup \{\perp\}$, where $\perp$ is a special symbol indicating rejection. Similarly, Bob outputs a key $R_B \in \{0,1\}^m \cup \{\perp\}$ at the end of the right execution $(P_E(E), P_B(X,W))$. We let E' denote the final view of Eve, which includes E and the communication transcripts of both executions $(P_A(X,Y), P_E(E))$ and $(P_E(E), P_B(X,W)$. We can now define the security of (P_A, P_B).

Definition 7.0.10. [KR09b] An interactive protocol (P_A, P_B), executed by Alice and Bob on a communication channel fully controlled by an active adversary Eve, is a (k, m, ϵ)-*privacy amplification protocol* if it satisfies the following properties whenever $H_\infty(X|E) \geq k$:

1. Correctness. If Eve is passive, then $\Pr[R_A = R_B \wedge R_A \neq \perp \wedge R_B \neq \perp] = 1$.
2. Robustness. We start by defining the notion of *pre-application* robustness, which states that even if Eve is active, $\Pr[R_A \neq R_B \wedge R_A \neq \perp \wedge R_B \neq \perp] \leqslant \epsilon$.

   The stronger notion of *post-application* robustness is defined similarly, except Eve is additionally given the key R_A the moment she completed the left execution (P_A, P_E), and the key R_B the moment she completed the right execution (P_E, P_B). For example, if Eve completed the left execution before the right execution, she may try to use R_A to force Bob to output a different key $R_B \notin \{R_A, \perp\}$, and vice versa.
3. Extraction. Given a string $r \in \{0,1\}^m \cup \{\perp\}$, let $\mathsf{purify}(r)$ be $\perp$ if $r = \perp$, and otherwise replace $r \neq \perp$ by a fresh m-bit random string U_m: $\mathsf{purify}(r) \leftarrow U_m$. Letting E' denote Eve's view of the protocol, we require that

   $$\Delta((R_A, E'), (\mathsf{purify}(R_A), E')) \leq \epsilon \quad \text{and} \quad \Delta((R_B, E'), (\mathsf{purify}(R_B), E')) \leq \epsilon$$

   Namely, whenever a party does not reject, its key looks like a fresh random string to Eve.

The quantity $k - m$ is called the *entropy loss* and the quantity $\log(1/\epsilon)$ is called the *security parameter* of the protocol.

Previously, Maurer and Wolf [MW97b] gave a one-round protocol which works when the entropy rate of the weakly-random secret X is bigger than $2/3$. This was later improved by Dodis, Katz, Reyzin, and Smith [DKRS06] to work for entropy rate bigger than $1/2$. However in both cases the resulting nearly-uniform secret key R is significantly shorter than the min-entropy of X. Dodis and Wichs [DW09] showed that there is no one-round protocol for entropy rate less than $1/2$. Renner and Wolf [RW03] gave the first protocol which works for entropy rate below $1/2$. Kanukurthi and Reyzin [KR09b] simplified their protocol and showed that the protocol can run in $O(s)$ rounds and achieve entropy loss $O(s^2)$ to achieve security parameter s. Dodis and Wichs [DW09] improved the number of rounds to 2 but did not improve the entropy loss. Chandran, Kanukurthi, Ostrovsky, and Reyzin [CKOR10] improved the entropy loss to $O(s)$ but the number of rounds remained $O(s)$. On the other hand, it was shown in [DW09] that non-constructively there exists a 2-round protocol with entropy loss $O(s)$. We also note that all the results above assume that each party has access to local uniform random bits.

Here we give two improvements over previous results. First, we show how to construct a privacy amplification protocol even if Alice and Bob only have access to local weak random sources, while all previous results assume that they have access to local uniform random bits. Second, In the case where Alice and Bob have access to local uniform random bits, we give protocols that improve various parameters, such as round complexity and entropy loss.

7.1 Some Previous Results that We Need

For the results in this chapter, we need the following lemma.

Lemma 7.1.1. *Assume we have 3 random variables X_1, Y_1, Y_2 such that $|Y_1 - Y_2| \le \epsilon$. Then there exists a random variable X_2 with the same support of X_1, such that*

$$|(X_1, Y_1) - (X_2, Y_2)| \le \epsilon.$$

Proof. We construct the random variable X_2 and the distribution (X_2, Y_2) as follows. For any y, consider $\Pr[Y_1 = y]$, $\Pr[Y_2 = y]$ and the distribution $(X_1, Y_1 = y)$. Let $\delta = \Pr[Y_1 = y] - \Pr[Y_2 = y]$. If $\delta \ge 0$, then we do the following:

1. Define an arbitrary order on the set of the support of X_1.
2. While $\delta > 0$, pick a new x from the support according to the above order and let $p = \Pr[X_1 = x, Y_1 = y]$.
3. Let $\Pr[X_2 = x, Y_2 = y] = p - min(p, \delta)$.
4. Let $\delta = \delta - min(p, \delta)$.
5. When $\delta = 0$, for all the rest x, let $\Pr[X_2 = x, Y_2 = y] = \Pr[X_1 = x, Y_1 = y]$.

If $\delta < 0$, then we do the following:

1. Pick an arbitrary x from the support of X_1 and let $p = \Pr[X_1 = x, Y_1 = y]$.

2. Let $\Pr[X_2 = x, Y_2 = y] = p - \delta$.

3. For all the other x in the support of X_1, let $\Pr[X_2 = x, Y_2 = y] = \Pr[X_1 = x, Y_1 = y]$.

It is easy to see that the distribution (X_2, Y_2) has marginal distribution Y_2 and $|(X_1, Y_1) - (X_2, Y_2)| = |Y_1 - Y_2| \leq \epsilon$.

$\square$

We also need the definition of an interactive authentication protocol. In such a protocol, Alice takes a message m as input and tries to authenticate the message to Bob over the channel. Bob obtains message m_B at the end of the protocol. We now give the formal definition of such a protocol.

Definition 7.1.2. [KR09a, CKOR10] An interactive protocol (P_A, P_B) played by Alice and Bob on a communication channel fully controlled by an active adversary Eve, is a (k, ℓ)-*interactive authentication protocol* if it satisfies the following properties whenever $H_\infty(W) \geq k$:

1. Correctness. If Eve is passive, then $\Pr[m_B = m] = 1$.

2. Robustness. The probability that the following experiment outputs "Eve wins" is at most $2^{-\ell}$: sample w from W; let v_a, v_b be the communication upon execution of (P_A, P_B) with Eve actively controlling the channel, and let $m_B = P_B(w, v_b, y)$. Output "Eve wins" if $(m_B \neq m \wedge m_B \neq \perp)$.

   Again ℓ is called the *security parameter* of the protocol.

In [KR09a], it is shown an interactive authentication protocol can be used to construct a privacy amplification protocol. Specifically, we have the following theorem.

Theorem 7.1.3 ([KR09a])**.** *Suppose there exists an efficient (k, ℓ) interactive authentication protocol for messages of length $\Theta(\ell + \log n)$, then there exits an efficient $(k, \lambda_k, 2^{-\ell})$ privacy amplification protocol.*

7.2 Privacy Amplification with Local Weak Random Sources

In this section we construct protocols for privacy amplification where Alice and Bob only have access to local weak random sources. We show that

1. Non constructively, we can do as good as if Alice and Bob have access to local random bits. Specifically, if Alice and Bob have two independent (n, k) sources and they share an independent (n, k) source, then there is a (possibly inefficient) protocol that achieves privacy amplification up to security parameter $\Omega(k)$.

2. If Alice and Bob have two independent $(n, (\frac{1}{2} + \delta)n)$ sources and they share an independent (n, k) source, then there is an explicit protocol that achieves privacy amplification up to security parameter $k^{\Omega(1)}$.

3. If Alice and Bob have two independent $(n, \delta n)$ sources and they share an independent (n, k) source, then there is an explicit protocol that achieves privacy amplification up to security parameter $\Omega(\log k)$.

Specifically, we have the following theorems.

Theorem 7.2.1. *For all positive integers n, k where $k > \log(n)$, assume that Alice and Bob have two independent local (n, k) sources, and they share an independent (n, k) source W. Then non-constructively there exists a $(k, k - O(\log n + \log(1/\epsilon)), \epsilon)$ privacy amplification protocol.*

Theorem 7.2.2. *For all positive integers n, k where $k \geq \text{polylog}(n)$ and any constant $0 < \delta < 1$, assume that Alice and Bob have two independent local $(n, (1/2 + \delta)n)$ sources, and they share an independent (n, k) source W. Then there exists an efficient $(k, k - k^{\Omega(1)}, 2^{-k^{\Omega(1)}}, 2^{-k^{\Omega(1)}})$ privacy amplification protocol.*

Theorem 7.2.3. *For all positive integers n, k where $k \geq \text{polylog}(n)$ and any constant $0 < \delta < 1$, assume that Alice and Bob have two independent local $(n, \delta n)$ sources, and they share an independent (n, k) source W. Then there exists an efficient $(k, k - k^{\Omega(1)}, 1/\text{poly}(k), 1/\text{poly}(k))$ privacy amplification protocol.*

Our results are the first results to give protocols that achieve privacy amplification when Alice and Bob only have access to local weak random sources.

7.2.1 Overview of the Constructions

Here what we do is to try to reduce the case to where Alice and Bob have access to local private random bits. In other words, we want to design a protocol such that at the end of the protocol, Alice and Bob end up with nearly private and random bits, while their shared secret W still has a lot of entropy left. Non-constructively, this is simple, because non-constructively there exists strong two-source extractors for min-entropy as small as $k > \log n$. If we have a strong two-source extractor, then Alice and Bob just each applies this extractor to his or her own source and W. By the property of the strong two-source extractor, even conditioned on W, their outputs are close to uniform. Moreover, conditioned on W their outputs are deterministic functions of their own sources, and are thus independent. Eve also knows nothing about their outputs since all computations are private. Thus we are done. In another case where Alice and Bob each has an independent source with entropy $k = (1/2 + \delta)n$, a construction of Raz [Raz05] serves as a strong two source extractor. Thus in this case we have an explicit protocol.

The hard case is where Alice and Bob only have independent sources with entropy $k = \delta n$. Our starting point here is how we can construct an extractor for these three sources X, Y and W. In other words, let's first forget about the communication problem and see how we can get a 3-source extractor.

Since X has linear min-entropy, a standard approach would be to convert X into a somewhere high entropy (say entropy rate 0.9) source $\bar{X}$, using the condenser based on the sum-product theorem [BKS+05, Zuc07]. $\bar{X}$ is a matrix with a constant number of rows such that at least one of the rows has entropy rate 0.9. Once we have this, we can apply Raz's extractor to each row of $\bar{X}$ and W, and we get a somewhere random source with a constant number of rows. Now we can extract from such a source and an independent weak random source using the two-source extractor in [BRSW06].

So now how do we use these ideas in the case where Alice and Bob are separated by a channel controlled by an active adversary Eve? As a first step, we still convert X and Y into somewhere high entropy (say entropy rate 0.9) sources $\bar{X}$ and $\bar{Y}$ with $D = O(1)$ rows. Next we apply Raz's extractor to each row of $\bar{X}$ and W, and each row of $\bar{Y}$ and W. Thus we get two somewhere random sources SR_x and SR_y. Note that since Raz's extractor is a strong two-source extractor, the random rows in SR_x and SR_y are close to being independent of W. This is important for us.

Next, we have Alice authenticate a string to Bob. To do this, we want to use the authentication protocol we discussed in the previous section. However, now Alice and Bob don't have access to random bits. The important observation here is that they have *somewhere random sources*. In particular, the random rows of SR_x and SR_y can be used as seeds for a strong seeded extractor in the authentication protocol (since they are independent of W). Of course we don't know which row is the random row, thus we take a slice from the somewhere random sources with small width (so that X and Y don't lose much entropy) and use these slices in the authentication protocol. We call these slices X_1 and Y_1. How do we use them? We have Alice and Bob announce their slices to each other and each time they communicate, they compute prefixes of the outputs of an extractor. They then check if the prefixes they received match the prefixes they compute locally. Only this time we apply the extractor to W using each row of the slice as a seed. Thus the output of the extractor is also a matrix of D rows. Since the random row of the slice is close to being independent of W, the output of the extractor is also somewhere random. Next, we increase the length of the prefix by a factor of $2D$, because each time Alice or Bob reveals a matrix of D rows. Now it can be shown that to answer a challenge, Eve has to come up with the random row in the output of the extractor, whose length is larger than the total number of bits revealed so far. Therefore Eve can only succeed with a small probability. Given this protocol, Alice uses it to send another small slice of SR_x to Bob, and Bob uses this slice to extract random bits from his own source. We call this slice X_2.

There are two problems with the above discussion. First, the small slice X_2 sent by Alice may not be independent of W or the random row of the extractor output. Second, since each time the length of the prefix increases by a factor of $2D$ and each time we want fresh entropy in the extractor output, Alice can only send $\alpha \log k$ bits with some $\alpha < 1$, where k is the entropy of W. With this small number of bits it's not clear how Bob can extract random bits from his own source.

For the first problem, we show that although the small slice X_2 sent by Alice may not be independent of W or the random row of the extractor output, with high probability over the fixings of X_2, the random row of the extractor output has very high min-entropy. This is mainly because the length of X_2 is very small compared to the extractor output. Thus a typical fixing of it doesn't reduce the entropy of the random row of the extractor output by much. Now we can show that with correct parameters, the high min-entropy row of the extractor output still suffices for authentication. Thus for a typical value of the small slice, the success probability of Eve changing it is still small. Note now Eve may be able to actually change a small probability mass of the slice sent by Alice, but that doesn't hurt us much. This is different from the case where Alice and Bob have local random bits. For the second problem, luckily Bob also has a somewhere random source SR_y. Thus we can take a small slice Y_2 of SR_y so that the two-source extractor from [BRSW06] can be used to extract random bits from these two sources.

Now suppose that Bob correctly received the small slice X_2 sent by Alice, and Bob takes a small slice Y_2 of his somewhere random source. Let the output of the [BRSW06] extractor be R. We first fix W and now X_2 and Y_2 are deterministic functions of X and Y respectively, and are thus independent. Moreover X_2 is somewhere random. Thus R is close to uniform. Furthermore, since the two source extractor from [BRSW06] is strong, we can now fix Y_2 and conditioned on this fixing, R is still close to uniform. Now R is a deterministic function of X. Note that now all strings revealed by Bob are functions of Y_1 and X_1 (since W is fixed), and Y_1 is a deterministic function of Y and has small size. Thus we can further fix Y_1 and conditioned on this fixing, R is still close to uniform and is independent of Y. Moreover Y has a lot of entropy left and all the strings revealed by Bob are now deterministic functions of X_1. Therefore now we can apply a strong seeded extractor to Y and R and Bob obtains S_y. Note that we can condition on R and S_y is still close to uniform by the strong extractor property. Now S_y is a deterministic function of Y and is thus independent of all the transcripts revealed so far, and X. Thus Bob has obtained random bits that are close to uniform and private.

We actually cheated a little bit above, because again the size of R is very small compared to Y. Thus we won't be able to apply a seeded extractor to Y and R. However we can fix this problem by taking a slice Y_3 of SR_y. The size of Y_3 is much larger compared to the length of the transcript, but much smaller compared to Y. It is actually a slice with width $k^{\Omega(1)}$ (R will have size $\Omega(\log k)$). Since in the analysis we fix W, Y_3 is a deterministic function of Y, and the random row of Y_3 still has a lot of entropy left conditioned on the fixings of the transcript. Therefore we can apply the strong seeded extractor to Y_3 and R, and the above analysis about Bob obtaining private random bits still holds.

By symmetry Alice can also take a slice X_3 of SR_x and apply a strong seeded extractor to X_3 and R, and the above argument would also work for Alice. Therefore now Bob can use the authentication protocol to send R to Alice, and Alice applies the extractor to X_3 and R. By the same discussion above Eve may be able to change only a small probability mass of R, and this doesn't hurt us much. Thus at the end of the protocol Alice and Bob end up with nearly private and uniform random bits, while their shared secret W still has a lot of entropy left. Thus we have reduced the problem to the case where Alice and Bob have access to local uniform random bits, and previous results can be used to construct a privacy amplification protocol. However since we only manage to send $\Omega(\log k)$ bits from Alice to Bob, the error of the extractor and thus the security parameter of the protocol is $\Omega(\log k)$.

7.2.2 The Constructions and the Analysis

In this section we present the protocols and the formal analysis.

7.2.2.1 Non Constructive Results

First we show that non-constructively, this can be done. In fact, we can essentially reduce the problem to the case where Alice and Bob have local random bits. First we have the following theorem, that can be easily proved by the probabilistic method:

Theorem 7.2.4. *(Two source extractor) For all positive integers n, k such that $k > \log n$, there exists a function* $\mathsf{TExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^m$ *and* $0 < \epsilon < 1$ *such that* $m = \Omega(k)$, $\epsilon = 2^{-\Omega(k)}$ *and if* X, Y *are two independent* (n,k)*-sources, then*

$$|(X, \mathsf{TExt}(X,Y)) - (X, U_m)| \leq \epsilon$$

and

$$|(Y, \mathsf{TExt}(X,Y)) - (Y, U_m)| \leq \epsilon$$

Now we have the following protocol.

Protocol NExtract(x, y, w):

- Alice has a weak random source X, Bob has an independent weak random source Y, and they share an independent weak random source W. All these sources have min-entropy $k > \mathrm{polylog}(n)$.
- Let TExt be the strong two source extractor from Theorem 7.2.4.

1. Alice and Bob each applies TExt to his or her own source and W.
2. Alice obtains $S_x = \mathsf{TExt}(X, W)$ and Bob obtains $S_y = \mathsf{TExt}(Y, W)$, each outputting $\Omega(k)$ bits.

Now we have the following theorem.

Theorem 7.2.5.

$$|(S_x, S_y, W) - (U_x, U_y, W)| \leq 2^{-\Omega(k)},$$

where here (U_x, U_y) *is the uniform distribution independent of* W.

Proof. By Theorem 7.2.4 and a standard averaging argument, with probability $1 - \sqrt{\epsilon}$ over the fixings of W, S_x is $\sqrt{\epsilon}$-close to uniform, where $\epsilon = 2^{-\Omega(k)}$. Similarly, with probability $1 - \sqrt{\epsilon}$ over the fixings of W, S_y is $\sqrt{\epsilon}$-close to uniform. Thus with probability $1 - 2\sqrt{\epsilon}$ over the fixings of W, both S_x and S_y are $\sqrt{\epsilon}$-close to uniform. Note that after fixing W, S_x is a function of X and S_y is a function of Y. Thus they are independent. Therefore with probability $1 - 2\sqrt{\epsilon}$ over the fixings of W, (S_x, S_y) is $2\sqrt{\epsilon}$-close to uniform. Thus we have

$$|(S_x, S_y, W) - (U_x, U_y, W)| \leq 2^{-\Omega(k)}.$$

■

Now all we need to do is to plug in the non-explicit optimal privacy amplification in [DW09] to obtain Theorem 7.2.1.

7.2.2.2 Weak random sources with entropy rate $> 1/2$

Now we study a simple case where Alice and Bob's weak random sources have entropy rate $> 1/2$. In this case, we show that we can also reduce the problem to the case where Alice and Bob have local random bits. The reason is that we have strong two-source extractors for such sources, namely Raz's extractor from Theorem 3.5.7.

First we have the following protocol.

Protocol ExtractH(x, y, w):

- Alice has a weak random source X, Bob has an independent weak random source Y, and they share an independent weak random source W. Both X and Y have min-entropy $(1/2 + \delta)n$ and W has min-entropy $k > \text{polylog}(n)$.
- Let Raz be the strong two source extractor from Theorem 3.5.7.

1. Alice and Bob each applies Raz to his or her own source and W.
2. Alice obtains $S_x = \mathsf{Raz}(X, W)$ and Bob obtains $S_y = \mathsf{Raz}(Y, W)$, each outputting $\Omega(k)$ bits.

Now we have the following theorem.

Theorem 7.2.6.

$$|(S_x, S_y, W) - (U_x, U_y, W)| \leq 2^{-\Omega(k)},$$

where here (U_x, U_y) is the uniform distribution independent of W.

Proof. Essentially repeat the proof in the previous section. ■

Again, all we need to do now is to plug in any privacy amplification protocol in [RW03, KR09a, DW09, CKOR10] to obtain Theorem 7.2.2.

7.2.2.3 Weak random sources with linear min-entropy

In this section we relax the assumption and only require Alice and Bob have weak random sources with arbitrarily linear min-entropy. More specifically, we assume that Alice and Bob each has a local $(n, \delta n)$ source for some constant $0 < \delta < 1$. We assume the shared source is an (n, k) source with $k \geq \text{polylog}(n)$. Actually we can also deal with the case where the shared source has linear min-entropy but the local weak sources only have poly logarithmic entropy. This case is quite similar, and thus omitted.

The protocol

Here we give a protocol for Alice and Bob to extract private local random bits. That is, in the end of the protocol, both Alice and Bob obtain local random bits that are close to uniform and independent of the shared weak random source, even in Eve's view. Moreover the shared weak source still has most of its entropy left.

We need the following definition about the slice of a concatenation of strings.

Definition 7.2.7. [Rao09] Given ℓ strings of length n, $x = x_1, \cdots, x_\ell$, define $\mathsf{Slice}(x, s)$ to be the string $x' = x'_1, \cdots, x'_\ell$ such that for each i x'_i is the prefix of x_i of length s.

Now we can describe our protocol. In this protocol when a party is authenticating a message to the other party, we do not use the error correcting code. Instead, we just convert the message to a string with a fixed number of 1's. One simple way to do this is map each bit 0 to 01 and map each bit 1 to 10. Thus the number of 1's in the authenticated message is known to both parties before they execute the protocol.

Protocol $\mathsf{Extract}(x, y, w)$:

- Alice has a weak random source X, Bob has an independent weak random source Y, and they share an independent weak random source W. Both X and Y have min-entropy δn and W has min-entropy $k > \text{polylog}(n)$.
- Let Zuc be the somewhere condenser from Theorem 3.5.3.
- Let Raz be the strong two source extractor from Theorem 3.5.7.
- Let SRGExt be the two source extractor from Theorem 3.5.11.
- Let Ext be a strong extractor as in Theorem 3.5.13.

- Let $0 < \gamma < 1$ be some constant.

1. Alice uses Zuc to convert X into a somewhere rate-.9 source $\bar{X}$, with D rows for some constant $D > 1$. Similarly Bob also converts Y into a somewhere rate-.9 source $\bar{Y}$ with D rows.

2. Alice applies Raz to each row of $\bar{X}$ and W and obtains a somewhere random source SR_x, with each row outputting k^γ bits. Similarly Bob also applies Raz to each row of $\bar{Y}$ and W and obtains a somewhere random source SR_y, with each row outputting k^γ bits.

3. Alice produces 3 strings: $X_1 = \mathsf{Slice}(SR_x, c\log n)$, $X_2 = \mathsf{Slice}(SR_x, \mu\log k)$ and $X_3 = \mathsf{Slice}(SR_x, k^\beta)$ for some parameters $c > 1$, $0 < \mu < 1$ and $0 < \beta < 1$ to be chosen later. Bob also produces 3 strings: $Y_1 = \mathsf{Slice}(SR_y, c\log n)$, $Y_2 = \mathsf{Slice}(SR_y, \mu\log k)$ and $Y_3 = \mathsf{Slice}(SR_y, k^\beta)$.

4. Alice announces x_1 to Bob and Bob announces y_1 to Alice. Alice then computes $r_y = \mathsf{Ext}(w, y_1)$ and Bob computes $r_x = \mathsf{Ext}(w, x_1)$, where the function Ext is applied to w and each row of x_1, y_1, and each output string has length k^γ.

5. Alice converts x_2 to a string m_x with a fixed number of 1's. Let the length of the string be t (note $t = O(\log k)$). Alice then authenticates m_x to Bob by doing the following:

6. Define three set of integers as $C_{1i} = (4D)^{3i-2}c\log n, C_{2i} = (4D)^{3i-1}c\log n, C_{3i} = (4D)^{3i}c\log n$, where $i = 1, \cdots, 2t$.

7. For $i = 1$ to t do (authenticate x_2 to Bob):

   - If $m_{xi} = 0$, Alice sends $(0, \mathsf{Slice}(r_y, C_{1i}))$. Otherwise she sends $(1, \mathsf{Slice}(r_y, C_{2i}))$.
   - Bob receives the message and verifies $\mathsf{Slice}(r_y, C_{1i}) = \mathsf{Slice}(\mathsf{Ext}(w, y_1), C_{1i})$ in the 0 case and $\mathsf{Slice}(r_y, C_{2i}) = \mathsf{Slice}(\mathsf{Ext}(w, y_1), C_{2i})$ in the 1 case. If the verification does not go through, abort. Bob then sends $\mathsf{Slice}(r_x, C_{3i})$ to Alice.
   - Alice receives the message and verifies $\mathsf{Slice}(r_x, C_{3i}) = \mathsf{Slice}(\mathsf{Ext}(w, x_1), C_{3i})$.

8. When received t bits, Bob verifies that the number of ones in the received string is $\mathsf{wt}(m_x)$; aborts otherwise. Bob recovers x_2 from m_x.

9. Bob computes $r_3 = \mathsf{SRGExt}(y_2, x_2)$, outputting $\Omega(\log k)$ bits. Bob then computes $s_y = \mathsf{Ext}(y_3, r_3)$, outputting $k^{\Omega(1)}$ bits.

10. Bob converts r_3 to a string m_y with a fixed number of 1's. The length of the string is t'. Bob then authenticates m_y to Alice by doing the following:

11. For $i = t+1$ to $t+t'$ do (authenticate r_3 to Alice):

    - If $m_{y(i-t)} = 0$, Bob sends $(0, \mathsf{Slice}(r_x, C_{1i}))$. Otherwise he sends $(1, \mathsf{Slice}(r_x, C_{2i}))$.
    - Alice receives the message and verifies $\mathsf{Slice}(r_x, C_{1i}) = \mathsf{Slice}(\mathsf{Ext}(w, x_1), C_{1i})$ in the 0 case and $\mathsf{Slice}(r_x, C_{2i}) = \mathsf{Slice}(\mathsf{Ext}(w, x_1), C_{2i})$ in the 1 case. If the verification does not go through, abort. Alice then sends $\mathsf{Slice}(r_y, C_{3i})$ to Bob.
    - Bob receives the message and verifies $\mathsf{Slice}(r_y, C_{3i}) = \mathsf{Slice}(\mathsf{Ext}(w, y_1), C_{3i})$.

12. When received t' bits, Alice verifies that the number of ones in the received string is $\mathsf{wt}(m_y)$; aborts otherwise. Alice recovers r_3 from m_y.

13. Alice computes $s_x = \mathsf{Ext}(x_3, r_3)$, outputting $k^{\Omega(1)}$ bits.

Analysis of the protocol

We claim that S_x and S_y can now be treated as local private random bits of Alice and Bob. That is , they are close to being independent and uniform and independent of W, even in Eve's view. Specifically, we have the following theorem.

Theorem 7.2.8. *Let V denote the transcript of the whole protocol in Eve's view. Then if $S_x \neq \perp$ and $S_y \neq \perp$ (the protocol doesn't abort), we have*

$$|(S_x, S_y, W, V) - (U_x, U_y, W, V)| \leq 1/\text{poly}(k),$$

where here (U_x, U_y) is the uniform distribution independent of (W, V). Moreover, with probability $1 - 2^{-k^{\Omega(1)}}$ over the fixings of $V = v$, W has min-entropy $k - k^{\Omega(1)}$.

Proof. Without loss of generality assume that the first row of $\bar{X}$ and the first row of $\bar{Y}$ have entropy rate 0.9. Let the two rows be $\bar{X}_1$ and $\bar{Y}_1$. Thus by Theorem 3.5.7 we have

$$|(SR_{x1}, W) - (U_x, W)| = 2^{-\Omega(k)}$$

and

$$|(SR_{y1}, W) - (U_y, W)| = 2^{-\Omega(k)},$$

where SR_{x1} and SR_{y1} stand for the first rows of SR_x and SR_y respectively. Since conditioned on any fixing of $W = w$, SR_{x1} and SR_{y1} are functions of X and Y and are thus independent, we have

$$|(SR_{x1}, SR_{y1}, W) - (U_x, U_y, W)| = 2^{-\Omega(k)}. \tag{7.1}$$

Note that the length of r_3 is less than the length of x_2. Thus $t' < t$ and therefore the protocol runs for at most $2t = O(\log k)$ rounds. Also in the protocol $\mathsf{Ext}(W, X_{11})$ and $\mathsf{Ext}(W, Y_{11})$ output at most $(4D)^{6t} c \log n = k^{\Omega(1)} \log n$ bits. We choose μ s.t. this number is at most k^{γ}, thus we have enough entropy in W for the outputs. Therefore by Equation 7.1,

$$|(\mathsf{Ext}(W, X_{11}), \mathsf{Ext}(W, Y_{11})) - (U'_x, U'_y)| = 2^{-\Omega(k)} + 1/\text{poly}(n) = 1/\text{poly}(n).$$

Note that now the random variable that Alice is trying to send to Bob, X_2, and the random variables X_1, Y_1 that have already been revealed, may not be (close to) independent of $(\mathsf{Ext}(W, X_{11}), \mathsf{Ext}(W, Y_{11}))$. We first show in this case the probability that Eve can successfully change a string x_2 to a different string is small. To show this, we have the following lemma.

Lemma 7.2.9. *Assume that* $(\mathsf{Ext}(W, X_{11}), \mathsf{Ext}(W, Y_{11}))$ *is* ϵ_0*-close to uniform. Let* X_1 *and* Y_1 *be as in the protocol. Let* M *be any random variable with at most* $D \log n$ *bits and Alice uses the protocol to authenticate* M *to Bob. Then the probability that Eve can successfully change a string* m *to a different string is bounded above by* $1/\text{poly}(n) + \epsilon_0$*, where the probability is over* M *and the random variables used to transfer* M*.*

Proof. Let $\bar{R_x} = \mathsf{Ext}(W, X_1)$, $\bar{R_y} = \mathsf{Ext}(W, Y_1)$ and $\bar{R_{x1}}$, $\bar{R_{y1}}$ be the first rows of $\bar{R_x}$, $\bar{R_y}$ respectively. Thus $\bar{R_{x1}} = \mathsf{Ext}(W, X_{11})$ and $\bar{R_{y1}} = \mathsf{Ext}(W, Y_{11})$. Let R_x and R_y be the actual random variables computed by Bob and Alice respectively. We want to deal with the ideal case where $\bar{R_{x1}}, \bar{R_{y1}}$ is uniform instead of ϵ_0-close to uniform. Note that $(M, X_1, Y_1, \bar{R_x}, \bar{R_y}, R_x, R_y)$ are all the random variables used by Alice to authenticate M to Bob. Thus by Lemma 7.1.1 we first construct another distribution $(M', X'_1, Y'_1, \bar{R'_x}, \bar{R'_y}, R'_x, R'_y, \bar{R'_{x1}}, \bar{R'_{y1}})$ where $(\bar{R'_{x1}}, \bar{R'_{y1}})$ is uniform and

$$|(M, X_1, Y_1, \bar{R_x}, \bar{R_y}, R_x, R_y, \bar{R_{x1}}, \bar{R_{y1}}) - (M', X'_1, Y'_1, \bar{R'_x}, \bar{R'_y}, R'_x, R'_y, \bar{R'_{x1}}, \bar{R'_{y1}})| \leq \epsilon_0.$$

From now on we will continue the discussion as if $(M, X_1, Y_1, \bar{R_x}, \bar{R_y}, R_x, R_y, \bar{R_{x1}}, \bar{R_{y1}}) = (M', X'_1, Y'_1, \bar{R'_x}, \bar{R'_y}, R'_x, R'_y, \bar{R'_{x1}}, \bar{R'_{y1}})$. We can do this because in the analysis all we use are the sizes of $M', X'_1, Y'_1, \bar{R'_x}, \bar{R'_y}, R'_x, R'_y, \bar{R'_{x1}}, \bar{R'_{y1}}$, which are the same as those of $M, X_1, Y_1, \bar{R_x}, \bar{R_y}, R_x, R_y, \bar{R_{x1}}, \bar{R_{y1}}$ by Lemma 7.1.1. Thus the success probability of Eve can only differ by at most ϵ_0.

Now note that the length of m is at most $D \log n$. Thus by Lemma 3.5.14 we have

$$\Pr_M[H_\infty(\mathsf{Ext}(W, X_{11})|M = m) \geq (4D)^{6t} c \log n - D \log n - D \log n] \geq 1 - 2^{-D \log n}.$$

That is,

$$\Pr_M[H_\infty(\mathsf{Ext}(W, X_{11})|M = m) \geq (4D)^{6t} c \log n - 2D \log n] \geq 1 - 1/\mathrm{poly}(n).$$

Similarly

$$\Pr_M[H_\infty(\mathsf{Ext}(W, Y_{11})|M = m) \geq (4D)^{6t} c \log n - 2D \log n] \geq 1 - 1/\mathrm{poly}(n).$$

We show that when m is a string s.t. both $(\mathsf{Ext}(W, X_{11})|M = m)$ and $(\mathsf{Ext}(W, Y_{11})|M = m)$ have min-entropy at least $(4D)^{6t} c \log n - 2D \log n$, the success probability that Eve can change m without being detected is $1/\mathrm{poly}(n)$. By the union bound this happens with probability $1 - 1/\mathrm{poly}(n)$.

To see this, we first prove the following lemma.

Lemma 7.2.10. *In order to change m to a different string, Eve has to come up with at least one challenge.*

Proof. To change m to a different string, Eve must take a series of operations. We consider two cases.

- Case 1: The operations that Eve made include insertion or deletion. In this case the first such operation must incur a challenge. To see this, let j be the round right before the insertion or deletion. Thus at the end of round j, Alice has announced at most a total of $DC_{2j} + cD \log n = C_{3j}/4 + cD \log n$ bits. Similarly Bob has announced at most a total of $DC_{3j} + cD \log n = C_{1(j+1)}/4 + cD \log n$ bits. If it's an insertion, Eve has to come up with at least $C_{1(j+1)} = (4D)^{3j+1} c \log n$ random bits to avoid detection, and we see that

$$C_{1(j+1)} - (C_{3j}/4 + cD \log n) - (C_{1(j+1)}/4 + cD \log n) - D \log n > 4cD \log n.$$

If it's a deletion, then Alice has announced at most a total of $DC_{2(j+1)}+cD\log n = C_{3(j+1)}/4+cD\log n$ bits and Bob has announced a total of $DC_{3j}+cD\log n = C_{1(j+1)}/4+cD\log n$ bits. Eve has to come up with at least $C_{3(j+1)} = (4D)^{3j+3}c\log n$ random bits to avoid detection, and we see that

$$C_{3(j+1)} - (C_{3(j+1)}/4 + cD\log n) - (C_{1(j+1)}/4 + cD\log n) - D\log n > 4cD\log n.$$

- Case 2: The operations that Eve made do not include insertion or deletion. In this case, since the number of 1's in the message is known to Bob, Eve must make at least one operation of changing 0 to 1 and at least one operation of changing 1 to 0. Then the operation of changing 0 to 1 will incur a challenge. To see this, let j be the current round(since Eve does not make operations of insertion and deletion, the round number is the same for Alice, Bob and Eve). Thus now Alice has announced a total of $DC_{1j}+cD\log n = C_{2j}/4+cD\log n$ bits while Bob has announced a total of $DC_{3(j-1)}+cD\log n = C_{1j}/4+cD\log n$ bits. Eve has to come up with at least $C_{2j} = (4D)^{3j-1}c\log n$ random bits to avoid detection, and we see that

$$C_{2j} - (C_{2j}/4 + cD\log n) - (C_{1j}/4 + cD\log n) - D\log n > 4cD\log n.$$

■

Now let j be the round that Eve has to answer the first challenge. Let B_j stand for the random variable of all the strings that have been revealed by Alice and Bob till now, and let l_b be the length of the string b_j. Let A_j denote the random variable that Eve is trying to come up with, and let l_a be the length of the string a. Thus we have just shown that $l_a \geq l_b + 4cD\log n$.

Since both $\mathsf{Ext}(W, X_{11})|M = m)$ and $\mathsf{Ext}(W, Y_{11})|M = m)$ have min-entropy at least $(4D)^{6t}c\log n - 2D\log n$, A has min-entropy $l_a - 2D\log n$. Thus by Lemma 3.5.14,

$$\Pr_B[H_\infty(A|B=b) \geq l_a - 2D\log n - l_b - D\log n] \geq 1 - 2^{-D\log n}.$$

Thus

$$\Pr_B[H_\infty(A|B=b) \geq D\log n] \geq 1 - 1/\mathrm{poly}(n).$$

Therefore the probability that Eve can successfully change the string is bounded from above by $1/\mathrm{poly}(n) + 2^{-D\log n} = 1/\mathrm{poly}(n)$.

Thus, going back to the case where $(\mathsf{Ext}(W, X_{11}), \mathsf{Ext}(W, Y_{11}))$ is ϵ_0-close to uniform, the success probability of Eve is bounded from above by $1/\mathrm{poly}(n) + \epsilon_0$. $\blacksquare$

Thus the success probability of Eve changing x_2 to a different string is bounded from above by $1/\mathrm{poly}(n) + 1/\mathrm{poly}(n) = 1/\mathrm{poly}(n)$. Note this probability is also over X_2. By a standard averaging argument, with probability $1 - 1/\mathrm{poly}(n)$ over X_2, the success probability of Eve changing x_2 to a different string is at most $1/\mathrm{poly}(n)$.

Now Bob obtains a random variable X_2'. Note that X_2' is not exactly X_2 since Eve may be able to change X_2 for a probability mass of $\epsilon = 1/\mathrm{poly}(n)$. Assume for now that Bob obtains X_2 instead of X_2'. Now we fix $W = w$. Note that after this fixing, X_1, X_2 are functions of X and Y_1, Y_2 are functions of Y. By Theorem 3.5.7, with probability $1 - 2^{-\Omega(k)}$ over the fixings of $W = w$, X_2 is $2^{-\Omega(k)}$-close to being a somewhere random source, and so is Y_2. Moreover X_2 and Y_2 are independent. Thus by Theorem 3.5.11, we have that for a typical fixing of $W = w$,

$$|(X_2, R_3) - (X_2, U_m)| < \epsilon_1 \tag{7.2}$$

and

$$|(Y_2, R_3) - (Y_2, U_m)| < \epsilon_1, \tag{7.3}$$

where $\epsilon_1 = 2^{-\Omega(k)} + 2^{-\Omega(\log k)} = 1/\mathrm{poly}(k)$.

We then further fix $Y_2 = y_2$. By Equation 7.3 with probability $1 - \sqrt{\epsilon_1}$ over the fixings of $Y_2 = y_2$, R_3 is $\sqrt{\epsilon_1}$-close to uniform. Further note that after this fixing R_3 is a deterministic function of X, and Y_1 is a deterministic function of Y. Thus we can further fix $Y_1 = y_1$ and R_3 is still $\sqrt{\epsilon_1}$-close to uniform. Note that y_1 has length $cD \log n$ and Y_3 has min-entropy k^{β}. Thus by Lemma 3.5.14 we have that with probability $1 - 1/\mathrm{poly}(n)$ over the fixings of $Y_1 = y_1$, Y_3 has min-entropy $0.9k^{\beta}$. Thus we have shown that

[Condition 1] With probability $1 - 2^{-\Omega(k)} - \sqrt{\epsilon_1} - 1/\mathrm{poly}(n) = 1 - 1/\mathrm{poly}(k)$ over the fixings of $W = w, Y_2 = y_2, Y_1 = y_1$, R_3 is $\sqrt{\epsilon_1}$-close to uniform, Y_3 has min-entropy k^{β} and R_3 and Y_3 are independent.

Now let's consider the case where Bob obtains X_2' instead of X_2 and Bob computes R_3' instead of R_3. Note that Eve can only change a $\epsilon = 1/\mathrm{poly}(n)$ probability mass of X_2. For a fixed $W = w, Y_1 = y_1$(note that y_2 is a slice of y_1), let E_{w,y_1} denote the event that Eve changes a $\sqrt{\epsilon}$ probability mass of $X_2|(W = w, Y_1 = y_1)$. By a standard averaging argument we have

$$\Pr_{W,Y_1}[E_{w,y_1}] \leq \sqrt{\epsilon}.$$

Now consider a typical fixing of $W = w, Y_1 = y_1$ where the event E_{w,y_1} does not happen and **Condition 1** holds. This happens with probability $1 - 1/\text{poly}(k) - \sqrt{\epsilon} = 1 - 1/\text{poly}(k)$. Note since Condition 1 holds, after this fixing R_3 and Y_3 are independent and R_3 is a deterministic function of X_2 (and X). Now Eve can change a probability mass of $\sqrt{\epsilon}$ here, but all strings revealed by Bob are fixed and W are fixed. Thus whatever Eve does, the resulting R_3' is a function of X and is still independent of Y_3. Moreover since Eve can only change a probability mass of $\sqrt{\epsilon}$, R_3' is $\sqrt{\epsilon_1} + \sqrt{\epsilon} = 1/\text{poly}(k)$-close to uniform. Therefore we have shown that

[**Condition 2**] With probability $1 - 1/\text{poly}(k)$ over the fixings of $W = w, Y_2 = y_2, Y_1 = y_1$, R_3' is $1/\text{poly}(k)$-close to uniform, Y_3 has min-entropy $0.9k^{\beta}$ and R_3' and Y_3 are independent.

Therefore by the property of the strong extractor Ext, we have

$$|(S_y, R_3') - (U, R_3')| < 1/\text{poly}(k).$$

Note that we have fixed $W = w, Y_2 = y_2, Y_1 = y_1$, and we can now further fix $R_3' = r_3'$. After this fixing S_y is just a function of Y and is independent of X. Thus we have fixed all possible information that Eve could know about Y and S_y is still close to uniform. Therefore S_y can be treated as local private random bits of Bob.

Now again by Lemma 7.2.9 Bob can authenticate R_3' to Alice such that Eve can only successfully change a probability mass of $\epsilon = 1/\text{poly}(n)$ of R_3'. Suppose Alice obtains R_3''. Now we fix $W = w$ and let E_w stand for the event that Eve changes a $\sqrt{\epsilon}$ probability mass of $X_2|(W = w)$. By a standard averaging argument we have

$$\Pr_W[E_w] \leq \sqrt{\epsilon}.$$

Now for a typical fixing of $W = w$ where both X_2 and Y_2 are close to a somewhere random source and Eve changes less than a $\sqrt{\epsilon}$ probability mass of $X_2|(W = w)$, X_2 is a function of X, Y_2 is a function of Y and are thus independent. By Equation 7.2 with probability $1 - \sqrt{\epsilon_1}$ over the fixings of $X_2 = x_2$, R_3 is $\sqrt{\epsilon_1}$-close to uniform. Thus for a further typical fixing of $X_2 = x_2$ where X_2 is not changed by Eve and R_3 is close to uniform, R_3(and R_3') is a function of Y and is independent of X. Therefore we can further fix $X_1 = x_1$ and R_3' is still close to uniform. Note that x_1 has length $cD \log n$ and X_3 has min-entropy k^β. Thus by Lemma 3.5.14 we have that with probability $1 - 1/\text{poly}(n)$ over the fixings of $X_1 = x_1$, X_3 has min-entropy $0.9k^\beta$.

Now Eve can change a $\epsilon = 1/\text{poly}(n)$ probability mass of R'_3. Let E_{w,x_2} stand for the event that Eve changes a $\sqrt{\epsilon}$ probability mass of $R'_3|(W = w, X_2 = x_2)$. By a standard averaging argument we have

$$\Pr_{W,X_2}[E_{w,x_2}] \leq \sqrt{\epsilon}.$$

Thus for a typical fixing of $(W = w, X_2 = x_2)$, Eve changes less than $\sqrt{\epsilon}$ probability mass of $R'_3|(W = w, X_2 = x_2)$. Since now all strings revealed by Alice and W are fixed, no matter what Eve does, the resulting R''_3 is a function of Y and is still independent of X_3. Moreover since Eve can only change a probability mass of $\sqrt{\epsilon}$, R''_3 is $\sqrt{\epsilon_1} + \sqrt{\epsilon} = 1/\text{poly}(k)$-close to uniform. Note the probability of typical fixings of $(W = w, X_2 = x_2)$ is at least $1 - \sqrt{\epsilon} - \sqrt{\epsilon_1} - \sqrt{\epsilon} - \sqrt{\epsilon} = 1 - 1/\text{poly}(k)$. Therefore we have shown that

[Condition 3] With probability $1-1/\text{poly}(k)$ over the fixings of $W = w, X_2 = x_2, X_1 = x_1$, R''_3 is $1/\text{poly}(k)$-close to uniform, X_3 has min-entropy $0.9k^\beta$ and R''_3 and X_3 are independent.

Therefore by the property of the strong extractor Ext, we have

$$|(S_x, R''_3) - (U, R''_3)| < 1/\text{poly}(k).$$

Note that we have fixed $W = w, X_2 = x_2, X_1 = x_1$, and we can now further fix $R''_3 = r''_3$. After this fixing S_x is just a function of X and is independent of Y, and is thus also independent of S_y (which now is a function of Y). Thus we have fixed all possible information that Eve could know about X and S_x is still close to uniform. Therefore S_x can be treated as local private random bits of Bob.

Therefore, we have eventually shown that

$$|(S_x, S_y, X_1, Y_1, W) - (U_x, U_y, X_1, Y_1, W)| \leq 1/\text{poly}(k).$$

Note that now the entire transcript V up till now is a deterministic functions of W, X_1, Y_1. Therefore we also have

$$|(S_x, S_y, V, W) - (U_x, U_y, V, W)| \leq 1/\text{poly}(k).$$

Note that the transcript has length at most k^γ. Therefore by Lemma 3.5.14 with probability $1-2^{-k^{\Omega(1)}}$ over the fixings of the transcript, W still has min-entropy at least $k-k^{\Omega(1)}$. Thus the theorem is proved. ■

Now all we need to do is to plug in any privacy amplification protocol in [RW03, KR09a, DW09, CKOR10] to obtain Theorem 7.2.3.

7.3 Privacy Amplification with Local Uniform Random Bits

In this section we show how to improve previous results in various settings. Our improvements rely on the construction of non-malleable extractors in Section 6.4.

Dodis and Wichs [DW09] showed that such extractors imply privacy amplification protocols with 2 rounds and optimal entropy loss. As a direct corollary of Theorem 6.4.2 and the protocol of Dodis and Wichs, we obtain a 2-round protocol for privacy amplification with optimal entropy loss, when the entropy rate is $1/2+\alpha$ for any $\alpha>0$. This improves the significant entropy loss in the one-round protocols of Dodis, Katz, Reyzin, and Smith [DKRS06] and Kanukurthi and Reyzin [KR08].

Next, we use our non-malleable extractor to give a constant-round privacy amplification protocol with optimal entropy loss, when the entropy rate is δ for any constant $\delta>0$. This significantly improves the round complexity of [KR09b] and [CKOR10]. It also significantly improves the entropy loss of [DW09], at the price of a slightly larger but comparable round complexity ($O(1)$ vs. 2). Our result is stated as follows.

Theorem 7.3.1. *Under conjecture 3.2.2, for any constant $0<\delta<1$ and error $2^{-\Omega(\delta n)}<\epsilon<1/n$, there exists a polynomial-time, constant-round $(\delta n, \delta n - O(\log(1/\epsilon)), \varepsilon)$-secure protocol for privacy amplification. More specifically, the protocol takes number of rounds $\mathrm{poly}(1/\delta)$, and achieves entropy loss $\mathrm{poly}(1/\delta)\log(1/\epsilon)$.*

7.3.1 Overview of the Protocol for Privacy Amplification

We first describe Dodis and Wichs' optimal two-round protocol using a non-malleable extractor. The protocol also uses a cryptographic primitive: a one-time message authentication code (MAC). Roughly speaking, a MAC uses a private uniformly random key R to produce a tag T for a message m, such that without knowing the key, the probability that an adversary can guess the correct tag T' for another message $m'\neq m$ is small, even given m and T.

Now assume that we have a non-malleable extractor nmExt that works for any (n, k)-source X. Then there is a very natural two-round privacy amplification protocol. In the first round Alice chooses a fresh random string Y and sends it to Bob. Bob receives a possibly modified string Y'. They then compute $R = \mathsf{nmExt}(X, Y)$ and $R' = \mathsf{nmExt}(X, Y')$ respectively. In the second round, Bob chooses a fresh random string W' and sends it to Alice, together with $T' = \mathsf{MAC}_{R'}(W')$ by using R' as the MAC key. Alice receives a possibly modified version (W, T), and she checks if $T = \mathsf{MAC}_R(W)$. If not, then Alice aborts; otherwise they compute outputs $Z = \mathsf{Ext}(X, W)$ and $Z' = \mathsf{Ext}(X, W')$ respectively, where Ext is a seeded strong extractor.

The analysis of the above protocol is also simple. If Eve does not change Y, then $R = R'$ and is (close to) uniform. Therefore by the property of the MAC the probability that Eve can change W' without being detected is very small. On the other hand if Eve changes Y, then by the property of the non-malleable extractor, one finds that R' is (close to) independent of R. Thus in this case, again the probability that Eve can change W' without being detected is very small. In fact, in this case Eve cannot even guess the correct MAC for W' with a significant probability.

The above protocol is nice, except that we only have non-malleable extractors for entropy rate $> 1/2$. As a direct corollary this gives our 2-round privacy amplification protocol for entropy rate $> 1/2$. To get a protocol for arbitrary positive entropy rate, we have to do more work.

We start by converting the shared weak random source X into a somewhere high min-entropy rate source. The conversion uses recent condensers built from sum-product theorems. Specifically, any n-bit weak random source with linear min-entropy can be converted into a matrix with a constant number of rows, such that at least one row has entropy rate 0.9.[1] Moreover each row still has $\Theta(n)$ bits. Note that since Alice and Bob apply the same function to the shared weak random source, they now also share the same rows.

Now it is natural to try the two-round protocol for each row and hope that it works on the row with high min-entropy rate. More specifically, for each row i we have a two round protocol that produces R_i, R'_i in the first round and Z_i, Z'_i in the second round. Now let g be the first row that has min-entropy rate 0.9. We hope that $Z_g = Z'_g$ with high probability, and further that Z_g, Z'_g are close to uniform and private. This is indeed the case if we run the two round protocol for each row sequentially (namely we run it for the first row, and then the second row, the third row, and so on), and we can argue as follows.

[1]In fact, the result is (close to) a convex combination of such matrices. For simplicity, however, we can assume that the result is just one such matrix, since it does not affect the analysis.

Assume the security parameter we need to achieve is s, so each of R_i, R'_i has $O(\log n + s)$ bits by the property of the MAC. As long as s is not too large, we can fix all these random variables up to row $g-1$, and argue that row g still has min-entropy rate $> 1/2$ (since each row has $\Theta(n)$ bits we can actually achieve a security parameter up to $s = \Omega(n)$). Note that we have essentially fixed all the information about X that can be leaked to Eve. Therefore now for row g the protocol succeeds and thus $Z_g = Z'_g$ with high probability, and Z_g, Z'_g are close to uniform and private.

However, we don't know which row is the good row. We now modify the above protocol to ensure that, once we reach the first good row g, for all subsequent rows i, with $i > g$, we will have that $Z_i = Z'_i$ with high probability, and further Z_i, Z'_i are close to uniform and private. If this is true then we can just use the output for the last row as the final output.

To achieve this, the crucial observation is that once we reach a row $i-1$ such that $Z_{i-1} = Z'_{i-1}$, and Z_{i-1}, Z'_{i-1} are close to uniform and private, then Z_{i-1} can be used as a MAC key to authenticate W'_i for the next row. Now if $W'_i = W_i$ for row i, then $Z_i = Z'_i$ and Z_i, Z'_i will also be close to uniform and private. Therefore, we modify the two-round protocol so that in the second round for row i, not only do we use $T'_{i1} = \mathsf{MAC}_{R'_i}(W'_i)$ to authenticate W'_i, but also we use $T'_{i2} = \mathsf{MAC}_{Z'_{i-1}}(W'_i)$ to authenticate W'_i.

This would have worked given that $Z_{i-1} = Z'_{i-1}$, and Z_{i-1}, Z'_{i-1} are close to uniform and private, except for another complication. The problem is that now $T'_{i1} = \mathsf{MAC}_{R'_i}(W'_i)$ could leak information about Z_{i-1} to Eve, so Z_{i-1} is no longer private. Fortunately, there are known constructions of MACs that work even when the key is not uniform, but instead only has large enough average conditional min-entropy in the adversary's view. Specifically, Theorem 7.3.3 indicates that the security parameter of this MAC is roughly the average conditional min-entropy of the key minus half the key length, and the key length is roughly twice as long as the length of the tag. Therefore, we can choose a small tag length for $T'_{i1} = \mathsf{MAC}_{R'_i}(W'_i)$, and a large tag length for $T'_{i2} = \mathsf{MAC}_{Z'_{i-1}}(W'_i)$. For example, if the tag length for T'_{i1} is $2s$, and the tag length for T'_{i2} is $4s$, then the key length for T'_{i2} is $8s$. Thus the average min-entropy of Z_{i-1} conditioned on T'_{i1} is $8s - 2s = 6s$, and we can still achieve a security parameter of $6s - 4s = 2s$.

Finally, the discussion so far implicitly assumed that Eve follows a natural "synchronous" scheduling, where she never tries to get one party out-of-sync with another party. To solve this problem, after each Phase i Bob performs a "liveness" test, where Alice has to respond to a fresh extractor challenge from Bob to convince Bob that Alice is still "present" in this round. This ensures that if Bob completes the protocol, Alice was "in-sync" with Bob throughout. However, Eve might be able to make Alice be out-of-sync with Bob, causing Alice to output a non-random key (and Bob reject). To solve this last problem, we add one more round at the end which ensures that Alice always outputs a random key (and Bob either outputs the same key or rejects).

With this modification, the complete protocol is depicted in Figure 7.2. Essentially, for the first good row, the property of the non-malleable extractor guarantees that Eve cannot change W'_g with significant probability. For all subsequent rows, by using the output Z'_{i-1} from the previous row as the MAC key, the property of the MAC guarantees that Eve cannot change W'_i with significant probability. Therefore, the output for the last row can be used to authenticate the last seed of the extractor chosen by Alice (for the reason mentioned above) to produce the final output.

Finally, we note that our final protocol has $O(1)$ rounds and achieves asymptotically optimal entropy loss is $O(s + \log n)$, for security parameter s.

7.3.2 The Construction and the Analysis

First we need the definition of a MAC. One-time message authentication codes (MACs) use a shared random key to authenticate a message in the information-theoretic setting.

Definition 7.3.2. [KR09b] A function family $\{\mathsf{MAC}_R : \{0,1\}^\ell \to \{0,1\}^v\}$ is a ϵ-secure one-time MAC for messages of length ℓ with tags of length v if for any $M \in \{0,1\}^\ell$ and any function (adversary) $A : \{0,1\}^v \to \{0,1\}^\ell \times \{0,1\}^v$,

$$\Pr_R[\mathsf{MAC}_R(M') = T' \wedge M' \neq M | (M', T') = A(\mathsf{MAC}_R(M))] \leq \epsilon,$$

where R is the uniform distribution over $\{0,1\}^n$.

Theorem 7.3.3 ([KR09b])**.** *For any message length d and tag length v, there exists an efficient family of $(\lceil \frac{d}{v} \rceil 2^{-v})$-secure* MAC*s with key length $\ell = 2v$. In particular, this* MAC *is ε-secure when $v = \log d + \log(1/\epsilon)$.*
More generally, this MAC *is also enjoys the following security guarantee, even if Eve has partial information E about its key R. Let (R, E) be any joint distribution. Then, for all attackers A_1 and A_2,*

$$\Pr_{(R,E)}[\mathsf{MAC}_R(W') = T' \wedge W' \neq W \mid W = A_1(E),\ (W', T') = A_2(\mathsf{MAC}_R(W), E)] \leq \left\lceil \frac{d}{v} \right\rceil 2^{v - H_\infty(R|E)}.$$

(In the special case when $R \equiv U_{2v}$ and independent of E, we get the original bound.)

Finally, we will also need to use any strong (k, ϵ)-extractor with with optimal entropy loss $O(\log(1/\epsilon))$. In principle, the seed length d of such an extractor can be sub-linear in the source length n, which will reduce the communication complexity of our protocol. However, since our final protocol will also use a non-malleable extractor nmExt, and our construction of the latter has linear-length seed, using a small-seed for Ext will not result in any asymptotic savings in communication complexity. In particular, we might as well use extremely simple extractors from the leftover hash lemma, having optimal entropy loss $2\log(1/\epsilon)$ and a linear-length seed.

Case of $k > n/2$ Given a security parameter s, Dodis and Wichs showed that a non-malleable extractor which extracts at least $2\log n + 2s + 4$ number of bits with error $\epsilon = 2^{-s-2}$ yields a two-round protocol for privacy amplification with optimal entropy loss. The protocol is depicted in Figure 7.1.

Alice: X	Eve: E	Bob: X
Sample random Y.		
	$Y \longrightarrow Y'$	
		Sample random W'. $R' = \mathsf{nmExt}(X; Y')$. $T' = \mathsf{MAC}_{R'}(W')$. Set final $R_B = \mathsf{Ext}(X; W')$.
	$(W, T) \longleftarrow (W', T')$	
$R = \mathsf{nmExt}(X; Y)$ **If** $T \neq \mathsf{MAC}_R(W)$ *reject.* Set final $R_A = \mathsf{Ext}(X; W)$.		

Figure 7.1: 2-round Privacy Amplification Protocol for $H_\infty(X|E) > n/2$.

Using the bound from Theorem 6.4.6 and setting $\varepsilon = 2^{-s}$ and $m = s$, we get the following theorem.

Theorem 7.3.4. *Under Conjecture 3.2.2 with constant c, for any $s > 0$ there is a polynomial time computable (k, ϵ)-non-malleable extractor with $m = s$ and $\epsilon = 2^{-s}$, as long as $k \geq n/2 + (c/2)\log n + 4s + b$, where b is a large enough constant.*

Using this theorem, we obtain the following.

Theorem 7.3.5. *Under Conjecture 3.2.2 with constant c, there is a polynomial-time two-round protocol for privacy amplification with security parameter s and entropy loss $O(\log n + s)$, when the min-entropy k of the n-bit secret satisfies $k \geq n/2 + (c/2 + 8)\log n + 8s + b$, where b is a large enough constant.*

Case of $k = \delta n$ Now we give our privacy amplification protocol for the case of arbitrarily linear min-entropy.

Now we give our privacy amplification protocol for the setting when $H_\infty(X|E) = k \geqslant \delta n$. We assume that the error ϵ we seek satisfies $2^{-\Omega(\delta n)} < \epsilon < 1/n$. In the description below, it will be convenient to introduce an "auxiliary" security parameter s. Eventually, we will set $s = \log(C/\epsilon) + O(1) = \log(1/\epsilon) + O(1)$, so that $O(C)/2^s < \epsilon$, for a sufficiently large $O(C)$ constant related to the number of "bad" events we will need to account for. We will need the following building blocks:

- Let $\mathsf{Cond} : \{0,1\}^n \to (\{0,1\}^{n'})^C$ be a rate-$(\delta \to 0.9, 2^{-s})$-somewhere-condenser. Specifically, we will use the one from Theorem 3.5.3, where $C = \mathrm{poly}(1/\delta) = O(1)$, $n' = \mathrm{poly}(\delta)n = \Omega(n)$ and $2^{-s} \gg 2^{-\Omega(\delta n)}$.
- Let $\mathsf{nmExt} : \{0,1\}^{n'} \times \{0,1\}^{d'} \to \{0,1\}^{m'}$ be a $(0.9n', 2^{-s})$-non-malleable extractor. Specifically, we will use the one from Theorem 7.3.4 (which is legal since $0.9n' \gg n'/2 + O(\log n') + 8s + O(1)$) and set the output length $m' = 4s$) (see the description of MAC below for more on m'.)
- Let $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a $(k', 2^{-s})$-extractor with optimal entropy loss $k' - m = O(s)$. Specifically, we will set $k' = k - (7C + 11)s = k - O(s)$, which means that $m = k - O(s)$ as well. We will use the notation $\mathsf{Ext}_{a..b}(X; W)$, where $1 \leqslant a \leqslant b \leqslant m$, to denote the substring of extracted bits from bit position a to bit position b. We assume the seed length $d \leqslant n$ (e.g., by using a universal hash function, but more seed-efficient extractors will work too, reducing the communication complexity).
- Let MAC be the one-time, 2^{-s}-secure MAC for d-bit messages, whose key length $\ell' = m'$ (the output length of nmExt). Using the construction from Theorem 7.3.3, we set the tag length $v' = s + \log d \leqslant 2s$ (since $d \leqslant n \leqslant 1/\epsilon \leqslant 2^s$), which means that the key length $\ell' = m' = 2v' \leqslant 4s$.
- Let lrMAC be the another one-time ("leakage-resilient") MAC for d-bit messages, but with tag length $v = 2v' \leqslant 4s$ and key length $\ell = 2v \leqslant 8s$. We will later use the second part of Theorem 7.3.3 to argue good security of this MAC even when v' bits of partial information about its key is leaked to the attacker. To not confuse the two MACs, we will use Z (instead of R) to denote the key of lrMAC and L (instead of T) to denote the tag of lrMAC.

Using the above building blocks, the protocol in given in Figure 7.2. To emphasize the presence of Eve, we will use 'prime' to denote all the protocol values seen or generated by Bob; e.g., Bob picks W'_1, but Alice sees potentially different W_1, etc. Also, for any random variable G used in describing our protocol, we use the notation $G = \perp$ to indicate that G was never assigned any value, because the party who was supposed to assign G rejected earlier. The case of final keys R_A and R_B becomes a special case of this convention.

Our protocol proceeds in $C+1$ Phases. During the first C Phases, we run C sequential copies of the two-round protocol for the entropy-rate greater than 1/2 case (see Figure 7.1), but use the derived secret X_i (output by the somewhere-condenser) instead of X during the i-th run. Intuitively, since one of the values X_i is expected to have entropy rate above 1/2, we hope that the key Z_i extracted in this Phase is secret and uniform. However, there are several complications we must resolve to complete this template into a secure protocol.

The first complication is that Eve might not choose to execute its run with Alice in a "synchronous" manner with its execution with Bob. We prevent such behavior of Eve by introducing "liveness tests", where after each Phase Alice has to prove that she participated *during* that Phase. Such tests were implicit in the original paper of Renner and Wolf [RW03], and made explicit by Khanakurthi and Reyzin [KR09b]. Each liveness test (except for the last one in Phase $C+1$, to be discussed) consists of Bob sending Alice a seed W'_i for the extractor Ext (which is anyway sent during the i-th Phase), and Alice responding with the first s bits of the extracted output. Intuitively, although Eve may choose to maul the extracted seed (which might be possible for all Phases, where the entropy rate of X_i is below 1/2), Eve cannot predict the correct output without asking Alice *something*. And since Bob does uses a new liveness test between every two Phases, this effectively forces Eve to follow a natural "synchronous" interleaving between the left and the right executions.

The second complication comes from the fact that after a "good" (rate above 1/2) Phase i is completed, the remaining phases might use low-rate sources $X_{i+1}, \ldots, X_C$. Hence, one needs a mechanism to make sure that once a good key is extracted in some *a-priori unknown* phase, good keys will be extracted in future phases as well, even if the remaining derived sources X_i have low entropy-rate. This is done by using a second message authentication code lrMAC, keyed by a value Z'_{i-1} extracted by Bob in the previous Phase $(i-1)$, to authenticated the seed W'_i sent in Phase i. The only subtlety is that Bob still sends the original MAC of W'_i, and this MAC might be correlated with the previous extracted key Z_{i-1} (especially if the Phase i uses "bad-rate" X_i). Luckily, by using the "leakage-resilient" property of our second MAC (stated in Theorem 7.3.3),

Alice: X	Eve: E	Bob: X
$(X_1, \dots X_C) = \mathsf{Cond}(X)$. Sample random Y_1.	$\boxed{\textbf{Phase } 1}$	$(X_1, \dots X_C) = \mathsf{Cond}(X)$.
	$Y_1 \longrightarrow Y_1'$	
		Sample random W_1'. $R_1' = \mathsf{nmExt}(X_1; Y_1')$. $T_1' = \mathsf{MAC}_{R_1'}(W_1')$.
	$(W_1, T_1) \longleftarrow (W_1', T_1')$	
$R_1 = \mathsf{nmExt}(X_1; Y_1)$ **If** $T_1 \neq \mathsf{MAC}_{R_1}(W_1)$ *reject.* $Z_1 = \mathsf{Ext}_{s+1..s+\ell}(X; W_1)$.		
	$\boxed{\textbf{Phases } 2\textbf{..}C}$	
For $i = 2$ **to** C Sample random Y_i. $S_{i-1} = \mathsf{Ext}_{1..s}(X; W_{i-1})$.		**For** $i = 2$ **to** C Sample random W_i'.
	$(S_{i-1}, Y_i) \longrightarrow (S_{i-1}', Y_i')$	
		If $S_{i-1}' \neq \mathsf{Ext}_{1..s}(X; W_{i-1}')$ *reject.* $Z_{i-1}' = \mathsf{Ext}_{s+1..s+\ell}(X; W_{i-1}')$. $L_i' = \mathsf{lrMAC}_{Z_{i-1}'}(W_i')$. $R_i' = \mathsf{nmExt}(X_i; Y_i')$. $T_i' = \mathsf{MAC}_{R_i'}(W_i')$.
	$(W_i, T_i, L_i) \longleftarrow (W_i', T_i', L_i')$	
If $L_i \neq \mathsf{lrMAC}_{Z_{i-1}}(W_i)$ *reject.* $R_i = \mathsf{nmExt}(X_i; Y_i)$. **If** $T_i \neq \mathsf{MAC}_{R_i}(W_i)$ *reject.* $Z_i = \mathsf{Ext}_{s+1..s+\ell}(X; W_i)$. **EndFor**		**EndFor**
	$\boxed{\textbf{Phase } C+1}$	
Re-assign $Z_C = \mathsf{Ext}_{1..m'}(X; W_C)$. Sample random W_{C+1}. $S_C = \mathsf{MAC}_{Z_C}(W_{C+1})$		$Z_C' = \mathsf{Ext}_{1..m'}(X; W_C')$
	$(S_C, W_{C+1}) \longrightarrow (S_C', W_{C+1}')$	
Set final $R_A = \mathsf{Ext}(X; W_{C+1})$.		**If** $S_C' \neq \mathsf{MAC}_{Z_C'}(W_{C+1}')$ *reject.* Set final $R_B = \mathsf{Ext}(X; W_{C+1}')$.

Figure 7.2: $(2C+1)$-round Privacy Amplification Protocol for $H_\infty(X|E) > \delta n$.

and setting the parameters accordingly), we can ensure that Z'_{i-1} has enough entropy to withstand the "leakage" of the original MAC of W'_i.

The template above already ensures the *robustness* of the protocol, if we were to extract the key Z_C (or Z'_C for Bob) derived at the end of Phase C. Unfortunately, it does not necessarily ensure that Alice outputs a *random* key (i.e., it does not guarantee the extraction property for Alice). Specifically, by making Alice's execution run faster than Bob's execution, it might be possible for Eve to make Alice successfully accept a non-random seed W_C, resulting in non-random key Z_C. Intuitively, since all the X_i's except for one might have low entropy rate, our only hope to argue security should come from the non-malleability on nmExt in the "good" Phase i. However, since Bob is behind (say, at Phase $j < i$) Alice during the good Phase i, Bob will use a wrong source X_j for the non-malleable extractor, and we cannot use the non-malleability of nmExt to argue that Eve cannot fool Alice into accepting a wrong seed W_i (and, then, wrong $W_{i+1}, \ldots, W_C$). Of course, in this case we know Bob will eventually reject, since Eve won't be able to answer the remaining liveness tests. However, Alice's key Z_C is still non-random, violating extraction.

This is the reason for introducing the last Phase $C + 1$. During this phase Alice (rather than Bob) picks the last seed W_{C+1} and uses it to extract her the final key R_A. Therefore, R_A is now guaranteed to be random. However, now we need to show how to preserve robustness and Bob's extraction. This is done by Alice sending the MAC of W_{C+1} using they key Z_C she extracted during the previous round. (We call this MAC S_C rather than T_{C+1}, since it also serves as a liveness test for Alice during Phase $(C + 1)$.) From the previous discussion, we know that, with high probability, (a) either Z_C is non-random from Eve's perspective, but then Bob will almost certainly reject (ensuring robustness and preserving Bob's extraction); or (b) $Z_C = Z'_C$ is random and secret from Eve, in which case the standard MAC security suffices to ensure both robustness and Bob's extraction.

We detail the formal proof following the above intuition in the next section, which also establishes the desired parameters promised by Theorem 7.3.1.

7.3.2.1 Security Proof of Our Protocol (Proof of Theorem 7.3.1)

We start by noticing that our protocol takes $2C + 1 = \mathrm{poly}(1/\delta) = O(1)$ rounds and achieves entropy loss $k - m = O(Cs) = O(\mathrm{poly}(1/\delta) \log(1/\epsilon))$, as claimed. Also, the protocol obviously satisfies the correctness requirement.

We will also assume that the side information E is empty (or fixed to a constant), since by Lemma 2.3.16, with probability $1 - 2^{-s}$, $H_\infty(X|E = e) \geqslant \delta n - s$, which will not affect any of our bounds. Before proving robustness and extraction properties of our protocol, we start with the following simple observation.

Lemma 7.3.6. *Let E' be Eve's view at the end of her attack (without the keys R_A and R_B used in the post-application robustness experiment). Then, for any deterministic functions f and g, we have*

$$H_\infty(f(X) \mid g(E')) \geqslant H_\infty(f(X)) - (7C-3)s$$

In particular, recalling that $k' = H_\infty(X) - (7C+11)s$, we have $H_\infty(X|g(E')) \geqslant k' + 14s$.

Proof. Clearly, if it sufficient to prove the claim for g being identity, as it gives the predictor the most information to guess $f(X)$. Also notice that, at best, if neither party rejects, Eve's view $E' = (\vec{Y}, \vec{S}, \vec{W}', \vec{T}', \vec{L}', W_{C+1})$, where $\vec{Y} = \{Y_1, \ldots, Y_C\}$, $\vec{S} = \{S_1, \ldots, S_C\}$, $\vec{W}' = \{W'_1, \ldots, W'_C\}$, $\vec{T}' = \{T'_1, \ldots, T'_C\}$ and $\vec{L}' = \{L'_2, \ldots, L'_C\}$. Since $\vec{Y}$, $\vec{W}'$ and W_{C+1} are independent of X (and, thus, $f(X)$), using Lemma 2.3.17 and recalling $|S_i| = s$ for $i < C$, $|S_C| = |T'_i| = v' \leqslant 2s$, $|L'_i| = v \leqslant 4s$, we have

$$\begin{aligned} H_\infty(f(X)|E') &\geqslant H_\infty(f(X)|(\vec{Y}, \vec{W}', W_{C+1})) - |\vec{S}| - |\vec{T}'| - |\vec{L}'| \\ &= H_\infty(f(X)) - (C-1)s - v' - Cv' - (C-1)v \\ &\geqslant H_\infty(f(X)) - (C-1)s - 2(C+1)s - (C-1)4s \\ &= H_\infty(f(X)) - (7C-3)s \end{aligned}$$

□

Next, we will argue the extraction property for Alice.

Lemma 7.3.7.

$$\Delta((R_A, E'), (\mathsf{purify}(R_A), E')) \leqslant 2^{-s+1}$$

Proof. Since $\mathsf{purify}(R_A) = R_A$ when Alice rejects (i.e., $R_A = \perp$), it is sufficient to show that Alice's key is close to uniform conditioned on Alice not rejecting, i.e.

$$\Delta((\mathsf{Ext}(X; W_{C+1}), E'), (U_m, E')) \leqslant 2^{-s+1} \tag{7.4}$$

By Lemma 7.3.6, $H_\infty(X|E') \geqslant k' + 14s$. Using Lemma 2.3.16, we get that

$$\Pr_{e' \leftarrow E'}[H_\infty(X|E' = e') \geqslant k'] \geqslant 1 - 2^{-s}.$$

Since Ext is $(k', 2^{-s})$-extractor, Equation (7.4) immediately follows the triangle inequality and the security of the extractor, by conditioning on whether or not $H_\infty(X|E' = e') \geqslant k'$. ☐

Next, we notice that in order to violate either robustness of Bob's extraction, Eve must make Bob accept (i.e., $R_B \neq \perp$). Therefore, we start by examining how Eve might cause Bob to accept. Notice, since Alice sends $C + 1$ messages, including the first and the last message, Eve can make $C + 1$ calls to Alice, which we call $Alice_1, \ldots, Alice_{C+1}$, where, for each call $Alice_i$, $1 \leqslant i \leqslant C + 1$, Eve gets back the message sent by Alice during Phase i. Additionally, Alice also computes her key R_A in response to $Alice_{C+1}$ (and gives R_A to Eve, in addition to S_C and W_{C+1}, for post-application robustness). Similarly, Eve can also make $C + 1$ calls to Bob, denoted $Bob_1, \ldots, Bob_{C+1}$, where each call Bob_i expects as input the message that Alice supposedly sent to Bob in Phase i. When $i \leqslant C$, Bob responds to such a message with his own message in Phase i. When $i = C + 1$, Bob computes his key R_B (and gives R_B to Eve for post-application robustness). Clearly, the $(C + 1)$ calls to Alice must be made in order, and the same the $(C + 1)$ calls to Bob. However, a malicious Eve might attempt to interleave the calls in some adversarial manner to make Bob accept. We say that Eve is *synchronous* if he makes his oracle calls in the ("synchronous") order $Alice_1, Bob_1, Alice_2, Bob_2, \ldots, Alice_{C+1}, Bob_{C+1}$. We notice that, without loss of generality, Eve always starts by making the $Alice_1()$ call, since this call has no inputs Eve needs to provide. Namely, Eve must as well find out the values Y_1 first, and, if she wants, delay using this value until later. With this convention in mind, we show that Eve *must be synchronous in order to make Bob accept.*

Lemma 7.3.8.

$$\Pr[R_B \neq \perp \wedge \text{Eve is not synchronous}] \leqslant \frac{3C}{2^s} \tag{7.5}$$

Proof. As we said, we assume Eve always makes the call $Alice_1$ first. After that, Eve makes $C + 1$ calls to Bob and C calls to Alice in some order. We claim that for every $1 \leqslant i \leqslant C$, Eve must make at least one call to some $Alice_j$ in between two successive calls Bob_i and Bob_{i+1}. If we show this (with total failure probability from Equation (7.5)), Eve must be synchronous, since the synchronous scheduling is the only scheduling that starts with $Alice_1$ and has a fresh call to Alice between Bob_1 and Bob_2, Bob_2 and Bob_3, …, Bob_C and Bob_{C+1}.

Given $1 \leqslant i \leqslant C$, let F_i denote the event that Eve's scheduling of calls made two successive calls Bob_i and Bob_{i+1} without a fresh call to some $Alice_j$, and Bob does not reject after the call Bob_{i+1}. We claim that $\Pr[F_i] \leqslant 3/2^s$. The bound from Equation (7.5) then follows by simply taking the union bound over all i. We consider two cases:

Case 1: $1 \leqslant i < C$. In this case, after the call $Bob_i(\cdot, \cdot)$ is made, Bob picks a fresh seed W'_i, and returns it as part of the output. By assumption, Eve immediately makes a call $Bob_{i+1}(S'_i, \cdot)$, without any intermediate calls to Alice, and Bob rejects if $S'_i \neq \mathsf{Ext}_{1\ldots s}(X; W'_i)$. Thus, to establish our claim it is sufficient to show that $\Pr[S'_i \neq \mathsf{Ext}_{1\ldots s}(X; W'_i)] \leqslant 3/2^s$. Intuitively, the bound on $\Pr[F_i]$ now follows from the fact that Ext is a good (strong) $(k', 2^{-s})$-extractor, since, conditioned on Eve's information so far, the s-bit value $\mathsf{Ext}_{1\ldots s}(X; W'_i)$ is 2^{-s}-close to random, and, hence, cannot be predicted with probability better that $2^{-s} + 2^{-s}$ (the third 2^{-s} is due to Lemma 2.3.16, since our extractor is assumed to be worst case, and is not needed for universal hash function extractors [DORS08]).

A bit more formally, let E_i be Eve's view before the call to Bob_i is made, and $E'_i = (E_i, W'_i, T'_i, L'_i)$ be Eve's view after the call to Bob_i is made. We notice that E'_i is a deterministic function of $E^*_i = (E_i, Z'_{i-1}, R'_i)$ and W'_i, since $L'_i = \mathsf{lrMAC}_{Z'_{i-1}}(W'_i)$ and $T'_i = \mathsf{MAC}_{R'_i}(W_i)$. Moreover, W'_i is freshly chosen even conditioned on E^*_i. Thus, $\Pr[F_i] \leqslant \Pr[Eve(E^*_i, W'_i) = \mathsf{Ext}_{1..s}(X; W'_i)]$, where W'_i is independent of (X, E^*_i). We also note that $H_\infty(X|E_i)) \geqslant k' + 14s$, by Lemma 7.3.6, since E_i is a function of E'. Thus, $H_\infty(X|E^*_i) \geqslant H_\infty(X|E_i) - |Z'_{i-1}| - |R'_i| \geqslant k' + 14s - 4s - 8s = k' + 2s$. Using Lemma 2.3.16, $\Pr_{e^*_i}[H_\infty(X|E^*_i = e^*_i) \geqslant k'] \geqslant 1 - 2^{-s}$, and the rest follows from the fact that in this case $(W'_i, \mathsf{Ext}_{1..s}(X; W'_i))$ is 2^{-s}-close to (W'_i, U_s), as mentioned earlier.

Case 2: $i = C$. In this case, after the call $Bob_C(\cdot, \cdot)$ is made, Bob picks a fresh seed W'_C, and returns it as part of the output. By assumption, Eve immediately makes a call $Bob_{i+1}(S'_C, W'_{C+1})$, without any intermediate calls to Alice, and Bob rejects if $S'_C \neq \mathsf{MAC}_{Z'_C}(W'_{C+1})$, where $Z'_C = \mathsf{Ext}_{1\ldots m'}(X; W'_i)$. Thus, to establish our claim it is sufficient to show that $\Pr[S'_C \neq \mathsf{MAC}_{Z'_C}(W'_{C+1})] \leqslant 3/2^s$. Completely similar to the previous case, we can argue that the value Z'_C used by Bob is 2^{1-s}-close to $U_{m'}$ conditioned on Eve's view so far. Moreover, the 2^{-s}-security of MAC ensures that, when the key Z'_C is truly uniform, Eve cannot successfully forge a valid tag $\mathsf{MAC}_{Z'_C}(W'_{C+1})$ of any (even adversarially chosen) message W'_{C+1} with probability greater than 2^{-s}, completing the proof of this case as well. □

Therefore, from now on *we assume that Eve is indeed synchronous.* Moreover, since Eve must make Bob accept, we assume Eve finishes the both left and right execution (with the last call to Bob_{C+1}, hoping that Bob will accept). Also, by Theorem 3.5.3, we have that $(X_1, \cdots, X_C)$ is $2^{-\Omega(\delta n)}$-close to a somewhere rate-0.9 source. Thus, we will ignore the error and think of $(X_1, \cdots, X_C)$ as indeed being a somewhere rate-0.9 source, as it only adds $2^{-\Omega(\delta n)} \ll 2^{-s}$ to the total probability of error. Also, it is sufficient to show robustness and extraction for Bob properties assuming that $(X_1, \cdots, X_C)$ is an *elementary* somewhere rate-0.9 source, since $(X_1, \cdots, X_C)$

is a convex combination of elementary somewhere rate-0.9 sources. Hence, from now one we assume that some "good" index $1 \leqslant g \leqslant C$ satisfies $H_\infty(X_g) \geqslant 0.9n'$. We stress that this index g is not known to Alice and Bob, but could be known to Eve. We start by showing that, with high probability, Eve must forward a correct seed $W_g = W'_g$ to Alice in the "good" Phase g.

Lemma 7.3.9. *Assuming Eve is synchronous,*

$$\Pr[R_B \neq \perp \wedge W_g \neq W'_g] \leqslant \frac{3}{2^s} \tag{7.6}$$

Proof. Let E'_{g-1} be Eve's view before the call to $Alice_g$. Note that X_g is a deterministic function of X, and $(E'_{g-1}, S_{g-1}, L'_g)$ is a deterministic function of Eve's transcript E'. Thus, by Lemma 7.3.6,

$$\begin{aligned} H_\infty(X_g|(E'_{g-1}, S_{g-1}, L'_g)) &\geqslant H_\infty(X_g) - (7C-3)s \\ &\geqslant 0.9n' - (7C-3)s \\ &= (n'/2 + O(\log n') + 8s + O(1)) + s - (0.4n' - O(Cs + \log n)) \\ &\geqslant (n'/2 + O(\log n') + 8s + O(1)) + s \end{aligned}$$

where the last inequality follows since $n' = \mathrm{poly}(1/\delta)n \gg O(Cs + \log n)$. By Lemma 2.3.16, with probability $1 - 2^{-s}$ over the fixings of E'_{g-1}, S_{g-1}, L'_g, the min-entropy of X_g conditioned on these fixings is at least $n'/2 + O(\log n') + 8s + O(1)$. Notice also that the seed Y_g is independent of E'_{g-1}, S_{g-1}, L'_g. Moreover, for the argument in this lemma, we will "prematurely" give Eve the value L'_g already after the call to $Alice_g$ (instead of waiting to get it from the call to Bob_g). Let us now summarize the resulting task of Eve in order to make $W_g \neq W'_g$, and argue that Eve is unlikely to succeed.

After the call to $Alice_g$, with high probability the min-entropy of X_g conditioned on Eve's view is greater than $n'/2+O(\log n')+8s+O(1)$, so that we can apply the non-malleability guarantees of nmExt given by Theorem 7.3.4. Alice then picks a random seed Y_g for nmExt and gives it to Eve. (Synchronous) Eve then forwards some related seed Y'_g to Bob_g (and another value S'_{g-1} that we ignore here), and learns some message W'_g and the tag T'_g of W'_g, under key $R'_g = \mathsf{nmExt}(X_g; Y'_g)$ (recall, we assume Eve already knows L'_g from before). To win the game, Eve must produce a value $W_g \neq W'_g$ and a valid tag T_g of W_g under the original key $R_g = \mathsf{nmExt}(X_g; Y_g)$.

We consider two cases. First, if Eve sets $Y'_g = Y_g$, then $R_g = R'_g$ is 2^{-s}-close to uniform by Theorem 7.3.4. Now, if R_g was truly uniform, by the one-time unforgeability of MAC, the probability that Eve can produce a valid tag T_g of a new message $W_g \neq W'_g$ is at most 2^{-s}. Hence, Eve cannot succeed with probability more that 2^{-s+1} even with R_g which is 2^{-s}-close to uniform, implying the bound stated in the lemma (since we also lost 2^{-s} by using Lemma 2.3.16 at the beginning).

On the other hand, if Eve makes $Y'_g \neq Y_g$, Theorem 7.3.4 implies that $\Delta((R_g, R'_g), (U_{m'}, R'_g)) \leqslant 2^{-s}$. Thus, the tag T'_g under R'_g is almost completely useless in predicting the tag of W_g under (nearly random) R_g. Therefore, by 2^{-s} security of MAC, once again the probability that Eve can successfully change W'_g without being detected is at most 2^{-s+1} (giving again the final bound $3/2^s$). □

Now, we want to show that, once Eve forwards correct $W_g = W'_g$ to Alice in Phase g, Eve must forward correct seeds $W_i = W'_i$ in all future phases $i = g+1, \ldots, C$. We start by the following observation, which states that the derived keys Z'_{i-1} used by Bob in lrMAC look random to Eve *whenever Eve forwards a correct key* $W_{i-1} = W'_{i-1}$ to Alice.

Lemma 7.3.10. *Assume Eve is synchronous,* $2 \leqslant i \leqslant C$, *and Eve forwards a correct value* $W_{i-1} = W'_{i-1}$ *to Alice during her call to* $Alice_i$. *Also, let* E_i *be Eve's view after the call to* $Alice_i(W_{i-1}, \cdot, \cdot)$. *Then*

$$\Delta((Z'_{i-1}, E_i), (U_\ell, E_i)) \leqslant \frac{3}{2^s} \tag{7.7}$$

Proof. Notice that $E_i = (E_{i-1}, W'_{i-1}, T'_{i-1}, L'_{i-1}, S_{i-1}, Y_i)$, where E_{i-1} is Eve's view after the call to $Alice_{i-1}$. For convenience, we replace the two tags T'_{i-1}, L'_{i-1} of W'_{i-1} by the corresponding MAC keys R'_{i-1}, Z'_{i-2}, respectively, since this gives Eve only more information. Also, since $W_{i-1} = W'_{i-1}$, we know that the value $S_{i-1} = \mathsf{Ext}_{1..s}(X; W_{i-1}) = \mathsf{Ext}_{1..s}(X; W'_{i-1})$. Recalling that $Z'_{i-1} = \mathsf{Ext}_{s+1..s+\ell}(X; W'_{i-1})$, and denoting "side information" by $E^*_i = (E_{i-1}, R'_{i-1}, Z'_{i-2}, Y_i)$, it is enough to argue

$$\Delta((E^*_i, W'_{i-1}, \mathsf{Ext}_{1..s}(X; W'_{i-1}), \mathsf{Ext}_{s+1..s+\ell}(X; W'_{i-1}))\ ,\ (E^*_i, W'_{i-1}, \mathsf{Ext}_{1..s}(X; W'_{i-1}), U_\ell)) \leqslant \frac{3}{2^s} \tag{7.8}$$

where we notice that E^*_i is *independent* of the choice of random W'_{i-1}. In turn, Equation (7.8) follows from the fact that Ext is $(k', 2^{-s})$-extractor provided we can show that $H_\infty(X|E^*_i) \geqslant k+s$. Indeed, the first error term 2^{-s} comes from Lemma 2.3.16 to argue that $\Pr_{e^*_i}[H_\infty(X|E^*_i = e^*_i) \geqslant k] \geqslant 1-2^{-s}$, and the other two error terms follow from the triangle inequality and the security of the extractor (first time applies on the first s extracted bits, and then on all $s+\ell$ extracted bits).

So we show that $H_\infty(X|E_i^*) \geqslant k+s$.

$$\begin{aligned} H_\infty(X|E_i^*) &= H_\infty(X|E_{i-1}, R'_{i-1}, Z'_{i-2}, Y_i) \\ &\geqslant H_\infty(X|E_{i-1}, Y_i) - |R'_{i-1}| - |Z'_{i-2}| \\ &= H_\infty(X|E_{i-1}) - m' - \ell \\ &\geqslant k' + 14s - 4s - 8s \\ &= k' + 2s \end{aligned}$$

where the first inequality used Lemma 2.3.17, the second equality used the fact that Y_i is independent of (X, E_{i-1}), and the second inequality used Lemma 7.3.6, since E_{i-1} is deterministic function of E'. □

Next, we use Lemma 7.3.9 and Lemma 7.3.10 to show that, with high probability, Alice and Bob must agree on the same key $Z_C = Z'_C$ when they reach the last Phase $(C+1)$.

Lemma 7.3.11. *Assuming Eve is synchronous,*

$$\Pr[R_B \neq \perp \wedge Z_C \neq Z'_C] \leqslant \frac{4C}{2^s} \tag{7.9}$$

Proof. Since $Z_C = \mathsf{Ext}_{1\ldots m'}(X; W_C)$ and $Z'_C = \mathsf{Ext}_{1\ldots m'}(X; W'_C)$, we get

$$\begin{aligned} \Pr[R_B \neq \perp \wedge Z_C \neq Z'_C] &\leqslant \Pr[R_B \neq \perp \wedge W_C \neq W'_C] \\ &\leqslant \Pr[R_B \neq \perp \wedge W_g \neq W'_g] + \sum_{i=g+1}^{C} \Pr[R_B \neq \perp \wedge W_{i-1} = W'_{i-1} \wedge \ W_i \neq W'_i] \\ &\leqslant \frac{3}{2^s} + (C-1) \cdot \max_{i>g} Pr[R_B \neq \perp \wedge W_{i-1} = W'_{i-1} \wedge \ W_i \neq W'_i] \end{aligned}$$

where the second inequality states that in order for $W_C \neq W'_C$, either we must already have $W_g \neq W'_g$ (which, by Lemma 7.3.9, happens with probability at most $3/2^s$), or there must be some initial Phase $i > g$ where $W_{i-1} = W'_{i-1}$ still, but $W_i \neq W'_i$. Thus, to establish Equation (7.9), it suffices to show that, for any Phase $g < i \leqslant C$,

$$Pr[R_B \neq \perp \wedge W_{i-1} = W'_{i-1} \wedge \ W_i \neq W'_i] \leqslant \frac{4}{2^s} \tag{7.10}$$

Intuitively, this property follows from the unforgeability of lrMAC, since Eve must be able to forge a valid tag L_i of $W_i \neq W'_i$, given a valid tag of W'_i (under the same $Z_{i-1} = Z'_{i-1}$ since $W_{i-1} =$

W'_{i-1}). The subtlety comes from the fact that Eve also learns the v'-bit value $T'_i = \mathsf{MAC}_{R'_i}(W'_i)$, which could conceivably be correlated with the key Z'_{i-1} for lrMAC. Luckily, since the tag length v of lrMAC is twice as large as v', Theorem 7.3.3 states that lrMAC is still unforgeable despite this potential "key leakage".

More formally, if Eve forwards a correct value $W_{i-1} = W'_{i-1}$, both Alice and Bob use the same key $Z'_{i-1} = Z_{i-1} = \mathsf{Ext}_{s+1..s+\ell}(X; W'_{i-1})$ to lrMAC during Phase i. Moreover, by Lemma 7.3.10, we know that this key Z_{i-1} looks random to Eve right before the call to Bob_i: $\Delta((Z'_{i-1}, E_i), (U_\ell, E_i)) \leqslant \frac{3}{2^s}$, where E_i is Eve's view after the call to $Alice_i(W_{i-1}, \cdot, \cdot)$. After the call to Bob_i, Eve learns the tag L'_i of W'_i, and also a v'-bit value T', which, for all we know, might be correlated with the key Z'_{i-1}. Therefore, to argue the bound in Equation (7.10), it is sufficient to argue that Eve can succeed with probability at most 2^{-s} in the following "truncated" experiment. After the call to $Alice_i$, the actual key Z'_{i-1} is replaced by uniform $Z^*_{i-1} \leftarrow U_\ell$. Then a random message W'_i is chosen, its tag L'_i is given to Eve, and Eve is also allowed to obtain arbitrary v' bits of information about Z^*_{i-1}. Eve succeeds if she can produce a valid tag L_i (under Z^*_{i-1}) of a different message $W_i \neq W'_i$. This is precisely the precondition of the second part of Theorem 7.3.3, where $H_\infty(Z^*_{i-1}|E) \geqslant \ell - v' = 2v - v/2 = 3v/2$. Hence, Eve's probability of success is at most $d2^{v-3v/2} = d2^{-v/2} = d2^{-v'} \leqslant 2^{-s}$. □

We need one more observation before we can finally argue Bob's extraction and robustness. Namely, at the end of Phase C, (synchronous) Eve has almost no information about the authentication key Z'_C used by the Bob (and Alice, by Lemma 7.3.11) in the final Phase $C+1$.

Lemma 7.3.12. *Assume Eve is synchronous, and let E'_C be Eve's view after the call to Bob_C. Then*

$$\Delta((Z'_C, E'_C \mid R_B \neq \perp), (U_{m'}, E'_C \mid R_B \neq \perp)) \leqslant \frac{2}{2^s} \tag{7.11}$$

Additionally, $H_\infty(X|(E'_C, Z'_C)) \geqslant k' + 10s$.

Proof. The proof is similar to, but simpler than, the proof of Lemma 7.3.10. We notice that $E'_C = (E_C, W'_C, T'_C, L'_C)$, where E_C is Eve's view after the call to $Alice_C$. For convenience, we replace the two tags T'_C, L'_C of W'_C by the corresponding MAC keys R'_C, Z'_{C-1}, respectively, since this gives Eve only more information. Recalling that $Z'_C = \mathsf{Ext}_{1..m'}(X; W'_C)$, and denoting "side information" by $E^*_C = (E_C, R'_C, Z'_{C-1})$, it is enough to argue

$$\Delta((E^*_C, W'_C, \mathsf{Ext}_{1..m'}(X; W'_C))\ ,\ (E^*_C, W'_C, U_{m'})) \leqslant \frac{2}{2^s} \tag{7.12}$$

where we notice that E^*_C is *independent* of the choice of random W'_C. In turn, Equation (7.12) follows from the fact that Ext is $(k', 2^{-s})$-extractor provided we can show that $H_\infty(X|E^*_C) \geqslant k+s$, where the extra error term 2^{-s} comes from Lemma 2.3.16 to argue that $\Pr_{e^*_C}[H_\infty(X|E^*_C = e^*_C) \geqslant k] \geqslant 1-2^{-s}$.

So we show that $H_\infty(X|E^*_C) \geqslant k+s$.

$$\begin{aligned} H_\infty(X|E^*_C) &= H_\infty(X|E_C, R'_C, Z'_{C-1}) \\ &\geqslant H_\infty(X|E_C) - |R'_C| - |Z'_{C-2}| \\ &= H_\infty(X|E_C) - m' - \ell \\ &\geqslant k' + 14s - 4s - 8s \\ &= k' + 2s \end{aligned}$$

where the first inequality used Lemma 2.3.17, and the second inequality used Lemma 7.3.6, since E_C is deterministic function of E'.

The final claim $H_\infty(X|(E'_C, Z'_C)) \geqslant k' + 10s$ follows from Lemma 2.3.17 and fact that $H_\infty(X|E'_C) \geqslant k' + 14s$ (Lemma 7.3.6) and $|Z'_C| = m' \leqslant 4s$. □

Lemma 7.3.11 and Lemma 7.3.12 imply that, in order for the synchronous Eve to have a non-trivial chance to make Bob accept, at the end of Phase C Alice and Bob must agree on a key $Z_C = Z'_C$ which looks random to Eve. Moreover, X still has a lot of entropy given Z'_C and Eve's view so far. Thus, to show both (post-application) robustness and extraction for Bob, it is sufficient to show these properties for a very simply one-round key agreement protocol, which emulates the final Phase $(C+1)$ of our protocol with Alice and Bob sharing a key $Z_C = Z'_C$ which is assumed to be random and independent from Eve's view so far. We start with post-application robustness.

Post-Application Robustness: To cause Bob output a different key than Alice in Phase $(C+1)$, Eve must modify Alice seed W_{C+1} to $W'_{C+1} \neq W_{C+1}$, and then forge a valid tag S'_C of W'_{C+1} under the shared key $Z_C = Z'_C$. For pre-application robustness, the unforgeability of MAC immediately implies that Eve's probability of success is at most 2^{-s}. However, in the post-application robustness experiment, Eve is additionally given Alice's final key $R_A = \mathsf{Ext}(X; W_{C+1})$. Luckily, since X has more than $k' + s$ bits of min-entropy *even conditioned of the MAC key* Z_C, security of the extractor implies that that the joint distribution of Z_C and R_A looks like a pair of independent random strings. In particular, Eve still cannot change the value of the seed W_{C+1} in Phase $(C+1)$, despite being additionally given Alice's key R_A, since that key looks random and independent of the MAC key $Z_C = Z'_C$.

Extraction for Bob: We just argued (pre-application) robustness of our protocol, which — for synchronous Eve — means that if Bob does not reject, then, with high probability, he outputs the same key $R_B = \mathsf{Ext}(X; W'_{C+1})$ as Alice's key $R_A = \mathsf{Ext}(X; W_{C+1})$. Thus, Bob's extraction is implied by Alice's extraction, which was already argued in Lemma 7.3.7. Alternatively, Alice's extraction can be seen directly, as she chooses a fresh seed W_{C+1} and $H_\infty(X|E'_C, Z_C) \geqslant k' + 10s$.

This concludes the proof of Theorem 7.3.1.

Bibliography

[AFWZ95] Noga Alon, Uriel Feige, Avi Wigderson, and David Zuckerman. Derandomized graph products. *Computational Complexity*, 5(1):60–75, 1995.

[AN93] Noga Alon and Moni Naor. Coin-flipping games immune against linear-sized coalitions. *SIAM Journal on Computing*, 22(2):403–417, April 1993.

[BIW04] Boaz Barak, R. Impagliazzo, and Avi Wigderson. Extracting randomness using few independent sources. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 384–393, 2004.

[BKS$^+$05] Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: New constructions of condensers, Ramsey graphs, dispersers, and extractors. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 1–10, 2005.

[BKS$^+$10] Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of reed-muller codes. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, 2010.

[BN00] Ravi B. Boppana and Babu O. Narayanan. Perfect-information leader election with optimal resilience. *SIAM J. Comput*, 29(4):1304–1320, 2000.

[BO83] Michael Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols (extended abstract). In *Proceedings of the Second Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, pages 27–30, Montreal, Quebec, Canada, 17–19 August 1983.

[BOL78] Michael Ben-Or and Nathan Linial. Collective coin flipping. *Randomness and Computation*, 1978.

[Bou05] Jean Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 1:1–32, 2005.

[Bou07] Jean Bourgain. On the construction of affine-source extractors. *Geometric and Functional Analysis*, 2007.

[BRSW06] Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2 source dispersers for $n^{o(1)}$ entropy and Ramsey graphs beating the Frankl-Wilson construction. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, 2006.

[BSK09] Eli Ben-Sasson and Swastik Kopparty. Affine dispersers from subspace polynomials. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 2009.

[BT85] G. Bracha and S. Toueg. Asynchronous consensus and broadcast protocols. *Journal of the ACM*, 32:824–840, 1985.

[Can97] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *CRYPTO '97*, 1997.

[CC85] Benny Chor and Brian A. Coan. A simple and efficient randomized byzantine agreement algorithm. *IEEE Transactions on Software Engineering*, 11(6):531–539, June 1985.

[CG88] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

[CKOR10] Nishanth Chandran, Bhavana Kanukurthi, Rafail Ostrovsky, and Leonid Reyzin. Privacy amplification with asymptotically optimal entropy loss. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, 2010.

[CL95] Jason Cooper and Nathan Linial. Fast perfect-information leader-election protocols with linear immunity. *Combinatorica*, 15(3):319–332, 1995.

[CPS07] R. Canetti, R. Pass, and A. Shelat. How to use an imperfect reference string. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 249–259, 2007.

[DKRS06] Yevgeniy Dodis, Jonathan Katz, Leonid Reyzin, and Adam Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. In *CRYPTO*, 2006.

[DKSS09] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009.

[DLWZ11] Yevgeniy Dodis, Xin Li, Trevor D. Wooley, and David Zuckerman. Privacy amplification and non-malleable extractors via character sums. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, 2011.

[DO03] Yevgeniy Dodis and R. Oliveira. On extracting private randomness over a public channel. In *RANDOM 2003*, pages 196–205, 2003.

[DOPS04] Yevgeniy Dodis, Shien Jin Ong, Manoj Prabhakaran, and Amit Sahai. On the (im)possibility of cryptography with imperfect randomness. In *FOCS04*, pages 196–205, 2004.

[DORS08] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38:97–139, 2008.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008.

[DS02] Yevgeniy Dodis and Joel Spencer. On the (non)universality of the one-time pad. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, page 376. IEEE Computer Society, 2002.

[DW08] Zeev Dvir and Avi Wigderson. Kakeya sets, new mergers and old extractors. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008.

[DW09] Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, page 601610, 2009.

[Fei99] Uriel Feige. Noncryptographic selection protocols. In IEEE, editor, *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 142–152. IEEE Computer Society Press, 1999.

[Gil98] David Gillman. A chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27(4):1203–1220, August 1998.

[GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 25–32, 1989.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, 1987.

[Gol95] Oded Goldreich. Three xor-lemmas - an exposition. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(56), 1995.

[Gol09] Oded Goldreich. A candidate counterexample to the easy cylinders conjecture. *ECCC Report TR09-028*, 2009.

[GPV06] Shafi Goldwasser, Elan Pavlov, and Vinod Vaikuntanathan. Fault-tolerant distributed computing in full-information networks. In *FOCS*, pages 15–26. IEEE Computer Society, 2006.

[GR05] Ariel Gabizon and Ran Raz. Deterministic extractors for affine sources over large fields. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, 2005.

[GRS04] Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic extractors for bit-fixing sources by obtaining an independent seed. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, 2004.

[GSV05] Shafi Goldwasser, Madhu Sudan, and Vinod Vaikuntanathan. Distributed computing with imperfect randomness. In *DISC 2005*, 2005.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM*, 56(4), 2009.

[HB78] D.R. Heath-Brown. Almost-primes in arithmetic progressions and short intervals. *Math. Proc. Cambridge Philos. Soc.*, 83:357–375, 1978.

[HB92] D.R. Heath-Brown. Zero-free regions for Dirichlet L-functions, and the least prime in an arithmetic progression. *Proc. London Math. Soc.*, 64:265–338, 1992.

[HS10] Elad Haramaty and Amir Shpilka. On the structure of cubic and quartic polynomials. Technical Report TR09-080, ECCC, 20010.

[ILL89] R. Impagliazzo, L. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 12–24, 1989.

[IW97] R. Impagliazzo and A. Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 220–229, 1997.

[Jus72] J. Justensen. A class of constructive asymptotically good algebraic codes. *IEEE Trans. Info. Theory.*, 18:652656, 1972.

[KKK$^+$08] B. Kapron, D. Kempe, V. King, J. Saia, and V. Sanwalani. Fast asynchronous Byzantine agreement and leader election with full information. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1038–1047, 2008.

[KLR09] Yael Kalai, Xin Li, and Anup Rao. 2-source extractors under computational assumptions and cryptography with defective randomness. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009.

[KLRZ08] Yael Tauman Kalai, Xin Li, Anup Rao, and David Zuckerman. Network extractor protocols. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008.

[KR08] B. Kanukurthi and L. Reyzin. An improved robust fuzzy extractor. In *SCN*, pages 156–171, 2008.

[KR09a] B. Kanukurthi and L. Reyzin. Key agreement from close secrets over unsecured channels. In *EUROCRYPT*, 2009.

[KR09b] B. Kanukurthi and L. Reyzin. Key agreement from close secrets over unsecured channels. In *EUROCRYPT*, pages 206–223, 2009.

[KRVZ06] Jesse Kamp, Anup Rao, Salil Vadhan, and David Zuckerman. Deterministic extractors for small space sources. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, 2006.

[KZ07] Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM Journal on Computing*, 36(5), 2007.

[LPS88] Alexander Lubotzky, R. Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[LRVW03] C. J. Lu, Omer Reingold, Salil Vadhan, and Avi Wigderson. Extractors: Optimal up to constant factors. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pages 602–611, 2003.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[MW97a] Ueli Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In *Advances in Cryptology — CRYPTO '97*, volume 1294, pages 307–321, August 1997.

[MW97b] Ueli M. Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In *CRYPTO '97*, 1997.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, October 1994.

[ORV94] Rafail Ostrovsky, Sridhar Rajagopalan, and Umesh Vazirani. Simple and efficient leader election in the full information model. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 234–242, Montréal, Québec, Canada, 23–25 May 1994.

[Pie09] Krzysztof Pietrzak. A leakage-resilient mode of operation. In *Advances in Cryptology — EUROCRYPT 2009*, 2009.

[Pip87] Nicholas Pippenger. Sorting and selecting in rounds. *SIAM Journal on Computing*, 16(6):1032–1038, 1987.

[Rab83] Michael O. Rabin. Randomized Byzantine generals. In *24th Annual Symposium on Foundations of Computer Science*, pages 403–409. IEEE, 7–9 November 1983.

[Rao06] Anup Rao. Extractors for a constant number of polynomially small min-entropy independent sources. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, 2006.

[Rao07a] A. Rao. An exposition of Bourgain's 2-source extractor. Technical Report TR07-034, Electronic Colloquium on Computational Complexity, 2007.

[Rao07b] Anup Rao. *Randomness Extractors for Independent Sources and Applications.* PhD thesis, UT-Austin, 2007.

[Rao09] Anup Rao. Extractors for low-weight affine sources. In *Proc. of the 24th CCC*, 2009.

[Raz05] Ran Raz. Extractors with weak random seeds. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 11–20, 2005.

[RRV99] Ran Raz, Omer Reingold, and Salil Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 149–158, 1999.

[RRV02] Ran Raz, Omer Reingold, and Salil Vadhan. Extracting all the randomness and reducing the error in trevisan's extractors. *JCSS*, 65(1):97–128, 2002.

[RTS97] Jaikumar Radhakrishnan and Amnon Ta-Shma. Tight bounds for depth-two superconcentrators. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 585–594, 1997.

[RW03] R. Renner and S. Wolf. Unconditional authenticity and privacy from an arbitrarily weak secret. In *CRYPTO*, pages 78–95, 2003.

[RZ01] Alexander Russell and David Zuckerman. Perfect information leader election in $\log^* n + O(1)$ rounds. *Journal of Computer and System Sciences*, 63(4):612–626, 2001.

[RZ08] Anup Rao and David Zuckerman. Extractors for 3 uneven length sources. In *Random 2008*, 2008.

[Sak89] Michael Saks. A robust noncryptographic protocol for collective coin flipping. *SIAM Journal on Discrete Mathematics*, 2(2):240–244, May 1989.

[Sch76] W.M. Schmidt. *Equations over Finite Fields. An Elementary Approach*, volume 536 of *Lecture Notes in Mathematics*. Springer-Verlag, 1976.

[SU05] Ronen Shaltiel and Chris Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *Journal of the ACM*, 52:172–216, 2005.

[SV86] Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33:75–87, 1986.

[Ter99] A. Terras. *Fourier Analysis on Finite Groups and Applications.* Cambridge University Press, 1999.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, pages 860–879, 2001.

[TUZ01] Amnon Ta-Shma, Chris Umans, and David Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 143–152, 2001.

[TV00] Luca Trevisan and Salil Vadhan. Extracting randomness from samplable distributions. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, 2000.

[TZ04] Amnon Ta-Shma and David Zuckerman. Extractor codes. *IEEE Transactions on Information Theory*, 50, 2004.

[Uma05] Christopher Umans. Reconstructive dispersers and hitting set generators. In *APPROX-RANDOM*, volume 3624, pages 460–471, 2005.

[Vaz85] Umesh Vazirani. Towards a strong communication complexity theory or generating quasi-random sequences from two communicating slightly-random sources (extended abstract). In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, 1985.

[Vin07] Le Anh Vinh. Szemeredi-trotter type theorem and sum-product estimate in finite fields. In *Arxiv*, 2007.

[VW08] Emanuele Viola and Avi Wigderson. Norms, xor lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(7):137–168, 2008.

[Wee05] Hoeteck Wee. On obfuscating point functions. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, 2005.

[Wei48] A. Weil. On some exponential sums. *Proceedings of the National Academy of Sciences*, 34:204–207, 1948.

[Xyl09] T. Xylouris. On Linnik's constant. Technical report, Arxiv, 2009. arXiv:0906.2749.

[Yeh10] Amir Yehudayoff. Affine extractors over prime fields. *Manuscript*, 2010.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, 11:345–367, 1997.

[Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Theory of Computing*, pages 103–128, 2007.