The Dissertation Committee for Anup Rao
certifies that this is the approved version of the following dissertation:

Randomness Extractors for Independent Sources and Applications

Committee:

David Zuckerman, Supervisor

Boaz Barak

Anna Gal

Adam Klivans

Charles Gregory Plaxton

Randomness Extractors for Independent Sources and Applications

by

Anup Rao, B.S.;B.S.

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

May 2007

For Austin

Acknowledgments

Many people have contributed to the work that appears in this thesis. The quality of my ideas and my productivity as a researcher were directly related to the kinds of people I have interacted with, both professionally and socially. I'd like to thank my co-authors for the good work they put into the research that this thesis is about. Most of the work that I did was only possible because of useful conversations that I had with others, so I'd like to thank everyone who played the role of sounding board for my ideas.

Perhaps the hardest challenge for me during my graduate career was finding ways to motivate myself when I wasn't making much progress. It was in times like these that my advisor had the biggest impact on me. David Zuckerman's support and encouragement gave me the confidence I needed to persist in thinking about the problems I was most interested in. I suspect that I would have settled for a less challenging research goal if it wasn't for his efforts. I have tried to emulate David's attitude to research, and I'd like to think that some of it has rubbed off on me over the years.

Apart from David, I have had many mentors during my time as a graduate student and have often been surprised at the effort that many of them put in. A major event during my student life was the arrival of Adam Klivans at UT Austin. He brought with him an infectious enthusiasm for research that immediately transformed the department at UT for me, making it much more enjoyable, both professionally and socially. I have been lucky to have had the chance to work with Avi Wigderson as a student. Avi has taught me a lot so far and I hope to learn even more from him in the future. Boaz Barak, Jaydev Misra and many others liberally doled out advice that made me wiser, or at the very least made me feel wiser, than I could have on my own.

I have been very fortunate to have spent time at several institutions other than UT Austin as a graduate student. Salil Vadhan and Madhu Sudan gave me the opportunity to visit and

participate in activities at Harvard and MIT. Boaz Barak organized a trip to Princeton that led to a significant part of the work in this thesis. Avi Wigderson set up several visits to the Institute for Advanced Study. He and Ran Raz arranged a visit to the Weizmann Institute of Science, where I was able to make good friends and develop relationships that I'm sure will last for a long time. I'd like to thank all of them for enriching my student life.

I owe a lot to my teachers and several graduate students in theoretical computer science at Georgia Tech, who got me interested in mathematics and theoretical computer science as an undergraduate. Thanks to Yan Ding for giving me my first reading assignment — a paper on extractors. It was the beginning of a research path that has so far resulted in the work that appears in this thesis.

Finally, I would like to thank my friends, my parents, my brother and those people who have been closest to me during these years. My friends made sure that I'd have fun, regardless of how well my work was going. My family's unwavering, rock-solid, (blind) faith in my ability to achieve helped drive me. Their advice and support has been invaluable to me.

Anup Rao

The University of Texas at Austin
May 2007

Randomness Extractors for Independent Sources and Applications

Publication No. ____________________

Anup Rao, Ph.D.
The University of Texas at Austin, 2007

Supervisor: David Zuckerman

The use of randomized algorithms and protocols is ubiquitous in computer science. Randomized solutions are typically faster and simpler than deterministic ones for the same problem. In addition, many computational problems (for example in cryptography and distributed computing) are impossible to solve without access to randomness.

In computer science, access to randomness is usually modeled as access to a string of uncorrelated uniformly random bits. Although it is widely believed that many physical phenomena are inherently unpredictable, there is a gap between the computer science model of randomness and what is actually available. It is not clear where one could find such a source of uniformly distributed bits. In practice, computers generate random bits in ad-hoc ways, with no guarantees on the quality of their distribution. One aim of this thesis is to close this gap and identify the weakest assumption on the source of randomness that would still permit the use of randomized algorithms and protocols.

This is achieved by building *randomness extractors.* A randomness extractor is an algorithm

that computes a function $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$, with the property that for any defective source of randomness X satisfying minimal assumptions, $\mathsf{Ext}(X)$ is close to uniformly distributed. Such an algorithm would allow us to use a compromised source of randomness to obtain truly random bits, which we could then use in our original application.

Randomness extractors are interesting in their own right as combinatorial objects that *look random* in strong ways. They fall into the class of objects whose existence is easy to check using the probabilistic method (i.e., almost all functions are good randomness extractors), yet finding explicit examples of a single such object is non-trivial. Expander graphs, error correcting codes, hard functions, epsilon biased sets and Ramsey graphs are just a few examples of other such objects. Finding explicit examples of extractors is part of the bigger project in the area of *derandomization* of constructing such objects which can be used to reduce the dependence of computer science solutions on randomness. These objects are often used as basic building blocks to solve problems in computer science.

The main results of this thesis are:

Extractors for Independent Sources The central model that we study is the model of *independent sources*. Here the only assumption we make (beyond the necessary one that the source of randomness has some entropy/unpredictability), is that the source can be broken up into two or more independent parts. We show how to deterministically extract true randomness from such sources as long as a constant (as small as 3) number of sources is available with a small amount of entropy.

Extractors for Small Space Sources In this model we assume that the source is generated by a computationally bounded processes — a bounded width branching program or an algorithm that uses small memory. This seems like a plausible model for sources of randomness produced by a defective physical device. We build on our work on extractors for independent sources to obtain extractors for such sources.

Extractors for Low Weight Affine Sources In this model, we assume that the source gives a random point from some unknown low dimensional affine subspace with a low-weight basis. This model generalizes the well studied model of bit-fixing sources. We give new extractors for this model that have exponentially small error, a parameter that is important for an application in cryptography. The techniques that go into solving this problem are inspired by the techniques that give our extractors for independent sources.

Ramsey Graphs A Ramsey graph is a graph that has no large clique or independent set. We show how to use our extractors and many other ideas to construct new explicit Ramsey graphs that avoid cliques and independent sets of the smallest size to date.

Distributed Computing with Weak Randomness Finally, we give an application of extractors for independent sources to distributed computing. We give new protocols for Byzantine Agreement and Leader Election that work when the players involved only have access to defective sources of randomness, even in the presence of completely adversarial behavior at many players and limited adversarial behavior at *every* player. In fact, we show how to simulate any distributed computing protocol that assumes that each player has access to private truly random bits, with the aid of defective sources of randomness.

Contents

Acknowledgments v

Abstract vii

Chapter 1 Introduction 1

1.1 Toy Examples . . . 2
1.2 Main Results . . . 5
1.2.1 Seeded Extractors . . . 6
1.2.2 Independent Sources . . . 6
1.2.3 Small Space Sources . . . 7
1.2.4 Affine Sources . . . 8
1.2.5 From Extractors to Ramsey Graphs . . . 8
1.2.6 An Application — Protocols in Distributed Computing . . . 9
1.3 Structure of This Thesis . . . 9

Chapter 2 Basic Definitions and Building Blocks 10

2.1 Min-Entropy and Sources of Randomness . . . 10
2.1.1 Block Sources and Conditional entropy. . . . 13
2.1.2 Somewhere Random Sources . . . 15
2.1.3 Affine Sources . . . 16
2.1.4 Small Space Sources . . . 17
2.2 Convex Combinations . . . 17
2.3 Extractors, Dispersers and other Manipulators of Independent Sources . . . 19
2.4 Network Extractors . . . 21
2.5 Extractors via The Probabilistic Method . . . 22
2.6 Previous Work that We Use . . . 23
2.6.1 Seeded Extractors . . . 23
2.6.2 Extractors for Two Independent Sources . . . 24
2.6.3 Extractors for Affine sources . . . 24

Chapter 3 Condensers for Somewhere Random Sources 25

3.1 Condensing Aligned Independent Somewhere Random Sources . . . 26
3.1.1 A simple condenser for 2 somewhere random sources . . . 26
3.1.2 Better Condensers . . . 27
3.1.3 A strong condenser for 2 somewhere random sources . . . 30

3.2 Condensing When Only Some Sources Are Good . . . 35
3.3 Condensing Affine Somewhere Random Sources . . . 37

Chapter 4 Extractors for Independent Sources **40**
4.1 Previous Results and Overview of Our Results . . . 41
4.1.1 Results in This Chapter . . . 41
4.2 2-Source Extractors via the Probabilistic Method . . . 43
4.3 Multisource Dispersers vs Ramsey Hypergraphs . . . 43
4.4 The Basic Construction . . . 44
4.5 Extractor for one block source and one general source . . . 47
4.5.1 Achieving Small Error . . . 48
4.5.2 Extractor for general source and an SR-source with few rows . . . 50
4.5.3 Proof of the main theorem . . . 55
4.6 3-source Extractors for Polynomially Small Min-entropy . . . 56
4.6.1 Converting Two Independent Sources Into A Block Source . . . 57
4.6.2 Putting it all together . . . 59

Chapter 5 Extractors for Small Space Sources **62**
5.0.3 Total-Entropy Independent Sources . . . 64
5.0.4 Organization . . . 67
5.1 Preliminaries . . . 67
5.1.1 Classes of Sources . . . 67
5.1.2 Seeded Extractors . . . 68
5.2 Small-Space Sources As Convex Combinations Of Independent Sources . . . 69
5.3 Extracting From Total-Entropy Independent Sources . . . 70
5.4 Extracting From Polynomial Entropy Rate . . . 71
5.4.1 Extracting From The Intermediate Model . . . 72
5.4.2 Condensing To Aligned Sources With High Somewhere-Min-Entropy . . . 74
5.5 Extractors For Total-Entropy Independent Sources With Many Short Smaller Sources 75
5.5.1 Random Walks . . . 76
5.5.2 Reducing to Flat Total-Entropy Independent Sources . . . 76
5.5.3 Extracting From Flat Total-Entropy Independent Sources . . . 77
5.6 Extracting More Bits From Total-Entropy Independent Sources . . . 81
5.6.1 Seed Obtainers . . . 81
5.6.2 Constructing Samplers . . . 82
5.6.3 Extractors From Seed Obtainers . . . 83
5.6.4 Extractors For Smaller Entropy . . . 85
5.7 Extractors For Small Space Sources via The Probabilistic Method . . . 86
5.7.1 Small-Space Sources . . . 86
5.7.2 Total-Entropy Independent Sources . . . 87
5.8 Doing Better For Width Two . . . 89
5.8.1 Extracting From Previous-Bit Sources . . . 89
5.8.2 Restricted Width Two Sources vs Previous-Bit Sources . . . 90

Chapter 6 Extractors for Low-Weight Affine Sources 95
6.1 Extractors for Affine Sources via the Probabilistic Method . . . 95
6.2 Previous Work and Our Results . . . 95
6.3 Techniques . . . 96
6.4 Preliminaries . . . 98
6.5 Converting Low-Weight Affine Sources into Affine Somewhere Random Sources . . . 99
6.6 The Extractor . . . 100

Chapter 7 Explicit Ramsey Graphs 101
7.0.1 Our Results . . . 103
7.0.2 Techniques . . . 104
7.1 A Somewhere Extractor with exponentially small error . . . 107
7.2 The Disperser . . . 109
7.2.1 Informal Overview of Construction and Analysis . . . 109
7.2.2 Parameters . . . 120
7.2.3 Formal Construction . . . 121
7.2.4 Formal Analysis . . . 123

Chapter 8 Distributed Computing with Weak Randomness 133
8.0.5 Network Extractors . . . 134
8.0.6 Leader Election and Collective Coin Flipping . . . 135
8.0.7 Byzantine Agreement . . . 135
8.0.8 Previous Work and Our Results . . . 135
8.0.9 Techniques . . . 137
8.1 Synchronous Network Extractors . . . 138
8.1.1 General Weak Sources . . . 139
8.2 Asynchronous Network Extractors . . . 144
8.3 A Lower Bound . . . 145
8.4 Applications to Distributed Computing . . . 146
8.4.1 Collective Coin-Flipping and Leader Election . . . 146
8.4.2 Byzantine Agreement . . . 147

Chapter 9 Conclusions and Future Directions 149
9.0.3 Potential Improvements to Our Constructions . . . 150
9.0.4 New Techniques . . . 150

Appendices 150

Appendix A A 3-Source Extractor for slightly sublinear entropy 151

Appendix B Basic Fourier Analysis 152
B.1 Notation . . . 152
B.2 Inner product and Norms . . . 152
B.3 The Cauchy Schwartz Inequality . . . 153
B.4 Characters and Discrete Fourier Basis . . . 153

B.5 Distributions as Functions 154

Appendix C Bourgain's 2-Source Extractor **155**

C.1 Line Point Incidences 155

C.2 Bourgain's Extractor 155

C.2.1 Review: The Hadamard Extractor 156

C.2.2 Bourgain's Extractor 157

Appendix D XOR lemma for abelian groups **162**

Appendix E Every 2-Source Extractor is Strong **166**

Bibliography **167**

Vita **174**

Chapter 1

Introduction

This thesis is about finding efficient ways to *extract* randomness from defective *sources* of randomness. A *randomness extractor* is an efficient algorithm computing a function

$$\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$$

that takes as input bits that come from a defective source of randomness and outputs bits that are close to uniformly random. There are at least two reasons why designing good randomness extractors is a worthwhile goal.

First, randomness is an essential resource for solving computational problems. Randomized algorithms and protocols play key roles in data structures, cryptography, distributed computing, load balancing and many other areas. We direct the interested reader to the book [MR95] for many examples. These algorithms are typically faster and simpler than their deterministic counterparts. In addition, problems often admit randomized solutions even though no deterministic solution is possible. For example, many problems that arise in cryptography and distributed computing are impossible to solve without access to randomness.

Unfortunately, there is a big gap between the computer science model of randomness and what is actually available. In computer science, it is usually assumed that we have access to a long string of unbiased uniformly random bits. Although it is widely acknowledged that the universe is unpredictable, and physicists believe that many natural phenomena contain randomness, it is unclear where such a string of uniformly distributed bits can be obtained. In practice, computers are programmed to generate "random" bits in some ad-hoc way with no provable guarantee on the quality of the randomness. Even worse, applications in computer science often make stronger assumptions on the randomness that they use. For example, public key cryptography is based on the assumption that keys can be generated by sampling a uniformly random *secret* element from some set. But what happens if an adversary can subvert this assumption? Can we ensure that security is not compromised even if the adversary is able to learn some information about the randomness that goes into the system?

These kinds of questions illustrate the importance of determining the weakest assumption on the source of randomness under which we can still perform the task at hand. A randomness extractor would give a generic way to weaken these assumptions and close the gap between how randomness is modeled in computer science and what is physically available. We could use the extractor to extract truly random bits from the compromised source of randomness, and then use the extracted bits in the original application.

A second reason for finding explicit constructions of randomness extractors is that these are functions that satisfy strong *random-like* combinatorial properties — non-trivial properties that random functions satisfy with high probability. When the model for the weak source is one that makes extraction feasible, a random function is typically a good extractor with high probability (i.e., most functions turn out to be good extractors). Expander graphs, samplers, hard functions, error-correcting codes, pseudorandom generators and epsilon biased sets are other examples of such objects. Finding explicit extractors is part of the larger project of constructing objects that exhibit such combinatorial properties. These objects give generic ways to *derandomize* randomized solutions to problems; they are useful in reducing the amount of randomness used in randomized solutions. This project relates to some of the central questions in complexity theory and computer science. For instance, showing that $P \neq NP$ amounts to giving an example of a hard function in NP which is not computable in P. Giving an example of another kind of hard function would show that $BPP = P$ [NW94] (i.e., that every randomized algorithm has an efficient deterministic counterpart), another central open question in computer science.

Not surprisingly, explicit constructions of these objects have found a wide variety of applications in computer science. These objects tend to have close connections with each other. Explicit constructions of extractors thus give insight into building deterministic objects that *look random*.

Constructions of randomness extractors have been used to get constructions of communication networks and good expander graphs [WZ99, CRVW02], error correcting codes [TZ04, Gur03], cryptographic protocols [Lu04, Vad04], data structures [MNSW98] and samplers [Zuc97]. In this thesis we use our extractors for independent sources to get constructions of Ramsey graphs (Chapter 7) and new protocols in distributed computing (Chapter 8).

1.1 Toy Examples

Before diving into discussing our research, we give a few examples that give a feel for *extractor problems*. Our discussion will be vague and will leave out many details, but we hope that it will help build a picture for what an *extractor problem* is, and how it can arise (at least in toy worlds). The reader may choose to skip this section if she'd rather get to the actual models we consider.

Biased coin Say two people are trying to settle a dispute by tossing a coin, yet each of them doesn't trust the other to use a coin which is fair. Can they *simulate* a fair coin toss with the aid of a defective biased coin?

One solution to this problem is to toss the coin n times and then compute the parity of the coin tosses, i.e., they decide that the outcome of the simulated coin toss is heads exactly when an even number of the n coin tosses give heads. It can be shown that if the original coin had a $1/2 + \epsilon$ chance of giving heads, the simulated coin has a $1/2 + (2\epsilon)^n/2$ chance[1] of giving heads. We can choose n to be so large that the adversary's advantage is reduced to something that might be insignificant, *assuming we know what ϵ is.*

An alternate solution due to von-Neumann [vN51] is cleaner: we group the sequence of coin tosses into pairs of coin tosses. We ignore all pairs which are of the form tail-tail or head-head and consider the first pair which isn't of this form. The simulated coin is a head exactly when this first pair is a tail-head. It is easy to check that now the simulated coin is *completely*

[1] This is an exercise.

unbiased, regardless of what the bias of the original coin was, since tail-head is as likely as head-tail regardless of the original bias. Of course there is a small chance that we will have to wait a very long time to get an output for the simulated coin. von-Neumann's solution seems particularly appealing because to use it we require a weaker assumption on the source of randomness. It is sufficient that all coin tosses are independent and have the same bias, we need not know what the actual bias of the coins is.

On the other hand, suppose we tried to speed up this process by tossing n different coins, presumably with different biases, *at the same time* and then do some computation to simulate a single coin toss. In this case von-Neumann's approach no longer works, since it relied crucially on each toss (or at least pairs of tosses) having the same bias. However, it's easy to see that the original approach of computing the parity of the coins still gives that the probability of the simulated coin being heads is $1/2 + (1/2)\prod_{i=1}^{n}(2\epsilon_i)$, if the i'th coin has a $1/2 + \epsilon_i$ probability of showing a head. Thus which solution should be used crucially depends on the type of defect in the coins.

What if we had loaded dice instead of coins to work with? Would computing the sum of n loaded dice modulo 6 generate a value that is close to uniform?[2]

Exposure Resilience Suppose Alice and Bob are in different locations and are trying to agree on a password for a joint bank account that they are trying to create. One way for them to do this would be for Alice to generate a random password (say a random n-bit string) and transmit this to Bob. A third person, Charlie, may intercept their communication and learn their password. Alice and Bob could prevent this via cryptography — Alice could encrypt the password and transmit the encryption to Bob, who would then decrypt it. But is there some way for them to agree on a password in a way that's information theoretically secure? Of course if Charlie intercepts all of their communication, this is impossible to achieve, since he'd be able to learn whatever Bob can learn. But what if Charlie only gets to see some t of the transmitted bits? In this case something can be done — Alice and Bob can agree that their password is the parity of the transmitted bits. If Charlie manages to miss a single bit of the transmission, then the parity remains completely unpredictable to him.

This is a pretty good solution, except that it only gives Alice and Bob a way to generate a single bit password. We could get 2 bits by breaking up the n transmitted bits into two $n/2$-bit strings and computing the parity of each part. This would work as long as Charlie gets to see less than $n/2$ total bits of the transmission. In this way we can get a scheme that generates n/t bits as long as Charlie only gets to see t bits of the transmission.

Another idea is to use polynomials over a finite field $\mathbb{F}$. Let $x_1, x_2, \ldots, x_{2h-t}$ be distinct points in a finite field. We could interpret the transmitted bits as the evaluations of a random degree h polynomial f at $h+1$ points: $f(x_1), f(x_2), \ldots, f(x_{h+1})$. Then we could make the real password $f(x_{h+2}), \ldots, f(x_{2h+2-t})$. Now if the adversary learns the evaluations at any t points, the evaluations at any other $h+1-t$ points look uniform, so this would work. The problem with this approach is the alphabet size: to find so many distinct point in a finite field, the size of the field has to be at least $\log h$. If n is the number of bits in the transmission, the adversary can read just $n/|\mathbb{F}| \sim n/\log n$ bits to ruin our solution. This scheme does have

[2] Yes, barring pathological cases (for instance every one of the dice could only give values from the same subgroup of $\mathbb{Z}_6$, in which case the sum will also lie in the subgroup). This is a harder exercise. A hint is to use Lemma D.0.11.

the advantage that we can even allow for the adversary to corrupt some of the bits of the transmission and recover (by choosing the degree of the transmitted polynomial to be smaller than h, so that we are transmitting a Reed Solomon codeword).

A smarter way to solve this problem was proposed by Kamp and Zuckerman [KZ03] — the final password is simply the sum of the bits modulo M, where M is some integer of size roughly $\sqrt{n-t}$. It can be shown that for any fixing of the t bits that Charlie receives, the sum of the bits modulo M corresponds to taking a random walk on a cycle of size M, with an adversarially chosen starting point. Such a walk mixes to close to uniform in roughly M^2 steps. This gives $m = \log M = \Omega(\log(n-t))$ bits which are 2^{-m} close to uniform, for *any* t. No other result with exponentially small error was known for the case when $n - t < \sqrt{n}$. In Chapter 6 we give a new scheme that gives $m = (n-t)^{\Omega(1)}$ bits which are 2^{-m} close to uniform, when $n - t$ is polylogarithmic in n.

Leader Election Suppose a group of c people want to play a board game and need to pick the person who gets to start the game. Unfortunately, they have no access to a public source of randomness that they can all trust. How can they select who gets to start in a fair manner? One possible solution is this: they could each write down a secret integer on separate pieces of paper and fold them. Then they could gather all the pieces of paper and compute the sum of these integers modulo c to determine which of the players gets to start. This solution guarantees that no group of $c-1$ players can control who gets to start[3].

Now what if we even want to tolerate the players getting additional information about the numbers that other players selected? If some player manages to learn the last digit of each of the integers that the other players selected by sneaking quick glances while they're writing, he could potentially use this information to give himself an edge. For instance, if c is even, knowing whether the last digits are even or odd would be enough for the player to control whether the final sum is even or odd modulo c. Of course we cannot hope to do anything if any player manages to learn *everything* about the other players' secret numbers. Can we find a protocol whose outcome cannot be controlled by a large coalition of players, even if they learn a lot of information about the honest players' numbers?

The problem with our last solution was that the even integers in $\mathbb{Z}_c$ form a subgroup of $\mathbb{Z}_c$. A coalition can use just a small amount of information (whether or not the secret integers lie in this even subgroup) to get a lot of information about the sum. This suggests the following solution: we could ask that each player write down an integer and then compute the sum of the integers modulo p, for a very large prime p. We could then reduce this sum modulo c to pick the winning player. Since the additive group of integers modulo p doesn't have any non-trivial subgroups, cheating players shouldn't be able to force the outcome to lie in a subgroup. This seems like it might be a step in the right direction; certainly as long as an adversary doesn't learn enough information about each of the honest players' integers, he cannot force the final outcome into a small set (note that any single integer that's not 0 mod p generates the entire group). Unfortunately, this solution has a pretty bad quantitative performance.

To see this, suppose the first $c-1$ players write down the integers $x_1, x_2, \ldots, x_{c-1}$. Fix a large constant $c < T < p$ and consider the integers $y_i = c^{-1}Tx_i \mod p$. We can express every such

[3] This is different from the standard *full information* model often used in distributed computing, since we are forcing all players to commit to values before revealing any of their integers.

$y_i = a_i T + b_i$, where here $0 \leq b_i < T$ and $0 \leq a_i < p/T$. Note that $x_i = ca_i + cT^{-1}b_i \mod p$. Suppose the c'th player manages to learn or guess $b_1, \ldots, b_{c-1}$, which is only $(c-1)\log T$ bits of information. He can then set $x_c = -cT^{-1}\sum_{i=1}^{c-1} b_i \mod p$. This guarantees that $\sum_{i=1}^{c} x_i = c\sum_{i=1}^{c-1} a_i + 0 \mod p$. Since each $a_i < p/T$, we get that $c\sum_{i=1}^{c-1} a_i < cp/T < p$, which guarantees that the result is a multiple of c even after reducing it modulo p. Thus, even learning a constant number of the bits about each player's number can allow one player to control the outcome[4].

It turns out that there is a simple protocol for this problem. We require each player to pick an integer which is not $0 \mod p$. Then we multiply each of the integers in $\mathbb{Z}_p$ and reduce this product modulo c to pick the winning player. It can be shown that this works, even if all but a constant (independent of c) number of the players collude together, and even if the cheating players learn 99% of the information about each of the honest players' secrets[5]. In fact, if $c-2$ players collude and learn less than 49% of the other two players' numbers, then the result is still uniformly distributed from their point of view[6].

This solution is discussed more thoroughly in Chapter 5.

Each of these examples involved constructing a *randomness extractor* for a specific class of sources. In the case of our first example (the biased coin), the source was a defective or maliciously set up device. In the case of our other two examples, the source was a distribution obtained by conditioning the uniform distribution on some type of event.

1.2 Main Results

Let us now be more precise about the actual problems that we solve in this thesis. We are interested in building *randomness extractors* — efficient algorithms computing a function

$$\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$$

that takes as input bits that come from a defective source of randomness and outputs bits that are close to uniformly random.

To formalize the problem of randomness extraction, we must decide on a model for the types of defective sources that the extractors can handle. If we intend to extract m random bits, information theoretic considerations show that the source must contain at least m bits of entropy. The goal is to construct extractors which output the most number of random bits for a source with given entropy, have small error and work for very general classes of sources.

We start with a very general (in fact, we shall soon see that it's too general) class: *weak sources* [CG88]. The only constraint on a weak source is a necessary constraint — that the source has some entropy. We say that a source that supplies n-bit samples has *min-entropy* k if the probability of getting any particular string from the source is at most 2^{-k}. Such a source is called an (n,k)-source. Unfortunately, it is easy to see that there is no deterministic extractor that can

[4] The intuition for this example is that giving the value of $x \in \mathbb{Z}_p$ modulo c is the same as giving which of the intervals $[1, p/c], [p/c, 2p/c], \ldots, [p(c-1)/c, p]$ contains $c^{-1}x$. If the cheating player finds out where each $c^{-1}x_i$ lies upto a $1/T$ accuracy, she can predict where the sum will lie accurately enough to control the outcome of the protocol. It is easy to modify the scheme so that the last player can force any outcome she wants.

[5] We are not aware of an easy proof of this fact. It is implied by Theorem 5.3.3.

[6] This fact seems to have a much easier proof — Theorem C.2.1.

extract from any weak source. If $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}$ is any such purported extractor, let S be the bigger set from the pre-images $\mathsf{Ext}^{-1}(0)$ and $\mathsf{Ext}^{-1}(1)$. We see that Ext must fail to extract from the distribution which is uniform over the set S, even though this source has min-entropy at least $n - 1$[7].

Given that it is impossible to find a single extractor that works for every source with sufficient entropy, our goal is to look for additional constraints on the source of randomness that will make randomness extraction feasible, yet be general enough to still be useful.

1.2.1 Seeded Extractors

Much work has focussed on the task of simulating probabilistic algorithms with weak random sources [VV85, CG88, Zuc96, SSZ98, ACRT99]. This work resulted in the introduction, by Nisan and Zuckerman [NZ96], of what we shall refer to in this thesis as a *seeded* extractor. Here the assumption is that the source consists of a sample from a weak source and an additional much shorter independent *seed* of truly uniformly random bits.

When simulating probabilistic algorithms with weak random sources, the need for truly random bits can be eliminated by enumerating over all choices of the seed. Seeded extractors have turned out to have a wide variety of other applications and were found to be closely related to many other important pseudorandom objects. Thus, they were the main focus of attention in the area of randomness extraction in the 90's, with a variety of very efficient constructions. For any $n, k \in \mathbb{N}$ we now know how to construct extractors that can extract a constant fraction of k bits which are almost uniformly random using a very short (only a constant multiple of $\log n$) length seed from any (n, k)-source [LRVW03, GUV07].

Unfortunately, the trick of running over all seeds to a seeded extractor does not give a generic way to make any protocol or algorithm that needs access to randomness (for more than efficiency) function with the aid of a weak source. These extractors cannot be used in applications like cryptography or distributed computing protocols.

We direct the interested reader to [Nis96, NT99, Sha02] for surveys of the origins, applications and constructions of seeded extractors.

1.2.2 Independent Sources

Perhaps the most natural restriction on the randomness (first considered by [SV86, Vaz85, CG88]) is to assume that we have access to two or more *independent* sources of randomness. Finding extractors for this class of sources is the central topic of this thesis. It is conceivable that we could find truly independent sources of randomness in nature. For instance, perhaps the bits coming from the operation of the operating system of the computer are independent of the bits coming from the computer's clock. At the very least, the assumption of independence between a few parts of the source is much weaker than the assumption that the source gives bits that are *really* close to uniform (which asserts in particular that every single bit of the source is independent of every other one).

Another way to model sources of randomness, that is relevant to cryptography, is to imagine that the source is the result of some partial information being revealed to an adversary. Suppose

[7] If the min-entropy is significantly larger (not that there's much room to play with) than $n - 1$, then the input itself must be statistically close to uniform. For instance if the min-entropy is $n - \epsilon$, the input is $(2^\epsilon - 1)$-close to uniform.

we are in the situation where for some function $f : \{0,1\}^n \to \{0,1\}^t$, the adversary is capable of learning $f(x)$ of our truly random string x. This may give the adversary some information, but if t is smaller than n, we expect that the adversary doesn't know *everything* about x. Can we then find a function $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ with the property that $\mathsf{Ext}(x)$ looks completely random to the adversary? This task turns out to be closely related to the problem of constructing randomness extractors for arbitrary sources with some entropy. Indeed, just like above, we can easily prove that it is impossible to find such an Ext that works for *every* f — the adversary can simply choose f to be the function Ext itself, then it is clear that she can predict the bits that we extract.

In this view, assuming that the source consists of several independent parts turns out to correspond to constraining the adversary's function f to be a function of low communication complexity — we assume that the sample x is broken into several parts and we are dealing with several adversaries, each of whom have access to only one part of the source. Then we bound the total number of bits they can communicate with each other to compute f. We discuss this view further in Chapter 8, where we use it to get distributed computing protocols that cannot be disrupted by such adversaries.

In Chapter 4, we discuss our work in constructing extractors for independent sources, showing how to extract randomness from as few as 3 independent sources, even when the entropy is polynomially small in the length of each of the sources. The results in this chapter are based on work with Boaz Barak, Xin Li, Ronen Shaltiel, Avi Wigderson and David Zuckerman [Rao06, BRSW06, LRZ07].

1.2.3 Small Space Sources

Another natural way to constrain the source is to assume that it was generated by some computationally bounded process. This type of source was first considered by Trevisan and Vadhan [TV00], who gave extractors for sources that are generated by small circuits. In our work, we consider the model where the source is generated by a bounded width branching program or an algorithm with a small amount of memory. This seems to be a plausible model for physical random sources since it seems believable that a defective physical device has only a small amount of state that creates correlations across the random bits that it is generating. This model also generalizes a number of previously studied models for sources. In the "adversarial" view discussed above, this model corresponds to constraining the adversary's function f to be a function that can be computed by a machine with small memory and one way access to the random string.

In Chapter 5 we show how to build extractors for this model. We give polynomial-time, deterministic randomness extractors for sources generated in small space, where we model space s sources on $\{0,1\}^n$ as sources generated by width 2^s branching programs. For every constant $\delta > 0$, our algorithm extracts $.99\delta n$ bits that are exponentially close to uniform (in variation distance) from space s sources of min-entropy δn, where $s = \Omega(n)$. In addition, assuming an efficient deterministic algorithm for finding large primes, there is a constant $\eta > 0$ such that for any $\zeta > n^{-\eta}$, our algorithm extracts $m = (\delta - \zeta)n$ bits that are exponentially close to uniform from space s sources with min-entropy δn, where $s = \Omega(\beta^3 n)$. Previously, nothing was known for $\delta \leq 1/2$, even for space 0.

Our results are obtained by reducing the problem to that of extracting from a class of sources that is very close to independent sources. These results are based on work with Jesse Kamp, Salil Vadhan and David Zuckerman [KRVZ06].

1.2.4 Affine Sources

This is a source that gives a uniform point from some unknown low dimensional subspace. In the "adversarial" view, this corresponds to constraining the adversary's function f to be a linear function. For example, the adversary might have gotten access to some t of the bits of our random string. We make partial progress towards building an extractor for such sources in Chapter 6, where we construct an extractor for a smaller class of sources called *low weight affine* source. Here the assumption is that the affine subspace has a basis which has many low weight vectors. In particular, our extractors give the best known extractors for *bit-fixing* sources, which are weight one sources. Although there isn't a direct connection to the case of independent sources, our extractors are obtained using techniques that are very similar to those used to get extractors for independent sources. These extractors have applications in cryptography.

1.2.5 From Extractors to Ramsey Graphs

One important reason why the independent sources model is interesting is its connection to explicit constructions of *Ramsey Graphs*. A graph on N vertices is called a K-*Ramsey Graph* if it contains no clique or independent set of size K. In 1947 Erdős published his paper inaugurating the *Probabilistic Method* with a few examples, including a proof that *most* graphs on $N = 2^n$ vertices are $2n$-Ramsey. The quest for constructing such graphs explicitly has existed ever since and led to some beautiful mathematics.

Another way to view 2-source extractors is as boolean matrices that look random in a strong sense: Every 2-source extractor for entropy k gives an $N \times N$ boolean matrix in which every $K \times K$ minor has roughly the same number of 1's and 0's, with $N = 2^n, K = 2^k$. When the matrix satisfies the weaker property that every such minor is not monochromatic (the fraction of 1's and 0's need not be roughly the same), we call the function a 2-source *disperser*. Viewed as the adjacency matrix of a bipartite graph, every 2-source disperser gives a construction of a K-Ramsey bipartite Graph, since there cannot be any bipartite cliques or independent sets of size $K \times K$. It turns out that there is a simple way to turn every bipartite Ramsey Graph into a Ramsey graph with almost the same parameters.

The best explicit Ramsey Graph construction to date before the results in this thesis was obtained in 1981 by Frankl and Wilson [FW81], who used intersection theorems for set systems to construct N-vertex graphs that are $2^{\sqrt{n \log n}}$-Ramsey. The best explicit 2-source disperser was a construction that worked for min-entropy $k = o(n)$, by Barak, Kindler, Shaltiel, Sudakov and Wigderson [BKS$^+$05].

In Chapter 7 of this thesis, we give an explicit disperser for two independent sources on n bits, each of min-entropy $k = 2^{\log^{1-\alpha_0} n}$, for some small constant $\alpha_0 > 0$. Put differently, setting $N = 2^n$ and $K = 2^k$, we construct explicit $N \times N$ Boolean matrices for which no $K \times K$ submatrix is monochromatic. Viewed as adjacency matrices of bipartite graphs, this gives an explicit construction of K-Ramsey *bipartite* graphs of size N. Thus we give explicit construction of both Ramsey graphs and 2-source dispersers that give better bounds.

These results were made possible by our earlier work on constructing good extractors for independent source. A key ingredient which allows us to beat the barrier of $k = \sqrt{n}$ is a new (and more complicated) variant of the *challenge-response mechanism* of Barak et al. [BKS$^+$05] that allows us to find the min-entropy concentrations in a source of *low* min-entropy.

Our results are based on work with Boaz Barak, Ronen Shaltiel and Avi Wigderson [BRSW06].

1.2.6 An Application — Protocols in Distributed Computing

Finally, in Chapter 8, we discuss how to use ideas that go into our extractor constructions to design new distributed computing protocols.

We design several efficient one-round *network extractor protocols*, which extract private randomness over a network with faulty players when each player has a single, weak random source of sufficient min-entropy. As a corollary, we derive efficient protocols for Byzantine agreement and leader election (and hence the equivalent collective coin-flipping) in the full information model. Our robust protocols run in just one more round than the corresponding protocols with perfect randomness.

In a synchronous network, if each player's weak source has min-entropy rate greater than $1/2$, then we essentially match the bounds for perfect randomness: Byzantine agreement tolerating a $1/3 - \alpha$ fraction faulty players, and leader election tolerating a $1/2 - \alpha$ fraction faulty players, for any constant $\alpha > 0$. In a synchronous network, if each player's n-bit source of randomness has $n^{\Omega(1)}$ min-entropy, then the bounds drop to $1/4 - \alpha$ and $1/3 - \alpha$, respectively. In an asynchronous network, if each player has access to a source with polynomial min-entropy (though $1/3$ of the players need shorter sources than the others), then our Byzantine agreement protocol tolerates a $1/18 - \alpha$ fraction of faulty players.

These results are based on work with Xin Li and David Zuckerman [LRZ07].

1.3 Structure of This Thesis

In Chapter 2 we give some basic definitions and results from previous works that our results depend on. We recommend that the reader skim this chapter on the first reading, returning to it whenever clarification is needed about a particular definition.

Many of our results rely on new ways to manipulate a special class of sources called *somewhere random sources*. In Chapter 3 we discuss these techniques. The rest of the chapters are devoted to presenting the constructions we have discussed above.

Chapter 2

Basic Definitions and Building Blocks

This chapter is intended to be a comprehensive listing of the formal definitions and basic facts that will be used throughout this thesis. We also list here many technical lemmas that are used repeatedly in this thesis. We recommend that the reader skim this chapter in a first reading, and return to it when a concept needs clarification in later chapters. We will usually repeat definitions in the later chapters when concepts are used for the first time.

Throughout this thesis, we will use capital letters to denote distributions and sets. We will usually use the same lowercase letter to denote an instantiation of the capital letter, for e.g. for a set X, we would use x to denote an element in X.

We use the convention that $N = 2^n$, $M = 2^m$ and $K = 2^k$.

All logarithms are meant to be base 2, unless we explicitly state otherwise.

We will use U_m to denote the uniform distribution on the set $\{0,1\}^m$.

Often in technical parts of this thesis, we will use constants like 0.9 or 0.1 where we could really use any sufficiently large or small constant that is close to 1 or 0. We do this because it simplifies the presentation by reducing the number of additional variables we will need to introduce.

2.1 Min-Entropy and Sources of Randomness

We will be concerned with the treatment of various kinds of distributions that are *nice* in that they contain a sufficient amount of usable randomness. Early works on extractors went through several ways to measure how much usable randomness a source contains, until they eventually converged onto a definition suggested by Chor and Goldreich [CG88].

Definition 2.1.1. [CG88] The *min-entropy* of a distribution X, denoted $H_\infty(X)$, is k if the heaviest point under X has weight 2^{-k}, i.e.,

$$H_\infty(X) \overset{def}{=} \min_{x \in \mathsf{supp}(X)} \log\left(\frac{1}{X(x)}\right)$$

The *min-entropy rate* of a distribution X on $\{0,1\}^n$ is $H_\infty(X)/n$.

Definition 2.1.2. An (n,k)*-source* is a distribution X over $\{0,1\}^n$ with $H_\infty(X) \geq k$.

Note that for any deterministic function $f : \{0,1\}^n \to \{0,1\}^m$ and any distribution X, $H_\infty(f(X)) \leq H_\infty(X)$, since the image of every point under f is at least as heavy as the point in

X. Thus, if we intend to extract m bits of randomness from a source, the source must *necessarily* have min-entropy m[1].

It is worthwhile to compare the notion of min-entropy with the more familiar notion of *Shannon* entropy: $H(X) = \sum_{x \in \mathsf{supp}(X)} \frac{X(x)}{\log(X(x))}$. The difference between the two measures is that the min-entropy of X gives a bound on the amount of information that *every* point in the support of the distribution carries, while the Shannon entropy measures the *average* amount of information that points in the support of a distribution carry. Clearly we have that $H_\infty(X) \leq H(X)$, thus min-entropy is a more stringent measure of the amount of randomness in a source. Still, it turns out that min-entropy shares many nice properties that Shannon entropy has. For instance, it is clear that for two random variables X, Y in the same probability space, $H_\infty(XY) \geq \max\{H_\infty(X), H_\infty(Y)\}$.

Many of our arguments will involve *conditioning* random variables on certain events and then arguing about the corresponding conditional distribution. It will be useful to have the following definition:

Definition 2.1.3 (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 exists a set $A \subseteq \{0,1\}^n$ such that $(X|A) = X'$ and $\Pr[X \in A] \geq 2^{-d}$.

Fact 2.1.4 (Deficiency of sub-subsources.)**.** *Let X' be a subsource of X with deficiency d' and let X'' be a subsource of X' with deficiency d''. Then X'' is a subsource of X with deficiency $d' + d''$.*

Sometimes the distributions we get are not exactly the distributions we want, but they may be close enough. We will measure how close distributions are by their *statistical distance*:

Definition 2.1.5. 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 shall 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 2.1.6. *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 $|g(D) - g(F)| \leq \epsilon$.*

Proposition 2.1.7. *Let A, B be two independent random variables over $\{0,1\}^n$. Then $|A \oplus B - U_n| \leq \max\{|A, U_n|, |B, U_n|\}$, where here $\oplus$ is the bitwise xor function.*

Lemma 2.1.8. *Let X, Y, and V be distributions over Ω such that X is ϵ-close to uniform and $Y = \gamma \cdot V + (1 - \gamma) \cdot X$. Then Y is $(\gamma + \epsilon)$-close to uniform.*

A fact that we will often use is that any *typical* event cannot *steal* a lot of min-entropy from a source, in the following sense:

Proposition 2.1.9. *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 $H_\infty(X') = k - d$.*

[1] If the source has min-entropy $m - \log(1/\epsilon)$, the output of our distribution must be $(1 - \epsilon)$-far from the uniform distribution in terms of statistical difference.

Proof. For every point $x \in \mathsf{supp}(X|A)$, note that $\Pr[X = x|X \in A] = \Pr[X = x]/\Pr[A]$, from which the proposition is obvious. □

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.

Proposition 2.1.10 (Fixing a function)**.** *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*

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

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, and choosing $\ell = m$ we get the second part.

□

The following lemma will also be useful:

Lemma 2.1.11. *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 2.1.9, 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 $\Pr[X_{|\overline{S}} \in H] \leq 2^{-l}]$.

□

In some situations we will have a source that is statistically close to having high min-entropy, but not close enough. We can use the following lemma to lose something in the entropy to get 0 error on some subsource.

Lemma 2.1.12. *Let X be a random variable over $\{0,1\}^n$ s.t. X is $1/4$-close to an (n,k) source. Then there is a deficiency 2 subsource $X' \subseteq X$ s.t. X' is a $(n, k-3)$ source.*

Proof. Let t be a parameter that we will pick later. Let $H \subseteq \mathsf{Supp}(X)$ be defined as $H \overset{def}{=} \{x \in \mathsf{Supp}(X)| \Pr[X = x] > 2^{-t}\}$. H is the set of heavy points of the distribution X. By the definition of H, $|H| \leq 2^t$.

Now we have that $\Pr[X \in H] - 2^{-k}|H| \leq 1/4$, since X is $1/4$-close to a source with min-entropy k. This implies that $\Pr[X \in H] \leq 1/4 + 2^{-k}|H| \leq 1/4 + 2^{t-k}$.

Now consider the subsource $X' \subseteq X$ defined to be $X|X \in (\mathsf{Supp}(X) \setminus H)$. For every $x \in \mathsf{Supp}(X')$, we get that

$\Pr[X' = x] = \Pr[X = x|X \notin H] \leq \frac{\Pr[X=x]}{\Pr[X \notin H]} \leq \frac{2^{-t}}{1-(1/4+2^{t-k})}$

Setting $t = k - 2$, we get that $\Pr[X' = x] \leq \frac{2^{-k+2}}{1-(1/4+2^{-2})} \leq 2^{-k+3}$.

□

We will need the following lemma to reduce the error in the constructions.

Lemma 2.1.13 ([BIW04]). *Let $Z^1, \ldots, Z^v$ be independent distributions over $\{0,1\}^k$ with $|Z^i - U_k| < \epsilon$ for every $i = 1, \ldots, v$. Then*

$$|Z^1 \oplus Z^2 \oplus \cdots \oplus Z^v - U_k| < \epsilon^v$$

The following lemma gives a sufficient condition to lowerbound the min-entropy of a source.

Lemma 2.1.14 ([GUV07]). *Let X be a random variable taking values is a set of size larger than 2^k such that for every set S of size less than $\epsilon 2^k$, $\Pr[X \in S] < \epsilon$. Then X is ϵ-close to having min-entropy k.*

Proof. First note that $|\mathsf{supp}(X)| \geq \epsilon 2^k$, or else the hypothesis of the lemma is contradicted by setting $S = \mathsf{supp}(X)$.

Let S be the $\epsilon 2^k$ heaviest elements under X. Then for every $x \notin S$ we must have that $\Pr[X = x < 2^{-k}]$, or else every element in S would have weight greater than 2^{-k}, which would contradict the hypothesis. Thus, the set of elements that have weight more than 2^{-k} are hit with probability at most ϵ, which implies that X is ϵ-close to having min-entropy k. □

2.1.1 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 2.1.15 (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 for all $x_1 \in X^1, \ldots, x_{i-1} \in X^{i-1}$, $H_\infty(X^i | X^1 = x_1, \ldots, X^{i-1} = x_{i-1}) \geq k_i$, i.e., each block has high min-entropy even conditioned on the previous blocks. If $k_1 = k_2 = \cdots = k_{\mathsf{C}}$ then 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 $X_1 = x_1$,...,$X_i = x_1$. For $1 \leq i < j \leq \mathsf{t}$, we denote by $X_{i,\ldots,j}$ the projection of X into the blocks $X_i, \ldots, X_j$. We have the following facts about such sources:

Lemma 2.1.16 (Typical prefixes). *Let $X = X_1, \cdots, X_{\mathsf{t}}$ be a random variable divided into t blocks, let $X' = X|A$ be a deficiency d subsource of X, and let ℓ be some number. Then for every $1 \leq i \leq \mathsf{t}$, with probability at least $1 - 2^{-\ell}$, a random prefix $x_1, \ldots, x_i$ in X' satisfies $\Pr[X \in A | x_1, \ldots, x_i] \geq 2^{-d-\ell}$.*

Proof. We denote by X^1 the first i blocks of X. Let B be the event over X^1 that $\Pr[X \in A | X^1] < 2^{-d-\ell}$. We need to prove that $\Pr[B|A] < 2^{-\ell}$ but this follows since $\Pr[B|A] = \frac{\Pr[A \cap B]}{\Pr[A]} \leq 2^d \Pr[A \cap B]$. However $\Pr[A \cap B] \leq \Pr[A|B] = \sum_{x \in B} \Pr[A|X^1 = x] Pr[X^1 = x|B] < 2^{-d-\ell}$. □

As a corollary we get the following

Corollary 2.1.17 (Subsource of block sources)**.** *Let $X = X_1, \cdots, X_{\mathsf{C}}$ be a $k_1, k_2, \ldots, k_{\mathsf{C}}$-entropy C-block source (i.e., for every $x_1, \ldots, x_i \in \mathsf{Supp}(X_{1,\ldots,i})$, $H_\infty(X_{i+1}|X_{1,\ldots,i} = x_1, \ldots, x_i) \geq k_{i+1}$) and X' be a deficiency d subsource of X. Then X' is $\mathsf{C}2^{-l}$ statistically close to being a $k_1 - d - l, k_2 - d - l, \ldots, k_{\mathsf{C}} - d - l$ block source. In fact, there is a subsource $Y \subset X'$ of density $1 - \mathsf{C}2^{-l}$ which is such a block source.*

Proof. Let $X' = X|A$ and define B to be following the event over X': $x = x_1, \ldots, x_{\mathsf{C}} \in B$ if for some $i \in [\mathsf{C}]$, $\Pr[X \in A|x_1, \ldots, x_i] < 2^{-d-l}$. By Lemma 2.1.16, $\Pr[X' \in B] < \mathsf{C}2^{-l}$. However, for every $x = x_1, \ldots, x_{\mathsf{C}} \in \bar{B} = A \setminus B$, we get that $Y' = X'_{i+1}|x_1, \ldots, x_{i-1}$ is a source with $H_\infty(Y') \geq H_\infty(Y) - d - l \geq k - d - l$. Hence $X'|\bar{B}$ is a $k - d - l$-block source of distance $\mathsf{C}2^{-l}$ from X'. □

If $X = X_1, \cdots, X_{\mathsf{t}}$ is a source divided into t blocks then in general, the entropy of X_i conditioned on some prefix $x_1, \ldots, x_{i-1}$ can depend on the choice of prefix. However, the following lemma tells us that we can restrict to a low deficiency subsource on which this entropy is always roughly the same, regardless of the prefix. Thus we can talk about the conditional entropy of a block X_i without referring to a particular prefix of it.

Lemma 2.1.18 (Fixing entropies)**.** *Let $X = X_1, X_2, \cdots, X_{\mathsf{t}}$ be a t-block random variable over $\{0,1\}^n$, and let $0 = \tau_1 < \tau_2 < \cdots < \tau_{c+1} = n$ be some numbers. Then, there is a deficiency $\mathsf{t}^2 \log c$ subsource X' of X and a sequence $\bar{e} = e_1, \ldots, e_{\mathsf{t}} \in [c]^{\mathsf{t}}$ such that for every $0 < i \leq \mathsf{t}$ and every sequence $x_1, \ldots, x_{i-1} \in \mathsf{Supp}(X'_{1,\ldots,i-1})$, we have that*

$$\tau_{e_i} \leq H_\infty(X'_i|x_1, \ldots, x_{i-1}) \leq \tau_{e_i+1} \tag{2.1}$$

Proof. We prove this by induction. Suppose this is true for up to $\mathsf{t} - 1$ block 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 $X_{2,\ldots,i}|x_1$. By the induction hypothesis there exists a $(\mathsf{t}-1)^2 \log c$ deficiency subsource $Y'(x_1)$ of $Y(x_1)$ source and $\bar{e}(x_1) \in [c]^{\mathsf{t}-1}$ the sequence such that $Y'(x_1)$ satisfies Equation 2.1 with respect to $\bar{e}(x_1)$. Define the function $f : X_1 \to [c]^{\mathsf{t}-1}$ that maps x_1 to $\bar{e}(x_1)$ and pick a subsource X'_1 of X_1 of deficiency $(\mathsf{t}-1) \log c$ such that f is constant on X'_1. That is, there are some values $e_2, \ldots, e_{\mathsf{t}} \in [c]^{\mathsf{t}-1}$ such that $F(x_1) = e_2, \ldots, e_{\mathsf{t}}$ with probability 1. We let the source X' be X conditioned on the event that for $x_1, \ldots, x_{\mathsf{t}} \in X'$, $x_1 \in X'_1$ and $x_2, \ldots, x_{\mathsf{t}} \in Y(x_1)$.

The deficiency of X' is indeed at most $(\mathsf{t} - 1) \log c + (\mathsf{t} - 1)^2 \log c < \mathsf{t}^2 \log c$. □

Corollary 2.1.19. *If X in the lemma above is a k-source, and $\bar{e}$ is as in the conclusion of the lemma, we must have that $\sum_{i=1}^{t} \tau_{e_i+1} \geq k - t^2 \log c$.*

Proof. If this was not the case, we could find some string in the support of X which is too heavy (simply take the heaviest string allowed in each successive block). □

The following lemma is useful to prove that a distribution is close to being a block source.

Lemma 2.1.20. *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 having min-entropy k_i, with $\sum_i \epsilon_i < 1/10$. Then for every $\ell > 10 \log t$ we must have that X is $t(\sum_{i=1}^{t} \epsilon_i + 2^{-\ell+1})$-close to being a block source, where each block X_i has min-entropy $k_i - n_{i-1} - 1 - 2\ell$.*

Proof. We will need to define the notion of a *submeasure*. 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.

Given a submeasure on $\{0,1\}^n$, we say that it is ϵ-close to having min-entropy k, if

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

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 $\Pr[M \in A] = \sum_{m \in A} M(m)$.

Returning to the lemma, let us define some submeasures: define $M_{t+1} = X$.

For $i = t, t-1, t-2, \ldots, 1$, define

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

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

Now define the sets $B_i = \{m \in \{0,1\}^{n_t} : M_i(m) \neq M_{i+1}(m)\}$. Set $B = \cup_i B_i$. Then note that $\Pr[X \in B] \leq \sum_{i=2}^{t} \Pr[M_{i+1} \in B_i]$.

If $C_i = \{m : M_{i+1}(m^i) > 2^{-k_1+\ell}\}$, we see that $\Pr[M_{i+1} \in C_i] \leq \epsilon_i + 2^{-\ell}$, since M_{i+1}^i is ϵ_i-close to min-entropy k_i. If $D_i = \{m : M_{i+1}(m^i) < 2^{-n_i-\ell}\}$. Get also get $\Pr[M_{i+1} \in D_i] < 2^{-\ell}$ by the union bound.

Thus, by the union bound, we get that $\Pr[X \in B] \leq \sum_{i=1}^{t} \Pr[M_{i+1} \in B_i] \leq \sum_{i=1}^{t} \Pr[M_{i+1} \in C_i] + \Pr[M_{i+1} \in D_i] \leq t \sum_i \epsilon_i + 2t2^{-\ell}$.

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

2.1.2 Somewhere Random Sources

Much of our work relies on ways to manipulate a class of sources called *somewhere random sources*. We discuss these in more detail in Chapter 3.

Definition 2.1.21 (Somewhere $\mathcal{P}$ sources)**.** Let $\mathcal{P}$ be a property of sources. A source X is $t \times r$ *somewhere-$\mathcal{P}$* if it is a distribution on $t \times r$ boolean matrices such that there is an i for which the i'th row X_i has property $\mathcal{P}$. If the property $\mathcal{P}$ is that the source is uniformly random, we will call the source somewhere random (SR-source for short).

Note that every $t \times r$ somewhere random source must have min-entropy at least r, since the random row itself has min-entropy r.

Definition 2.1.22. We will say that a collection of somewhere-$\mathcal{P}$ sources is *aligned* if there is some i for which the i'th row of every SR-source in the collection is uniformly distributed.

Definition 2.1.23 (Weak somewhere random sources)**.** A source X is $(t \times r)$ *k-somewhere-random* (*k-SR-source* for short) if it is a somewhere $\mathcal{P}$ source where a source has property $\mathcal{P}$ if it has min-entropy k.

Often we will need to apply a function to every row of a somewhere source. We will adopt the following convention: if $f : \{0,1\}^r \times \{0,1\}^r \to \{0,1\}^m$ is a function and a, b are samples from $t \times r$ somewhere sources, $f(\vec{a}, \vec{b})$ refers to the $t \times m$ string whose first row is obtained by applying f to the first rows of a, b and so on. Similarly, if a is an element of $\{0,1\}^r$ and b is a sample from a $t \times r$ somewhere source, $f(a, \vec{b})$ refers to the $t \times m$ matrix whose ith row is $f(a, b_i)$.

Many times we will treat a sample of a somewhere random source as a set of strings, one string from each row of the source.

Sometimes our constructions will need to take a small subset of the bits of a somewhere random source, called a *slice*:

Definition 2.1.24. Given ℓ strings of length n, $x = x_1, \ldots, x_n$, define $\mathsf{Slice}(x, w)$ to be the string $x' = x'_1, \ldots, x'_n$ such that for each i x'_i is the prefix of x_i of length w.

2.1.3 Affine Sources

Definition 2.1.25 (Affine Source)**.** A source X is called an *affine source* if it gives uniformly random point in some affine subspace $V \subset \mathbb{F}^n$ of a vector space over a finite field $\mathbb{F}$.

Note that for an affine source X, $H_\infty(X) = H(X)$, i.e., the min-entropy and entropy are the same.

Definition 2.1.26 (Affine Somewhere Random Source)**.** A source X is called an *affine $t \times r$ somewhere random source* if it is a distribution over $t \times r$ matrices with entries from a finite field $\mathbb{F}$, such one row X_i of the source is uniformly distributed.

The following basic lemma will be key to our results about affine sources:

Lemma 2.1.27 (Affine Conditioning)**.** *Let X be any affine source on $\{0,1\}^n$ with entropy k. Let $L : \{0,1\}^n \to \{0,1\}^m$ be any linear function. Then there exist independent affine sources A, B such that:*

- *$H(A) \leq m$.*
- *$H(B) \geq k - m$.*
- *$X = A + B$.*
- *For every $b \in \mathsf{supp}(B)$, $L(b) = 0$.*

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). Let B be the linear source whose support is $\{x \in \mathsf{supp}(x) : L(x) = 0\}$. Let $b_1, \ldots, b_t$ be a basis for B. Then we can complete this basis to get a basis for X. Let A be the span of the basis vectors in the completed basis that are not in B. Thus $X = B + A$.

Note that $H(A) \leq H(L(A))$ since $L(a) \neq 0$ for every $a \in \mathsf{supp}(A)$. Thus, $H(A) \leq m$. This then implies that $H(B) \geq H(X) - H(A) \geq k - m$. □

2.1.4 Small Space Sources

Definition 2.1.28. A *space s source* X on $\{0,1\}^n$ is a source generated by a width 2^s branching program. That is, the branching program is viewed as a layered graph with $n+1$ layers with a single start vertex in the first layer and 2^s vertices in each subsequent layer. Each edge is labeled with a probability and a bit value. From a single vertex we can have multiple edges corresponding to the same output bit. The source is generated by taking a random walk starting from the start vertex and outputting the bit values on every edge.

2.2 Convex Combinations

We say that a distribution X is a *convex combination* of distributions $X_1, X_2, \dots$ if there exist positive constants $\alpha_1, \alpha_2, \dots$ with the property that

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

For example, if X, Y are random variables in the same probability space, then we see that the distribution of X is a convex combination of the distributions $X|Y = y$, for every $y \in \mathsf{supp}(Y)$.

A key observation that is essential to many of our results is that random variables that are convex combinations of sources with some good property are usually good themselves. This is captured in the following easy propositions:

Definition 2.2.1. Let $\mathcal{P}$ be a property of sources. Let X be some random variable over some universe. We will say that X is a *convex combination* of sources with property $\mathcal{P}$ if there exists some random variable I over an arbitrary universe s.t. for all $i \in \mathsf{supp}(I)$, $X|I = i$ has property $\mathcal{P}$.

The following proposition is used implicitly in many of our arguments:

Proposition 2.2.2 (Preservation of properties under convex combination)**.** *Let A, B, Q be random variables 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 $|A - B| < \epsilon_1 + \epsilon_2$.

Proof.

$$\begin{aligned}
&|A - B| \\
&= \sum_{x \in \mathsf{Supp}(A) \cup \mathsf{Supp}(B)} \left| \Pr[A = x] - \Pr[B = x] \right| \\
&\leq \sum_{q \in \mathsf{Supp}(Q)} \Pr[Q = q] \sum_{x \in \mathsf{Supp}(A) \cup \mathsf{Supp}(B)} \left| \Pr[A = x | Q = q] - \Pr[B = x | Q = q] \right| \\
&\leq \epsilon_2 + \sum_{\text{good } q \in \mathsf{Supp}(Q)} \Pr[Q = q] \epsilon_1 \\
&\leq \epsilon_1 + \epsilon_2
\end{aligned}$$

□

This proposition will be used to simply the proofs that many of our constructions are *strong*. Usually it is easy to show that our output distributions are convex combinations of distributions that are close to having the strong property. Here we show that this implies that the output actually has the strong property. Think of X in the lemma below as one of the inputs, Q as some internal random variable, O as the output of our extractor/disperser and U as some independent random variable that is uniformly distributed. Then the following lemma is just a special case of the above proposition. In words it just says that if X is close to being independent of O for each subsource in the convex combination, it is fact close to being independent of O after the convex combination.

Lemma 2.2.3 (Preservation of strongness under convex combination)**.** *Let X, O, U, Q be random variables over the same finite probability space, with U, O both random variables over $\{0,1\}^m$. Let $\epsilon_1, \epsilon_2 < 1$ be constants s.t. :*

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

i.e., conditioned on Q being fixed and good*, X, O is statistically close to X, U.*
Then we get that $|X, O - X, U| < \epsilon_1 + \epsilon_2$.

To see why this last proposition is useful, imagine that X is the output of some extractor and Z is the uniform distribution. Then the proposition asserts that to show that the output of the extractor is close to uniform, it is sufficient to argue that the output is a convex combination of distributions that are close to uniform.

Proposition 2.2.4. *Let X, I be random variables s.t. X is a convex combination of random variables $\{X_i\}_{i \in I}$. Let f be some function s.t. for all $i \in I$, $f(X_i)$ is a convex combination of sources that have some property $\mathcal{P}$. Then $f(X)$ is a convex combination of sources that have property $\mathcal{P}$.*

Typically, f in the above proposition will be an extractor or some intermediate algorithm. We use the proposition to argue that if we need to argue that the output of the distribution of the extractor is close to uniform, it suffices to break the input source into a convex combination of *nice* sources, and then argue that on each of these nice sources, the function does what we want it to do. Then the above proposition implies that the final output is a convex combination of distributions that are close to uniform, and hence the entire output distribution is close to uniform.

Given any set $S \subset \{0,1\}^n$, and a distribution

$$\Pr[X = x] = \begin{cases} 1/|S| & x \in S \\ 0 & x \notin S \end{cases}$$

we call this distribution the *flat* distribution over S. The following fact will sometimes be useful:

Fact 2.2.5. *Every distribution X with min-entropy at least k is a convex combination of flat distributions with min-entropy k.*

This implies in particular that any extractor that is designed to work for min-entropy k will work even if the min-entropy is greater than k.

2.3 Extractors, Dispersers and other Manipulators of Independent Sources

In this section we define some of the objects we will later use and construct. All of these objects will take one or more inputs and produce one output, such that under particular guarantees on the distribution of the input, we'll get some other guarantee on the distribution of the output. Various interpretation of this vague sentence lead to objects called extractors and dispersers.

Definition 2.3.1 (C-source extractor)**.** Let $n_1, n_2, \ldots, n_C, k_1, k_2, \ldots, k_C, m, \epsilon$ be some numbers. A function $\mathsf{Ext} : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \cdots \times \{0,1\}^{n_C} \to \{0,1\}^m$ is called a *C-source extractor* with $k_1, k_2, \ldots, k_C$ min-entropy requirement, and error ϵ if for every independent sources $X_1, X_2, \ldots, X_C$ over $\{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \cdots \times \{0,1\}^{n_C}$ satisfying

$$\forall i, H_\infty(X_i) \geq k_i \tag{2.2}$$

it holds that

$$\left| \mathsf{Ext}(X, Y) - U_m \right| \leq \epsilon \tag{2.3}$$

We will say that a function is a C-source extractor for min-entropy k if it satisfies the above definition with $k_1 = k_2 = \cdots = k_C = k$.

A *seeded extractor* is just a special case of the above definition:

Definition 2.3.2 (Seeded Extractor)**.** A function $\mathsf{Ext} : \{0,1\}^{n_1 \times n_2} \to \{0,1\}^m$ is called a *seeded extractor* for min-entropy k and error ϵ if it is a 2-source extractor with k, n_2 min-entropy requirement and error ϵ.

A non-trivial construction will satisfy of course $n_2 \ll m$ (and hence also $n_2 \ll k_1 < n$). Thus, two source extractors are strictly more powerful than seeded extractors. However, the reason seeded extractors are more popular is that they suffice for many applications, and that (even after the work in this thesis) the explicit constructions for seeded extractors have much better parameters than the explicit constructions for 2-source extractors with $k_1 \ll n_1$, $k_2 \ll n_2$. (Note that this is not the case for *non-explicit* construction, where 2-source extractors with similar parameters to the best possible seeded extractors can be shown to exist using the probabilistic method.)

Variants. We'll use many variants of extractors in this thesis to various similar combinatorial objects. Most of the variants are obtained by giving different the conditions on the input (Equation 2.2) and the guarantee on the output (Equation 2.3). Some of the variants we will consider will be:

Dispersers. In dispersers, the output guarantee (Equation 2.3) is replaced with $|\mathsf{Supp}(\mathsf{Ext}(X, Y))| \geq (1 - \epsilon)2^m$,

Somewhere extractors. In *somewhere extractors* the output guarantee (Equation 2.3) is replaced with the requirement that $|\mathsf{Ext}(X, Y) - Z| < \epsilon$ where Z is a *somewhere random source* of $t \times m$ rows for some parameter t.

Extractors for block sources. In *extractors for block sources* the input requirement (Equation 2.2) is replaced with requirement that X and Y are *block sources* of specific parameters. Similarly we will define extractors for other families of inputs (i.e., somewhere random sources) and extractors where each input should come from a different family.

Strong extractors. Many of these definition have also a *strong* variant, and typically constructions for extractors also achieve this strong variant. A 2-source extractor is *strong in the first input* if the output requirement (Equation 2.3) is replaced with $\Pr_{x \leftarrow_{\mathrm{R}} X}[|\mathsf{Ext}(x, Y) - U_m| \geq \epsilon] \leq \epsilon$. This is equivalent (upto replacing ϵ by $\sqrt{\epsilon}$) to the statement:

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

where here X is independent of U_m.

We define an extractor to be strong in the second input similarly. If the extractor is strong in both inputs, we simply say that it is *strong*.

We say that a seeded extractor is *strong* if it is strong in the second input. Another way to view a strong seeded extractor is as a family of 2^t deterministic extractors, one for each seed in $\{0,1\}^t$. In this view, the strong seeded extractor property asserts that for any fixed source X, most functions in the family are good deterministic extractors for the source.

We say that a seeded extractor is *linear* if for every fixing of the seed, the resulting function is a linear function over $GF(2)$.

Remark 2.3.3 (Input lengths)**.** Whenever we have a C-source extractor $\mathsf{Ext} : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \cdots \times \{0,1\}^{n_C} \to \{0,1\}^m$ with inputs lengths $n_1, n_2, \ldots, n_C$ and min-entropy requirement $k_1, k_2, \ldots, k_C$ we can always invoke it on shorter sources with the same entropy, by simply padding it with zeros. For example if we have an extractor with $n_1 = \cdots = n_C$ we can still invoke it on inputs of *unequal* length by padding one of the inputs. The same observation holds for the other source types we'll use, namely *block* and *somewhere random* sources, if the padding is done in the appropriate way (i.e., pad each block for block sources, add all zero rows for somewhere random sources), and also holds for all the other extractor-like objects we consider (dispersers, somewhere extractors, and their subsource variant). In the following, whenever we invoke an extractor on inputs shorter than its "official" input length, this means that we use such a padding scheme.

The following proposition is immediate:

Proposition 2.3.4. *Let* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a strong seeded extractor for min-entropy* k *with error* ϵ*. Let* X *be any* (n,k) *source. Let* $\{0,1\}^d = \{s_1, s_2, \ldots, s_{2^d}\}$*. Then the matrix whose i'th row is* $\mathsf{Ext}(X, s_i)$ *is* ϵ*-close to a* $2^d \times m$ *somewhere random source.*

Observe that if we use a strong seeded extractor to turn several general sources into somewhere random sources, doing this actually gives us aligned somewhere random sources. If the strong extractor that we used to convert the input general sources to SR-sources has error ϵ, at most ϵ fraction of the rows in each source are not ϵ-close to uniform. Thus, if we are given u sources, as long as $u\epsilon < 1$, we will have one aligned row in *every* source that is ϵ-close to uniform. These sources are $u\epsilon$-close to being the distribution of independent aligned SR-sources. This gives us the following proposition:

Proposition 2.3.5. *Let* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a seeded* (k,ϵ) *strong extractor. Let* $X^1, \ldots, X^u$ *be independent* (n,k) *sources, with* $u\epsilon < 1$*. Let* $\{0,1\}^d = \{s_1, s_2, \ldots, s_{2^d}\}$*. Let* Z^i *denote* $\mathsf{Ext}(X^i, s_1), \mathsf{Ext}(X^i, s_2), \cdots, \mathsf{Ext}(X^i, s_{2^d})$*. Then* $Z^1, \cdots, Z^u$ *is* $u\epsilon$*-close to the distribution of* u *independent aligned* $2^d \times m$ *SR-sources.*

If Ext is a strong seeded extractor with seed length $O(\log n)$ and output length m, we can use Proposition 2.3.5 to convert u independent sources into u independent aligned $\mathrm{poly}(n) \times m$ SR-sources.

It will be useful to have the following proposition:

Proposition 2.3.6. *Let* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a strong linear seeded extractor with error* $\epsilon < 1/2$*. Let* X *be any affine source with entropy* k*. Then,*

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

Proof. Note that if X is an affine source, for every linear function $L : \{0,1\}^n \to \{0,1\}^m$, $L(X)$ is also an affine source. Thus we have that $|L(X) - U_m| = 0$ or $|L(X) - U_m| \geq 1/2$. Since for every fixed u, $\mathsf{Ext}(\cdot, u)$ is a linear function, this implies that:

$$\begin{aligned}
&\Pr_{u \leftarrow_R U_d}[|\mathsf{Ext}(X,u) - U_m| = 0] \\
&\qquad = \Pr_{u \leftarrow_R U_d}[|\mathsf{Ext}(X,u) - U_m| < 1/2] \\
&\qquad \geq \Pr_{u \leftarrow_R U_d}[|\mathsf{Ext}(X,u) - U_m| < \epsilon] \\
&\qquad \geq 1 - \epsilon
\end{aligned}$$

□

2.4 Network Extractors

In Chapter 8, we discuss the construction of *network extractors*.

Before defining this, we fix some notation. Player i begins with an input weakly-random sample $x_i \in \{0,1\}^n$ and ends in possession of a hopefully-random sample $z_i \in \{0,1\}^m$. Let b be the concatenation of all publicly broadcasted strings in the protocol. Capital letters such as Z_i and B denote these strings viewed as random variables.

Definition 2.4.1 (Network Extractor)**.** A protocol is a (t, g, ϵ) *network extractor* for min-entropy k if for any min-entropy k independent sources $X_1, \ldots, X_n$ over $\{0,1\}^n$ and any choice of t faulty players, after running the protocol, the number of players 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 say that a protocol is a *network extractor for block sources* if each player is assumed to have access to a block source.

2.5 Extractors via The Probabilistic Method

In this section we give a standard argument that shows that a randomly chosen function is an extractor for any class of sources that has *low complexity* in some sense, as long as the sources in the class are close to having high min-entropy.

Theorem 2.5.1. *Suppose we have a set $\mathcal{X}$ of random sources on $\{0,1\}^n$ and $\epsilon > 0$ such that $\forall X \in \mathcal{X}$, there is a source X' with $|X' - X| \leq \frac{\epsilon}{2}$ and $H_\infty(X') \geq k$. Then, with probability $1 - 1/2^{2^m}|\mathcal{X}|$ a function chosen uniformly at random is an extractor for $\mathcal{X}$ as long as $k \geq \log(2^m + \log|\mathcal{X}|) + 2\log(1/\epsilon) + O(1)$. In particular, as long as $k \geq \log\log|\mathcal{X}| + 2\log(1/\epsilon) + O(1)$, we can extract $m = k - 2\log(1/\epsilon) - O(1)$ bits.*

We need the following Chernoff bound to prove Theorem 2.5.1.

Lemma 2.5.2. *Let $Z_1, \ldots, Z_r$ be independent indicator random variables such that $\Pr[Z_1 = 1] = p_i$. Let $Z = \sum_{i=1}^n a_i Z_i$ where $0 \leq a_i \leq 1$ for all i, and let $\mu = \mathbb{E}[Z]$. Then for any $0 < \epsilon \leq 1$*

$$\Pr[|Z - \mu| \geq \epsilon\mu] < 2\exp(-\mu\epsilon^2/3).$$

Proof. (of Theorem 2.5.1) We'll first use Lemma 2.5.2 to show that a random function is a good extractor for a single source, and then apply the union bound.

Let $f : \{0,1\}^n \to \{0,1\}^m$ be chosen uniformly at random from all functions from n bits to m bits. Fix $X \in \mathcal{X}$ and $S \subset \{0,1\}^m$. Let X' be such that $|X' - X| \leq \epsilon/2$ and $H_\infty(X') \geq k$. Let Z_x be the indicator random variable for whether $f(x) \in S$. Let

$$Z = 2^k \Pr_{x \leftarrow_{\mathrm{R}} X'}[f(x) \in S] = 2^k \sum_{x \in \mathsf{supp}(X')} \Pr[X' = x] Z_x$$

Since the function f is chosen uniformly at random, $E[Z] = 2^k|S|/2^m$. Thus we can apply

Lemma 2.5.2 to get

$$\Pr_f\left[\left|\Pr_{x\in X'}[f(x)\in S]-\frac{|S|}{2^m}\right|\geq \epsilon'\frac{|S|}{2^m}\right]=\Pr_f\left[\left|Z-\frac{2^k|S|}{2^m}\right|\geq \epsilon'\frac{2^k|S|}{2^m}\right]\leq 2\exp\left(-\epsilon'^2\frac{2^k|S|}{3\cdot 2^m}\right)$$

Making the change of variables $\epsilon' = \epsilon 2^m/|S|$, we get that for any fixed set S, we proved that

$$\Pr_f[|\Pr[f(X')\in S]-\Pr[U_m\in S]|\geq \epsilon/2]\leq 2\exp\left(-\left(\frac{\epsilon 2^m}{2|S|}\right)^2\frac{2^k|S|}{3\cdot 2^m}\right)=2\exp\left(-\frac{\epsilon^2 2^k 2^m}{12|S|}\right)$$

Recall that $|f(X')-U_m|=\max_S\{|\Pr[f(X')\in S]-|S|/2^m|\}$. By the union bound over all sets $S\subset\{0,1\}^m$ and all $X\in\mathcal{X}$, and since $2^m/|S|\geq 1$,

$$\Pr_f[\max_S\{|f(X')-U_m|\geq \epsilon/2\}]\leq 2\exp\left(-\epsilon^2 2^k/12\right)2^{2^m}|\mathcal{X}|$$

Now whenever f does satisfy $|f(X')-U_m|<\epsilon/2$, we have that $|f(X)-U_m|<\epsilon/2+\epsilon/2=\epsilon$. Setting the above error to $1/2^{2^m}|\mathcal{X}|$ and solving for k, we get that a function chosen uniformly at random is an extractor for $|\mathcal{X}|$ with probability $1-1/2^{2^m}|\mathcal{X}|$ as long as $k\geq\log(2^m+\log|\mathcal{X}|)+2\log(1/\epsilon)+O(1)$. In particular, as long as $k\geq\log\log|\mathcal{X}|+2\log(1/\epsilon)+O(1)$, we can extract $m=k-2\log(1/\epsilon)-O(1)$ bits.

□

2.6 Previous Work that We Use

2.6.1 Seeded Extractors

Here we list the previous constructions of seeded extractors that we will use in this thesis.

Theorem 2.6.1 ([LRVW03]). *For any constant $\alpha\in(0,1)$, every $n\in\mathbb{N}$ and $k\leq n$ and every $\epsilon\in(0,1)$ where $\epsilon > exp(-\sqrt{k})$, there is an explicit (k,ϵ) seeded extractor* $\mathsf{Ext}:\{0,1\}^n\times\{0,1\}^{O(\log n+\log(n/k)\log(1/\epsilon))}\to\{0,1\}^{(1-\alpha)k}$.

Theorem 2.6.2 ([Tre01, RRV02]). *For every $n,k,m\in\mathbb{N}$ and $\epsilon>0$, such that $m\leq k\leq n$, there is an explicit (k,ϵ)-strong seeded extractor* $\mathsf{Ext}:\{0,1\}^n\times\{0,1\}^d\to\{0,1\}^m$ *with* $d=O\left(\frac{\log^2(n/\epsilon)}{\log(k/m)}\right)$.

We shall be interested in the following two instantiations of this theorem, obtained by setting the parameters appropriately:

Corollary 2.6.3 ([Tre01, RRV02]). *For every $n\in\mathbb{N}$, constants $r>0,\gamma<1$, there is an explicit (n^γ,n^{-r})-strong seeded extractor* $\mathsf{Ext}:\{0,1\}^n\times\{0,1\}^d\to\{0,1\}^{n^{\gamma'}}$ *with* $d=O(\log(n))$.

Corollary 2.6.4 ([Tre01, RRV02]). *For every $n,k\in\mathbb{N}$, there is an explicit (k,ϵ)-strong seeded extractor* $\mathsf{Ext}:\{0,1\}^n\times\{0,1\}^d\to\{0,1\}^{\Omega(k)}$ *with* $d=O(\log^2(n/\epsilon))$.

The first instantiation will be used when we need an extractor that has a good seed length. The second will be used when we need an extractor that has good output length.

If we need to get almost all of the randomness in the source out, the following corollary is available. This extractor also happens to be a *linear* seeded extractor over $GF(2)$, i.e., for every fixing of the seed, the extractor is a linear function on the source.

Corollary 2.6.5 ([Tre01, RRV02]). *For every $n, k \in \mathbb{N}$, $\epsilon > 0$, there is an explicit strong seeded extractor* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-O(\log^3(n/\epsilon))}$ *for min-entropy k and error ϵ, with $d = O(\log^3(n/\epsilon))$.*

2.6.2 Extractors for Two Independent Sources

Theorem 2.6.6 ([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 $2^{-1.5m}$.*

Bourgain constructed a strong 2-source extractor for min-entropy rate slightly less than half, for which we have written an exposition in Appendix C.

Theorem 2.6.7 ([Bou05]). *There exists a universal constant $\gamma > 0$ and a polynomial time computable function* $\mathsf{Bou} : (\{0,1\}^n)^2 \to \{0,1\}^m$ *s.t. if X, Y are two independent $(n, (1/2 - \gamma)n)$ sources, $\mathbb{E}_Y[\|\mathsf{Bou}(X,Y) - U_m\|_{\ell^1}] < \epsilon$, with $\epsilon = 2^{-\Omega(n)}, m = \Omega(n)$.*

In fact, it's easy to show that Bourgain's extractor can be extended to the more general case when we have several independent sources, but only two of them have entropy. We sketch the proof for this in Appendix C.

Theorem 2.6.8. *There exists a universal constant $\gamma > 0$ and a polynomial time computable function* $\mathsf{Bou} : (\{0,1\}^n)^2 \to \{0,1\}^m$ *s.t. if $X_1, X_2, \ldots, X_t$ are t independent n-bit sources with min-entropies $k_1, k_2, \ldots, k_t$ and $k_i + k_j \geq 2n(1/2 - \gamma)$ for some $i, j \in [t]$,*

$$\mathbb{E}_{X_i}[\|\mathsf{Bou}(X_1, \ldots, X_t) - U_m\|_{\ell^1}] < \epsilon$$

$$\mathbb{E}_{X_j}[\|\mathsf{Bou}(X_1, \ldots, X_t) - U_m\|_{\ell^1}] < \epsilon$$

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

2.6.3 Extractors for Affine sources

Theorem 2.6.9 ([Bou07]). *For every constant $\delta > 0$, there exist constants $\gamma, \beta > 0$ and a polynomial time computable function* $\mathsf{Bou} : \{0,1\}^n \to \{0,1\}^{\beta n}$ *s.t. for every affine source X of entropy δn,* $\mathsf{Bou}(X)$ *is $2^{-\gamma n}$-close to uniform.*

Chapter 3

Condensers for Somewhere Random Sources

A common theme in most of the results in this thesis is that they all rely on ways to manipulate a special class of sources of randomness, called *somewhere random* sources. In this chapter we discuss these basic techniques. The reader may choose to skip ahead to one of the later chapters where the techniques from this chapter are applied, to get more motivation for the algorithms designed in this chapter.

We start with the definition of a somewhere random source.

Definition 3.0.10 (Somewhere Random Sources)**.** [TS96][1] A source X is $t \times r$ *somewhere-random* (*SR-source* for short) if it is a distribution on $t \times r$ boolean matrices, s.t. one of the rows in the matrix is uniformly random.

Every other row in the matrix may depend on the random row in arbitrary ways. To give some motivation for this definition, let us step back and reexamine our ultimate goal. What we are trying to do is find the most general *class* of sources of randomness that would admit a *single* deterministic polynomial time extractor algorithm. Consider the following informal definition:

Informal Definition 3.0.11 ((d, ϵ)-extractable sources)**.** We say that randomness is (d, ϵ)*-extractable* from a class of sources $\mathcal{C}$, if there exists a polynomial time algorithm that computes a function $F : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that for every source $X \in \mathcal{C}$, there exists an advice string $a \in \{0,1\}^d$ such that $F(X, a)$ is ϵ-close to uniform.

We can rephrase the extractor project as trying to come up with the most general class of sources $\mathcal{C}$ from which randomness is $(0, \epsilon)$-extractable. Somewhere random sources are intimately connected with this definition, since they are *complete* for the class of sources from which randomness is (d, ϵ)-extractable, in the following sense:

- For every class $\mathcal{C}$ from which randomness is (d, ϵ)-extractable, there exists a polynomial time algorithm that converts every X belonging to $\mathcal{C}$ into a $2^d \times m$ somewhere random source (or

[1] This definition is slightly different from the original one used by Ta-Shma [TS96]. The original definition considered the closure under convex combinations of the class defined here (i.e., convex combinations of sources that have one random row). We use this definition because we can do so without loss of generality and it considerably simplifies the presentation.

at least something that is ϵ-close). We can do this by running F on the sample from the source 2^d times, once with every possible advice string a, i.e., the a'th row of the somewhere random source is $F(X, a)$.

- The class of $2^d \times m$ somewhere random sources is itself a class from which randomness is trivially $(d, 0)$-extractable — the required advice string a simply tells the algorithm F which row of the somewhere random source to output.

Given the above discussion, it is natural to lower our goals and ask only for a class of sources from which randomness is (d, ϵ) extractable for small d, with the hope that we can incrementally lower the required d, until we bring it down to 0. Indeed, this broad idea, which we refer to as *condensing*[2] somewhere random sources, plays an important role in most of our results.

Luckily, earlier work on constructing *strong seeded extractors* (Section 2.3) already gives us a starting point to turn this vague idea into reality. Any polynomial time computable strong seeded extractor can be seen as evidence that randomness is (d, ϵ)-extractable from *every* (n, k) source, for small d, ϵ. This is captured in the following proposition, which is immediate from definitions:

Proposition 3.0.12. *Let* $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \rightarrow \{0,1\}^m$ *be a strong seeded* (k, ϵ) *extractor. Let* X *be any* (n, k) *source. Let* $\{0,1\}^d = \{s_1, s_2, \ldots, s_{2^d}\}$. *Then* $\mathsf{Ext}(X, s_1), \mathsf{Ext}(X, s_2), \cdots, \mathsf{Ext}(X, s_{2^d})$ *is* ϵ*-close to a* $2^d \times m$ *SR-source.*

Using any good seeded strong extractor with seed length $O(\log n)$ (for instance Theorem 2.6.1), we can turn any (n, k) source into a somewhere random source with $\mathrm{poly}(n)$ rows.

In the rest of this chapter we give several constructions involving somewhere random sources. The common theme in all of them will be exactly as we discussed above. We will build simple algorithms that can turn somewhere random sources that have many rows into somewhere random sources with fewer rows. Sometimes we will even be able to reduce the number of the rows in the output to 1, which means the algorithm is an extractor.

3.1 Condensing Aligned Independent Somewhere Random Sources

Now we discuss how to build condensers for multiple independent somewhere random sources. We will give ways to combine several such independent sources into fewer somewhere random sources that have fewer rows.

An important concept we will need is that of *aligned* somewhere random sources.

Definition 3.1.1. We will say that a collection of $t \times r$ SR-sources $X^1, \ldots, X^u$ is *aligned* if there is some i for which the i'th row of every SR-source in the collection is uniformly distributed.

3.1.1 A simple condenser for 2 somewhere random sources

The first condenser we consider is extremely simple — we use the xor function to condense two independent aligned $t \times r$ somewhere random sources into one $t/2 \times r$ somewhere random source.

[2] The term *condensing* is adapted from the literature on seeded extractors (see [RSW00]). In that context it was used to denote the analogous operation of increasing the min-entropy rate of a weak random source.

Algorithm 3.1.2 ($\mathsf{XORCondense}(x, y)$).

Input: x, y two $t \times r$ matrices.
Output: z, a $\lceil t/2 \rceil \times r$ matrix.

```
1. For all $i = 1, 2, \ldots, \lceil t/2 \rceil$ and $j = 1, 2, \ldots, r$, set $z_{i,j} = x_{2i-1,j} \oplus y_{\min\{2i,t\},j}$.
```

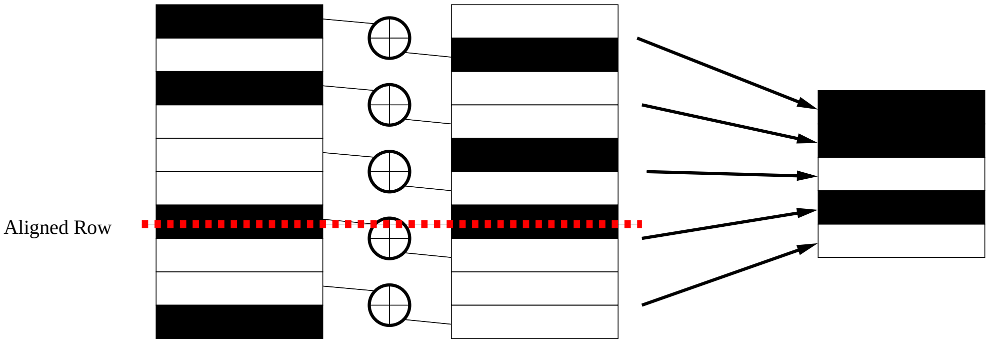

Figure 3.1: $\mathsf{XORCondense}$ — A condenser for two independent aligned somewhere random sources. Black rows correspond to rows that are uniform.

Proving that the above algorithm is a condenser is easy:

Proposition 3.1.3. *If X, Y are two independent aligned $t \times r$ somewhere random sources, $\mathsf{XORCondense}(X, Y)$ is a $\lceil t/2 \rceil \times r$ somewhere random source.*

Proof. Since X, Y are aligned, there must be an index h for which h'th rows X_h and Y_h are both uniform. Then we get that the $\lceil h/2 \rceil$'th row of $\mathsf{XORCondense}(X, Y)$ is uniform, since it is obtained by computing the bitwise xor of a sample that is uniform with something that is independent of it. □

Remark 3.1.4. Of course, we could have used any group operation (instead of the bitwise xor) to achieve the same effect for this algorithm.

The above construction suggests a general approach to building condensers for somewhere random sources — given any extractor for C independent aligned $c \times r$ SR-sources, we can get a condenser that condenses C independent aligned $t \times r$ SR-sources into a single $\lceil t/c \rceil \times m$ SR-source, where m is the output length of the extractor. We simply break up the rows of the somewhere random sources into $\lceil t/c \rceil$ equal groups, each containing at most c rows, and apply the extractor to each group to get each of the rows of the output.

3.1.2 Better Condensers

In this section we show how to use the basic condensers that we have developed so far to get a much better condenser for aligned independent somewhere random sources. We will construct a condenser that can condense just two independent aligned $t \times r$ somewhere random sources into a single $\lceil t/r^{0.7} \rceil \times (r - r^{0.9})$ somewhere random source. To illustrate the main ideas, we first give a somewhat simpler condenser for three independent aligned somewhere random sources. As we saw in our previous discussion, to do this, it suffices to build an extractor for three independent aligned $r^{0.7} \times r$ somewhere random sources that outputs $r - r^{0.9}$ bits. If we had such an extractor, we could get the promised condenser by breaking up the rows of our somewhere random sources into groups of $r^{0.7}$ rows and then applying our extractor to each group.

Our extractor itself will be built by repeatedly condensing the sources we are working with, with a crucial difference from our previous constructions — each step will retain as many independent sources as it started out with. In each step we will manipulate the three independent SR-sources we are working with to get three new SR-sources that are still (essentially) independent, aligned and have fewer rows than we started out with. Repeating this procedure, we will eventually be left with three somewhere random sources each having only one row.

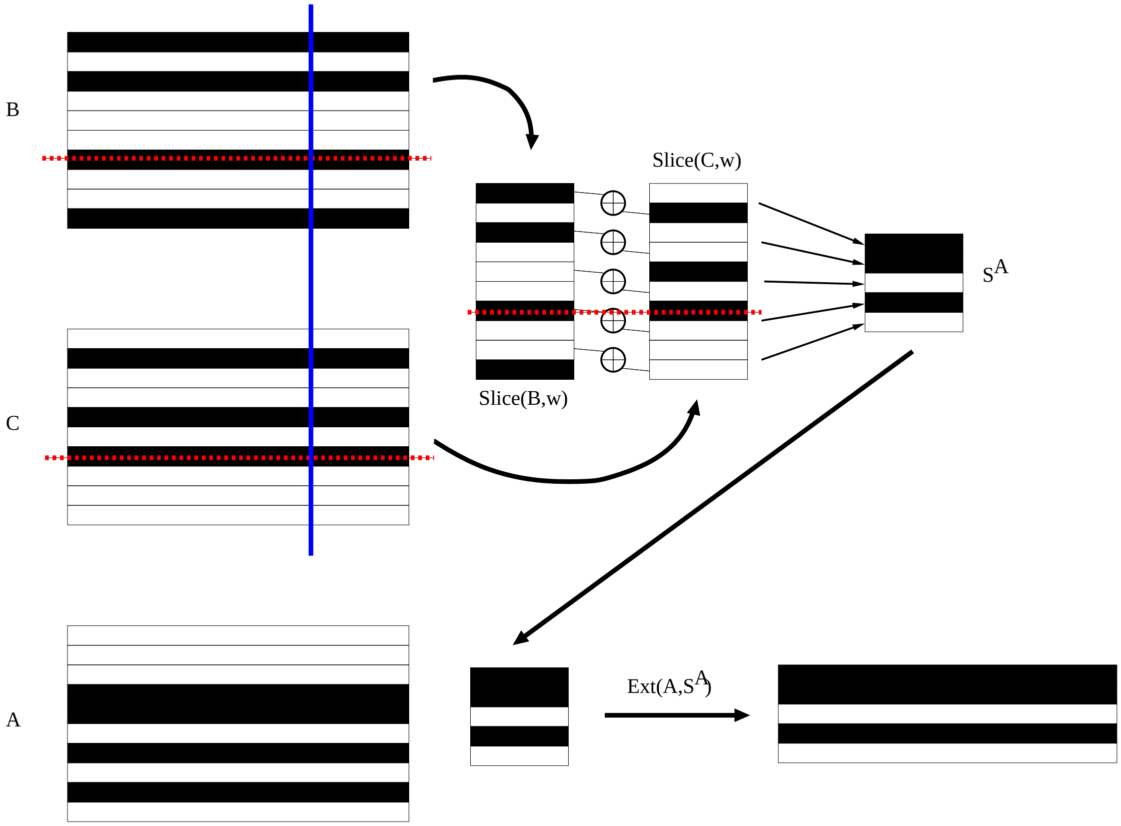

Figure 3.2: Using two sources to condense one source

Now let us describe one condensing step. Assume we are given three aligned independent $t \times r$ somewhere random sources A, B, C, with $t \leq r^{0.7}$. We first describe our construction in words. A precise description of the algorithm follows our word description. Given the three matrices coming from A, B, C, we output 3 new matrices. Each of the output matrices will be associated with one of the inputs, in the sense that the entropy for that output will come from the associated input source. To condense the source A, we take small slices $\mathsf{Slice}(B, w)$ and $\mathsf{Slice}(C, w)$ of the other two sources and then combine them together by $\mathsf{XORCondense}(\mathsf{Slice}(B, w), \mathsf{Slice}(C, w))$ to get a single somewhere random source with $t/2$ rows. We then use each row of this somewhere random source as a seed to a strong seeded extractor, to extract from A. In this way, we turn A into a somewhere random source with half the number of rows that we started out with.

In the same way, we use the slices $\mathsf{Slice}(B, w)$ and $\mathsf{Slice}(A, w)$ to condense C, and the slices $\mathsf{Slice}(A, w)$ and $\mathsf{Slice}(C, w)$ to condense A.

Conditioned on all the small sections of the sources that we've used as seed, we show that the condensing succeeds, we obtain 3 new SR-sources that have half the number of rows as the original sources. Since we're conditioning on the only part that's involved in the interactions between the sources, after conditioning, the output of the condensing step is a collection of independent sources. Iterating this condensing process, we will eventually obtain a single string that is statistically close to uniformly distributed.

Now let us be more precise.

Algorithm 3.1.5 ($\mathsf{ThreeCondense}(a, b, c)$)**.**

Input: a, b, c three $t \times r$ matrices with $t \leq r^{0.7}$
Output: x, y, z three $\lceil t/2 \rceil \times m$ matrices with $m \geq r - r^{0.9}$.

Sub-Routines and Parameters:
Let $w = r^{0.1}$.
Let $\mathsf{Ext} : \{0,1\}^{rt} \times \{0,1\}^{w} \rightarrow \{0,1\}^{m}$ be the strong seeded extractor from Corollary 2.6.5, set up to extract $m = r - r^{0.9}$ bits from a min-entropy $r - 100wr^{0.7}$ source with error $\epsilon = 2^{-r^{\Omega(1)}}$.
We will use Algorithm 3.1.2 — $\mathsf{XORCondense}$, set up to condense two $t \times r$ somewhere random sources.

1. Set $s^a = \mathsf{XORCondense}(\mathsf{Slice}(B, w), \mathsf{Slice}(C, w))$.
2. Set $s^b = \mathsf{XORCondense}(\mathsf{Slice}(A, w), \mathsf{Slice}(C, w))$.
3. Set $s^c = \mathsf{XORCondense}(\mathsf{Slice}(B, w), \mathsf{Slice}(A, w))$.
4. Let x be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{Ext}(a, s^a_i)$, i.e., it is obtained by applying Ext to a with the i'th row of s^a as seed.
5. Let y be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{Ext}(b, s^b_i)$
6. Let z be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{Ext}(c, s^c_i)$

Lemma 3.1.6. *If A, B, C are three independent aligned $t \times r$ somewhere random sources with $t \leq r^{0.7}$, the output of $\mathsf{ThreeCondense}(A, B, C)$ is $2^{-r^{\Omega(1)}}$-close to a convex combination of three independent aligned $\lceil t/2 \rceil \times m$ somewhere random sources.*

Proof. Define $A' = \mathsf{Slice}(A, w), B' = \mathsf{Slice}(B, w), C' = \mathsf{Slice}(C, w)$.

We will show that for typical a', b', c', the output of $\mathsf{ThreeCondense}(A, B, C)|A' = a', B' = b', C' = c'$ is $2^{-r^{\Omega(1)}}$-close to three independent aligned somewhere random sources.

We will bound the probability of several bad events. The first bad event is that a slice could steal too much entropy from the corresponding source. Since, each slice is a distribution on only $tw < r^{0.7}w$ bits, Proposition 2.1.10 implies that the probability that this happens is small:

$$\Pr_{a' \leftarrow_{\mathrm{R}} A'}[H_\infty(A|A'=a') \leq r - 100wr^{0.7}] < 2^{-99wr^{0.7}}$$

Note that A', B', C' are three independent aligned $\lceil t/2 \rceil \times w$ somewhere random sources. We get that S^a, S^b, S^c are three aligned $\lceil t/2 \rceil \times w$ somewhere random sources (though they aren't independent). Further, we have that S^a is independent of A, S^b is independent of B, and S^c is independent of C. Let h be the index of a good aligned row in these sources. Now for any a' that does not steal too much entropy, we call $s^a \in \mathsf{supp}(S^a)$ a bad seed for a', if $|\mathsf{Ext}(A|A'=a', s^a_h) - U_m| \geq \epsilon$. Then, since S^a is completely independent of A, by the definition of the strong seeded extractor Ext, we have that

$$\Pr_{s^a \leftarrow_{\mathrm{R}} S^a}[|\mathsf{Ext}(A|A'=a', s^a_h) - U_m| \geq \epsilon] < \epsilon$$

Thus, by the union bound, our arguments imply that with probability at least $1 - (3\epsilon + 3 \cdot 2^{-99wr^{0.7}}) = 1 - 2^{-r^{\Omega(1)}}$, each of a', b', c' doesn't steal too much entropy from the corresponding source and each of s^a, s^b, s^c are good seeds for a', b', c' respectively. We get that with probability $1 - 2^{-r^{\Omega(1)}}$ over the choice of a', b', c', the output of $\mathsf{ThreeCondense}(A, B, C)|A'=a', B'=b', C'=c'$ is $2^{-r^{\Omega(1)}}$-close to three independent aligned $\lceil t/2 \rceil \times m$ somewhere random sources. □

Given this lemma, it's easy to see how to get the extractor. We repeatedly condense the three sources until the number of rows drops to one. Since we reduce the number of rows by a factor of 2 every time, we only need to repeat this process $\log r$ times. Thus the error does not go up by a significant amount. Similarly, the output length remains at least $r - r^{0.95}$. We obtain the following thm:

Theorem 3.1.7 (Good Condenser). *There exists a polynomial time computable function* $\mathsf{Cond} : (\{0,1\}^{tr})^3 \rightarrow \{0,1\}^{tm}$ *with the property that given any A, B, C — three independent aligned $t \times r$ somewhere random sources, the output* $\mathsf{Cond}(A, B, C)$ *is $2^{-r^{\Omega(1)}}$-close to a $\lceil t/r^{0.7} \rceil \times m$ somewhere random source, with $m = 3r - r^{0.95}$.*

3.1.3 A strong condenser for 2 somewhere random sources

One weakness of the above constructions is that they aren't *strong*, i.e., they guarantees nothing about the output if we fix one of the inputs[3]. Some of our results will crucially rely on having a good strong condenser for two sources.

Definition 3.1.8 (Strong Condenser). We say that a function Cond that takes two $t \times r$ boolean matrices as input and outputs a $t' \times r'$ boolean matrix is a *strong condenser* for two independent aligned somewhere random sources with error ϵ if for every two independent aligned $t \times r$ somewhere random sources X, Y it holds that,

$$\Pr_{x \leftarrow_{\mathrm{R}} X}[\mathsf{Cond}(x, Y) \text{ is } \epsilon\text{-close to a } t' \times r' \text{ somewhere random source}] \geq 1 - \epsilon$$

[3] The 3 source condenser can be modified slightly to get a strong variant.

and

$$\Pr_{y \leftarrow_{\mathrm{R}} Y}[\mathsf{Cond}(X, y) \text{ is } \epsilon\text{-close to a } t' \times r' \text{ somewhere random source}] \geq 1 - \epsilon$$

In this section, we will build a strong condenser that can condense two independent aligned $t \times r$ somewhere random sources into a single $\lceil t/r^{0.7} \rceil \times r$ somewhere random source. As we discussed above, any extractor for two independent aligned $r^{0.7} \times r$ somewhere random sources would give us a condenser for two independent aligned $t \times r$ somewhere random sources with the parameters we need, so we will just construct such an extractor.

As in our last construction, we will obtain our extractor by repeated condensing, except this time we will not use $\mathsf{XORCondense}$ from Algorithm 3.1.2. Instead, we will need to use a strong extractor — Theorem 2.6.7. This gives a strong extractor for any two independent sources with min-entropy rate $1/2$; in particular it gives an extractor for two independent aligned $2 \times r$ somewhere random sources. The following algorithm is essentially obtained (at least conceptually), by replacing the use of $\mathsf{XORCondense}$ in Algorithm 3.1.5 with the two source extractor from Theorem 2.6.7.

Algorithm 3.1.9 ($\mathsf{TwoCondense}(a, b)$)**.**

Input: a, b — two $t \times r$ matrices with $t \leq r^{0.7}$.
Output: x, y — two $\lceil t/2 \rceil \times m$ matrices, with $m = r - r^{0.9}$.

Sub-Routines and Parameters:
Let $w = r^{0.1}$.
Let $\mathsf{Ext} : \{0,1\}^{rt} \times \{0,1\}^w \to \{0,1\}^m$ be the strong seeded extractor from Corollary 2.6.5, set up to extract $m = r - r^{0.9}$ bits from a min-entropy $r - 100wr^{0.7}$ source with error $\epsilon = 2^{-r^{\Omega(1)}}$.
Let $\mathsf{Bou} : \{0,1\}^{2w} \times \{0,1\}^{2w} \to \{0,1\}^d$ be the extractor from Theorem 2.6.7.
Recall the definition of a *slice* of a somewhere random source — Definition 2.1.24.

1. Let z^a be the $\lceil t/2 \rceil \times 2r$ matrix obtained by concatenating pairs of rows in $\mathsf{Slice}(a, w)$, i.e., the i'th row is $\mathsf{Slice}(a, w)_{2i-1}, \mathsf{Slice}(a, w)_{\min\{2i,t\}}$. Similarly, let z^b be the $\lceil t/2 \rceil \times 2r$ matrix whose i'th row is $\mathsf{Slice}(b, w)_{2i-1}, \mathsf{Slice}(b, w)_{\min\{2i,t\}}$.
2. Let s be the $\lceil t/2 \rceil \times d$ matrix whose i'th row is $\mathsf{Bou}(z^a_i, z^b_i)$.
3. Let x be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{Ext}(a, s_i)$. Similarly let y be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{Ext}(b, s_i)$.

The following lemma shows that indeed, for any A, B as above, the output X is essentially the result of condensing A, and the output Y is the result of condensing B.

Lemma 3.1.10. *For any two independent aligned $t \times r$ somewhere random sources A, B, with $t \leq r^{0.7}$, let X, Y be the outputs of* $\mathsf{TwoCondense}(A, B)$*. Then there exists a random variable Z and $\epsilon = 2^{-r^{\Omega(1)}}$ with the property that:*

- *X is a deterministic function of Z and A.*

- *Y is a deterministic function of Z and B.*
- $\Pr_{z \leftarrow_R Z}[(X,Y)|Z=z$ *are ϵ-close to two aligned somewhere random sources*$] \geq 1-\epsilon$.

Proof. Let A, B be any two independent aligned $t \times r$ SR-sources.

As in the algorithm, set Z^a to be the distribution on $\lceil t/2 \rceil \times 2r$ matrices obtained by concatenating pairs of rows in $\mathsf{Slice}(A,w)$, i.e., the i'th row is $\mathsf{Slice}(A,w)_{2i-1}, \mathsf{Slice}(A,w)_{\min\{2i,t\}}$. Similarly, let Z^b be the $\lceil t/2 \rceil \times 2r$ matrix whose i'th row is $\mathsf{Slice}(B,w)_{2i-1}, \mathsf{Slice}(B,w)_{\min\{2i,t\}}$.

Then we see immediately that X is a deterministic function of (Z,A) and Y is a deterministic function of (Z,B). To prove the remaining conclusion, we once again use the union bound.

First we bound the probability that the slices Z^a, Z^b steal too much entropy from the corresponding sources. As before, since Z^a and Z^b are distributions on only wt bits Proposition 2.1.10 gives

$$\Pr_{z^a \leftarrow_{\mathrm{R}} Z^a}[H_\infty(A|Z^a=z^a) \leq r - 100wr^{0.7}] < 2^{-99wr^{0.7}}$$

$$\Pr_{z^b \leftarrow_{\mathrm{R}} Z^b}[H_\infty(B|Z^b=z^b) \leq r - 100wr^{0.7}] < 2^{-99wr^{0.7}}$$

Observe that since A, B were independent aligned somewhere random sources, there must exist an index h for which Z^a_h, Z^b_h are two independent sources with min-entropy rate at least $1/2$. Fix such an h.

Set $\beta = 2^{-\Omega(w)}$ to be the error of Bourgain's extractor. Call $z^b \in \mathsf{supp}(Z^b)$ a *bad slice for* Bou, if $|\mathsf{Bou}(Z^a_h, z^b_h) - U_m| \geq \beta$. Since Bou is a strong extractor, we get that

$$\Pr_{z^b \leftarrow_{\mathrm{R}} Z^b}[|\mathsf{Bou}(Z^a_h, z^b_h) - U_m| \geq \beta] < \beta$$

Similarly,

$$\Pr_{z^a \leftarrow_{\mathrm{R}} Z^a}[|\mathsf{Bou}(Z^b_h, z^a_h) - U_m| \geq \beta] < \beta$$

Set γ to be the error of the seeded extractor Ext. For any z^b for which z^b is not a bad slice for Bou and z^b does not steal too much entropy, we see that

$$\begin{aligned}&\Pr_{u \leftarrow_{\mathrm{R}} U_d}[|\mathsf{Ext}(B|Z^b=z^b, u) - U_m| \geq \gamma] < \gamma \\ \Rightarrow &\Pr_{z^a \leftarrow_{\mathrm{R}} Z^a}[|\mathsf{Ext}(B|Z^b=z^b, \mathsf{Bou}(z^a_h, z^b_h)) - U_m| \geq \gamma] < \beta + \gamma\end{aligned}$$

since $|\mathsf{Bou}(Z^a_h, z^b_h) - U_m| < \beta$.

Now, as long as z^b doesn't steal too much entropy from Z^b, and $\mathsf{Bou}(z^a_h, z^b_h)$ gives a good seed to extract from $B|Z^b=z^b$, we get that $\mathsf{Ext}(B|Z^b=z^b, \mathsf{Bou}(z^a_h, z^b_h))$ is close to uniform. Thus, by the union bound, our previous discussion gives the bound

$$\begin{aligned}
&\Pr_{z \leftarrow_{\mathrm{R}} Z}[|\mathsf{Ext}(X_h|Z = z, \mathsf{Bou}(z_h^a, z_h^b)) - U_m| \geq \gamma] \\
&= \Pr_{z \leftarrow_{\mathrm{R}} Z}[|\mathsf{Ext}(B|Z = z, \mathsf{Bou}(z_h^a, z_h^b)) - U_m| \geq \gamma] \\
&= \Pr_{z^a \leftarrow_{\mathrm{R}} Z^a, z^b \leftarrow_{\mathrm{R}} Z^b}[|\mathsf{Ext}(B|Z^b = z^b, \mathsf{Bou}(z_h^a, z_h^b)) - U_m| \geq \gamma] \\
&< \gamma + 2^{-99rw} + \beta + \gamma \\
&< 2^{-r^{\Omega(1)}}
\end{aligned}$$

By the union bound, we get this conclusion holds for both sources, simultaneously with probability at least $1 - 2^{-r^{\Omega(1)}}$, as desired.

□

Given this condenser, we can simply apply it repeatedly to get an extractor.

Algorithm 3.1.11 ($\mathsf{StrongSRExt}(a, b)$)**.**

Input: a, b — two $t \times r$ matrices with $t \leq r^{0.7}$.
Output: z — an m bit string, with $m \geq r - r^{0.95}$.

```
1. If $a, b$ have only one row each, output the bitwise xor $a \oplus b$.
2. Else, set $x, y$ to be the output of $\mathsf{TwoCondense}(a, b)$.
3. Set $a = x, b = y$ and go to the first step.
```

Note that the loop runs at most $\log t$ times, since after that, the number of rows is reduced to 1. Thus the output has the promised length.

Given our work so far, the following theorem is easy:

Theorem 3.1.12. *Let A, B be two independent aligned $t \times r$ somewhere random sources with $t \leq r^{0.7}$. Then we have that*

$$\Pr_{a \leftarrow_R A}[|\mathsf{StrongSRExt}(a, B) - U_m| \geq \gamma] < \gamma$$

and

$$\Pr_{b \leftarrow_R B}[|\mathsf{StrongSRExt}(A, b) - U_m| \geq \gamma] < \gamma$$

for $\gamma = 2^{-r^{\Omega(1)}}$.

Proof. Let ϵ be the error in Lemma 3.1.10.

We prove that the theorem holds with $\gamma = 3\epsilon \log t$ by induction on t. When $t = 1$, the theorem is trivially true.

When $t > 1$, by Lemma 3.1.10, we get that there exists a random variable Z such that

- X is a deterministic function of Z and A.
- Y is a deterministic function of Z and B.
- $\Pr_{z \leftarrow_{\mathrm{R}} Z}[(X,Y)|Z{=}z$ are ϵ-close to two aligned somewhere random sources$] \geq 1-\epsilon$.

Then we have that

$$\begin{aligned}
&\Pr_{a \leftarrow_{\mathrm{R}} A}[|\mathsf{StrongSRExt}(a,B) - U_m| \geq 3\epsilon \log t] \\
&\Pr_{a \leftarrow_{\mathrm{R}} A}[|\mathsf{StrongSRExt}(X,Y)|A{=}a - U_m| \geq 3\epsilon \log t] \\
&= \sum_{z \in \mathsf{supp}(Z)} \Pr[Z=z] \Pr_{x \leftarrow_{\mathrm{R}} X|Z=z}[|\mathsf{StrongSRExt}(x,Y|Z{=}z) - U_m| \geq 3\epsilon \log t] \qquad (3.1)
\end{aligned}$$

Now, when z is such that $(X,Y)|Z{=}z$ are ϵ-close to two aligned somewhere random sources, we get by induction that

$$\begin{aligned}
&\Pr_{x \leftarrow_{\mathrm{R}} X|Z=z}[|\mathsf{StrongSRExt}(x,Y|Z{=}z) - U_m| \geq 3\epsilon \log t] \\
&\leq \Pr_{x \leftarrow_{\mathrm{R}} X|Z=z}[|\mathsf{StrongSRExt}(x,Y|Z{=}z) - U_m| \geq 3\epsilon \log \lceil t/2 \rceil] \\
&\leq 3\epsilon \log \lceil t/2 \rceil + \epsilon \qquad (3.2)
\end{aligned}$$

Since $\Pr_{z \leftarrow_{\mathrm{R}} Z}[(X,Y)|Z{=}z$ are ϵ-close to two aligned somewhere random sources$] \geq 1-\epsilon$, Equation 3.1 and Equation 3.2 imply that

$$\Pr_{a \leftarrow_{\mathrm{R}} A}[|\mathsf{StrongSRExt}(a,B) - U_m| \geq 3\epsilon \log t] < 3\epsilon \log \lceil t/2 \rceil + \epsilon + \epsilon < 3\epsilon \log t$$

The proof that the extractor is strong with respect to the other input is similar.

■

Given this extractor and our discussion above, we get a strong condenser for aligned somewhere random sources.

Theorem 3.1.13 (Strong Condenser)**.** *There exists a polynomial time computable function* Cond *that maps two $t \times r$ matrices to a single $\lceil t/r^{0.7} \rceil \times r - r^{0.95}$ matrix, and a constant $\alpha > 0$ such that for every two independent aligned $t \times r$ somewhere random sources X, Y, we have that*

$$\Pr_{x \leftarrow_R X}[\mathsf{Cond}(x,Y) \textit{ is } 2^{-r^{\alpha}}\textit{-close to a somewhere random source}] \geq 1 - 2^{-r^{\alpha}}$$

and

$$\Pr_{y \leftarrow_R Y}[\mathsf{Cond}(X,y) \textit{ is } 2^{-r^{\alpha}}\textit{-close to a somewhere random source}] \geq 1 - 2^{-r^{\alpha}}$$

Remark 3.1.14. We note that in this last theorem the constant 0.7 can be replaced with any constant $0 < \gamma < 1$. This would simply have the effect of changing the constants 0.9 and α to some other constants in the interval $(0,1)$.

Theorem 3.1.15. *For every constant $\gamma < 1$ and n, n', t with $t = t(n, n')$ s.t. $t < n^{\gamma}$ and $t < n'^{\gamma}$ there exists a constant $\alpha < 1$ and a polynomial time computable strong extractor $\overline{\mathsf{2SRExt}} : \{0,1\}^{tn} \times \{0,1\}^{tn'} \to \{0,1\}^m$ s.t. that succeeds when X is a $t \times n$ SR-source and Y is an independent aligned $t \times n'$ SR-source, with $m = \min[n, n'] - \min[n, n']^{\alpha}$ and error $2^{-\min[n,n']^{1-\alpha}}$.*

3.2 Condensing When Only Some Sources Are Good

In this section, we will use a slight twist on the main ideas from Section 3.1 to get a condenser that is strictly stronger than the one we obtained there — the condenser will function even if only *some* of the sources in the input have entropy. The theorem we will prove is this one:

Theorem 3.2.1 (Strong Condenser)**.** *For every t, r, C, with $C \leq r^{100}$, there exists a polynomial time computable function $\mathsf{Cond} : (\{0,1\}^{tr})^C \to \{0,1\}^{\lceil t/r^{0.7} \rceil (r - r^{0.95})}$ that maps a C tuple of $t \times r$ matrices to a single $\lceil t/r^{0.7} \rceil \times r - r^{0.95}$ matrix, and a constant $\alpha > 0$ such that if $X^1, \ldots, X^C$ are independent random variables over $t \times r$ matrices with the property that some X^i, X^j are independent aligned somewhere random sources, then*

$$\Pr_{x^h \leftarrow_R X^h, h \neq j}[\mathsf{Cond}(x^1, \ldots, X^j, \ldots, x^C) \textit{ is } 2^{-r^{\alpha}}\textit{-close to a somewhere random source}] \geq 1 - 2^{-r^{\alpha}}$$

and

$$\Pr_{x^h \leftarrow_R X^h, h \neq j}[\mathsf{Cond}(x^1, \ldots, X^i, \ldots, x^C) \textit{ is } 2^{-r^{\alpha}}\textit{-close to a somewhere random source}] \geq 1 - 2^{-r^{\alpha}}$$

The algorithm Cond above is just a slight variation of the algorithm that we developed for Theorem 3.1.13. As usual, it will suffice for us to build an extractor for a C tuple of $t \times r$ independent sources when we only have the promise that two of them are aligned independent somewhere random sources. It turns out, that Bourgain's extractor can be modified to give an extractor that can extract from *any* number of sources

Our extractor will be built exactly as in Subsection 3.1.3, except that at the bottom, we will use Theorem 2.6.8 instead of Theorem 2.6.7. To clarify, let us give the algorithm that corresponds to Algorithm 3.1.9 in the last section.

Algorithm 3.2.2 ($\mathsf{ManyCondense}(x^1, x^2, \ldots, x^C)$)**.**

Input: $x^1, x^2, \ldots, x^C$ — C tuple of $t \times r$ matrices with $t \le r^{0.7}$.
Output: $y^1, y^2, \ldots, y^C$ — C tuple of $\lceil t/2 \rceil \times m$ matrices, with $m = r - r^{0.9}$.

Sub-Routines and Parameters:
Let $w = r^{0.1}$.
Let $\mathsf{Ext} : \{0,1\}^{rt} \times \{0,1\}^w \to \{0,1\}^m$ be the strong seeded extractor from Corollary 2.6.5, set up to extract $m = r - r^{0.9}$ bits from a min-entropy $r - 100wr^{0.7}$ source with error $\epsilon = 2^{-r^{\Omega(1)}}$.
Let $\mathsf{Bou} : \{0,1\}^{2w} \times \{0,1\}^{2w} \to \{0,1\}^d$ be the extractor from Theorem 2.6.8.
Recall the definition of a *slice* of a somewhere random source — Definition 2.1.24.

```
1. For each $i = 1, 2, \ldots, C$, let $z^i$ be the $\lceil t/2 \rceil \times 2r$ matrix obtained by concatenating pairs
   of rows in $\mathsf{Slice}(x^i, w)$, i.e., the $j$'th row is $\mathsf{Slice}(x^i, w)_{2j-1}, \mathsf{Slice}(x^i, w)_{\min\{2j,t\}}$.
2. Let $s$ be the $\lceil t/2 \rceil \times d$ matrix whose $j$'th row is $\mathsf{Bou}(z^1_j, z^2_j, \ldots, z^C_j)$.
3. For each $i = 1, 2, \ldots, C$, let $y^i$ be the $\lceil t/2 \rceil \times m$ matrix whose $j$'th row is $\mathsf{Ext}(x^i, s_j)$.
```

Just as in the previous subsection, we have the analogous lemma:

Lemma 3.2.3. *For any C independent sources on $t \times r$ matrices $X^1, \ldots, X^C$, with $t \le r^{0.7}$,$C \le r^{100}$ and the guarantee that at least two of them X^i, X^j are aligned somewhere random sources, let $Y^1, \ldots, Y^C$ be the outputs of $\mathsf{ManyCondense}(X^1, \ldots, X^C)$. Then there exists a random variable Z and $\epsilon = 2^{-r^{\Omega(1)}}$ with the property that:*

- *For every $h = 1, 2, \ldots, C$, Y^h is a deterministic function of Z and X^h.*
- $\Pr_{z \leftarrow_R Z}[(Y^i, Y^j)|Z = z$ *are ϵ-close to two aligned somewhere random sources*$] \ge 1 - \epsilon$.

The proof of the lemma is essentially the same as the proof of Lemma 3.1.10, so we omit it. Just as in that case, we can define Z to be the concatenation of all the slices used in the interaction between the sources. Then we can use the union bound over $O(C)$ events to bound the probability in the last item of the conclusion.

Next, we define the following algorithm, analogous to Algorithm 3.1.11.

Algorithm 3.2.4 ($\mathsf{StrongSRExt}(a, b)$)**.**

Input: $x^1, \ldots, x^C$ — a C tuple of $t \times r$ matrices with $t \le r^{0.7}$.
Output: z — an m bit string, with $m \ge r - r^{0.95}$.

```
1. If $x^1, \ldots, x^C$ have only one row each, output the bitwise xor $x^1 \oplus x^2 \oplus \cdots \oplus x^c$.
2. Else, set $y^1, \ldots, y^c$ to be the output of $\mathsf{ManyCondense}(x^1, \ldots, x^c)$.
3. For $i = 1, 2, \ldots, C$, set $x^i = y^i$ and go to the first step.
```

The proof for the following theorem is the analogue of Theorem 3.1.12:

Theorem 3.2.5. *Let $X^1, \dots, X^C$ be C independent distributions on $t \times r$ matrices with $t \le r^{0.7}$, $C \le r^{100}$, and the guarantee that (X^i, X^j) are aligned somewhere random sources for some i, j. Then we have that*

$$\Pr_{\textit{For every } h \neq j, x^h \leftarrow_R X^h}[|\mathsf{StrongSRExt}(x^1, \dots, X^j, \dots, x_C) - U_m| \ge \gamma] < \gamma$$

and

$$\Pr_{\textit{For every } h \neq i, x^h \leftarrow_R X^h}[|\mathsf{StrongSRExt}(x^1, \dots, X^j, \dots, x_C) - U_m| \ge \gamma] < \gamma$$

for $\gamma = 2^{-r^{\Omega(1)}}$.

Again, the proof is omitted since it is essentially the same as the proof of Theorem 3.1.12. Given this theorem, Theorem 3.2.1 follows.

3.3 Condensing Affine Somewhere Random Sources

Recall that we say that a source is *affine* with entropy k, if it is uniformly distributed on some k dimensional affine subspace of an dimensional vector space over a finite field $\mathbb{F}$. We say that a source is an $t \times r$ affine somewhere random source, if it is an affine source over $t \times r$ matrices with entries in $\mathbb{F}$, with the property that one of the rows of the matrix is uniformly random. The formal definitions are in Section 2.1.3. Throughout this section, we work over the finite field $GF(2)$.

In this section, we show how the technique of condensing independent somewhere random sources can be extended to apply to affine somewhere random sources. The final theorem we will prove is the following:

Theorem 3.3.1 (Affine Somewhere Random Extractor)**.** *There exists a polynomial time computable function $\mathsf{Ext} : \{0,1\}^{rt} \to \{0,1\}^{r-r^{0.9}}$ with the property that for every affine $t \times r$ somewhere random source X with $t \le r^{0.7}$, $\mathsf{AffineExt}(X)$ is $2^{-r^{\Omega(1)}}$-close to uniform.*

The arguments that will go into proving this theorem are reminiscent of the arguments we used in the previous sections of this chapter. We shall rely on two earlier works to get our results. The first is a construction of a linear seeded extractor, mentioned in Corollary 2.6.5. The second is a construction of an affine source extractor for any constant entropy rate — Theorem 2.6.9.

As in the previous sections, we will obtain our extractor by repeatedly condensing the source we are working with. We do this with the following algorithm:

Algorithm 3.3.2 ($\mathsf{AffineCondense}(x)$).

Input: x — a $t \times r$ matrix with $t \leq r^{0.7}$.
Output: y — a $\lceil t/2 \rceil \times m$ matrix, with $m = r - r^{0.9}$.

Sub-Routines and Parameters:
Let $w = r^{0.1}$.
Let $\mathsf{Ext} : \{0,1\}^{rt} \times \{0,1\}^{w} \to \{0,1\}^{m}$ be the strong seeded extractor from Corollary 2.6.5, set up to extract $m = r - r^{0.9}$ bits from a min-entropy $r - 100wr^{0.7}$ source with error $\epsilon = 2^{-r^{\Omega(1)}}$.
Let $\mathsf{Bou} : \{0,1\}^{2w} \times \{0,1\}^{2w} \to \{0,1\}^{d}$ be the extractor from Theorem 2.6.9, set up to extract from entropy rate $1/2$.
Recall the definition of a *slice* of a somewhere random source — Definition 2.1.24.

1. Let z be the $\lceil t/2 \rceil \times 2r$ matrix obtained by concatenating pairs of rows in $\mathsf{Slice}(x, w)$, i.e., the i'th row z_i is $\mathsf{Slice}(x, w)_{2i-1}, \mathsf{Slice}(x, w)_{\min\{2i,t\}}$

2. Let s be the $\lceil t/2 \rceil \times d$ matrix whose i'th row is $\mathsf{Bou}(z_i)$.

3. Let y be the $\lceil t/2 \rceil \times m$ matrix whose i'th row is $\mathsf{Ext}(x, s_i)$.

We can then show that the output of this algorithm is close to a convex combination of affine somewhere random sources:

Lemma 3.3.3. *For any affine $t \times r$ somewhere random source X, with $t \leq r^{0.7}$, then $\mathsf{AffineCondense}(X)$ is $2^{-r^{\Omega(1)}}$-close to a convex combination of affine somewhere random sources.*

Proof. Let $Z = \mathsf{Slice}(X, w)$ as in the algorithm. Then note that $\mathsf{Slice}(\cdot, w)$ is a linear function. Thus, by Lemma 2.1.27, there must exist affine sources A, B with $X = A + B$, $H(B) \geq r - tw$, and $\mathsf{Slice}(B, w)$ is the all zero matrix with probability 1. In particular, this implies that $Z = \mathsf{Slice}(X, w) = \mathsf{Slice}(A, w)$ is independent of B.

Now, since X was somewhere random, there must exist an index h for which Z_h is an affine source with min-entropy rate $1/2$. Then, if β is the error of Bou, we get that:

$$|\mathsf{Bou}(Z_h) - U_d| < \beta \tag{3.3}$$

Since Ext is a linear seeded extractor, for any $u \in \{0,1\}^d$ we have that $\mathsf{Ext}(X, u) = \mathsf{Ext}(A + B, u) = \mathsf{Ext}(A, u) + \mathsf{Ext}(B, u)$. Note that for every fixing of Z, the output the algorithm is a linear function of the rest of the source. Thus $Y|Z = z$ is affine. All that remains to be shown is that with high probability over the choice of $z \leftarrow_{\mathrm{R}} Z$, the source $Y|Z = z$ is also somewhere random.

By Proposition 2.3.6, we get that

$$\begin{aligned}
&\Pr_{u \leftarrow_{\mathrm{R}} U_d}[|\mathsf{Ext}(B, u) - U_m| > 0] < \epsilon \\
&\Rightarrow \Pr_{s_h \leftarrow_{\mathrm{R}} \mathsf{Bou}(Z_h)}[|\mathsf{Ext}(B, s_h) - U_m| > 0] < \epsilon + \beta
\end{aligned} \tag{3.4}$$

Since B is independent of Z, we have that for any $z \in \mathsf{supp}(Z), u \in \{0,1\}^d$, $\mathsf{Ext}(X,u)|Z = z = \mathsf{Ext}(B,u) + (\mathsf{Ext}(A,u)|Z = z$. Since A is completely determined by Z, $\mathsf{Ext}(X,u)|Z = z$ is uniform exactly when $\mathsf{Ext}(B,u)$ is uniform.

$$\begin{aligned}
&\Pr_{z \leftarrow_{\mathrm{R}} Z}[|\mathsf{Ext}(X|Z = z, \mathsf{Bou}(z_h)) - U_m| > 0] \\
&\le \Pr_{z \leftarrow_{\mathrm{R}} Z}[|\mathsf{Ext}(B, \mathsf{Bou}(z_h)) - U_m| > 0] \\
&< \epsilon + \beta && \text{by Equation 3.4} \\
&= 2^{-r^{\Omega(1)}}
\end{aligned}$$

This completes the proof.

□

Given this condenser, we can use it repeatedly to get an extractor.

Algorithm 3.3.4 ($\mathsf{AffineSRExt}(x)$)**.**

Input: x — a $t \times r$ matrix with $t \le r^{0.7}$.
Output: z — an m bit string, with $m \ge r - r^{0.95}$.

```
1. If $x$ has only one row, output $x$.
2. Else, set $y$ to be the output of $\mathsf{AffineCondense}(x)$.
3. Set $x = y$ and go to the first step.
```

It's clear that the extractor succeeds. We will need to run $\mathsf{AffineCondense}$ at most $\log t$ times, which is insignificant compared to the error in each step and the reduction in the length of each of the rows. This completes the proof of Theorem 3.3.1.

Chapter 4

Extractors for Independent Sources

A natural model for a source that would allow extraction to be feasible is to assume that the source consists of two or more independent parts, each with sufficient entropy. We say that a function Ext is a C-source extractor for entropy k if given any C independent sources with entropy k, $X_1, \ldots, X_{\mathsf{C}}$, $\mathsf{Ext}(X_1, \ldots, X_{\mathsf{C}})$ is close to being uniformly random.

Definition 4.0.5. A function $\mathsf{IndepExt} : (\{0,1\}^n)^{\mathsf{C}} \to \{0,1\}^m$ is an *extractor for* C *independent sources* with min-entropy k and error ϵ if for any independent (n, k) sources $X^1, \ldots, X^{\mathsf{C}}$ we have

$$|\mathsf{IndepExt}(X_1, \ldots, X_{\mathsf{C}}) - U_m| < \epsilon$$

Another way to view 2-source extractors is as boolean matrices that look random in a strong sense: Every 2-source extractor for entropy k gives an $N \times N$ boolean matrix in which every $K \times K$ minor has roughly the same number of 1's and 0's, with $N = 2^n, K = 2^k$.

The independent sources model is one of the earliest models studied [SV86, Vaz85, CG88]. The probabilistic method shows that most functions are 2-source extractors requiring entropy that is just logarithmic in the total length of each of the sources (we give the proof in Section 4.2). Explicit constructions are still very far from achieving this kind of performance. The classical Lindsey Lemma gives a 2-source extractor for sources on n bits with entropy $n/2$ (we revisit this theorem in the appendix – Theorem C.2.1). No significant progress was made in improving the entropy requirements over this, until recently. In the last few years, sparked by new results in arithmetic combinatorics [BKT04], there were several results.

One important reason why this model is interesting is its connection to explicit constructions of Ramsey Graphs. Every function with two inputs can be viewed as a coloring of the corresponding complete bipartite graph. When the function is an extractor for 2 independent sources, the extractor property guarantees that this coloring gives a bipartite Ramsey Graph, i.e., a two colored complete bipartite graph with no large monochromatic bipartite clique. It is easy to convert any bipartite Ramsey Graph into a regular Ramsey Graph, so this immediately gives explicit constructions of Ramsey Graphs. When the extractor requires a few (say constant u) number of sources, the corresponding coloring can be used to efficiently construct a u-uniform Ramsey Hypergraph. This connection is discussed in Section 4.3.

4.1 Previous Results and Overview of Our Results

The problem of extracting from several independent sources was first considered by Chor and Goldreich [CG88][1]. They demonstrated extractors for 2 independent $(n, (1/2 + \alpha)n)$-sources, for all constant $\alpha \in (0, 1/2]$.

Since then there had not been much success in improving the entropy requirements until Barak, Impagliazzo and Wigderson [BIW04] showed how to extract from a constant number of independent $(n, \delta n)$-sources, where δ (the *min-entropy rate* of the source) is allowed to be any arbitrarily small constant. The number of sources used depends on δ. Subsequently Barak et al. [BKS+05] showed how to extract a constant number of bits with constant error from 3 $(n, \delta n)$-sources, where δ is an arbitrarily small constant. In this work they also present 2-source *dispersers* (a disperser is an object similar to but somewhat weaker than an extractor) that output a constant number of bits with constant error and work for min-entropy rate δ where δ is an arbitrarily small constant.

Raz [Raz05] gave an extractor for 2 independent sources where one source needs to have min-entropy rate greater than and bounded away from 1/2 and the other source may have polylogarithmically small min-entropy. In this case his extractor can extract a linear fraction of the min-entropy with exponentially small error. Improving the 3 source extractor of Barak et al., he constructed an extractor for 3 independent sources where one source must have constant min-entropy rate and the other two need polylogarithmic min-entropy. In this case his extractor can extract a constant number of bits with constant error.

Bourgain [Bou05] (an exposition of this result is in Appendix C gave another extractor for 2 independent sources. His extractor can extract from 2 $(n, (1/2 - \alpha_0)n)$-sources, where α_0 is some small universal constant. This is the first extractor to break the 1/2 min-entropy rate barrier for 2 sources. His extractor outputs a linear fraction of the min-entropy, with exponentially small error.

Other than Raz's extractor for 2 sources, all of these recent results were made possible by new breakthroughs on the sum-product estimate for finite fields [BKT04, Kon03], a result from additive number theory. A common feature of the work of Raz (in the case of 3 sources) [Raz05] and Barak et al. [BKS+05] is that they reduce the general problem of extracting from independent sources to the problem of extracting from independent sources that come from a much more restricted class, called *somewhere-random* sources. They then build extractors for these sources. A key step in our construction is building much better extractors for independent *somewhere-random* sources.

4.1.1 Results in This Chapter

In this chapter, we give several constructions of extractors for independent sources. These results are based on work with Boaz Barak, Xin Li, Ronen Shaltiel, Avi Wigderson and David Zuckerman [Rao06, BRSW06, LRZ07]. Here we list each of our results:

- We give a polynomial time computable extractor that extracts k random bits from $O(\frac{\log n}{\log k})$ independent (n, k)-sources with error 2^{-k^c} for any $k(n) > \log^4 n$ and some universal constant $c > 1$. An interesting setting of parameters is when $k = n^\gamma$ for some $0 < \gamma < 1$. In this case we get an extractor for a constant number of sources that extracts a constant fraction of the

[1] Santha and Vazirani [SV86, Vaz85] also considered extracting from independent sources, but the sources had additional restrictions placed on them.

Construction	**Min-Entropy** k	**Output**	**Error**	**Ref**
poly$(1/\delta)$-source extractor	δn	$\Theta(n)$	$2^{-\Omega(n)}$	[BIW04]
3-source extractor	δn, any constant δ	$\Theta(1)$	$O(1)$	[BKS$^+$05]
3-source extractor	One source: δn, any constant δ. Other sources may have polylog(n) min-entropy.	$\Theta(1)$	$O(1)$	[Raz05]
2-source extractor	One source: $(0.5+\alpha)n$, $\alpha > 0$. Other source may have $k =$ polylog(n) min-entropy.	$\Theta(k)$	$2^{-\Omega(k)}$	[Raz05]
2-source extractor	$(0.5-\alpha_0)n$ for some universal constant $\alpha_0 > 0$	$\Theta(n)$	$2^{-\Omega(n)}$	[Bou05], Appendix C
$O(1/\delta)$-source extractor	n^δ	$\Theta(k)$	$k^{-\Omega(1)}$	This thesis [Rao06]
$O(1/\delta)$-source extractor	n^δ	$\Theta(k)$	$2^{-k^{\Omega(1)}}$	This thesis [BRSW06]
3-source extractor (with constraints on input lengths)	n^δ for any constant δ (additional constraints apply)	$k - o(k)$	$2^{-k^{\Omega(1)}}$	This thesis [LRZ07]

Table 4.1: Performance of recent extractors for independent sources

total min-entropy with exponentially small error. Formally, the theorem we will prove[2] is the following:

Theorem 4.1.1. *There exist constants $c > 0$ and c' such that for every n, k with $k > \log^4 n$ there exists a polynomial time computable function $\mathsf{IndepExt} : (\{0,1\}^n)^{\mathsf{C}} \to \{0,1\}^k$ with $\mathsf{C} \leq c' \frac{\log n}{\log k}$ s.t. if $X^1, X^2, \ldots, X^u$ are independent (n,k) sources then*

$$|\mathsf{IndepExt}(X_1, \ldots, X_{\mathsf{C}}) - U_k| < 2^{-k^c}$$

- We give an extractor for 2 sources, under the assumption that just one of the sources is a *block source*. Block sources have been involved in many earlier works in extractors. Informally, a source is a block source if it can broken up into several blocks, such that every block has high enough min-entropy even conditioned on the event that all the previous blocks in the source are fixed to some value in their support. Two blocks in a block source aren't completely independent, but they do satisfy the property that the second block is hard to predict even if we know what the value of the first block is. The concatenation of several independent sources is of course a block source. Thus block-sources are a strictly more general class of sources than independent sources. It can be shown that there is no deterministic extractor for a single block source.

Theorem 4.1.2 (Block vs General Source Extractor Theorem 4.5.1)**.** *There is a polynomial time computable extractor $\mathsf{B} : (\{0,1\}^n)^2 \to \{0,1\}^m$ for 2 independent sources, one of which*

[2] We note that independent of this work, Chung and Vadhan [CV06] discovered how to improve the error of the original extractor in [Rao06] to make it exponentially small.

is a c-block-sources with block min-entropy k and the other a source of min-entropy k, with $m = \Omega(k)$, $c = O((\log n)/(\log k))$ and error at most $2^{-k^{\Omega(1)}}$.

- Finally, we show how to extract from just 3 sources when they have polynomially small entropy, as long as the sources satisfy some constraints on their sizes.

  Theorem 4.1.3. *There exists a constant h such that for every constant $\gamma > 1$, there is a polynomial time computable function $\mathsf{3Ext} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^{n^{\gamma/h}} \to \{0,1\}^{n^\gamma - o(n^\gamma)}$ which is an extractor for three sources with min-entropy requirements $k_1 = k_2 \geq n^\gamma, k_3 \geq \log^{10} n$ and error $2^{-\Omega(k_3)} + 2^{-k_1^{\Omega(1)}}$.*

Remark 4.1.4. Ronen Shaltiel showed how to improve the output length of any strong extractor [Sha06]. His techniques show how to get extractors which output $k - o(k)$ output bits, where now k is the *total* entropy in all sources, by paying a small price in the error of the extractors.

Several ideas go into our final extractor construction. There are essentially three constructions in this chapter. The first gives the basic extractor for $O(\log n/\log k)$ sources of min-entropy k with polynomially small error. The second is a twist on the first, that achieves the same parameters, but with exponentially small error. The final construction uses significantly different ideas (but still relies on the second result in a black box way) to give an extractor for just 3 sources, even when the min-entropy is low.

4.2 2-Source Extractors via the Probabilistic Method

Here we show that almost all functions are good 2-source extractors. Recall that every distribution with min-entropy k is a convex combination of of *flat* distributions with min-entropy k. Thus, it will suffice to show that most functions are good extractors for 2 independent flat sources.

We will do this via Theorem 2.5.1. Let $\mathcal{X}$ be the class of two independent flat (n, k) sources. Note that every source in this class has min-entropy $2k$. Then we get that $|\mathcal{X}| = \binom{2^n}{2^k}^2 \leq 2^{2n2^k}$. Thus $\log|\mathcal{X}| \leq 2n2^k$. The theorem then implies that as long as $\epsilon^2 2^k \geq 6 \log e(2^{m-k} + 2n)$, there exists a 2 source extractor with output length m and error ϵ. We see that we can afford to even set k to as low as $2\log n$ to extract $m = 2k - \log k - \log n$ random bits with constant error.

4.3 Multisource Dispersers vs Ramsey Hypergraphs

Closely related to extractors for independent sources is the notion of a *disperser* for independent sources.

Definition 4.3.1. A function $\mathsf{IDisp} : (\{0,1\}^n)^u \to \{0,1\}^m$ is an (k, ϵ) *u-source disperser* if for all sets $A_1, A_2, \ldots, A_u \subseteq \{0,1\}^n$, with $|A_1|, |A_2|, \ldots, |A_u| \geq 2^k$, $|\mathsf{IDisp}(A_1, \ldots, A_u)| \geq (1 - \epsilon)2^m$.

Here we outline how to convert any efficiently computable multisource disperser into an explicit Ramsey Hypergraph.

Proposition 4.3.2. *Let $\mathsf{IDisp} : (\{0,1\}^n)^u \to \{0,1\}$ be a (k, ϵ) u-source disperser. Then IDisp can be used to give an explicit u-uniform Ramsey Hypergraph on 2^n vertices that avoids monochromatic cliques of size $u2^k$.*

Proof Sketch: Consider the u-uniform hypergraph defined as follows: given any potential edge $\{a_1, a_2, \ldots, a_u\}$ of the graph, first sort the vertices according to some predetermined total order to ensure that $a_1 \geq a_2 \geq \cdots \geq a_u$ in this order. Then color the edge red if $\mathsf{IDisp}(a_1, a_2, \ldots, a_u) = 0$, else color it blue.

Now let S be any subset of the vertices of this graph of size $u2^k$. Then we can use the total order to partition the vertices of S into u sets $S_1, \ldots, S_u$ of size 2^k by taking the highest 2^k vertices in the total order as S_1, then the next 2^k vertices as S_2 and so on. By the disperser property of IDisp, $\mathsf{IDisp}(S_1, \ldots, S_u) = \{0, 1\}$. Thus S contains hyperedges of both colors.

□

4.4 The Basic Construction

In this section, we discuss the simplest of our extractors for independent sources. Many extractor constructions in the past have been based on the paradigm of iterative condensing [RSW00, TUZ01, CRVW02, LRVW03, BIW04]. The idea is to start with some distribution that has low min-entropy and apply a function (called a *condenser*) whose output has a better min-entropy rate. Repeating this process, we eventually obtain a distribution that has very high min-entropy rate. Then we can apply some other extractor which works for such a high min-entropy rate to obtain random bits. The extractor in this thesis can also be viewed as an example of this paradigm, with a slight twist.

We make progress by considering a more restricted model for sources called *somewhere random* sources (SR-sources for short), which we discussed extensively in Chapter 3. SR-sources were first introduced by by Ta-Shma [TS96]. An important concept that we introduced in that chapter is that of *aligned* SR-sources. Two SR-sources with the same number of rows are said to be *aligned* if there is an i such that the i'th row of both sources are distributed uniformly.

We will think of the number of rows of an SR-sources as a measure of the quality of the source. The fewer the number of rows, the better the quality is. Our construction will manipulate SR-sources. We will iteratively improve the quality (reduce the number of rows) of the SR-sources that we are working with until extracting randomness from them becomes easy.

Our construction will use *strong seeded extractors* as a basic tool. A strong seeded extractor can be viewed as a small family of deterministic functions (each function in the family indexed by a unique seed), such that for any fixed adversarially chosen source of randomness, almost all functions from the family are good extractors for that source. Several constructions of strong seeded extractors with seed length $O(\log n)$ (giving a family of polynomially many functions) are known (e.g. [LRVW03, Tre01]).

Now we describe some basic observations that go into the construction. We will then show how to put these together to get the high level view of our extractor construction (Figure 4.1).

Idea 1: General Sources can be turned into aligned SR-sources. A strong seeded extractor can be used to convert any general weak source into an SR-source. Given a sample from the weak source, we simply evaluate the extractor on the sample with all possible seeds, getting one row of the output for each fixed seed. For any fixed weak source, the strong extractor property guarantees that most seeds will give a distribution that is statistically close to uniform. As long as the seed length required by the extractor is $O(\log n)$, we get a polynomial time algorithm to convert any weak source to a distribution that is statistically close to an SR-source with poly(n) rows. A simple union bound argument can be used to show that if we

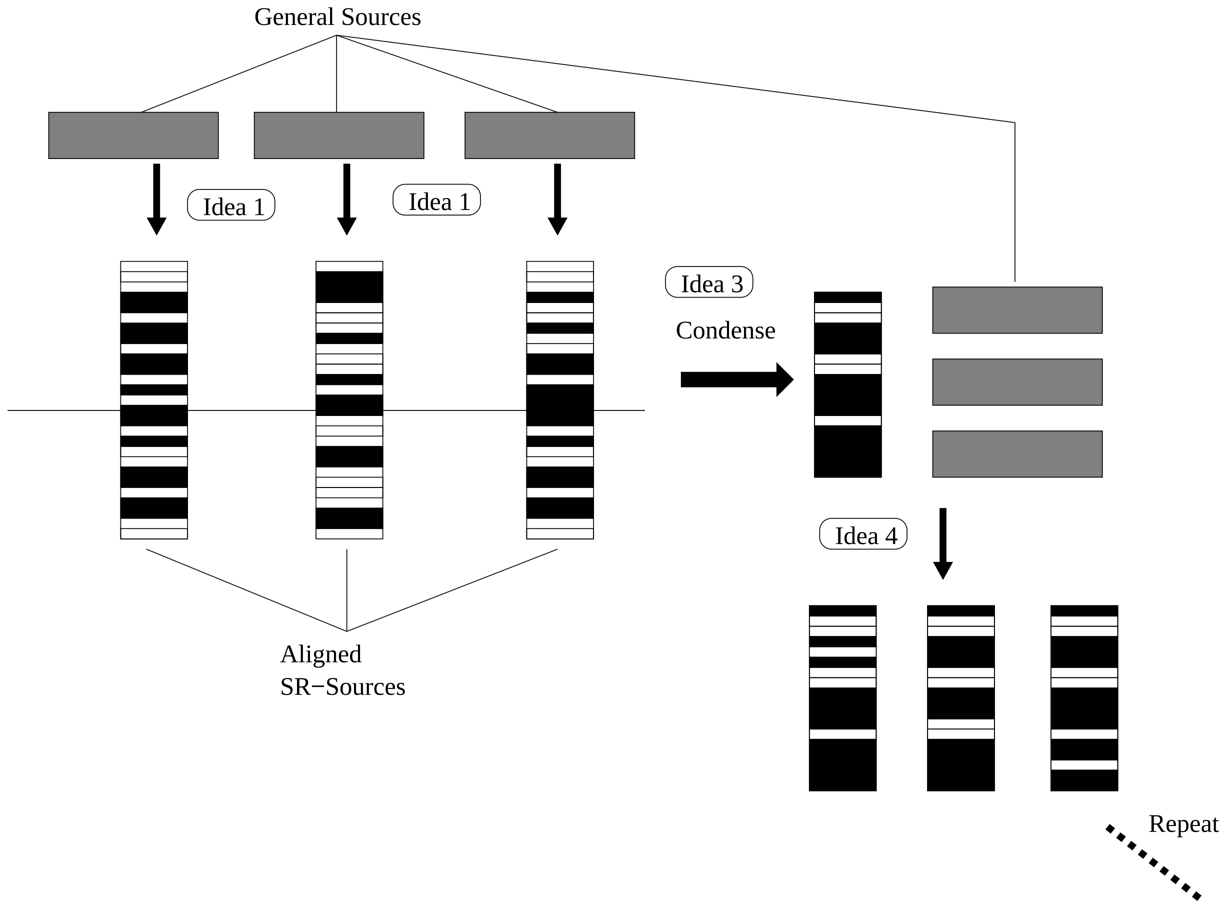

Figure 4.1: High level picture of the extractor

convert a constant number of independent sources to independent SR-sources in this way, the SR-sources we obtain are also aligned.

Idea 2: Extraction is easy from *high quality* independent aligned SR-sources. It is easy to extract from independent SR-sources when each source has very few rows relative to the length of each of the rows. In the extreme case, when an SR-source has just one row, it is a uniformly random string. A slightly more non-trivial example is when we have two independent aligned SR-sources, each with two rows. In this case it is easy to see that if we output the bitwise XOR of the first row of the first source with the second row of the second source, we get uniformly random bits. Building on these simple ideas, we will show how to build extractors for just 2 aligned SR-sources even when the number of rows is superconstant. We will be able to extract from such sources as long as the number of rows is significantly less than the length of each row. These results are discussed in Section 3.1 of Chapter 3.

Idea 3: Condensers for *low quality* SR-sources can be obtained via Idea 2. We build condensers for SR-sources in the following sense: given a few input independent aligned SR-sources, our condenser's output is essentially the distribution of independent aligned SR-sources with fewer rows. Suppose we have a construction of an extractor for c aligned SR-sources with

t' rows. Suppose we are given c aligned SR-sources, each with $t > t'$ rows. We can run our extractor with the first t' rows of all of the SR-sources to get a single output. Then we can repeat this with the next t' rows of each of the SR-sources. In this way we obtain t/t' outputs, one of which is guaranteed to be uniformly random, i.e., we obtain a new SR-source with t/t' rows. In this way, we obtain a condenser which given c independent SR-sources, outputs one SR-source with fewer rows. Again, this idea is discussed in depth in Section 3.1 of Chapter 3.

Idea 4: The quality of SR-sources can be transferred. A single SR-source S with t rows can be used to convert many other independent sources into SR-sources with t rows. Simply use the t rows of S as seeds with a strong seeded extractor to extract from each of the other independent sources. With high probability, the random row of S is a good seed to extract from all the other independent sources simultaneously. It turns out that the output we obtain in this way is close to a convex combination of independent aligned SR-sources, each with t rows. This observation can be interpreted as a way to *transfer* quality from a single SR-source to many other independent sources.

Given these observations, the high level informal view of our extractor construction is the following:

1. Use **Idea 1** to convert a constant number of independent sources into SR-sources with $t = \text{poly}(n)$ rows each.

2. Use **Idea 3** to condense these sources to get 1 SR-source S with much fewer rows t'. If $t' = 1$, stop and output the random row, else continue.

3. Using **Idea 4**, take a constant number of input independent sources and transfer the quality of S to these sources, to get a constant number of independent SR-sources with t' rows each.

4. Go to step 2.

The number of sources required depends on how quickly the number of rows in the SR-sources we are working with drop down to 1. We will give two condenser constructions. The first one is simpler (essentially based on the XOR extractor discussed in **Idea 2**), but only gives an extractor for $\log n$ sources. The second one is more involved, but gives an extractor for a constant number of sources when the min-entropy is polynomially small.

Now we describe our algorithm more precisely. The algorithm for our extractor is:

Algorithm 4.4.1 ($\mathsf{IndepExt}(x^1, \ldots, x^{3u})$).

Input: $x^1, \ldots, x^u$ — a u tuple of n bit strings.
Output: z — an m bit string, with $m = k - o(k)$.

Sub-Routines and Parameters:
Let $\mathsf{Ext}_1 : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a strong seeded extractor from Corollary 2.6.5, set up to extract $m = k - k^{0.5}$ bits from a min-entropy k source with error $\epsilon_1 = 1/\mathrm{poly}(n)$ and seed length $d = O(\log n)$.
Let $\mathsf{Ext}_2 : \{0,1\}^n \times \{0,1\}^{k/2} \to \{0,1\}^m$ be a strong seeded extractor from Corollary 2.6.5, set up to extract $m = k - k^{0.5}$ bits from a min-entropy k source with error $\epsilon_2 = 2^{-k^{\Omega(1)}}$.
Let Cond be the condenser from Theorem 3.1.7, set up to condense three independent aligned $t \times m$ somewhere random sources.

1. For $i = 1, 2, 3,\ j = 1, 2, \ldots, 2^d$, let a^i be the $2^d \times m$ matrix whose jth row is $a^i_j = \mathsf{Ext}_1(x^i, j)$, i.e., a^i is obtained from x^i by evaluating the seeded extractor Ext_1 on it with every possible seed.

2. For $l = 1, 2, \ldots, u$ repeat the following steps

   (a) Let s be the matrix $\mathsf{Cond}(a^1, a^2, a^3)$.

   (b) Let t' be the number of rows in s. If $t' = 1$, output s and terminate.

   (c) For $i = 1, 2, 3,\ j = 1, 2, \ldots, t'$, let $a^i_j = \mathsf{Ext}_2(x^{3l+i}, s_j)$.

Note that every time Cond is used, the number of rows in the somewhere random source we are working with get reduced by a factor of $m^{0.7}$. Thus in $O(\log n / \log k)$ steps, the number of rows is brought down to 1 and the algorithm terminates. This means that $u = O(\log n / \log k)$ for the algorithm to terminate.

Theorem 4.4.2. *Let $X^1, \ldots, X^u$ be independent (n, k) sources. Then $\mathsf{IndepExt}(X^1, \ldots, X^u)$ is $1/\mathrm{poly}(n)$ close to uniform.*

Proof. Let γ be the error of Cond. Recall that ϵ_1 is the error of Ext_1 and ϵ_2 is the error of Ext_2. We'll prove that during that during the l'th iteration of the loop, S is $\epsilon_1 + l(\gamma + 6\epsilon_2)$ close to being somewhere random.

We do this by induction. When $l = 1$ this is clearly true, by the properties of Cond. For higher l, note that S is obtained by using each row of the old S to extract from three new independent sources. The resulting 3 sources are then a convex combination of independent aligned somewhere random sources. The use of Ext_2 adds an error of $2\epsilon_2$ for each source. The use of Cond then adds another γ to the error. ■

4.5 Extractor for one block source and one general source

In this section we build on the ideas of the previous section to get an extractor that works for two sources, given an assumption on one of the sources. The assumption is that the first source is a block source (Definition 2.1.15), which means that it is divided into C blocks such that each block has entropy above a certain threshold *even conditioned on all previous blocks.* In particular, we note that the concatenation of independent sources is a block source. Thus, our algorithm is also an extractor for a few independent sources.

This algorithm has the added advantage that it gives an extractor with extremely small error — $2^{-k^{\Omega(1)}}$.

We will prove the following theorem:

Theorem 4.5.1 (Block vs General Source Extractor). *There exists constants c_1, c_2 such that for every n, n, k, with $k > \log^{10} n$ there exists a polynomial time computable function $\mathsf{BExt} : \{0,1\}^{\mathsf{C}n} \times \{0,1\}^n \to \{0,1\}^m$ with $\mathsf{C} = O(\frac{\log n}{\log k})$ s.t. , if $X = X^1, \cdots, X^{\mathsf{C}}$ is a k-block source and Y is an independent k-source*

$$|\mathsf{BExt}(X, Y) - U_m| < 2^{-k^{c_1}}$$

with $m = c_2 k$.

The low error guaranteed by this theorem is important for applications that require a *negligible* error. Since the concatenation of independent sources is a block source, an immediate corollary of the above theorem is a new extractor for independent sources with exponentially small error.

Corollary 4.5.2 (Independent Source Extractor). *There exists constants c_1, c_2 such that for every $n, , k$, with $k > \log^{10}(n)$ there exists a polynomial time computable function $\mathsf{BExt} : (\{0,1\}^n)^{\mathsf{C}} \to \{0,1\}^m$ with $\mathsf{C} = O(\frac{\log n}{\log k})$ s.t. , if $X^1, \ldots, X^{\mathsf{C}}$ are independent (n, k) sources,*

$$|\mathsf{BExt}(X_1, \ldots, X_{\mathsf{C}}) - U_m| < 2^{-k^{c_1}}$$

with $m = c_2 k$.

We will obtain our extractor by reducing the problem to the one of constructing an extractor for two independent aligned somewhere random sources, a problem that was solved in Chapter 3.

The remainder of this section is devoted to proving Theorem 4.5.1. We start with elaborating on the ideas that enable us to get lower error.

4.5.1 Achieving Small Error

We remark that the technique we apply towards achieving low error could also be applied to our earlier extractor construction. A somewhat similar observation was made by Chung and Vadhan [CV06], who noted that our extractor from [Rao06] for independent sources can more directly be shown to have low error.

We will first prove the following theorem, which gives an extractor for a block source and an independent *somewhere random* source. This extractor has low error.

Theorem 4.5.3 (Somewhere random vs Block Source Extractor). *There exist constants $\alpha, \beta, \gamma < 1$ such that for every n, t, k, with $k > \log^{10} t, k > \log^{10} n$ there is a polynomial time computable function $\mathsf{SRvsBExt} : \{0,1\}^{\mathsf{C}n} \times \{0,1\}^{tk} \to \{0,1\}^m$ with $\mathsf{C} = O(\frac{\log t}{\log k})$ s.t. , if $X = X^1, \cdots, X^{\mathsf{C}}$ is a $(k, \ldots, k)$ block source and Y is an independent $t \times k$ $(k - k^{\beta})$-SR-source,*

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

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

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

We defer the proof of this theorem to the next section.

Note that we can get an extractor for a block source and a general independent source from Theorem 4.5.3 by using the fact that a general source can be transformed into a somewhere random source (Proposition 2.3.4). However, this transformation seems to spoils the error as when transforming a general source into a somewhere random source with poly(n) rows it only guarantees that the random row is $1/\text{poly}(n)$-close to uniform (and this error is inherited by the final extractor).

Nevertheless, we will use this transformation and provide a better analysis for the final extractor (resulting in smaller error). In order to so that we will first prove a weaker version of Theorem 4.5.3 in which the success of the extractor is only guaranteed on a large subsource Y' of Y. We will then show how to use this to prove Theorem 4.5.1.

Theorem 4.5.4 (Block vs General Subsource Extractor). *There exist absolute constants $c_1, c_2, c_3 > 0$ such that for every n, k, with $k > \log^{10} n$ there exists a polynomial time computable function $\mathsf{BExt} : \{0,1\}^{\mathsf{C}n} \times \{0,1\}^n \to \{0,1\}^m$ with $\mathsf{C} = c_1 \frac{\log n}{\log k}$ s.t. , if $X = X^1, \cdots, X^{\mathsf{C}}$ is a k block source and Y is an independent (n,k)-source, there is a deficiency 2 subsource $Y' \subseteq Y$ s.t.*

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

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

where U_m is independent of X and Y, and for $m = c_2 k$ and $\epsilon = 2^{-k^{c_3}}$.

Proof. The idea is to reduce to the case of Theorem 4.5.3. We convert the general source Y into an SR-source. To do this we will use a strong seeded extractor and Proposition 2.3.4. If we use a strong seeded extractor that requires only $O(\log n)$ bits of seed, the SR-source that we get will have only poly(n) rows, and one of the rows is $1/n$-close to uniform. By Lemma 2.1.12, we can go to a deficiency 2 subsource $Y' \subseteq Y$ which has high entropy in some row. This is good enough to use our extractor from Theorem 4.5.3 and get the better error. ■

Proof of Theorem 4.5.1. We prove the theorem by showing that any extractor that satisfies the conclusions of Theorem 4.5.4 (i.e., low strong error on a subsource), must satisfy the seemingly stronger conclusions of Theorem 4.5.1.

Let BExt be the extractor from Theorem 4.5.4, set up to extract from a $k/2$ block source and a $k/2 - 2$ general source. Then we claim that when this extractor is run on a k block source and a k general source, it must succeed with much smaller error (on sources with min-entropy k).

Given the source X let $B_X \subset \{0,1\}^n$ be defined as $B_X \stackrel{def}{=} \{y : |\mathsf{BExt}(X, y) - U_m| \geq \epsilon\}$. Then,

Claim 4.5.5. $|B_X| < 2^{k/2}$

Proof. The argument for this is by contradiction. Suppose $|B_X| \geq 2^{k/2}$. Then define Z to be the source which picks a uniformly random element of B_X. By the definition of B_X, this implies that $|Z', \mathsf{BExt}(X, Z') - Z', U_m| \geq \epsilon$ for *any* subsource $Z' \subset Z$. This contradicts Theorem 4.5.4. ■

Thus $\Pr[Y \in B_X] < 2^{k/2-k} = 2^{-k/2}$.

This implies that $|\mathsf{BExt}(X,Y) - U_m| < \epsilon + 2^{-k/2}$, where ϵ is the ϵ from Theorem 4.5.4. ■

Remark 4.5.6. In fact the above proof actually implies the extractor from Theorem 4.5.1 is *strong* with respect to Y, i.e., $|Y, \mathsf{BExt}(X,Y) - Y, U_m| < \epsilon + 2^{-k/2}$.

4.5.2 Extractor for general source and an SR-source with few rows

Here we will construct the extractor for Theorem 4.5.3. The main step in our construction is the construction of an extractor for a general source and an independent SR-source which has *few* rows. Once we have such an extractor, it will be relatively easy to obtain our final extractor by iterated condensing of SR-sources.

First, we prove the following theorem:

Theorem 4.5.7. *There exist constants $\alpha, \beta < 1$ such that for every $n, k(n)$ with $k > \log^{10} n$, and constant $0 < \gamma < 1/2$, there is a polynomial time computable function* $\mathsf{BasicExt} : \{0,1\}^n \times \{0,1\}^{k^{\gamma+1}} \to \{0,1\}^m$ *s.t. 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

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

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

Proof. We are trying to build an extractor that can extract from one $k^\gamma \times k$ $k - k^\beta$-SR-source Y and an independent (n,k) source X. We will reduce this to the case of two independent aligned SR-sources with few rows, for which we can use Theorem 3.1.15. More precisely, the plan is to use the structure in the SR-source Y to impose structure on the source X and obtain two independent aligned somewhere random sources.

In the following discussion, the term *slice* refers to a subset of the bits coming from an SR-source that takes a few bits of the SR-source from every row (Definition 2.1.24). We also remind the reader of the following notation: if $f : \{0,1\}^r \times \{0,1\}^r \to \{0,1\}^m$ is a function and a, b are samples from $t \times r$ somewhere sources, $f(\vec{a}, \vec{b})$ refers to the $t \times m$ matrix whose ith row is $f(a_i, b_i)$. Similarly, if c is an element of $\{0,1\}^r$ and b is a sample from a $t \times r$ somewhere source, $f(c, \vec{b})$ refers to the $(t \times m)$ matrix whose ith row is $f(c, b_i)$.

The idea is to use a small slice of the somewhere random source to convert the general source into a somewhere random source. When we fix the slice that we used, we are left with two independent sources, both of which are (almost) somewhere random. We will then show we have essentially reduced the problem to the case of extracting from two independent somewhere random sources.

We first describe our algorithm in words, giving more intuition for why it should succeed.

High level description of the algorithm We first explain the high level intuition behind the algorithm. The first target in the above algorithm is to generate a list of candidate seeds (S) from the two sources, one of which will be close to uniformly random. To generate the list of seeds that we want, we will first take a small slice (Definition 2.1.24) of the bits from Y, i.e., we take $\mathsf{Slice}(Y, w)$, where w is a parameter that we will pick later (think of w as k^{μ} for small μ). We will be able to guarantee that at least one of the rows of $\mathsf{Slice}(Y, w)$ has high entropy (this follows from the fact that Y is somewhere high entropy). We can then use Raz's extractor Theorem 2.6.6 with these bits to extract from X. This gives us a $k^{\gamma} \times w'$ SR-source Q, where $w' = k^{\theta(1)} \gg w$ is some parameter that we will pick later. The two sources that we have now (Q and Y) are not independent. However, note that when we fix the slice of bits (S) that we used, we get two independent sources. In other words, Q and Y can be written as a convex combination of independent distributions.

We now turn our attention to the effect of fixing S on the randomness in Q and Y. We note that Y conditioned on the value of S could potentially lose entropy in its high entropy row. Still, we can expect this high entropy row to have about $k - k^{\beta} - wk^{\gamma}$ bits of entropy, since we fixed only wk^{γ} bits of Y in S. Furthermore, as Raz's extractor is strong for a typical fixing of S the good row has a good seed that extracts randomness from X. This means that Q is (close to) a somewhere random source for a typical fixing of S.

In the next step we take a wider slice of Y and call it $R = \mathsf{Slice}(Y, w'')$. Note that on fixing S to a typical value, we get that Q, R are two independent aligned somewhere high entropy sources. We then use Raz's extractor again to convert Q, R into a somewhere random source H, by applying the extractor to each pair of rows from Q, R. We indeed get that H has a row that is close to uniform. Furthermore, since Raz's extractor is strong, we will be able to guarantee that one of the rows in the resulting SR-source is independent of any of the two input sources X and Y. Thus, once we have H, we can use it with a strong seeded extractor to extract from both X and Y to get independent aligned SR-sources of the type that Theorem 3.1.15 can handle.

More precisely, it follows that for any fixing of Q and R, the distributions X' and Y' are independent. Thus it remains to be shown that both X' and Y' are somewhere random sources (at least for a typical fixing of Q and R). Let us consider the case of of X': For typical fixings of S we have that Q and R are independent sources. By the strongness of Raz's extractor we have that for a typical fixing of Q, the distribution H is (close to) somewhere random and is independent of X. Furthermore for a typical fixing of Q, X retains most of its entropy, thus the seeds in H can be used to extract from X and for a typical fixing of R the good row in H provides a seed that can extract randomness from X even conditioned on the previous fixings and thus we get that X' is (close to) somewhere random. Moving over to the case of Y', for typical fixings of S, Q and R are independent somewhere high entropy sources. Thus, for a typical fixing of R we have that H is a somewhere random source that is independent of Y. Furthermore for a typical such fixing Y still retains most of its entropy and thus we get that Y' is (close to) a somewhere random source.

More formally, the algorithm for our extractor is the following:

Algorithm 4.5.8 (BasicExt(x, y))**.**

Input: x, a sample from an (n, k) source and y a sample from a $(k^\gamma \times k)$ k^β-somewhere random source.
Output: z

Sub-Routines and Parameters:
Let $w, w', w'', l, d, \beta_1$ be parameters that we will pick later. These will satisfy $w'' > w > k^\gamma$ and $w - k^\gamma > w'$.
Let $\mathsf{Raz}_1 : \{0,1\}^n \times \{0,1\}^w \to \{0,1\}^{w'}$ be the extractor from Theorem 2.6.6 set up to extract w' bits from an (n, k) source, using a $(w, 0.9w)$ source as seed.
Let $\mathsf{Raz}_2 : \{0,1\}^{w'} \times \{0,1\}^{w''} \to \{0,1\}^d$ be the extractor from Theorem 2.6.6, set up to extract d bits from a (w', w') source and an independent $(w'', 0.9w'')$ source.
Let $\mathsf{Ext}_1 : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k-k^{\beta_1}}$ and $\mathsf{Ext}_2 : \{0,1\}^{k^{1+\gamma}} \times \{0,1\}^d \to \{0,1\}^{k-2k^{\beta_1}}$ be strong seeded extractors from Corollary 2.6.5, each set up to extract from min-entropy $k - k^{\beta_1}$ with error $2^{-k^{\Omega(1)}}$.
Let $\overline{\mathsf{2SRExt}} : \{0,1\}^{k^\gamma(k-2k^{\beta_1})} \times \{0,1\}^{k^\gamma(k-2k^{\beta_1})} \to \{0,1\}^m$ be the extractor from Theorem 3.1.15, set up to extract from two aligned $k^\gamma \times k - 2k^{\beta_1}$ SR-sources.
Let Slice be the function defined in Definition 2.1.24.

1. Set $s = \mathsf{Slice}(y, w)$.
2. Treating s as a list of k^γ seeds, use it to extract from x to get $q = \mathsf{Raz}_1(x, \vec{s})$. The result is a string with k^γ rows, each of length w'.
3. Set $r = \mathsf{Slice}(y, w'')$.
4. Let $h = \mathsf{Raz}_2(\vec{q}, \vec{r})$, i.e., h is a list of k^γ strings, where the ith string is $\mathsf{Raz}_2(q_i, r_i)$.
5. Let $x' = \mathsf{Ext}_1(x, \vec{h}), y' = \mathsf{Ext}_2(y, \vec{h})$.
6. Use $\overline{\mathsf{2SRExt}}$ to get $z = \overline{\mathsf{2SRExt}}(x', y')$.

Proof of correctness We will prove the following lemma:

Lemma 4.5.9. *For every (n, k) source X and a $k^\gamma \times k$ k^β-somewhere random source Y as in Theorem 4.5.7, we can pick $w, w', w'', l, d, \beta_1$ and a constant β s.t. (X, Y) is $2^{-k^{\Omega(1)}}$-close to a convex combination of sources s.t. for any source in the convex combination, (X', Y') in step 5 above:*

1. *X' is independent of Y'*
2. *X' is a $k^\gamma \times k - k^\beta$ SR-source*
3. *Y' is a $k^\gamma \times k - k^\beta$ SR-source*

Given the lemma, we have reduced the problem to one of extracting from aligned somewhere random sources. Theorem 4.5.7 then follows by the properties of $\overline{\mathsf{2SRExt}}$.

Proof of Lemma 4.5.9. We assume that we have some fixed random variables X, Y that satisfy the hypotheses of the lemma. We will make several claims about the various random variables involved in the construction, setting $w, w', w'', l, d, \beta_1$ along the way to ensure that our lemma is true. In the rest of this proof, a capital letter represents the random variable for the corresponding small letter in the construction above.

Recall that k^β (we are allowed to set $\beta < 1$ to anything we want) is the randomness deficiency of the random row in Y. Note that:

Claim 4.5.10. *For any $w > 2k^\beta$, S is 2^{-k^β} close to a $k^\gamma \times w$ $(w - 2k^\beta)$-SR-source*

Proof. This follows from an application of Lemma 2.1.11. ■

We set $w = k^{\alpha_1}$ for some constant α_1 s.t. $\alpha_1 + \gamma < 1$ and $\alpha_1 > \beta$ and set $w' = w/10$. Note that Theorem 2.6.6 does give an extractor for a $(w, w - 2k^\beta)$ source and an independent (n, k) source with output length $w/10$.

Now Q is correlated with both X and Y. However, when we fix S, Q becomes independent of Y, i.e.: $(X, Q)|S = s$ is independent of $Y|S = s$ for any s. Since Raz_1 is a strong extractor, Q still contains a random row for a typical fixing of S.

Claim 4.5.11. *There exists some constant $\alpha_2 < 1$ s.t.* $\Pr_{s \leftarrow_R S}[Q|S = s$ *is $2^{-k^{\alpha_2}}$ close to a $k^\gamma \times w'$ SR-source*$] > 1 - 2^{-k^{\alpha_2}}$.

Thus with high probability Q is independent upto convex combinations from Y.

Next, set $w'' = k^{\alpha_3}$, where $1 > \alpha_3 > \alpha_1 + \gamma$ is any constant. Now consider the random variable R.

Claim 4.5.12. *R is 2^{-k^β} close to a $k^\gamma \times w''$ $(w'' - 2k^\beta)$-SR-source.*

Proof. This follows from an application of Lemma 2.1.11. ■

Now we assume that R is in fact a $w'' - 2k^\beta$-SR-source (we will add 2^{-k^β} to the final error).

After we fix S, R can lose entropy in its random row, but not much. We can expect it to lose as many bits of entropy as there are in S, which is only $k^{\alpha_1+\gamma}$. Since we picked $w'' = k^{\alpha_3} \gg k^{\alpha_1+\gamma}$, we get that R still contains entropy.

Claim 4.5.13. $\Pr_{s \leftarrow_R S}[R|S = s$ *is a $k^\gamma \times w''$ $(w'' - 2k^{\alpha_3})$-SR-source* $] > 1 - 2^{-k^{\alpha_3}}$.

Proof. By Proposition 2.1.9, we get that $\Pr_{s \leftarrow_R S}[R|S = s$ is a $k^\gamma \times w''$ $(w'' - k^{\alpha_1+\beta} - l)$-SR-source $] > 1 - 2^l$. Setting $l = k^{\alpha_3}$ gives the claim. ■

Thus, upto a typical fixing of S, (Q, R) are statistically close to two aligned sources, Q a $k^\gamma \times w'$ SR-source, and R an independent $k^\gamma \times w''$ $(0.1w'')$-SR source. If we set $d = w'/10$, we see that our application of Raz_2 above succeeds. In the aligned good row, Raz_2 gets two independent (after fixing S) sources which are statistically close to having extremely high entropy.

The result of applying Raz_2 is the random variable H.

Claim 4.5.14. *H is $2^{-\Omega(d)}$ close to a $(k^\gamma, \Omega(d))$ SR-source.*

In addition, we argue that the random row of H is independent of both X and Y. Without loss of generality, assume that H^1 is the random row of H. Let $\alpha_4 > 0$ be a constant s.t. $2^{-k^{\alpha_4}}$ is an upperbound on the error of $\mathsf{Ext}_1, \mathsf{Ext}_2$. Then for a typical fixing of Q, R, we get that X, Y are independent sources, and the random row of H (which is determined by (Q, R)) is a good seed to extract from both sources.

Claim 4.5.15. *With high probability H contains a good seed to extract from each of the sources:*

$$\Pr_{(q,r)\leftarrow_R(Q,R)}[|\mathsf{Ext}_2((Y|R{=}r), h^1(q,r)) - U_m| \geq 2^{-k^{\alpha_4}}]] < 2^{-k^{\alpha_4}}$$

and

$$\Pr_{(q,r)\leftarrow_R(Q,R)}[|\mathsf{Ext}_1((X|S{=}s(r), Q{=}q), h^1(q,r)) - U_m| \geq 2^{-k^{\alpha_4}}] < 2^{-k^{\alpha_4}}$$

Proof. There are two ways in which the claim can fail. Either S, Q, R steal a lot of entropy from X, Y, or they produce a bad seed in H to extract from $X|S = s, Q = q$ or $Y|R = r$. Both events happen with small probability.

Specifically, we have that there exist constants β_1, β_2 s.t.

- By Proposition 2.1.10, $\Pr_{r\leftarrow_{\mathrm{R}}R}[H_\infty(Y|R = r) < k - k^{\beta_1}] < 2^{-k^{\beta_2}}$
- By Proposition 2.1.10, $\Pr_{(q,r)\leftarrow_{\mathrm{R}}R}[H_\infty(X|R = r, Q = q) < k - k^{\beta_1}] < 2^{-k^{\beta_2}}$
- By our earlier claims, $\Pr_{r\leftarrow_{\mathrm{R}}R}[H|R = r$ is $2^{-k^{\beta_2}}$-close to being somewhere random]
- By our earlier claims, $\Pr_{(s,q)\leftarrow_{\mathrm{R}}(S,Q)}[H|S = s, Q = q$ is $2^{-k^{\beta_2}}$-close to being somewhere random]
- By the properties of the strong seeded extractor Ext_1, for any s, q such that $H_\infty(X|S = s, Q = q) \geq k - k^{\beta_1}$ and $H|S = s, Q = q$ is $2^{-k^{\beta_2}}$-close to being somewhere random,

$$\Pr_{h\leftarrow_{\mathrm{R}}H|Q=q,S=s}[|\mathsf{Ext}_1((X|S = s, Q = q), (H|S = s, Q = q)) - U_m| \geq 2^{-k^{\beta_2}}] < 2 \cdot 2^{-k^{\beta_2}}$$

- By the properties of the strong seeded extractor Ext_2, for any r such that $H_\infty(Y|R = r) \geq k - k^{\beta_1}$ and $H|R = r$ is $2^{-k^{\beta_2}}$-close to being somewhere random,

$$\Pr_{h\leftarrow_{\mathrm{R}}H|R=r}[|\mathsf{Ext}_2((Y|R = r), (H|R = r)) - U_m| \geq 2^{-k^{\beta_2}}] < 2 \cdot 2^{-k^{\beta_2}}$$

Thus we can use the union bound to get our final estimate. ∎

∎

This concludes the proof of Theorem 4.5.7.

∎

4.5.3 Proof of the main theorem

We are finally ready to prove Theorem 4.5.3. We are given a block-wise source X and a somewhere high entropy source Y that has many rows (more than k rows). The high level idea is to run "condensing steps" where each condensing step consumes one block of X while reducing the number of rows in Y by dividing it by $k^{\Omega(1)}$. The precise details follow:

Proof of Theorem 4.5.3. As in Chapter 3, the theorem is obtained by repeated condensing of SR-sources. In each condensing step, we will consume one block of X to reduce the number of rows of the SR-source by a factor of $k^{\Omega(1)}$. Thus after $O(\log t/\log k)$ steps, we will have reduced the number of rows to just 1, at which point extraction becomes trivial.

Algorithm 4.5.16 ($\mathsf{Cond}(x, y)$)**.**

Input: $x = x^1, x^2, \cdots, x^{\mathsf{C}}$, a sample from a block source and y a sample from a $t \times k$ SR-source.
Output: $z = x^2, x^3, \cdots, x^{\mathsf{C}}$ and y' a $t/k^{\gamma} \times m$ sample that we will claim comes from a SR-source.

Sub-Routines and Parameters:
Set $\gamma \ll 1/2$ to some constant value. Let β be the constant guaranteed by Theorem 4.5.3.
For these γ, β, let $\mathsf{BasicExt}$ be the function promised by Theorem 4.5.7. Let m, ϵ be the output length and error of $\mathsf{BasicExt}$ respectively.

1. Partition the t rows of y equally into t/k^{γ} parts, each containing k^{γ} rows. Let $y^{(j)}$ denote the j'th such part.
2. For all $1 \le j \le t/k^{\gamma}$, let $y'_j = \mathsf{BasicExt}(x^1, y^{(j)})$.
3. Let y' be the string with rows $y'_1, y'_2, \ldots, y'_{t/k^{\gamma}}$.

Given $X = X^1, \cdots, X^{\mathsf{C}}$ and Y, the above algorithm uses X^1 to condense Y. Even though this introduces dependencies between X and Y, once we fix X^1, the two output distributions are once again independent. Formally we will argue that after applying the condenser, the output random variables Z and Y' above are statistically close to a convex combination of independent sources, where Z is a block source with one less block than X, and Y' is an SR-source with much fewer rows than Y.

Lemma 4.5.17. *Let X, Y be as above. Let ϵ be the error of* $\mathsf{BasicExt}$. *Then $(Z = X^2, \ldots, X^{\mathsf{C}}, Y')$ is $2\sqrt{\epsilon}$-close to a convex combination of sources where each source in the combination has*

1. *Z is a $(k, \ldots, k)$ block source*
2. *Y' is a $(t/k^{\gamma}, m)$ SR-source*
3. *Z is independent of Y'*

Proof. Let $h \in [t/k^{\gamma}]$ be such that $Y^{(h)}$ contains the random row. Consider the random variable X^1. We will call x^1 *good* if $|\mathsf{BasicExt}(Y^{(h)}, x^1) - U_m| < \sqrt{\epsilon}$, where m, ϵ are the output length and error of $\mathsf{BasicExt}$ respectively.

Then we make the following easy claims:

Claim 4.5.18. *For good x^1,*

1. *$Z|X^1 = x^1$ is a $(k, \ldots, k)$ block source*
2. *$Y'|X^1 = x^1$ is a $\sqrt{\epsilon}$-close to being a $((t/k^\gamma) \times m)$ SR-source*
3. *$Z|X^1 = x^1$ is independent of $Y'|X^1 = x^1$*

Proof. The first and third property are trivial. The second property is immediate from the definition of *good*. ■

Claim 4.5.19. $\Pr[X^1$ *is not* good $] < \sqrt{\epsilon}$

Proof. This is an immediate consequence of Theorem 4.5.7. ■

These two claims clearly imply the lemma.

■

Now we use Cond repeatedly until the second source contains just one row. At this point we use the one row with Raz's extractor from Theorem 2.6.6 with X to get the random bits.

To see that the bits obtained in this way are strong, first note that Raz's extractor is strong in both inputs. Let O be the random variable that denotes the output of our function $\mathsf{BExt}(X, Y)$. Let Q denote the concatenation of all the blocks of X that were consumed in the condensation process. Let U_m denote a random variable that is independent of both X, Y. Then we see that these variables satisfy the hypothesis of Lemma 2.2.3, i.e., on fixing Q to a good value, Raz's extractor guarantees that the output is independent of both inputs, thus we must have that the output is close to being independent of both inputs. The dominant error term in BExt comes from the first step, when we convert Y to an SR-source.

■

4.6 3-source Extractors for Polynomially Small Min-entropy

In this section we build on the results from the last section to give a new extractor for just 3 independent sources, even when the min-entropy is polynomially small. One way to use our results to obtain a new extractor for just 3 sources was noticed by Avi Wigderson. We have outlined that approach in Appendix A.

We shall now describe another 3 source extractor which can handle a much lower entropy level, under the assumption that one of the sources is much shorter than the others. Our extractor uses as a key component a randomness condenser, constructed by Guruswami, Umans and Vadhan [GUV07], which is in turn based on recent constructions of good list decodable codes ([GR06, PV05]), though we give a self contained proof of everything we need in this section.

First let us give a high level description of our algorithm and analysis. As we saw in the last section, although it seems hard to build extractors for two independent sources, the problem seems considerably easier when one of the sources is a block source. Indeed, our new algorithm will be obtained by reducing to this case. We will give an algorithm that given two independent sources, can turn them into a single block source, with many blocks. Once we have this algorithm, we will simply use one additional source and our extractor from the previous section to get random bits.

4.6.1 Converting Two Independent Sources Into A Block Source

Fix a finite field $\mathbb{F}$. The following algorithm is from [GUV07].

Algorithm 4.6.1 ($\mathsf{Cond}(f, y)$)**.**

Input: $f \in \mathbb{F}^t$, $y \in \mathbb{F}$ and an integer r.
Output: $z \in \mathbb{F}^r$.

Sub-Routines and Parameters:
Let $g \in \mathbb{F}[X]$ be an irreducible polynomial of degree $t+1$. Set $h = |\mathbb{F}|^{0.8\alpha}$ for some parameter α.

```
1. For every $i = 0, 1, \ldots, m-1$, let $f_i \in \mathbb{F}[x]$ be the polynomial $f^{h^i} \mod g$.
2. Output $f_0(y), f_1(y), \ldots, f_{r-1}(y)$.
```

Guruswami et al. were interested in building seeded condensers. So they used the above algorithm with y sampled uniformly at random. Below, we show that the algorithm above is useful even when y is a high min-entropy source. We can prove the following lemma, which is a slight generalization of a lemma in [GUV07]:

Lemma 4.6.2. *If F is a distribution on $\mathbb{F}^t$ with min-entropy k and Y is an independent distribution on $\mathbb{F}$ with min-entropy rate α and*

- $rt < \epsilon|\mathbb{F}|^{0.1\alpha}$
- $k > \log(2/\epsilon) + (0.8\alpha r)\log(|\mathbb{F}|)$

$\mathsf{Cond}(F, Y)$ is ϵ-close to having min-entropy rate 0.7α.

Remark 4.6.3. In order to avoid using too many variables, we have opted to use constants like 0.1 and 0.7 in the proof. We note that we can easily replace the constants $0.7, 0.8$ with constants that are arbitrarily close to 1, at the price of making 0.1 closer to 0.

Proof of Lemma 4.6.2. We will repeatedly use the basic fact that any non-zero polynomial of degree d can have at most d roots.

By Fact 2.2.5, it suffices to prove the lemma when F and Y are flat sources.

We will prove that the output is close to having high min-entropy via Lemma 2.1.14. To do this, we need to show that for every set $S \subset \mathbb{F}^r$ of size $\epsilon|\mathbb{F}|^{0.7\alpha r}$, $\Pr[\mathsf{Cond}(F, Y) \in S] < \epsilon$. Fix such a set S.

Let $Q(Z_1, \ldots, Z_r) \in \mathbb{F}[Z_1, \ldots, Z_r]$ be a non-zero r variate polynomial whose degree is at most $h-1$ in each variable, such that $Q(s) = 0$ for every $s \in S$. Such a polynomial must exist since the parameters have been set up to guarantee $h^r = |\mathbb{F}|^{0.8\alpha r} > |S| = \epsilon|\mathbb{F}|^{0.7r\alpha}$.

Now call $f \in \mathsf{supp}(F)$ bad for S if

$$\Pr_{y \leftarrow_{\mathrm{R}} Y}[\mathsf{Cond}(f, y) \in S] \geq \epsilon/2$$

We will bound the number of bad f's. Fix any such bad f. Then consider the univariate polynomial

$$R(X) = Q(f_0(X), f_1(X), \dots, f_{r-1}(X)) \in \mathbb{F}[X]$$

This polynomial has degree at most $tr(h-1)$. But $tr(h-1) < \epsilon|\mathbb{F}|^{0.1\alpha}|\mathbb{F}|^{0.8\alpha} < \epsilon|\mathbb{F}|^{\alpha}/2 = (\epsilon/2)|\mathsf{supp}(Y)|$, thus this polynomial must be the zero polynomial. In particular, this means that $R(X) = 0 \mod g(X)$. This in turn implies that f must be a root of the polynomial

$$Q'(Z) = Q(Z, Z^h, Z^{h^2}, \dots, Z^{h^{r-1}}) \in (\mathbb{F}[X]/g(X))[Z]$$

which is a univariate polynomial over the extension field $\mathbb{F}[X]/g(X)$, since $Q'(f(X)) = R(X) \mod g(X)$ by our choice of $f_0, \dots, f_{r-1}$.

Recall that Q had degree at most $h-1$ in each variable. This means that Q' has degree at most $h^r - 1$ and is non-zero, since no two monomials can clash when making the substitution Z^i for Z_i in Q. The number of bad f's can be at most $h^r - 1 < |\mathbb{F}|^{0.8\alpha r}$, since every bad f is a root of this low degree non-zero polynomial. This implies that $\Pr[F \text{ is bad}] < |\mathbb{F}|^{0.8\alpha r}/2^k < \epsilon/2$, since the constraint on k implies that $2^k > |\mathbb{F}|^{0.8\alpha r}2/\epsilon$.

By the union bound, $\Pr[\mathsf{Cond}(F,Y) \in S] \leq \Pr[F \text{ is bad}] + \Pr[\mathsf{Cond}(F,Y) \in S | F \text{ is not bad}] < \epsilon/2 + \epsilon/2 = \epsilon$.

□

Guruswami et al. [GUV07] were interested in constructing a seeded condenser, so they were interested in the special case of $\alpha = 1$ in the above lemma. When α is small, it seems like the lemma doesn't say anything useful, since the min-entropy rate of the output is bounded above by α. But note that the lemma works for a very wide range of r's. The above function is more than a condenser, it *spreads* the entropy out across the output. Specifically, if we look at the first r' symbols in the output, they must also have min-entropy rate close to 0.7α. We can use this to construct a block source with geometrically increasing block lengths, as in the following lemma:

Lemma 4.6.4. *Let $\mathsf{Cond}, F, Y, \alpha, r, t, \epsilon$ be as in Algorithm 4.6.1 and Lemma 4.6.2. Let $r_1, r_2, \dots, r_{\mathsf{C}} = r$ be positive integers. For $i = 1, 2, \dots, \mathsf{C}$, set Z^i to be the first r_i field elements in the output of $\mathsf{Cond}(F,Y)$. Then let $Z_1, \dots, Z_{\mathsf{C}}$ be such that $Z^i = Z_1, \dots, Z_i$ for every i. Then for every $\ell > 10 \log \mathsf{C}$ we have that $Z_1, Z_2, \dots, Z_{\mathsf{C}}$ is $\mathsf{C}(\mathsf{C}\epsilon + 2^{-\ell+1})$-close to being a block source with entropy $(0.7\alpha r_i - r_{i-1})\log(|\mathbb{F}|) - 1 - 2\ell$ in each block.*

Proof. We will apply Lemma 2.1.20.

Note that for each i, Z^i is simply the output of the condenser upto the first r_i elements. Since $r_i \leq r$, r_i satisfies the constraints of Lemma 4.6.2, so Z^i is ϵ close to having min-entropy rate 0.7α. □

We set parameters to get the following theorem:

Theorem 4.6.5. *There exists a polynomial time computable function $\mathsf{BlockConvert} : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \to \{0,1\}^{m_1} \times \{0,1\}^{m_2} \times \cdots \times \{0,1\}^{m_{\mathsf{C}}}$, such that for every min-entropy k_1 source X over $\{0,1\}^{n_1}$ and every min-entropy k_2 source Y over $\{0,1\}^{n_2}$ satisfying*

- $\mathsf{C}(\log \frac{10n_2}{k_2}) + 2\log(n_1) < 0.095k_2$
- $\sqrt{k_1} > k_2(10n_2/k_2)^{\mathsf{C}}$,

$\mathsf{BlockConvert}(X, Y)$ *is* $\mathsf{C}^2 2^{-\Omega(k_2)} + 2^{-k_1^{\Omega(1)}}$*-close to being a block source with* $\sum_i m_i \le (10n_2/k_2)^{\mathsf{C}}\sqrt{k_1}$ *and min-entropy* $2\sqrt{k_1}$ *in each block.*

Proof. We show how to set parameters and apply Lemma 4.6.4.

Set $\mathbb{F}$ to be the finite field of size 2^{n_2}. Set $t = n_1/n_2, \epsilon = 2^{-0.05k_2}$ and $k = k_1$.

Set $r_i = (10n_2/k_2)^i\sqrt{k_1}$, so $\sum_i m_i = r = k_1^{1/2}(10n_2/k_2)^{\mathsf{C}}$.

Using the first assumption,

$$rt = \sqrt{k_1}\left(\frac{10n_2}{k_2}\right)^{\mathsf{C}}\frac{n_1}{n_2} \le n_1^2\left(\frac{10n_2}{k_2}\right)^{\mathsf{C}} < 2^{0.095k_2} = 2^{-0.05k_2}2^{0.1k_2} = \epsilon|\mathbb{F}|^{0.1\alpha}$$

to satisfy the first constraint of Lemma 4.6.2.

We have that

$$k_1 = k > 1 + 0.05k_2 + 0.8 \cdot 10^{\mathsf{C}} k_2\sqrt{k_1}(n_2/k_2)^{\mathsf{C}-1} = \log(2/\epsilon) + (0.8r\alpha)\log(|\mathbb{F}|)$$

to satisfy the second constraint of Lemma 4.6.2.

Set $\ell = k_1^{0.1}$. Note that the second constraint implies that $\mathsf{C} < \log k_1$.

Then let us use the algorithm Cond as promised by Lemma 4.6.4 with the above settings. We get that the final output is $\mathsf{C}(\mathsf{C}\epsilon + 2^{-\ell+1}) \le 2^{-\Omega(k_2)} + 2^{-k_1^{\Omega(1)}}$- close to being a block source with min-entropy $(0.7\alpha r_i - r_{i-1})\log(|\mathbb{F}|) - 1 - 2\ell$ in each block. We can lower bound this as follows:

$$\begin{aligned}
&(0.7\alpha r_i - r_{i-1})\log(|\mathbb{F}|) - 1 - 2\ell \\
&= \left(0.7\frac{k_2}{n_2}\left(\frac{10n_2}{k_2}\right)^i\sqrt{k_1} - \left(\frac{10n_2}{k_2}\right)^{i-1}\sqrt{k_1}\right)n_2 - 1 - 2k_1^{0.1} \\
&= (0.7 \cdot 10 - 1)\left(\frac{10n_2}{k_2}\right)^{i-1} n_2\sqrt{k_1} - 1 - 2k_1^{0.1} \\
&= 6\left(\frac{10n_2}{k_2}\right)^{i-1} n_2\sqrt{k_1} - (1 + 2k_1^{0.1}) \\
&\ge 2\sqrt{k_1}
\end{aligned}$$

■

4.6.2 Putting it all together

All that remains is to put together the various components to get our extractor.

Algorithm 4.6.6 ($\mathsf{IndepExt}(a, b, c)$)**.**

Input: $a \in \{0,1\}^{n_1}, b \in \{0,1\}^{n_2}, c \in \{0,1\}^{n_3}$.
Output: $z \in \{0,1\}^m$ for a parameter m that we will set.

Sub-Routines and Parameters:
Let $\mathsf{BlockConvert}$ be the algorithm promised by Theorem 4.6.5, set up to operate on two sources with entropy k_1, k_2 and lengths n_1, n_2 respectively.
Let BExt be the algorithm promised by Theorem 4.5.1, set up to extract from a block source with C blocks of length $(10n_2/k_2)^{\mathsf{C}}\sqrt{k_1}$, each with entropy $\sqrt{k_1}$ conditioned on previous blocks, and an independent source with length n_3 and min-entropy k_3.

1. Run $\mathsf{BlockConvert}(a, b)$ to get the blocks $x = x_1, x_2, \ldots, x_{\mathsf{C}}$.
2. Output $\mathsf{BExt}(x, c)$.

We can then prove the following theorem.

Theorem 4.6.7 (Three Source Extractor)**.** *There exists a constant d and a polynomial time computable function $\mathsf{3Ext} : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^{n_3} \to \{0,1\}^m$ which is an extractor for three sources with min-entropy requirements $k_1, k_2, k_3 = \sqrt{k_1}$, error $2^{-\Omega(k_2)} + 2^{-k_1^{\Omega(1)}}$ and output length $m = k_1 - o(k_1)$ as long as:*

- $\frac{\log k_1}{\log n_2} > d\frac{\log(n_1+n_3)}{\log k_1}$
- $k_2 > d \log n_1$

Proof. Let t be a constant so that BExt requires $\mathsf{C} = t\log(n_1 + n_3)/\log(k_1)$ blocks to extract bits from an $(n_3, k_3 = \sqrt{k_1})$ source and an independent block source with blocks of length n_1, each with entropy $\sqrt{k_1}$ conditioned on previous blocks. The error of this extractor is promised to be $2^{-k_1^{\Omega(1)}}$.

We check each of the constraints needed for $\mathsf{BlockConvert}$ to succeed.

First we have that

$$\begin{aligned}
&\mathsf{C}\left(\log\frac{10n_2}{k_2}\right) + \log n_1 \\
&< \mathsf{C}10\log n_2 + \log n_1 \\
&\leq t\frac{\log(n_1+n_3)}{\log k_1}\log n_2 + \log n_1 \\
&\leq (t/d)\log k_1 + \log n_1 && \text{by the first assumption} \\
&< 0.095d\log n_1 && \text{for } d \text{ large enough} \\
&< 0.095k_2 && \text{by the second assumption}
\end{aligned}$$

For the next constraint,

$$\begin{aligned}
&\log(k_2(10n_2/k_2)^{\mathsf{C}}) \\
&= \mathsf{C}\log(10n_2/k_2) + \log k_2 \\
&\leq t\frac{\log(n_1+n_3)}{\log k_1}(\log(n_2) + \log 10) + \log n_1 \\
&< 3(t/d)\log k_1 && \text{by the first assumption} \\
&< (1/2)\log k_1 && \text{for } d \text{ large enough}
\end{aligned}$$

We are not yet done, since the algorithm above will only output $m_1 = \sqrt{k_1} - o(\sqrt{k_1})$ bits. However, we do have that:

$$\Pr_{x_1 \leftarrow_{\mathrm{R}} X_1}[|\mathsf{IndepExt}(x_1, Y, Z) - U_{m_1}| > 2^{-\Omega(k_2)} + 2^{-k_1^{\Omega(1)}}] < 2^{-\Omega(k_2)} + 2^{-k_1^{\Omega(1)}}$$

since BExt is strong.

Thus we have that $|X, \mathsf{IndepExt}(X, Y, Z) - X, U_{m_1}| < 2^{-\Omega(k_2)} + 2^{-k_1^{\Omega(1)}}$, which implies that if Ext is any strong seeded extractor set up to extract from a min-entropy k_1 source with seed length m_2, $\mathsf{Ext}(X, U_{m_1})$ is $2^{-\Omega(k_2)} + 2^{-k_1^{\Omega(1)}}$ close to $\mathsf{Ext}(X, \mathsf{IndepExt}(X, Y, Z))$. This is our final extractor.

■

One example of how to set parameters is in the following corollary:

Corollary 4.6.8. *There exists a constant h such that for every constant $\gamma > 1$, there is a polynomial time computable function $\mathsf{3Ext} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^{n^{\gamma/h}} \to \{0,1\}^{n^\gamma - o(n^\gamma)}$ which is an extractor for three sources with min-entropy requirements $k = n^\gamma, n^\gamma, \log^{10} n$, error $2^{-\Omega(\log^{10} n)}$.*

When the min-entropy of the sources gets lower, the first constraint in the theorem forces the length of the shorter source to be much shorter than the other two sources.

Chapter 5

Extractors for Small Space Sources

Trevisan and Vadhan [TV00] proposed the study of extraction from weak random sources that are generated by a process that has a bounded amount of computational resources. This seems to be a plausible model for physical random sources and generalizes a number of the previously studied models. They focused on the case that the source is sampled by either a small circuit or an algorithm with a limited running time. Their main result is a construction of polynomial-time extractors for such sources based on some strong but plausible complexity assumptions. It would be nice to have unconditional constructions (as well as ones that are more efficient and have better error). However, they showed that complexity assumptions are needed for the original model of sources generated by time-bounded algorithms. Thus, they suggested, as a research direction, that we might be able to construct unconditional extractors for sources generated by *space-bounded* algorithms. This model is our focus.

Small space sources are very general in that most other classes of sources that have been considered previously can be computed with a small amount of space. This includes von Neumann's model of a coin with unknown bias [vN51], Blum's finite Markov chain model [Blu84], symbol-fixing sources [KZ03], and sources that consist of many independent sources. Strong results in this last model will not follow directly from strong results in the small-space model, but our results do generalize, for example, the results of [BIW04]. In fact, the only model for which deterministic extractors have been given that appears unrelated to our model is "affine sources". Yet despite the small-space model being so natural, very little in the way of explicit constructions for such sources was known.

The first example of an explicit construction was due to Blum [Blu84], who showed how to extract from sources generated by a finite Markov chain with a constant number of states. His results generalized the earlier results of von Neumann [vN51] for extracting from an independent coin with unknown bias. However, the finite Markov chain model is very restricted; it has a constant-size description and the transitions must be the same at each time step.

The model for small-space sources we consider is similar to the one previously considered by Koenig and Maurer [KM04, KM05]. It is a generalization of the Markov chain model to time-dependent Markov chains, which yields a much richer class of sources. Our model of a space s source is basically a source generated by a width 2^s branching program. The exact model we consider is that at each step the process generating the source is in one of 2^s states. This can be modelled by a layered graph with each layer corresponding to a single time-step and consisting of vertices corresponding to each of the states. From each node v in layer i, the edges leaving v (going

Table 5.1: Small space extractors for sources on $\{0,1\}^n$ that extract 99% of the min-entropy. In this table c and C represent sufficiently small and large constants, respectively.

Reference	Min-entropy Rate	Space	Error
Theorem 5.0.9	$\delta \geq n^{-c}$	$c\delta^3 n$	$\exp(-n^c)$
Theorem 5.0.11	Any constant δ	cn	$\exp(-\tilde{\Omega}(n))$
Theorem 5.0.12	$\delta \geq C/\log n$	$c\delta \log n$	$\exp(-n^{.99})$

to layer $i+1$) are assigned a probability distribution as well as an output bit for each edge. Unlike in Blum's model [Blu84], the transitions can be different at each time-step.

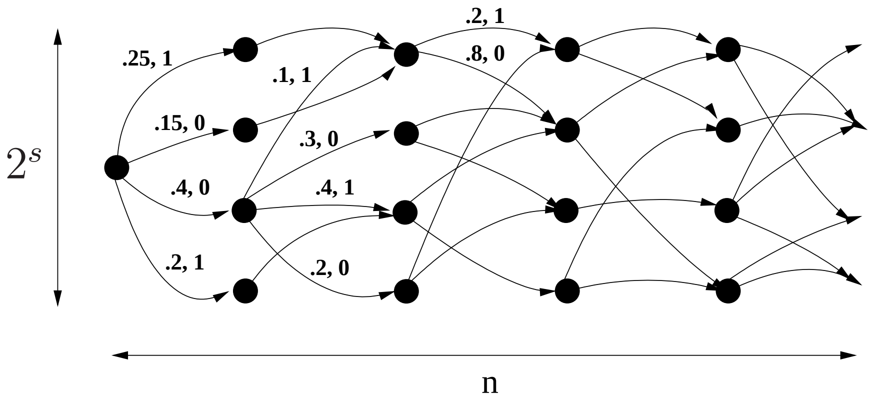

Figure 5.1: Part of a space $s = 2$ source

It can be shown using the probabilistic method that there exist extractors even when the space s is a constant fraction of the min-entropy k, even when the min-entropy is logarithmically small. Our goal is to provide *efficient* and *deterministic* constructions with parameters that come as close to these bounds as possible.

Koenig and Maurer [KM04, KM05] gave the first explicit constructions of extractors for space-bounded sources. Their extractors require the min-entropy rate to be least 1/2. We do not know of any other constructions for space-bounded sources, even space 0 sources, which are simply sources of independent bits each of which has a different, unknown, bias.

Our Results

The results in this chapter are based on work with Jesse Kamp, Salil Vadhan and David Zuckerman [KRVZ06].

For space s sources with min-entropy $k = \delta n$, we have several constructions, all of which are able to extract almost all of the entropy in the source. These extractors are summarized in Table 5.1. The first extracts whenever $\delta > n^{-\eta}$ for some fixed constant η and extracts almost all of the entropy.

Theorem 5.0.9. *Assume we can find primes with length in* $[r, 2r]$ *deterministically in time* $\mathrm{poly}(r)$*. Then there is a constant* $\eta > 0$ *such that for every* $n \in \mathbb{N}$*, and* $\delta > \zeta > n^{-\eta}$*, there is an polynomial-*

time computable ϵ-extractor $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ *for space s sources with min-entropy rate δ, where $s = \Omega(\zeta^3 n)$, $m = (\delta - \zeta)n$, and $\epsilon = 2^{-n^{\Omega(1)}}$.*

Remark 5.0.10. The assumption about finding primes follows from Cramer's conjecture on the density of primes [Cra37], together with the deterministic primality test of [AKS04].

We also have constructions that do not depend on the ability to find large primes. Though the parameters of these constructions are mostly subsumed by the previous construction, they are considerably simpler and achieve somewhat better error. For constant min-entropy rate sources, we have a construction that extracts any constant fraction of the entropy.

Theorem 5.0.11. *For any constants $\delta > \zeta > 0$ and every $n \in \mathbb{N}$, there is a polynomial-time computable ϵ-extractor* $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ *for space s sources with min-entropy rate δ, where $s = \Omega(n)$, $m = (\delta - \zeta)n$, and $\epsilon = 2^{-\Omega(n/\log^3 n)}$.*

The last extractor works with min-entropy rate as low as $\delta = \Omega(1/\log n)$ and space $O(\delta \log n)$.

Theorem 5.0.12. *For every $n \in \mathbb{N}$ and $\delta > \zeta > 28/\log n$ and $s \leq (\zeta \log n)/28$, there is a polynomial-time computable ϵ-extractor* $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^m$ *for space s sources with min-entropy rate δ, where $m = (\delta - \zeta)n$ and $\epsilon = \exp(-n/(2^{O(s/\zeta)} \cdot \log^5 n))$.*

In comparison to the previous results (e.g. [KM04, KM05]) we have reduced the min-entropy required from $n/2$ to $n^{1-\Omega(1)}$ (in Theorem 5.0.9). However, we are still far from what can be achieved nonconstructively, where we can extract when the min-entropy is logarithmically small. We also have a gap in terms of the space tolerated. Nonconstructively we can get s to be almost $\delta n/2$ while our results require s to be smaller than $\delta^3 n$.

In a partial attempt to close the entropy gap for the case of space 1 sources, we also have an extractor that extracts about $\Omega(k^2/n)$ bits from a more restricted model when $k > n^{0.81}$. The extra restriction is that the output bit is required to be the same as the state.

5.0.3 Total-Entropy Independent Sources

Our extractors for small-space sources are all obtained via a reduction from a new model of sources we introduce called *total-entropy independent sources*. The reduction we use is based on that of Koenig and Maurer [KM04, KM05], who used it to generalize extractors for two independent sources. Total-entropy independent sources consist of a string of r independent sources of length ℓ such that the total min-entropy of all r sources is at least k. Our reduction shows that optimal extractors for total-entropy independent sources are also essentially optimal extractors for small-space sources. In addition to being a natural model, these sources are a common generalization of two of the main models studied for seedless extraction, namely symbol-fixing sources and independent sources , which we proceed to discuss below.

Independent Sources

One of the most well-studied models of sources is that of extracting from a small number of *independent sources*, each of which has a certain amount of min-entropy. We have given several extractors for this model in Chapter 4.

However, this model requires that all of the sources have large entropy. This motivates our generalization of independent sources to total-entropy independent sources, where we only require that the *total* min-entropy over all of the sources is high. Another difference between what we consider is that the usual independent source model consists of few sources that are long, whereas total-entropy independent sources are interesting even if we have many short sources.

Oblivious Bit-Fixing and Symbol-Fixing Sources

Another particular class that has been studied a great deal is that of *bit-fixing sources*, where some subset of the bit-positions in the source are fixed and the rest are chosen uniformly at random. The first extractors for bit-fixing sources extracted perfectly random bits [CFG+85, CW89] and required the source to have a large number of random positions. Kamp and Zuckerman [KZ03] constructed extractors that worked for sources with a much smaller number of random bits, and we have improved their results in Chapter 6. They also generalized the notion of bit-fixing sources to symbol-fixing sources, where instead of bits the values are taken from a d symbol alphabet. Gabizon, Raz and Shaltiel [GRS04] gave a construction that converts any extractor for bit-fixing sources into one that extracts almost all of the randomness.

Total-entropy independent sources can be seen as a generalization of symbol-fixing sources, where each symbol is viewed as a separate source.[1] The difference is that instead of each symbol being only fixed or uniformly random, the symbols (sources) in total-entropy independent sources are allowed to have any distribution as long as the symbols are independent. Naturally, we place a lower bound on the total min-entropy rather than just the number of random positions. Usually, symbol-fixing sources are thought of as having many symbols that come from a small alphabet (e.g. $\{0, 1\}$). This restriction is not necessary to the definition, however, and here we consider the full range of parameters, including even the case that we have a constant number of symbols from an exponentially large "alphabet" (e.g. $\{0, 1\}^\ell$).

Our Results

Our extractors for total-entropy independent sources are all based on generalizing various techniques from extractors for independent and symbol-fixing sources.

Koenig and Maurer [KM04, KM05] showed how any extractor for two independent sources with certain algebraic properties can be translated into an extractor for many sources where only two of the sources have sufficient entropy. Their result generalizes to extractors for more than two sources. We show that this also yields extractors for independent-symbol sources. In particular, we apply this to extractors for independent sources that follow from the exponential sum estimates of Bourgain, Glibichuk, and Konyagin [BGK06] (see Bourgain [Bou05]), and thereby obtain extractors for total-entropy independent sources of any constant min-entropy rate. These extractors are quite simple. Each source is viewed as being an element of a finite field, and the output of the extractor is simply the product of these finite field elements.

We also show how to use ideas from Chapter 4 for extracting from several independent sources to get extractors for total-entropy independent sources that extract from sources of min-entropy $(r\ell)^{1-\Omega(1)}$.

When the smaller sources each have short length ℓ, we use ideas from the work of Kamp and Zuckerman [KZ03] about bit-fixing sources to construct extractors for total-entropy independent

[1] Though for ease of presentation we define total-entropy independent sources only over sources with alphabet size 2^ℓ, more generally the sources could be over alphabets of any size d, as with symbol-fixing sources. All of our results naturally generalize to this more general case.

Table 5.2: Total-entropy independent source extractors for sources on $(\{0,1\}^\ell)^r$ that extract 99% of the min-entropy. In this table c and C represent sufficiently small and large constants, respectively.

Reference	Min-entropy Rate	Error
Theorem 5.0.13	$\delta = (r\ell)^{-c}$	$\exp(-(r\ell)^c)$
Theorem 5.0.14	Any constant δ	$\exp(-\tilde{\Omega}(r\ell))$
Theorem 5.0.15	$\delta = C\frac{d\log^{3/2} r}{(r\ell)^{\frac{1}{2}-\gamma}}$	$\exp(-(r\ell)^{2\gamma})$
Theorem 5.0.16	$\delta = (2^\ell \log r)^C/r$	$(\delta r\ell)^{-c}$

sources with min-entropy k. We can extract many bits when $k > 2^\ell\sqrt{r\ell}$, and for $k = \Omega(2^{2\ell}\ell)$ we can still extract $\Omega(\log k)$ bits. The base extractor simply takes the sum of the sources modulo p for some $p > 2^\ell$, similar to the cycle walk extractor in [KZ03]. Using this extractor we can extract $\Omega(\log k)$ bits. To extract more bits when k is sufficiently large, we divide the source into blocks, apply the base extractor to each block, and then use the result to take a random walk on an expander as in [KZ03].

Unlike the first two extractors, the extractors obtained using this technique use the full generality of the total-entropy independent sources. In the first two constructions, using a Markov argument we can essentially first reduce the total-entropy independent sources into sources where some of the input sources have sufficiently high min-entropy while the rest may or may not have any min-entropy. These reductions also cause some entropy to be lost. In this last construction, however, we benefit even from those sources that have very little min-entropy. Thus we are able to take advantage of all of the entropy, which helps us extract from smaller values of k.

We also show how to generalize the construction of Gabizon et al. [GRS04] to total-entropy independent sources to enable us to extract more of the entropy. Note that we use it to improve not only the extractors based on [KZ03] (analogous to what was done in [GRS04] for bit-fixing sources), but also our extractors based on techniques developed for independent sources. Independently of our work, Shaltiel [Sha06] has recently generalized the ideas in [GRS04] to give a framework for constructing deterministic extractors which extract almost all of the entropy from extractors which extract fewer bits. Our extractor can be seen to fit inside this framework, although we cannot directly use his results as a black box to obtain our results.

Applying the technique based on [GRS04] to our extractors that use the independent sources techniques of Chapter 3, the results of [BGK06], and the bit-fixing source extractor from [KZ03], respectively, we get the following three theorems. These theorems are directly used to obtain the small-space extractors from Theorem 5.0.9, Theorem 5.0.11, and Theorem 5.0.12. Table 5.2 presents a summary of these extractors.

Theorem 5.0.13. *Assuming we can find primes with length in $[r, 2r]$ deterministically in time $\mathrm{poly}(r)$, there is a constant η such that for every $r, \ell \in \mathbb{N}$ and $\delta > \zeta > (r\ell)^{-\eta}$, there is a polynomial-time computable ϵ-extractor $\mathsf{Ext} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ for sets of r independent sources over $\{0,1\}^\ell$ with total min-entropy rate $\delta > \zeta$ where $m = (\delta - \zeta)r\ell$ and $\epsilon = \exp(-(r\ell)^{\Omega(1)})$.*

We note that in the independent sources model this extractor gives comparable results to the extractor from [BIW04] as a corollary.

The following extractor extracts a constant fraction of the entropy from any constant rate source.

Theorem 5.0.14. *For any constants $\delta > \zeta > 0$ and every $r \in \mathbb{N}$, there is a polynomial-time computable ϵ-extractor $\mathsf{Ext} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ for sets of r total min-entropy rate δ independent smaller sources over $\{0,1\}^\ell$, where $m = (\delta - \zeta)r\ell$ and $\epsilon = 2^{-\Omega((r\ell)/\log^3(r\ell))}$.*

For the following extractor we can take $\zeta = \tilde{O}(1/\sqrt{r})$ and can then extract randomness from sources with min-entropy rate as small as $\delta = \tilde{O}(1/\sqrt{r})$.

Theorem 5.0.15. *For every $r \in \mathbb{N}$, $1 \leq \ell \leq \frac{1}{2}\log r$ and $\zeta > \sqrt{2^{2\ell}\log^3 r/r\ell}$ there is a polynomial-time computable ϵ-extractor $\mathsf{Ext} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ for r total min-entropy rate $\delta > \zeta$ independent smaller sources over $\{0,1\}^\ell$ where $m = (\delta - \zeta)r\ell$ and $\epsilon = \exp(-\Omega((\zeta^2 r\ell)/(2^{2\ell}\log^3 r)))$.*

Gabizon et al. also give a technique which improves the output length of extractors that extract only $\Omega(\log k)$ bits. We show that this technique also generalizes to total-entropy independent sources, so we use it together with our extractor based on ideas from [KZ03] that extracts $\Omega(\log k)$ bits to get the following theorem. This theorem shows that even for polylogarithmic k, for small enough ℓ we can extract almost all of the min-entropy.

Theorem 5.0.16. *There exists a constant $C > 0$ such that for every $r \in \mathbb{N}$, $\ell \geq 1$, $k \geq (2^\ell \log r)^C$, there exists a polynomial-time computable ϵ-extractor $\mathsf{Ext} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ for r independent smaller sources over $\{0,1\}^\ell$ with total min-entropy k, where $m = k - k^{1-\Omega(1)}$ and $\epsilon = k^{-\Omega(1)}$.*

Using the probabilistic method, we show that there exist (nonconstructive) extractors that extract even when the min-entropy k is as small as $O(\ell + \log r)$. Note that we always need $k > \ell$, since otherwise all of the entropy could be in a single source, and thus extraction would be impossible. The extractor from Theorem 5.0.16 comes closest to meeting this bound on k, but only works for small ℓ and has suboptimal error, so there is still much room for improvement.

5.0.4 Organization

In Section 5.2 we describe our reduction from small-space sources to total-entropy independent sources. We then restrict our focus to extracting from total-entropy independent sources, starting with the basic extractors. In Section 5.3 we describe the extractor that provides the basis for the extractor from Theorem 5.0.14. In Section 5.4 we describe the extractor that provides the basis for the extractor from Theorem 5.0.13. In Section 5.5 we describe the extractors that provide the basis for the extractors from Theorem 5.0.15 and Theorem 5.0.16. Then in Section 5.6, we describe how to generalize the techniques of Gabizon et al. [GRS04] so that we can extract almost all of the entropy, and so achieve the theorems described in the introduction. Next, in Section 5.7, we give nonconstructive results on extractors for both small-space and total-entropy independent sources. Finally, in Section 5.8, we give the improved extractor for our more restrictive model of space 1 sources.

5.1 Preliminaries

5.1.1 Classes of Sources

We formally define the various classes of sources we will study.

Definition 5.1.1. A *space s source* X on $\{0,1\}^n$ is a source generated by a width 2^s branching program. That is, the branching program is viewed as a layered graph with $n+1$ layers with a single start vertex in the first layer and 2^s vertices in each subsequent layer. Each edge is labeled with a probability and a bit value. From a single vertex we can have multiple edges corresponding to the same output bit. The source is generated by taking a random walk starting from the start vertex and outputting the bit values on every edge.

This definition is very similar to the general Markov sources studied by Koenig and Maurer [KM04, KM05]. This is not quite the most general model of such sources imaginable, because we could consider sources that output a variable number of bits depending on which edge is chosen at each step, including possibly not outputting any bits. However, this restriction makes sense in light of the fact that we are primarily interested in sources of fixed length. In this case, the sources in the more general model can be transformed into our model by modifying the states appropriately.

The other important class of sources we study are independent sources.

Definition 5.1.2. A source consisting of r smaller sources on $\{0,1\}^\ell$ is an *independent source* on $(\{0,1\}^\ell)^r$ if each of the r smaller sources are independent. An independent source on $(\{0,1\}^\ell)^r$ has total-rate δ if the total min-entropy over all of the sources is $\delta r \ell$ and total-entropy k if the total min-entropy is k.

Definition 5.1.3. A source on $\{0,1\}^\ell$ is *flat* if it is uniformly distributed over a non-empty subset of $\{0,1\}^\ell$.

Note that when $\ell = 1$, a flat independent source is the same as an oblivious bit-fixing source.

Definition 5.1.4. Let X be a random variable taking values in $\{0,1\}^{t\times a}$, viewed as $t \times a$ matrices with entries in $\{0,1\}$. We say that X on $(\{0,1\}^a)^t$ is $(t \times a)$ *somewhere-random* [2] (*SR-source* for short) if it is a random variable on t rows of r bits each such that one of the rows of X is uniformly random. Every other row may depend on the random row in arbitrary ways. We will say that a collection $X_1, \dots, X_m$ of $(t \times a)$ SR-sources is *aligned* if there is some i for which the i'th row of each X_j is uniformly distributed.

We will also need a relaxed notion of the previous definition to where the "random" row is not completely random, but only has some min-entropy.

Definition 5.1.5. We say that a $(t \times a)$ source X on $(\{0,1\}^a)^t$ has *somewhere-min-entropy* k, if X has min-entropy k in one of its t rows. We will say that a collection $X_1, \dots, X_m$ of $(t \times a)$ somewhere-min-entropy k sources is *aligned* if there is some i for which the i'th row of each X_j has min-entropy k.

5.1.2 Seeded Extractors

We will also need to define what it means to have a seeded extractor for a given class of sources.

[2] This definition is slightly different from the original one used by Ta-Shma [TS96]. The original definition considered the closure under convex combinations of the class defined here (i.e., convex combinations of sources that have one random row). We use this definition because we can do so without loss of generality and it considerably simplifies the presentation.

Definition 5.1.6. A polynomial-time computable function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^m$ is a *seeded ϵ-extractor* for a set of random sources $\mathcal{X}$, if for every $X \in \mathcal{X}$, $\mathsf{Ext}(X, U_s)$ is ϵ-close to uniform. The extractor is called *strong* if for Y chosen according to U_s, $Y, \mathsf{Ext}(X, Y)$ is also ϵ-close to uniform.

We use the following seeded extractor in our constructions, which allows us to get almost all the randomness out.

Theorem 5.1.7 ([Tre01, RRV02])**.** *For every $n, k \in \mathbb{N}$, $\epsilon > 0$, there is a polynomial-time computable strong seeded ϵ-extractor $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^{k - O(\log^3(n/\epsilon))}$ for sources with min-entropy k, with $t = O(\log^3(n/\epsilon))$.*

5.2 Small-Space Sources As Convex Combinations Of Independent Sources

Here we show how small-space sources can be converted into convex combinations of independent sources. Thus we will be able to use our extractor constructions from subsequent sections to extract from small-space sources. The idea is simple: to extract from a space s source X, we divide the n bits in X into n/t blocks of size t. We view each block as a source on t bits. If we condition on the states of the source at the start of each block, all of the blocks become independent, so we end up with a set of n/t independent smaller sources on $\{0,1\}^t$. We show, using techniques similar to Koenig and Maurer [KM04, KM05], that with high probability these sources will have sufficient min-entropy.

Lemma 5.2.1. *Let X be a space s source on $\{0,1\}^n$ with min-entropy rate δ. Then for any $0 < \alpha < 1$, X is $2^{-\alpha\delta n/2}$-close to a convex combination of independent sources on $(\{0,1\}^\ell)^r$ with total-rate δ', where $\ell = 2s/(\alpha\delta)$, $r = \alpha\delta n/2s$ and $\delta' = (1-\alpha)\delta$.*

All of our extractors for small-space sources are obtained by combining Lemma 5.2.1 with the corresponding extractor for total-entropy independent sources. We note that the reduction in this lemma is only interesting when the min-entropy rate $\delta > 1/\sqrt{n}$, since otherwise the total entropy of the independent sources would be less than the length of an individual source. In this case all of the entropy could be in a single source and thus extraction would be impossible.

To prove Lemma 5.2.1 we use the following standard lemma (for a direct proof see Lemma 5 in Maurer and Wolf [MW97], although it has been used implicitly earlier in, e.g., [WZ99]).

Lemma 5.2.2. *Let X and Y be random variables and let $\mathcal{Y}$ denote the range of Y. Then for all $\epsilon > 0$*

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

Proof of Lemma 5.2.1. Divide X into $\alpha\delta n/2s$ blocks of size $2s/\alpha\delta$. Let Y represent the values of the initial states for each block. Then each $(X|Y=y)$ is a set of independent smaller sources with each block viewed as a smaller source of length $2s/(\alpha\delta)$. By Lemma 5.2.2, since $|\mathcal{Y}| = (2^s)^{(\alpha\delta n)/(2s)} = 2^{\alpha\delta n/2}$, with probability $1 - 2^{-\alpha\delta n/2}$ the sources $(X|Y=y)$ have min-entropy $(1-\alpha)\delta n$ and thus min-entropy rate $(1-\alpha)\delta$. □

5.3 Extracting From Total-Entropy Independent Sources

In this section, we show how to construct extractors for total-entropy independent sources using techniques from standard independent sources.

The following Markov-like lemma will be used to show that if we divide a source into blocks, many of the blocks will have a large entropy rate.

Lemma 5.3.1. *For any partition of a total-rate δ independent source on $(\{0,1\}^\ell)^r$ into t blocks of r/t smaller sources each, the number b of blocks with min-entropy rate greater than $\delta/2$ satisfies $b > \delta t/2$.*

Therefore we can view this source as a set of t independent smaller sources on $\{0,1\}^{\ell r/t}$ where at least $\delta t/2$ of the smaller sources have min-entropy rate greater than $\delta/2$.

Proof. We know that b blocks have min-entropy rate greater than $\delta/2$ and at most 1. In each of the remaining blocks the min-entropy rate is at most $\delta/2$. Since the total entropy rate is δ and min-entropies add for independent sources, $\delta \leq (b+(t-b)(\delta/2))/t$, which after a simple calculation gives the desired result. □

Once we are in this model, we can generalize the result from Koenig and Maurer [KM04, KM05] that states that any two source extractor of the form $f(x_1 \cdot x_2)$, where the x_i are elements of some group, can be extended to any number of sources where only two of the sources have sufficient min-entropy.

Lemma 5.3.2. *Let $(\mathcal{G}, *)$ be a group and let $Ext(x_1, x_2, \dots, x_b) := f(x_1 * x_2 \cdots * x_b)$ be an extractor for b independent sources over $\mathcal{G}$, each of which has min-entropy rate at least δ. Then $F(x_1, \dots, x_r) := f(x_1 * \cdots * x_r)$ is an extractor for r independent sources over $\mathcal{G}$, b of which have min-entropy rate at least δ.*

The proof is simple and is the same as in [KM04, KM05]. The key idea is that the r sources can be divided into b blocks, each of which contains exactly one of the high entropy sources, since the group operation cannot lower the entropy.

Bourgain, Glibichuk, and Konyagin [BGK06] gave bounds on the exponential sums of the function $f(x_1, \dots, x_b) = \prod_{i=1}^{b} x_i$ over large subsets of fields without large subfields, in particular $GF(p)$ and $GF(2^p)$. This estimate gives an extractor for b independent sources where each source has high entropy via Vazirani's XOR lemma [Vaz86].

Theorem 5.3.3 ([BGK06])**.** *Let the finite field K be either $GF(p)$ or $GF(2^p)$ for some prime p. Let $f(x_1, \dots, x_b) = \prod_{i=1}^{b} x_i$ and view the output of the function as an integer from 0 to $|K| - 1$. Then there exist functions $B(\delta)$ and $c(\delta)$ such that the function $\mathsf{BGK}(x_1, \dots, x_b) = \lfloor (2^m f(x_1, \dots, x_b))/|K| \rfloor$ (i.e., taking the m most significant bits of $f(x_1, \dots, x_b)/|K|$) is an ϵ-extractor for b independent min-entropy rate δ sources over K for $b \geq B(\delta)$, $m = \Theta(c(\delta) \log |K|)$, and $\epsilon = 2^{-\Omega(m)}$.*

Note that for constant δ, we can extract $\Theta(\log |K|)$ bits with only a constant number of sources. For $GF(p)$, [BGK06] make explicit the relationship between δ and the number of sources and entropy. It turns out in this case that we can handle slightly subconstant δ, down to $\delta = \Omega(1/(\log \log |K|)^{(1/C)})$ for some constant C. For $GF(2^p)$, it's not clear whether or not a similar result can be achieved.

Combining this theorem with Lemma 5.3.2, restricting the sources to be over the multiplicative group K^*, we get the following corollary.

Corollary 5.3.4. *Let the finite field K be either $GF(p)$ or $GF(2^p)$ for some prime p. Let $f(x_1, \dots, x_r) = \prod_{i=1}^r x_i$ and view the output of the function as a number from 0 to $|K| - 1$. Then there exist functions $B(\delta)$ and $c(\delta)$ such that the function $\mathsf{BGK}(x_1, \dots, x_r) = \lfloor (2^m f(x_1, \dots, x_r))/|K| \rfloor$ is an ϵ-extractor for r independent sources over K^*, at least $B(\delta)$ of which have min-entropy rate at least δ, and with $m = \Theta(c(\delta) \log |K|)$ and $\epsilon = 2^{-\Omega(m)}$.*

It will also be useful to formulate the following corollary.

Corollary 5.3.5. *For every constant $\delta > 0$, there exists a constant $v(\delta)$ and a polynomial time computable function $\mathsf{BGK} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ that is an ϵ-extractor for r independent sources on $\{0,1\}^\ell$, such that at least $v(\delta)$ of the sources have min-entropy rate δ where $m = \Omega(\ell)$ and $\epsilon = 2^{-\Omega(\ell)}$.*

Proof. Find the next smallest prime $p > \ell$ (we know $p \leq 2\ell$), and apply the extractor from Corollary 5.3.4 over $GF(2^p)$, viewing each source as being embedded in $GF(2^p)^*$. □

Now we can combine this extractor with Lemma 5.3.1 to get an extractor for independent sources with constant total min-entropy rate.

Theorem 5.3.6. *For any constant δ, we can construct a polynomial-time computable ϵ-extractor $\mathsf{Ext} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ for total-rate δ independent sources on $(\{0,1\}^\ell)^r$, with $m = \Theta(r\ell)$ and $\epsilon = 2^{-\Omega(m)}$. This extractor can be computed in time $\mathrm{poly}(r, \ell)$.*

Proof. Given an independent source $X = X_1, \dots, X_n$ on $(\{0,1\}^\ell)^r$, divide it into $t = 2B(\delta/2)/\delta$ blocks of r/t smaller sources each, where $B(\delta)$ is the constant from Corollary 5.3.4. Then by Lemma 5.3.1, we can view X as an independent sources on $(\{0,1\}^{\ell r/t})^t$, where at least $\delta t/2 = B(\delta/2)$ of the smaller sources have min-entropy rate at least $\delta/2$. Find the smallest prime $p > (r\ell)/t$. By Bertrand's postulate, $p \leq 2(r\ell)/t$, we can find such a prime in time $\mathrm{poly}(r, \ell)$ by brute force search. Then we can embed each of our smaller sources into $GF(2^p)^*$ and apply the extractor from Corollary 5.3.4 to get the stated result. ■

5.4 Extracting From Polynomial Entropy Rate

In this section we will show how to extract from total-entropy independent sources when the min-entropy of the sources is polynomially small. As in the previous section, we will reduce the problem to another model: we will try to extract from a few independent sources when just some of them have a polynomial amount of entropy, but we don't know exactly which ones. The probabilistic method shows that extractors exist for this model even when just two sources contain logarithmic min-entropy and the total number of sources is polynomially large. Our main theorem is as follows.

Theorem 5.4.1. *Assuming we can find primes with length in $[r, 2r]$ in time $\mathrm{poly}(r)$, there is a constant β such that there exists a polynomial-time computable ϵ-extractor $\mathsf{Ext} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ for total-rate $\delta \geq \ell^{-\beta}$ independent sources on $(\{0,1\}^\ell)^r$, where $n = \Theta(1/\delta^2)$, $m = \ell^{\Omega(1)}$ and $\epsilon = 2^{-\ell^{\Omega(1)}}$.*

We can also get the following corollary for when we have a larger number of smaller sources.

Corollary 5.4.2. *Assuming we can find primes with length in $[r, 2r]$ in time $\mathrm{poly}(r)$, there exists a constant η such that for any $\delta \geq (r\ell)^{-\eta}$, there exists a polynomial-time computable ϵ-extractor $\mathsf{Ext} : (\{0,1\}^\ell)^r \to \{0,1\}^m$ for total-rate δ independent sources on $(\{0,1\}^\ell)^r$, where $m = (\delta^2 r\ell)^{\Omega(1)}$ and $\epsilon = 2^{-(\delta^2 r\ell)^{\Omega(1)}}$.*

Proof. Divide the source into $\Theta(1/\delta^2)$ blocks of $\Theta(\delta^2 n)$ smaller sources each and apply Theorem 5.4.1. □

In this section we will describe a generic technique to turn any extractor for the model where a few smaller sources have min-entropy rate less than half into an extractor that can extract when the min-entropy is as small as $\ell^{1-\alpha_0}$ for some universal constant α_0. There are two major ingredients that will go into our construction:

- The first ingredient is based on recent constructions of randomness efficient condensers [BKS$^+$05, Raz05]. We use these condensers to transform a set of sources with polynomial min-entropy rate into a set of aligned sources with somewhere-min-entropy rate 0.9. An important property that we will need is that the length of each of the rows is much higher than the number of rows. We prove the following theorem in Section 5.4.2.

  Theorem 5.4.3. *Assume we can find primes with length in $[r, 2r]$ in time $\mathrm{poly}(r)$. Let $X_1, \ldots, X_B$ all be sources on $\{0,1\}^\ell$, for B a constant. Then for any small enough constant $\alpha > 0$ there exist constants $\gamma = \gamma(\alpha)$ and $\mu(\alpha) > 2\gamma$ and a polynomial time computable function $\mathsf{ACond} : \{0,1\}^\ell \to (\{0,1\}^{\ell^\mu})^{\ell^\gamma}$ such that if each X_i has min-entropy rate $\delta = \ell^{-\alpha}$, then*

  $$\mathsf{ACond}(X_1), \mathsf{ACond}(X_2), \ldots, \mathsf{ACond}(X_B)$$

  is $2^{-\Omega(\ell^{1-2\alpha})}$ close to a convex combination of sets of aligned somewhere-min-entropy rate 0.9 sources.

- The second ingredient is the technique of condensing independent SR-sources from Chapter 3. There we showed how to extract from independent sources with only a few of them being aligned SR-sources that have rows that are much longer than the number of rows. Formally, we get the following, proved in Section 3.2.

  Theorem 5.4.4. *For every constant $\gamma < 1$ there exists a polynomial time $2^{-\ell^{\Omega(1)}}$-extractor $\mathsf{SRExt} : (\{0,1\}^{\ell^{\gamma+1}})^u \to \{0,1\}^m$ for u independent sources, of which v are independent aligned $(\ell^\gamma \times \ell)$ SR-sources, where $m = \ell - \ell^{\Omega(1)}$.*

We will first describe how to use these two ingredients to extract from an intermediate model. Then we will see that total-entropy independent sources can be easily reduced to this intermediate model to prove Theorem 5.4.1.

5.4.1 Extracting From The Intermediate Model

The intermediate model we work with is defined as follows.

Definition 5.4.5. A (u, v, α) intermediate source X consists of u^2 smaller sources $X^1, \ldots, X^{u^2}$, each on $\{0,1\}^\ell$. These smaller sources are partitioned into u sets $S_1, \ldots, S_u$ such that v of the sets have the property that v of their sources have min-entropy at least $\ell^{1-\alpha}$.

Now we show that for certain constant v and $\alpha > 0$ we can extract from this model.

Theorem 5.4.6. *Assuming we can find primes with length in* $[r, 2r]$ *in time* $\mathrm{poly}(r)$, *for some constants* v *and* $\alpha > 0$ *there exists a polynomial time computable* $2^{-\ell^{\Omega(1)}}$*-extractor* IndepExt *for* (u, v, α) *intermediate sources, where* $m = \ell^{\Omega(1)}$.

Using this theorem together with Lemma 5.3.1, we can prove Theorem 5.4.1.

Proof of Theorem 5.4.1. Let $X = X_1, \ldots, X_r$ be an independent source on $(\{0,1\}^\ell)^r$ with total min-entropy rate $\delta \geq 4\ell^{-\alpha}$, where α is the constant from Theorem 5.4.6 and $n = u^2$ where u will be chosen later. Divide the source into u blocks with u smaller sources each. By Lemma 5.3.1, $\delta u/2$ of the blocks have min-entropy rate at least $\delta/2$. Now further divide each of the blocks into u sub-blocks of one smaller source each. By Lemma 5.3.1, for the blocks with min-entropy rate at least $\delta/2$, at least $\delta u/4$ of the sub-blocks have min-entropy rate $\delta/4 \geq \ell^{-\alpha}$. Let $u = 4v/\delta$, where v is the constant from Theorem 5.4.6. Then X is a (u, v, α) intermediate source satisfying the conditions of Theorem 5.4.6, which immediately gives us the theorem. ■

Here is the algorithm promised by Theorem 5.4.6:

Algorithm 5.4.7 (IndepExt$(x^1, \ldots, x^{u^2})$).

Input: $x^1, \ldots, x^{u^2}$ partitioned into sets $S_1, \ldots, S_u$.
Output: z.

Sub-Routines and Parameters:
Let v be a constant that we will pick later.
Let BGK be as in Corollary 5.3.5 - an extractor for independent sources when $v - 1$ of the smaller sources have min-entropy.
Let ACond be as in Theorem 5.4.3, letting $B = v^2$ - a condenser that converts sources with sublinear min-entropy into a convex combination of aligned sources with somewhere-min-entropy rate 0.9.
Let SRExt be as in Theorem 5.4.4 - an extractor for independent sources that works when just v of the inputs come from aligned SR-sources.
Set $\epsilon = 1/v^3$. Let α be a small enough constant to apply Theorem 5.4.3 with α in the hypothesis. Let γ be as in the conclusion of the theorem.

1. Compute $y^i = \mathsf{ACond}(x^i)$ for every source in the input. Let y^i_j denote the jth row of y^i.
2. For every $l \in [u]$, and every $j \in [2^{\ell^\gamma}]$, let b^l_j be the string obtained by applying BGK using the y^i_j from all $i \in S_l$ as input.

   We think of b^l as a sample from an SR-source with ℓ^γ rows, one for each seed s_i.
3. Output SRExt$(b^1, \ldots, b^u)$.

Proof of Theorem 5.4.6. If we restrict our attention to the v^2 high min-entropy smaller sources, from Theorem 5.4.3 we know that from the first step from these smaller sources is $2^{-\Omega(\ell^{1-2\alpha})}$ close to a convex combination of sets of aligned somewhere-min-entropy rate 0.9 sources.

Then in the second step, for each distribution in the convex combination BGK succeeds in extracting from the aligned min-entropy rate 0.9 row in each set.

Remark 5.4.8. Actually, we don't really need the Bourgain-Glibichuk-Konyagin extractor for this step. If the min-entropy is so high, it is easy to see that the generalized inner product function is an extractor. Still we use BGK since we will need it later on in the construction.

Thus the result of the first step in the algorithm is a distribution that is $2^{-\ell^{\Omega(1)}}$-close to a convex combination of collections of u independent smaller sources, v of which are independent aligned SR-sources.

Our extractor SRExt then extracts from each distribution in the convex combination, and thus extracts from the entire convex combination. So our algorithm succeeds in extracting from the input. ■

5.4.2 Condensing To Aligned Sources With High Somewhere-Min-Entropy

In this section we give the condenser from Theorem 5.4.3. The first ingredient we'll need is the following condenser from [Zuc06], which improves on the condenser from [BKS⁺05].

Lemma 5.4.9 ([Zuc06])**.** *Assuming we can find primes with length in $[r, 2r]$ in time $\mathrm{poly}(r)$, there exists a constant $\alpha > 0$ such that for any $t, \ell > 0$ there exists a polynomial-time computable condenser $\mathsf{Zuck} : \{0,1\}^\ell \to (\{0,1\}^{(2/3)^t \ell})^{2^t}$ such that if X has rate δ, $\mathsf{Zuck}(X)$ is $t2^{-\Omega(\alpha\delta\ell)}$ close to somewhere-min-entropy rate $\min((1+\alpha)^t\delta, 0.9)$.*

We'll also need to use the condenser from Raz's work [Raz05] with the improved analysis of Dvir and Raz (Lemma 3.2 in [DR05]), which shows that most of the output rows are statistically close to having high min-entropy.

Lemma 5.4.10 ([DR05])**.** *For any constant $c > 0$, there is a polynomial-time computable function $\mathsf{Raz} : (\{0,1\}^\ell)^r \to (\{0,1\}^{\Theta(\ell)})^{2^{\Theta(r)}}$ such that the following holds. If the input source X has somewhere-min-entropy rate δ, the output $\mathsf{Raz}(X)$ is $2^{-\Omega(\delta\ell)}$ close to a convex combination of distributions, each of which has the property that at least a $(1-c)$ fraction of its rows have min-entropy rate at least 0.9δ.*

Now we can apply the functions from the previous two lemmas in succession to transform any source with min-entropy rate δ into a convex combination of sources with high somewhere-min-entropy sources where almost all of the rows in the sources have high min-entropy.

Lemma 5.4.11. *Assuming we can find primes with length in $[r, 2r]$ in time $\mathrm{poly}(r)$, there exists a constant $\alpha > 0$ such that for any constants $t > 0$ and $c > 0$ there exists a polynomial-time computable function $\mathsf{Cond} : \{0,1\}^\ell \to (\{0,1\}^{\Theta((2/3)^t\ell)})^{2^{\Theta(2^t)}}$ such that the following holds. If the input source X has min-entropy rate δ, the output $\mathsf{Cond}(X)$ is $2^{-\Omega(\delta\ell)}$ close to a convex combination of distributions, each of which has the property that at least a $(1-c)$ fraction of its rows have min-entropy rate at least $\min(0.9\delta(1+\alpha)^t, 0.9)$.*

Proof. Let $\mathsf{Cond}(x) = \mathsf{Raz}(\mathsf{Zuck}(x))$. □

Corollary 5.4.12. *Assuming we can find primes with length in $[r, 2r]$ in time $\mathrm{poly}(r)$, there is a constant C such that for any constant $c > 0$ there exists a polynomial-time computable function $\mathsf{Cond} : \{0,1\}^\ell \to (\{0,1\}^{\Theta(\ell)})^C$ such that the following holds. If the input source X has min-entropy rate δ, then the output $\mathsf{Cond}(X)$ is $2^{-\Omega(\delta\ell)}$ close to a convex combination of distributions where each source in the convex combination has the property that at least a $(1-c)$ fraction of its rows have min-entropy rate at least* $\min(2\delta, 0.9)$.

Proof. Pick t large enough (but still constant) in Lemma 5.4.11 so that $0.9(1+\alpha)^t \geq 2$. Then $C = 2^{\Theta(2^t)}$. □

Now we can use this basic condenser to help prove Theorem 5.4.3. To do this, we apply this condenser to our input smaller sources and then recursively apply it to the outputs. We might think we could just apply the union bound to show that most of the output rows are aligned, but that is not true. However, we only need that one single row in the output is aligned, which we can accomplish by ensuring that at each step we have an aligned row, and then concentrating recursively on that aligned row.

Proof of Theorem 5.4.3. First, apply the function Cond from Corollary 5.4.12 to each X_i, choosing $c < \frac{1}{B}$. Then the output $\langle \mathsf{Cond}(X_1), \mathsf{Cond}(X_2), \ldots, \mathsf{Cond}(X_B)\rangle$ is $2^{-\Omega(\delta\ell)}$ close to a convex combination of distributions $Y = \sum_j \beta_j Y^{(j)}$, where $Y^{(j)} = \langle Y_1^{(j)}, Y_2^{(j)}, \ldots, Y_B^{(j)}\rangle$ and $\sum_j \beta_j = 1$. Each smaller source $Y_i^{(j)}$ has the property that at least a $(1-c)$ fraction of its rows have min-entropy rate at least 2δ. Now we restrict our attention to a single source in the convex combination $Y^{(j)}$. In this source at most $cB < 1$ fraction of the rows have a smaller source $Y_i^{(j)}$ with min-entropy rate less than 2δ in that row. Thus there is at least one row where the min-entropy rate for all the smaller sources is at least 2δ, i.e., the output is aligned with somewhere-min-entropy rate 2δ. Now we recursively apply Cond to each row in each output source. When we apply it to the aligned row, we'll get another aligned row with min-entropy rate 4δ. If we recursively do this t times, we end up close to a convex combination of a set of aligned sources with somewhere-min-entropy rate $2^t\delta$. If we let $t = \log(0.9/\delta) = \log(0.9\ell^\alpha)$, then these sources have somewhere-min-entropy rate 0.9. If we choose α small enough (depending on the constants in Corollary 5.4.12), then we can achieve $\mu > 2\gamma$, as desired. ■

5.5 Extractors For Total-Entropy Independent Sources With Many Short Smaller Sources

Now we show how for sources consisting of many smaller sources of length ℓ we can do better than the constructions in the previous sections by generalizing earlier constructions for symbol-fixing sources. The base extractor simply takes the sum of the smaller sources modulo p for some prime $p > 2^\ell$. Then we divide the source into blocks, apply the base extractor to each block, and then use the result to take a random walk on an expander as in [KZ03].

We will need the following definition from [KZ03].

Definition 5.5.1. An independent source on $(\{0,1\}^\ell)^r$ is a (k, ϵ)-*approximate symbol-fixing source* if k of the r smaller sources have distributions within an ℓ_2 distance ϵ of uniform.

These sources will be used as intermediate sources. We will transform the sources we wish to extract from into approximate symbol-fixing sources and then use the results of [KZ03] to extract from these sources.

5.5.1 Random Walks

Let $\lambda(P)$ be the second largest eigenvalue in absolute value of the transition matrix P for a random walk on a graph G. It is well known that the ℓ_2 distance from the uniform distribution decreases by a factor of $\lambda(P)$ for each uniform step of the random walk (see e.g. [Lov96]).

We will also need the following Lemma from [KZ03], which shows that we can use a random walk to extract from approximate symbol-fixing sources.

Lemma 5.5.2 ([KZ03])**.** *Let G be an undirected non-bipartite d-regular graph on M vertices with uniform transition matrix P. Suppose we take a walk on G for r steps, with the steps taken according to the symbols from a (k, ϵ)-approximate oblivious symbol-fixing sources on $[d]^r$. For any initial probability distribution , the variation distance from uniform at the end of the walk is at most $\frac{1}{2}(\lambda(P) + \epsilon\sqrt{d})^k\sqrt{M}$.*

Note that if $\lambda(P) + \epsilon\sqrt{d}$ is bounded above by a constant, as would happen if G were an expander and ϵ was small enough, then this immediately gives us a good extractor for approximate symbol-fixing sources. This is shown in the following proposition, which follows immediately from Lemma 5.5.2.

Proposition 5.5.3. *Let G be an undirected non-bipartite d-regular graph on 2^m vertices with uniform transition matrix P. Then we can construct a polynomial-time computable ϵ'-extractor for the set of (k, ϵ)-approximate oblivious symbol-fixing sources on $[d]^r$, where $\epsilon' = \frac{1}{2}(\lambda(P)+\epsilon\sqrt{d})^k 2^{m/2}$. This extractor simply uses the input from the source to take a random walk on G and outputs the label of the final vertex.*

5.5.2 Reducing to Flat Total-Entropy Independent Sources

It will be simpler to analyze our extractor for flat total-entropy independent sources. We show that any extractor that works for flat total-entropy independent sources also works for general total-entropy independent sources because any total-entropy independent source is close to a convex combination of flat independent sources with high total-entropy.

Lemma 5.5.4. *Any ϵ-extractor for the set of flat independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy $k/(2\log 3)$ is also an $(\epsilon + e^{-k/9})$-extractor for the set of independent sources on $(\{0,1\}^\ell)^r$ with min-entropy k.*

This lemma follows directly from the following lemma.

Lemma 5.5.5. *Any independent source $X = X_1, \dots, X_r$ on $(\{0,1\}^\ell)^r$ with total min-entropy k is $e^{-k/9}$-close to a convex combination of flat independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy $k/(2\log 3)$.*

Proof. Let $H_\infty(X_i) = k_i$ for all i. If $k_i \geq 1$, we can write X_i as a convex combination of flat sources with support size $\lfloor 2^{k_i} \rfloor$. Each of these flat sources has min-entropy $\log\lfloor 2^{k_i}\rfloor > \frac{k_i}{\log 3}$, since we lose the largest fraction of min-entropy from taking the floor when 2^{k_i} is nearly 3.

If $k_i < 1$, then we must have constant sources in our convex combination, so if we did as above, we'd lose up to a bit of entropy for each such i. Instead, suppose k' of the total entropy is contained in X_i with less than a bit of entropy each. Call this set $S \subseteq [r]$. Now suppose $k' \leq k/2$. In this case, we can write X_S as a convex combination of constant sources and we are still left with $(k-k')/\log 3 \geq k/(2\log 3)$ bits of entropy in each of our sources, as desired.

From now on we will assume $k' \geq k/2$. We will show we can write X_S as a convex combination of sources that with probability $1-\epsilon$ have min-entropy $k'/3$. For each $i \in S$, we can write X_i as a convex combination of flat sources with one or zero bits of entropy. The one bit sources are obtained by choosing uniformly between the most probable value and each of the other values for X_i. Each of these sources occurs with probability equal to twice the probability of the less probable value. Since the most probable value occurs with probability 2^{-k_i}, we get one bit of entropy with probability $2(1-2^{-k_i})$. Otherwise, X_i is fixed to the most probable value.

Now we can use a Chernoff bound to bound the entropy in the sources in the overall convex combination of sources for X_S. Let Y_i be an indicator random variable for the ith source having one bit of entropy. Then $Y = \sum Y_i$ is a random variable representing the total entropy. Note that $\mathbb{E}[Y] = \sum \mathbb{E}[Y_i] = \sum 2(1-2^{-k_i}) \geq \sum k_i = k'$, where the inequality is true because $k_i < 1$. Now we are ready to apply the Chernoff bound (Theorem A.1.13 in Alon and Spencer [AS92]).

$$\Pr[Y < (1-\lambda)k'] \leq \Pr[Y < (1-\lambda)\mathbb{E}[Y]] < e^{-\lambda^2(\sum(1-2^{-k_i}))} \leq e^{-\lambda^2\frac{k'}{2}} \leq e^{-\lambda^2\frac{k}{4}}$$

Setting $\lambda = 2/3$ we get the desired error bound $\epsilon = e^{-\frac{k}{9}}$. Then with probability $1-\epsilon$ we have at least $(k-k')/\log 3 + k'/3 \geq k/(2\log 3)$ bits of entropy, as desired. □

5.5.3 Extracting From Flat Total-Entropy Independent Sources

Now we show how to extract from flat total-entropy independent sources for small ℓ. Our initial extractor simply takes the sum modulo p of the individual sources, for some prime $p \geq 2^\ell$

Theorem 5.5.6. *Let $\ell \geq 1$ and $p \geq 2^\ell$ a prime. Then $Sum_p : (\{0,1\}^\ell)^r \to [p]$, where $Sum_p(x) = \sum_i x_i \mod p$, is an ϵ-extractor for the set of flat independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy k, where $\epsilon = \frac{1}{2}2^{-2k/p^2}\sqrt{p}$.*

Combining Theorem 5.5.6 with Lemma 5.5.4 we get an extractor for total-entropy independent sources.

Corollary 5.5.7. *Suppose $p \geq 2^\ell$ is a prime. Then Sum_p is an ϵ-extractor for the set of independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy $k \geq \Omega(p^2 \log p)$, where $\epsilon = 2^{-\Omega(k/p^2)}$.*

We will prove Theorem 5.5.6 via the following lemma, which will be useful later.

Lemma 5.5.8. *Let $\ell \geq 1$ and $p \geq 2^\ell$ a prime. Then for all sets of flat independent sources $X = X_1, \ldots, X_r$ on $(\{0,1\}^\ell)^r$ with min-entropy k, $Sum_p(x)$ has ℓ_2 distance from uniform at most $2^{-2k/p^2}$.*

It is well known that if X and Y are both distributed over a universe of size p, then $|X-Y| \leq \frac{1}{2}\sqrt{p}||X-Y||_2$. Theorem 5.5.6 then follows by combining this lemma with this relation between ℓ_2 and variation distance.

To analyze the distance from uniform of the sum modulo p, we use the following lemma that relates this distance to the additive characters of $\mathbb{Z}_p$. For $\mathbb{Z}_p$, where p is a prime, the ith additive character is defined as $\chi_j(a) \overset{def}{=} e^{\frac{2\pi i j a}{p}}$.

Lemma 5.5.9. *For any function $f : \{0,1\}^r \to \mathbb{Z}_p$ and random variable X over $\{0,1\}^r$,*

$$||f(X) - U_p||_2^2 = \frac{1}{p}\sum_{j=1}^{p-1} |\mathbb{E}[\chi_j(f(X))]|^2 < \max_{j\neq 0} |\mathbb{E}[\chi_j(f(X))]|^2,$$

where U_p denotes the uniform distribution over $\mathbb{Z}_p$.

Proof. Let $Y = f(X) - U_p$. The jth Fourier coefficient of Y is given by $\hat{Y}_j = \sum_{y=0}^{p-1} Y(y)\chi_j(y)$. By Parseval's Identity and using the fact that $\sum_{y=0}^{p-1} \chi_j(y) = 0$ when $j \neq 0$ we get

$$\begin{aligned}
||Y||_2^2 = \frac{1}{p}\sum_{j=0}^{p-1} |\hat{Y}_j|^2 &= \frac{1}{p}\sum_{j=0}^{p-1} \left|\sum_{y=0}^{p-1} Y(y)\chi_j(y)\right|^2 \\
&= \frac{1}{p}\sum_{j=0}^{p-1} \left|\sum_{y=0}^{p-1} \Pr[f(X) = y]\chi_j(y) - \frac{1}{p}\sum_{y=0}^{p-1}\chi_j(y)\right|^2 \\
&= \frac{1}{p}\sum_{j=1}^{p-1} |\mathbb{E}[\chi_j(f(X))]|^2 \\
&< \max_{j\neq 0} |\mathbb{E}[\chi_j(f(X))]|^2.
\end{aligned}$$

□

Using the previous lemma we can now prove Theorem 5.5.6.

Proof. Let $f(X) = \sum_{i=1}^r X_i$ and fix $j \neq 0$. Then $|\mathbb{E}[\chi_j(f(X))]|^2 = \prod_{i=1}^r |\mathbb{E}[\chi_j(X_i)]|^2$. Suppose X_i has min-entropy k_i, so $k = \sum_i k_i$. Then since each X_i is a flat source, X_i is uniformly distributed over $K_i = 2^{k_i}$ values. Our goal is to upper bound $|\mathbb{E}[\chi_j(X_i)]|^2$ over all possible choices of X_i. Doing so, we get

$$\begin{aligned}
|\mathbb{E}[\chi_j(X_i)]|^2 &\leq \max_{X_i:\mathbb{Z}_p\to\{0,1/K_i\},\sum_x X_i(x)=1} |\mathbb{E}[\chi_j(X_i)]|^2 \\
&= \max_{X_i:\mathbb{Z}_p\to\{0,1/K_i\},\sum_x X_i(x)=1} \left|\sum_{x\in\mathbb{Z}_p} X_i(x)\chi_j(x)\right|^2 \\
&= \max_{y,|y|=1}\left(\max_{X_i:\mathbb{Z}_p\to\{0,1/K_i\},\sum_x X_i(x)=1}\left(\left(\sum_{x\in\mathbb{Z}_p} X_i(x)\chi_j(x)\right)\odot y\right)^2\right) \\
&= \max_{X_i:\mathbb{Z}_p\to\{0,1/K_i\},\sum_x X_i(x)=1}\left(\max_{y,|y|=1}\left(\sum_{x\in\mathbb{Z}_p} X_i(x)(\chi_j(x)\odot y)\right)^2\right),
\end{aligned}$$

where $\odot$ denotes the complex dot product, where the complex numbers are viewed as two dimensional vectors, and the third line follows from the observation that the dot product is maximized when y is in the same direction as $(\sum_{x\in\mathbb{Z}_p} X_i(x)\chi_j(x))$, in which case we get exactly the square of the length. Now we further note that $\chi_j(x) \odot y$ is greatest for values of x for which $\chi_j(x)$ is closest to y. Thus we achieve the maximum when X_i is distributed over the K_i values closest to y. Without loss of generality we can assume these values correspond to $x = 0$ to $K_i - 1$ (since we only care about the magnitude). Thus

$$\begin{aligned}
|\mathbb{E}[\chi_j(X_i)]|^2 &\le \left| \frac{1}{K_i} \sum_{j=0}^{K_i-1} e^{\frac{2\pi i j}{p}} \right|^2 \\
&= \left| \frac{1}{K_i} \frac{1 - e^{\frac{2\pi i j K_i}{p}}}{1 - e^{\frac{2\pi i}{p}}} \right|^2 \\
&= \left| \frac{1}{K_i} \frac{e^{\frac{\pi i K_i}{p}}(e^{\frac{-\pi i K_i}{p}} + e^{\frac{\pi i K_i}{p}})}{e^{\frac{\pi i}{p}}(e^{\frac{-\pi i}{p}} + e^{\frac{\pi i}{p}})} \right|^2 \\
&= \left(\frac{1}{K_i} \frac{\sin(\frac{\pi K_i}{p})}{\sin(\frac{\pi}{p})} \right)^2 \\
&= \left(\frac{1}{K_i} \frac{\frac{\pi K_i}{p} \prod_{m=1}^{\infty}(1 - \frac{K_i^2}{p^2 m^2})}{\frac{\pi}{p} \prod_{m=1}^{\infty}(1 - \frac{1}{p^2 m^2})} \right)^2 \\
&= \left(\prod_{m=1}^{\infty} \left(1 - \frac{K_i^2 - 1}{p^2 m^2 - 1} \right) \right)^2 \\
&< \left(1 - \frac{K_i^2 - 1}{p^2 - 1} \right)^2 \\
&< e^{-2(K_i^2-1)(p^2-1)},
\end{aligned}$$

where in the fifth line we use the infinite product representation of sine.

So

$$\begin{aligned}
|\mathbb{E}[\chi_j(f(X))]|^2 &= \prod_{i=1}^{r} |\mathbb{E}[\chi_j(X_i)]|^2 \\
&< \prod_{i=1}^{r} e^{-2(K_i^2-1)/(p^2-1)} \\
&< e^{2r/p^2} e^{-2(\sum_i K_i^2)/p^2}.
\end{aligned}$$

By the power mean inequality, $\sum_{i=1}^{r} K_i^2 \ge r \cdot (\prod_{i=1}^{r} K_i)^{2/r} = r2^{2k/r}$. Thus

$$|\mathbb{E}[\chi_j(f(X))]|^2 < e^{-\frac{2r(2^{2k/r}-1)}{p^2}}$$

Let $k = \delta r$. Then this quantity is $e^{-(2k/p^2)((2^{2\delta}-1)/\delta)}$. Since $(2^{2\delta} - 1)/\delta$ is an increasing function of

δ and goes to $2 \ln 2$ as δ goes to 0, we have

$$|\mathbb{E}[\chi_j(f(X))]|^2 < e^{-(2k/p^2)((2^{2\delta}-1)/\delta)} < e^{-4(\ln 2)k/p^2} = 2^{-4\frac{k}{p^2}}$$

Then since by Lemma 5.5.9 $||f(X) - U_p||_2^2 < \max_{j \neq 0} |\mathbb{E}[\chi_j(f(X))]|^2$, $||f(X) - U_p||_2 < 2^{-2k/p^2}$. ■

Now we show that if we divide the source into blocks and take the sum modulo p for each block, we get a convex combination of approximate symbol-fixing sources, which we can then use an expander walk to extract from.

Lemma 5.5.10. *For any prime $p \geq 2^\ell$ and any t, any flat independent source X on $(\{0,1\}^\ell)^r$ with total min-entropy k can be transformed in polynomial-time into a $(k', 1/p)$-approximate oblivious symbol-fixing source $f(X)$ on $[p]^{r'}$, where $r' = k/(2p^2 \log p)$ and $k' = k^2/(4np^2 \log^2 p)$.*

Proof. First divide X into $\frac{k}{2t}$ blocks consisting of $\frac{2t}{k}r$ smaller sources, for $t = p^2 \log p$. Then for each block take the sum modulo p of the smaller sources in the block. Then $f(X)$ is the concatenation of the resulting symbols for each block.

By Lemma 5.3.1, the number of blocks with min-entropy at least t is greater than $\frac{k^2}{4tr\ell} > \frac{k^2}{4tr \log p}$. For each of these blocks, by Lemma 5.5.8, we mix within $2^{-t/p^2} = \frac{1}{p}$ of uniform. □

Now, as in [KZ03], we use $f(X)$ as defined above to take a random walk on an expander graph, which will mix to uniform by Lemma 5.5.2 and thus give us our extractor.

Theorem 5.5.11. *There exists an ϵ-extractor for the set of flat independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy k that outputs $m = \Omega(k^2/(r2^{2\ell}\ell))$ bits and has error $\epsilon = 2^{-m}$. This extractor is computable in time* $\mathrm{poly}(r, 2^\ell)$.

Proof. Let p be the least prime greater than 2^ℓ. Since by Bertrand's Postulate $p < 2 \cdot 2^\ell$, this can easily be done in polynomial time in 2^ℓ by exhaustive search. Given a source X, first apply $f(X)$ from Lemma 5.5.10 to get a $(k', 1/p)$-approximate oblivious symbol-fixing source on $[p]^{r'}$, where $r' = k/(2p^2 \log p)$ and $k' = k^2/(4rp^2 \log^2 p)$. Then apply the extractor from Proposition 5.5.3 to $f(X)$, taking the graph G to be a p regular expander graph on 2^m vertices (for m to be given later). Specifically, assume G has $\lambda(G) \leq \frac{1}{p^\alpha} - \frac{1}{\sqrt{p}}$ for some constant $\alpha < 1/2$. This can be achieved, for example, by taking G to be an $O(\log p)$ power of a constant degree expander with self loops added to make it degree p. Then by Proposition 5.5.3 $f(X)$ is within

$$\begin{aligned}
\epsilon &\leq \frac{1}{2}\left(\lambda(G) + \frac{1}{\sqrt{p}}\right)^{(k^2/4rp^2 \log^2 p)} 2^{m/2} \\
&< p^{-(\alpha k^2/4rp^2 \log^2 p)} 2^{m/2} \\
&= 2^{-((\alpha k^2/4rp^2 \log p) - (m/2))}
\end{aligned}$$

of uniform. Then let $m = \alpha k^2/6rp^2 \log p$ so then $\epsilon < 2^{-m}$. ■

Combining this theorem with our reduction from general to flat sources, we get that this same extractor works for general total-entropy independent sources.

Theorem 5.5.12. *There exists an ϵ-extractor for the set of independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy k that outputs $m = \Omega(k^2/r2^{2\ell}\ell)$ bits and has error $\epsilon = 2^{-m}$. This extractor is computable in time* $\mathrm{poly}(r, 2^\ell)$.

Proof. Combine Theorem 5.5.11 and Lemma 5.5.4. ■

5.6 Extracting More Bits From Total-Entropy Independent Sources

5.6.1 Seed Obtainers

Now that we have extractors for total-entropy independent sources, we can extract even more bits using the techniques that Gabizon et al. [GRS04] used to extract more bits out of oblivious bit-fixing sources. Assuming the entropy is high enough to use the extractors from Theorem 5.5.12, Theorem 5.3.6, or Corollary 5.4.2, we can extract almost all of the entropy. Their construction works by using an extractor for bit-fixing sources and a sampler to construct a seed obtainer. This seed obtainer outputs a source and a seed that is close to a convex combination of independent bit-fixing sources and uniform seeds. We generalize their definition of seed obtainer to total-entropy independent sources.

Definition 5.6.1. A function $F : (\{0,1\}^\ell)^r \to (\{0,1\}^\ell)^r \times \{0,1\}^d$ is a (k', ρ)-seed obtainer for all independent sources X on $(\{0,1\}^\ell)^r$ with total min-entropy k if the distribution $R = F(X)$ can be expressed as a convex combination of distributions $R = \eta Q + \sum_a \alpha_a R_a$ (where the coefficients η and α_a are nonnegative and $\eta + \sum_a \alpha_a = 1$) such that $\eta \leq \rho$ and for every a there exists an independent source Z_a on $(\{0,1\}^\ell)^r$ with min-entropy k' such that R_a is ρ-close to $Z_a \otimes U_d$.

Now, as in the bit-fixing case, we can use a seeded extractor for total-entropy independent sources together with a seed obtainer to construct a deterministic extractor for total-entropy independent sources. The proof for the following Theorem is the same as the proof for the bit-fixing case in [GRS04]. We include it here for the sake of completeness.

Theorem 5.6.2. *Let $F : (\{0,1\}^\ell)^r \to (\{0,1\}^\ell)^r \times \{0,1\}^t$ be a (k', ρ)-seed obtainer for independent sources X on $(\{0,1\}^\ell)^r$ with total min-entropy k. Let $E_1 : (\{0,1\}^\ell)^r \times \{0,1\}^d \to \{0,1\}^m$ be a seeded ϵ-extractor for independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy k. Then $E : (\{0,1\}^\ell)^r \to \{0,1\}^m$ defined by: $E(x) \overset{def}{=} E_1(F(x))$ is a deterministic $(\epsilon + 2\rho)$-extractor for independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy k.*

Proof. By the definition of a seed obtainer we have that $E(X) = \eta E_1(Q) + \sum_a \alpha_a E_1(R_a)$ for some $\eta \leq \rho$. For each a we have that R_a is ρ-close to $Z_a \otimes U_d$, so $E_1(R_a)$ is ρ-close to $E_1(Z_a \otimes U_d)$, which is itself ϵ-close to U_m since E_1 is an ϵ-extractor. Thus $E_1(R_a)$ is $(\epsilon + \rho)$-close to U_m, which implies that $E(X)$ is $(\epsilon + \rho)$-close to $\eta E_1(Q) + (1 - \eta) U_m$. Therefore by Lemma 2.1.8 we have that $E(X)$ is $(\eta + \epsilon + \rho)$-close to uniform. The lemma follows because $\eta \leq \rho$. ■

To construct seed obtainers, we need to extend the definition of averaging samplers from [GRS04] to general functions as follows. This definition is similar in spirit to that of Vadhan in [Vad04], except the sample size is not fixed and we both upper and lower bound the total value of the sample.

Definition 5.6.3. A function $Samp : \{0,1\}^t \to P([r])$ is a $(\delta, \theta_1, \theta_2, \gamma)$ averaging sampler if for every function $f : [r] \to [0,1]$ with average value $\frac{1}{r}\sum_i f(i) = \delta$, it holds that

$$\Pr_{w \leftarrow U_t}\left[\theta_1 \leq \sum_{i \in Samp(w)} f(i) \leq \theta_2\right] \geq 1 - \gamma.$$

When applying these samplers to total-entropy independent sources, we get the following lemma.

Lemma 5.6.4. *Let $Samp : \{0,1\}^t \to P([r])$ be a $(\delta, \delta_1 r, \delta_2 r, \gamma)$ averaging sampler. Then for any independent source X on $(\{0,1\}^\ell)^r$ with total min-entropy $k = \delta r \ell$, we have*

$$\Pr_{w \leftarrow U_t}[\delta_1 r\ell \leq H_\infty(X_{Samp(w)}) \leq \delta_2 r\ell] \geq 1 - \gamma.$$

Proof. Let $f(i) = H_\infty(X_i)/\ell$. □

Given these definitions, we can show that essentially the same construction from Gabizon et al. [GRS04] for bit-fixing seed obtainers works for total-entropy independent source seed obtainers.

Theorem 5.6.5. *Let $Samp : \{0,1\}^t \to P([r])$ be a $(\delta, \delta_1 r, \delta_2 r, \gamma)$ averaging sampler and $E : (\{0,1\}^\ell)^r \to \{0,1\}^m$ be an ϵ-extractor for independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy $k = \delta_1 r\ell$. Then $F : (\{0,1\}^\ell)^r \to (\{0,1\}^\ell)^r \times \{0,1\}^{m-t}$ defined as follows is a (k', ρ)-seed obtainer for independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy $k = \delta r\ell$ with $k' = (\delta - \delta_2)r\ell$ and $\rho = \max(\epsilon + \gamma, \epsilon \cdot 2^{t+1})$.*

The Construction of F:

- *Given $x \in (\{0,1\}^\ell)^r$ compute $z = E(x)$. Let $E_1(x)$ denote the first t bits of $E(x)$ and $E_2(x)$ denote the remaining $m - t$ bits.*
- *Let $T = Samp(E_1(x))$.*
- *Let $x' = x_{[r]\setminus T}$. If $|x'| < n$ we pad it with zeroes to get an r source long string.*
- *Let $y = E_2(x)$. Output x', y.*

The proof of this theorem is almost exactly the same as the proof in [GRS04], except substituting independent sources and the associated sampler and extractor for bit-fixing sources, so we omit it here. This theorem also follows from the main theorem of [Sha06].

5.6.2 Constructing Samplers

In order to use the seed obtainer construction to extract more bits, we first need a good averaging sampler. We will show that the same sampler construction given in Gabizon et al. [GRS04] generalizes to our definition. Our sampler works by generating d-wise independent variables $Z_1, \ldots, Z_r \in [b]$ and letting $Samp(U_t) = \{i | Z_i = 1\}$.

Lemma 5.6.6. *For all δ and integers r, b, t such that $b/r \leq \delta \leq 1$ and $6 \log r \leq t \leq \frac{\delta r \log r}{20b}$ there is a polynomial-time computable $(\delta, \frac{\delta r}{2b}, \frac{3\delta r}{b}, 2^{-\Omega(t/\log r)})$ averaging sampler $Samp : \{0,1\}^t \to P([r])$*

The following tail inequality for d-wise independent variables is due to Bellare and Rompel [BR94].

Theorem 5.6.7. *[BR94] Let $d \geq 6$ be an even integer. Suppose that $X_1, \ldots, X_r$ are d-wise independent random variables taking values in $[0,1]$. Let $Y = \sum_{1 \leq i \leq r} Y_i$, $\mu = \mathbb{E}[Y]$, and $A > 0$. Then*

$$\Pr[|Y - \mu| \geq A] \leq 8 \left(\frac{d\mu + d^2}{A^2}\right)^{d/2}$$

Proof. (of Lemma 5.6.6) Let d be the largest even integer such that $d \log r \leq t$ and let $q = \lfloor \log b \rfloor \leq \log r$. Use $d \log r$ random bits to generate r d-wise independent random variables $Z_1, \ldots, Z_r \in \{0,1\}^q$ using the construction from [CW79]. Fix $a \in \{0,1\}^q$. Let the random variable denoting the output of the sampler be $Samp(U_t) = \{i | Z_i = a\}$. For $1 \leq i \leq r$, define a random variable Y_i that is set to $f(i)$ if $i \in Samp(U_t)$ and 0 otherwise. Let $Y = \sum_i Y_i$ (note that Y is exactly the sum we wish to bound). Note that $\mu = \mathbb{E}[Y] = \delta r / 2^q$ and that the random variables $Y_1, \ldots, Y_r$ are d-wise independent. Applying Theorem 5.6.7 with $A = \delta r / 2b$,

$$\Pr[|Y - \mu| \geq A] \leq 8 \left(\frac{d \frac{\delta r}{2^q} + d^2}{A^2} \right)^{d/2}.$$

Note that

$$\begin{aligned} \{|Y - \mu| < A\} &\subseteq \{\frac{\delta r}{2^q} - A < Y < \frac{\delta r}{2^q} + A\} \subseteq \{\frac{\delta r}{b} - A < Y < \frac{2\delta r}{b} + A\} \\ &\subseteq \{\frac{\delta r}{2b} \leq Y \leq \frac{3\delta r}{b}\} = \{\frac{\delta r}{2b} \leq \sum_{i \in Samp(w)} f(i) \leq \frac{3\delta n}{b}\}. \end{aligned}$$

Note that $d \leq \frac{t}{\log r} \leq \frac{\delta r}{20b}$ by assumption. We conclude that

$$\begin{aligned} \Pr_{w \leftarrow U_t}\left[\frac{\delta r}{2b} \leq \sum_{i \in Samp(w)} f(i) \leq \frac{3\delta r}{b} \right] &\geq 1 - 8 \left(\frac{d \frac{\delta r}{2^q} + d^2}{(\delta r / 2b)^2} \right)^{d/2} \geq 1 - 8 \left(\frac{4b^2}{(\delta r)^2} \left(\frac{2d\delta r}{b} + \frac{d\delta r}{20b} \right) \right)^{d/2} \\ &\geq 1 - 8 \left(\frac{10db}{\delta r} \right)^{d/2} \geq 1 - 2^{-(d/2+3)} \geq 1 - 2^{-\Omega(t/\log r)} \end{aligned}$$

■

5.6.3 Extractors From Seed Obtainers

As in [GRS04] it will be convenient to combine Theorem 5.6.2 and Theorem 5.6.5 to get the following theorem.

Theorem 5.6.8. *Assume we have the following:*

- *A $(\delta, \delta_1 r, \delta_2 r, \gamma)$ averaging sampler $Samp : \{0,1\}^t \to P([r])$.*
- *A deterministic ϵ^*-extractor for total-rate δ_1 independent sources $E^* : (\{0,1\}^\ell)^r \to \{0,1\}^{m'}$.*
- *A seeded ϵ_1-extractor for total-rate $\delta - \delta_2$ independent sources $E_1 : (\{0,1\}^\ell)^r \times \{0,1\}^s \to \{0,1\}^m$, where $m' \geq s + t$.*

Then we get a deterministic ϵ-extractor for total-rate δ independent sources $E : (\{0,1\}^\ell)^r \to \{0,1\}^m$ where $\epsilon = \epsilon_1 + 3 \cdot \max(\epsilon^* + \gamma, \epsilon^* \cdot 2^{t+1})$.

We will use the following seeded extractor from Raz, Reingold, and Vadhan [RRV02].

Theorem 5.6.9. *[RRV02] For any r,k, and $\epsilon > 0$, there exists a ϵ-extractor $Ext : \{0,1\}^r \times \{0,1\}^s \to \{0,1\}^m$ for all sources with min-entropy k, where $m = k$ and $s = \Theta(\log^2 r \cdot \log(1/\epsilon) \cdot \log m)$.*

Combining the extractor from [RRV02] with the sampler from the previous section, we get the following general corollary, which shows how to transform a deterministic extractor that extracts just some of the min-entropy into one that extracts almost all of the min-entropy.

Corollary 5.6.10. *Let $\delta, \delta_1, \epsilon_1$ and integers r, t be such that $\delta_1 \geq 1/2r$ and $6 \log r \leq t \leq \frac{\delta_1 r \log r}{10}$. Also let $m = (\delta - 6\delta_1)r\ell$ and $s = \Theta(\log^2(r\ell) \cdot \log(1/\epsilon_1) \cdot \log m)$. Then given any deterministic ϵ^*-extractor for total-rate δ_1 independent sources $E^* : (\{0,1\}^\ell)^r \to \{0,1\}^{m'}$ with $m' \geq s + t$, we can construct an ϵ-extractor for total-rate δ independent sources $E : (\{0,1\}^\ell)^r \to \{0,1\}^m$ where $\epsilon = \epsilon_1 + 3 \cdot \max(\epsilon^* + 2^{-\Omega(t/\log r)}, \epsilon^* \cdot 2^{t+1})$.*

Proof. Combine Lemma 5.6.6 with $b = \delta/2\delta_1$, Theorem 5.6.9, and Theorem 5.6.8. □

Now we can use Corollary 5.6.10 together with our previous deterministic extractor construction from Theorem 5.5.12 to show how we can extract nearly all of the entropy from total-entropy independent sources with sufficiently high min-entropy, proving Theorem 5.0.15.

Proof of Theorem 5.0.15. Use the construction from Corollary 5.6.10 with the extractor from Theorem 5.5.12 as E^* and let $\epsilon_1 = 2^{-\Omega((\delta_1^2 r\ell)(2^{2\ell}\log^3 r))}$ and $t = \Omega(\frac{\delta_1^2}{2^{2\ell}} r\ell)$. Then it's not hard to see that (choosing appropriate constants) these values satisfy $6 \log r \leq t \leq \frac{\delta_1 r \log r}{10}$ and $m' \geq s + t$ for sufficiently large r. ■

The extractor for small-space sources from Theorem 5.0.12 is then obtained by combining Theorem 5.0.15 with Lemma 5.2.1.

We could also use a seed obtainer together with the extractor for constant rate sources from Theorem 5.3.6. This lets us extract any constant fraction of the entropy and proves Theorem 5.0.14.

Proof of Theorem 5.0.14. Use the construction from Corollary 5.6.10 with the extractor from Theorem 5.3.6 as E^* and let $\epsilon_1 = 2^{-\Omega((r\ell)/(\log^3(r\ell)))}$ and $t = \Theta(r \log(\min(2^\ell, r)))$. Then it's not hard to see that (choosing appropriate constants) these values satisfy $6 \log r \leq t \leq \frac{\delta_1 r \log r}{10}$ and $m' \geq s + t$ for sufficiently large r. ■

The extractor for small-space sources from Theorem 5.0.11 is then obtained by combining Theorem 5.0.15 with Lemma 5.2.1.

We can also apply this construction to the polynomial entropy rate extractor from Corollary 5.4.2, which proves Theorem 5.0.13.

Proof of Theorem 5.0.13. Use the construction from Corollary 5.6.10 with the extractor from Corollary 5.4.2 as E^* and let $\epsilon_1 = 2^{-(\delta_1^2 r\ell)^{\Omega(1)}/(\log^3(r\ell))}$ and $t = (\delta_1^2 r\ell)^{\Omega(1)}$. Then it's not hard to see that (choosing appropriate constants) these values satisfy $6 \log r \leq t \leq \frac{\delta_1 r \log r}{10}$ and $m' \geq s + t$ for sufficiently large r. ■

The extractor for small-space sources from Theorem 5.0.9 is then obtained by combining Theorem 5.0.13 with Lemma 5.2.1.

5.6.4 Extractors For Smaller Entropy

Gabizon et. al [GRS04] also showed how to use seed obtainers to extract more bits even when the initial extractor only extracts $\Theta(\log k)$ bits, which they're able to get from the cycle walk extractor from [KZ03]. We can generalize their construction to work for total-entropy independent sources, which together with our generalization of the cycle walk extractor allows us to extract more bits from smaller entropy rates.

In order to get a seed obtainer that can use only $\Theta(\log k)$ bits, we need both a sampler and a seeded extractor for total-entropy independent sources. To do so, as in [GRS04], we use d-wise ϵ-dependent random variables to both sample and partition. The proofs of the following two lemmas easily generalize the construction from [GRS04] in a similar way to our earlier sampler construction.

Lemma 5.6.11. *For any constant $0 < \alpha < 1$, there exist constants $c > 0$ and $0 < b < 1/2$ (both depending on α) such that for any $r \geq 16$ and $k = \delta r \ell \geq \log^c r$, the following holds. There is a polynomial-time computable $(\delta, \delta r/2k^b, 3\delta r/k^b, O(k^{-b}))$ sampler Samp : $\{0,1\}^t \to P([r])$ where $t = \alpha \cdot \log k$.*

Lemma 5.6.12. *Fix any constant $0 < \alpha < 1$. There exist constants $c > 0$ and $0 < b < 1/2$ (both depending on α) such that for any $r \geq 16$ and $k = \delta r \ell \geq \log^c r$, we can use $\alpha \cdot \log k$ random bits to explicitly partition $[r]$ into $m = \Theta(k^b)$ sets $T_1, \ldots, T_m$ such that for every function $f : [r] \to [0,1]$ with average value $\frac{1}{r}\sum_i f(i) = \delta$,*

$$\Pr\left[\forall i, \delta r/2k^b \leq \sum_{j \in T_i} f(j) \leq 3\delta r/k^b\right] \geq 1 - O(k^{-b}).$$

As in Lemma 5.6.6, this lemma implies that if we partition a total-rate δ independent source, with high probability each T_i has some min-entropy.

Corollary 5.6.13. *For any constant $0 < \alpha < 1$, there exist constants $c > 0$ and $0 < b < 1/2$ (both depending on α) such that for any $r \geq 16$ and $k \geq \log^c r$, the following holds. We can use $\alpha \cdot \log k$ random bits to explicitly partition $[r]$ into $m = \Theta(k^b)$ sets $T_1, \ldots, T_m$ such that for any independent sources X on $(\{0,1\}^\ell)^r$ with total min-entropy k,*

$$\Pr\left[\forall i, k^{1-b}/2 \leq H_\infty(X_{T_i}) \leq 3k^{1-b}\right] \geq 1 - O(k^{-b}).$$

Now we will use this partitioning to construct a seeded extractor for total-entropy independent sources that uses a small seed. As in [GRS04] once we partition the source, we apply an extractor to each part. The extractor we will use is our sum mod p extractor.

Theorem 5.6.14. *For any constant $0 < \alpha < 1$, there exist constants $c > 0$ and $0 < b < 1/2$ (both depending on α) such that for any $r \geq 16$, $k \geq \log^c r$, $0 < \delta \leq 1$ and $\ell \leq \log(k^{(1-b)/2}/\sqrt{\log k^{2b}})$, the following holds. There is a polynomial-time computable seeded ϵ-extractor $E : (\{0,1\}^\ell)^r \times \{0,1\}^s \to \{0,1\}^m$ for independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy $k = \delta r \ell$, with $s = \alpha \cdot \log k$, $m = \Theta(k^b \ell)$ and $\epsilon = O(k^{-b})$.*

Proof. As stated above, E works by first partitioning the input x into $m' = \Theta(k^b)$ parts $T_1, \ldots, T_{m'}$ using Corollary 5.6.13. Next we find the next largest prime $p \geq 2^\ell$, which by Bertrand's postulate is at most $2 \cdot 2^{\ell}$, so we can find it efficiently by brute force search. Then for each T_i we compute $z_i = \sum_{j \in T_i} x_j \mod p$ and output $z = z_i, \ldots, z_{m]}$.

Let Z be the distribution of the output string z. Let A be the "good" event that all sets T_i have entropy at least $k^{1-b}/2$. Then we decompose Z as

$$Z = \Pr[A^c] \cdot (Z|A^c) + \Pr[A] \cdot (Z|A).$$

Now by Corollary 5.6.13, $\Pr[A] \geq 1 - O(k^{-b})$. By Corollary 5.5.7, $(Z|A)$ is $m' \cdot 2^{-\Omega(k^{1-b}/2^{2\ell})}$ close to uniform. Since $\ell \leq \log(k^{(1-b)/2}/\sqrt{\log k^{2b}})$, $(Z|A)$ is $O(k^{-b})$ close to uniform. Thus by Lemma 2.1.8, Z is $O(k^{-b})$ close to uniform. ■

Now we are ready to combine these ingredients using Theorem 5.6.8 to get an improved extractor.

Theorem 5.6.15. *There exist constants $c > 0$ and $0 < b < 1/2$ such that for $k \geq \log^c r$ and $2^{\ell} \leq O(k^{(1-b)/2}/\sqrt{\log k^{2b}})$, the following holds. There exists a polynomial-time computable ϵ-extractor $E : (\{0,1\}^{\ell})^r \to \{0,1\}^m$ for independent sources on $(\{0,1\}^{\ell})^r$ with min-entropy k, where $m = \Theta(k^b \ell)$ and $\epsilon = O(k^{-b})$.*

Proof. Use Theorem 5.6.8 together with the sampler from Lemma 5.6.11, the deterministic extractor from Corollary 5.5.7, and the seeded extractor from Theorem 5.6.14 ■

This still doesn't get all of the entropy out of the source, but now we have a long enough output that we can use the seeded extractor from Theorem 5.6.9 to get the rest of the entropy, which proves Theorem 5.0.16.

Proof of Theorem 5.0.16. Use Theorem 5.6.8 together with the sampler from Lemma 5.6.11, the deterministic extractor from Theorem 5.6.15, and the seeded extractor from Theorem 5.6.9. ■

5.7 Extractors For Small Space Sources via The Probabilistic Method

In this section, we describe nonconstructive results for both small-space and total-entropy independent sources. We will use Theorem 2.5.1.

5.7.1 Small-Space Sources

Since the probabilities on the edges in small-space sources can be any real number in $[0, 1]$, there are an infinite number of such sources, and so we cannot directly apply Theorem 2.5.1. We instead introduce a more restricted model to which we can apply Theorem 2.5.1, and show that general small-space sources are close to convex combinations of this more restricted model. The more restricted model we consider restricts all probabilities to be a multiple of some α.

Definition 5.7.1. An α-approximate space s source is a space s source where the probabilities on all edges are multiples of α.

We'll show that any rate δ small-space source is a convex combination of α-approximate small-space sources, each of which is close to the original source. Thus any extractor that works on α-approximate sources that are close to having rate δ will also be an extractor for rate δ small-space sources.

Lemma 5.7.2. *Let X be a space s source on $\{0,1\}^n$ with min-entropy rate δ. The source X is a convex combination of α-approximate space s sources, each of which has distance at most $\alpha n 2^s$ to X.*

Proof. We can write X as a convex combination of sources X_a such that each X_a is obtained from X by replacing each edge probability p with either $\lfloor \frac{p}{\alpha} \rfloor \alpha$ or $(\lfloor \frac{p}{\alpha} \rfloor + 1)\alpha$.

We will show that X_a is close to X via a hybrid argument. Let X_a^i be the hybrid obtained by the first i bits having probabilities from X_a and the rest of the bits having probabilities from X. So $X = X_a^0$ and $X_a = X_a^n$. Then $|X - X_a| = |\sum_{i=1}^n (X_a^{i-1} - X_a^i)| \leq \sum_{i=1}^n |X_a^{i-1} - X_a^i|$.

For each term $|X_a^{i-1} - X_a^i|$ the only difference is in the probabilities on the edges in the ith layer, which each differ by at most α. We fix i and calculate this distance. Let $v_{i,j}$ denote the jth vertex in the ith layer. Let $q_{i-1,j}$ denote the probability of reaching $v_{i-1,j}$ in X_a and $p^0_{j,j'}$ $(p^1_{j,j'})$ denote the probability on the 0 (1) edge from $v_{i-1,j}$ to $v_{i,j'}$ in X. Then

$$|X_a^{i-1} - X_a^i| \leq \frac{1}{2} \sum_{j,j'} q_{i-1,j}((p^0_{j,j'} + \alpha - p^1_{j,j'}) + (p^1_{j,j'} + \alpha - p^1_{j,j'})) \leq \alpha \sum_{j'} \sum_{j} q_{i-1,j} = \alpha \sum_{j'} 1 = \alpha 2^s.$$

So the overall error is bounded by $|X - X_a| \leq \sum_{i=1}^n \alpha 2^s = \alpha n 2^s$. □

Lemma 5.7.3. *The number of α-approximate space s sources on $\{0,1\}^n$ is less than $2^{(s+1)2^s n/\alpha}$.*

Proof. First count the number of possible edge configurations from any given vertex. There are 2^{s+1} possible edges, since there is a 0 edge and a 1 edge for each of the 2^s vertices in the next layer. Since all probabilities are multiples of α, there are less than $2^{(s+1)/\alpha}$ ways to allocate probabilities to these edges. Since there are n layers and 2^s vertices at each layer, the total number of possible sources is $2^{(s+1)2^s n/\alpha}$. □

Now we invoke Theorem 2.5.1 to show that a random function is a good extractor for small-space sources.

Theorem 5.7.4. *For space s sources with min-entropy k, a function $f : \{0,1\}^n \to \{0,1\}^m$ chosen uniformly at random is an ϵ-extractor with output length $m = k - 2\log(1/\epsilon) - O(1)$ with probability at least $1 - 1/2^{2^m} 2^{(s+1)n^2 2^{2s+1}/\epsilon}$, as long as $k \geq 2s + 1 + \log(s+1) + 2\log n + 3\log(1/\epsilon) + O(1)$.*

This theorem says that extractors exist for sources with space almost as large as $k/2$ and with min-entropy as low as $\Theta(\log n)$.

Proof. First apply Lemma 5.7.2 with $\alpha = \epsilon/n2^{s+1}$ to show that the each small-space source X is a convex combination of α-approximate sources that are $\epsilon/2$ close to X. Then apply Theorem 2.5.1 to the set of α-approximate sources that are $\epsilon/2$ close to having min-entropy k, using Lemma 5.7.3 as the bound on the number of such sources (since this set is a subset of all α-approximate space s sources). Since each min-entropy k space s source is a convex combination of these α-approximate sources, the extractors given by Theorem 2.5.1 also work with these sources. ■

5.7.2 Total-Entropy Independent Sources

We can also apply Theorem 2.5.1 to total-entropy independent sources. Similarly to the small-space case, we define an intermediate model to reduce the number of sources.

Definition 5.7.5. An α-approximate independent source $X_1, \ldots, X_r$ on $(\{0,1\}^\ell)^r$ is an independent source such that $\forall y \in \{0,1\}^\ell$ and $\forall i$, $\Pr[X_i = y]$ is a multiple of α.

We use this model rather than flat independent sources because as we saw in Lemma 5.5.5, we can lose a constant fraction of the min-entropy when viewing an independent source as a convex combination of flat independent sources.

This lemma allows us to restrict our attention to α-approximate independent sources. We'll show that any total-rate δ independent-symbol source is a convex combination of α-approximate independent sources, each of which is close to the original source.

Lemma 5.7.6. *Let $X = X_1, \ldots, X_r$ be an total-rate δ independent source on $(\{0,1\}^\ell)^r$. The source X is a convex combination of α-approximate independent sources, each of which has distance at most $\frac{1}{2}\alpha r 2^\ell$ to X.*

Proof. We can write X as a convex combination of sources $X' = X'_1, \ldots, X'_r$ such that each X'_i is obtained from X_i by replacing each output probability $\Pr[X_i = y]$ with either $\lfloor \frac{p}{\alpha} \rfloor \alpha$ or $(\lfloor \frac{p}{\alpha} \rfloor + 1)\alpha$.

Now the distance

$$|X' - X| = \sum_{i=1}^{r} |X'_i - X_i| = \frac{1}{2} \sum_{i=1}^{r} \sum_{x \in (\{0,1\}^\ell)} |\Pr[X'_i = x] - \Pr[X_i = x]|$$
$$\leq \frac{1}{2} \sum_{i=1}^{r} \alpha 2^\ell = \frac{1}{2}\alpha r 2^\ell,$$

where the first inequality is because each string $x \in \{0,1\}^\ell$ contributes at most α error for each X_i. □

Lemma 5.7.7. *The number of α-approximate independent sources on $(\{0,1\}^\ell)^r$ is less than $2^{\frac{1}{\alpha} r \ell}$.*

Proof. Let $X = X_1, \ldots, X_r$ be an α-approximate total-rate δ independent source on $(\{0,1\}^\ell)^r$. Since there are 2^ℓ possible values for each X_i, each of which has a probability that is a multiple of α, there are less then $2^{\frac{\ell}{\alpha}}$ possible distributions for X_i. Thus there are less than $(2^{\frac{\ell}{\alpha}})^r = 2^{\frac{1}{\alpha} r \ell}$ possible distributions for X. □

Now we can apply Theorem 2.5.1 to show that a random function is a good extractor for total-rate δ independent sources.

Theorem 5.7.8. *For total-entropy k independent sources, a function $f : (\{0,1\}^\ell)^r \to \{0,1\}^m$ chosen uniformly at random is an ϵ-extractor with output length $m = k - 2\log(1/\epsilon) - O(1)$ with probability $1 - 1/2^{2^m} 2^{r^2 \ell 2^\ell / \epsilon}$ as long as $k \geq \ell + \log \ell + 2\log r + 3\log(1/\epsilon) + O(1)$.*

Note that the $k > \ell$ is necessary because otherwise all of the entropy could be contained within a single source, which we know is impossible to extract from. Thus, the bound in this theorem is close to the best we could hope for.

Proof. First apply Lemma 5.7.6 with $\alpha = \epsilon / r 2^\ell$ to show that the each total-entropy k independent source X is a convex combination of α-approximate total-entropy k independent sources that are $\epsilon/2$ close to X. Then apply Theorem 2.5.1 to the set of α-approximate total-entropy k independent sources that are $\epsilon/2$ close to having min-entropy k, using Lemma 5.7.7 as the bound on the number of such sources (since this set is a subset of all α-approximate independent sources). Since each total-entropy k independent source is a convex combination of these α-approximate sources, the extractors given by Theorem 2.5.1 also work with these sources. ■

5.8 Doing Better For Width Two

We consider the case of space 1 (width 2) sources where the output bit is restricted to be the same as the label of the next state, which we will call *restricted width two sources*. For such sources, we can improve our results by decreasing the alphabet size in the total-entropy independent sources. This will allow us to extract from smaller entropy rates. We will need the following class of sources.

Definition 5.8.1. A previous-bit source on $\{0,1\}^n$ with min-entropy k has at least k uniformly random bits and the rest of the bits are functions of the previous bit.

We will show that restricted width two sources are close to a convex combination of previous-bit sources, and then show that these previous bit sources can be converted into total-entropy independent sources with small alphabet size.

5.8.1 Extracting From Previous-Bit Sources

To convert a previous-bit source to a total-entropy independent source, we first divide the source into blocks as before, but instead of simply viewing each block as a binary number, we apply a function to reduce the alphabet size while still maintaining some of the entropy. Specifically, we will show that if a block has at least one random bit, then the output symbol will have at least one bit of entropy. The main lemma is as follows.

Lemma 5.8.2. *Any length n previous-bit source X with min-entropy k can be converted in polynomial time to a convex combination of flat independent sources on $(\{0,1\}^\ell)^r$ with min-entropy k', where $r = \frac{k}{2}$, $k' = k^2/4n$ and $\ell = \lceil \log\left(\frac{2n}{k}+1\right)\rceil$.*

The following lemma shows that any block that contains at least one random bit will give a random source.

Lemma 5.8.3. *We can construct a function $f : \{0,1\}^t \to \{0,1\}^{\lceil \log(t+1)\rceil}$ so that for any previous-bit source Y on $\{0,1\}^t$ with exactly one random bit, f attains different values depending on whether the random bit in Y is set to 0 or 1.*

Proof. For $0 \le i \le t$, let $z_i \in \mathbb{Z}_2^{\lceil \log(t+1)\rceil}$ be the standard representation of i as a vector over $\mathbb{Z}_2$. (More generally, we only require the z_i to be distinct vectors.) Then $f(y) = \sum_{i=1}^t y_i(z_i - z_{i-1})$.

Let y_0 (y_1) be Y with the random bit set to 0 (1). Now we show that $f(y_0) \neq f(y_1)$. We see that

$$f(y_0) - f(y_1) = \sum_{i=1}^{t} (y_{0i} - y_{1i})(z_i - z_{i-1}).$$

It's easy to see that $y_{0i} - y_{1i}$ will be 0 for all fixed bits and 1 whenever the random bit or its negation appears. For our sources, all appearances of the random bit must appear consecutively. This means that if the random bit appears from positions j through k, $f(y_0) - f(y_1) = z_k - z_{j-1}$, since all of the other terms cancel. Thus since $z_k \neq z_{j-1}$, $f(y_0) - f(y_1) \neq 0$. □

Now we can prove Lemma 5.8.2.

Proof. Divide X into $r = k/2$ blocks of size $n/r = 2n/k$. Then apply the function f from Lemma 5.8.3 to each block to get Y.

To see that this works, fix all of the random bits that cross between blocks. Also, for each block fix all but one of the random bits that are contained within the block. Now X is a convex combination of all of the sources given by every possible such fixing. Let X' be a source corresponding to one particular fixing. We will show that if we apply f to every block of X', we will get a source with enough random blocks. Any block of X' with a random source is a previous-bit source with one random bit, so we can apply Lemma 5.8.3 to see that the output of f on this block is uniformly chosen from among two different sources, as desired.

Now we just need to see how many blocks with at least one random bit there are. There can be at most r random bits that cross between blocks. So removing those bits we are left with at least $k - r = k/2$ random bits. These $k/2$ random bits must be contained in at least $k' = (k/2)/(n/r) = k^2/4n$ different blocks, which gives us the desired bound. □

Now we can combine Theorem 5.0.15 and Lemma 5.8.2 to get an extractor for previous-bit sources.

Theorem 5.8.4. *There exists a polynomial-time computable ϵ-extractor for the set of previous-bit sources of length n with min-entropy k that outputs $m = \frac{k^2}{8n}$ bits and has error $\epsilon = \exp(-\Omega(k^5/(n^4 \log(n/k) \log^3 k)))$.*

Proof. Given a source X, apply Lemma 5.8.2 to convert X into a convex combination of flat independent sources on $(\{0,1\}^\ell)^r$ with total min-entropy k', where $r = \frac{k}{2}$, $k' = \frac{k^2}{4n}$, and $\ell' = \lceil \log\left(\frac{2n}{k} + 1\right) \rceil$. Then apply the extractor from Theorem 5.0.15 with $\zeta = k^2/(48n \cdot r\ell)$. ■

5.8.2 Restricted Width Two Sources vs Previous-Bit Sources

To show we can extract from restricted width two sources, we will prove that these sources can be viewed as convex combinations of previous bit sources. With high probability, these previous-bit sources will have sufficient entropy so that our extractor from the previous section will work.

Lemma 5.8.5. *Any length n restricted width two source X with min-entropy k is a convex combination of length n previous bit sources Z_j so that with probability at least $1 - 2^{-k/4} - e^{-9k'^2/2n}$, the sources Z_j have at least $k' = \min(k/48 \log(n/k), k/96)$ random bits.*

To get our extractor, we just combine this lemma with the extractor from Theorem 5.8.4.

Theorem 5.8.6. *There exists a polynomial-time computable ϵ-extractor for the set of length n restricted width two sources with min-entropy k that outputs $m = \Omega(k^2/n(\max(\log(n/k), 1))^2)$ bits and has error $\epsilon = 2^{-\Omega((k')^5/(n^4 \log(n/k') \log^3 k')}$, where $k' = \min(k/48 \log(n/k), k/96)$.*

Proof. By Lemma 5.8.5 our source X is $2^{-k/4} + e^{-9k'^2/2n}$ close to a convex combination of length n previous-bit sources with $k' = \min(k/48 \log(n/k), k/96)$ random bits. We can then apply the extractor from Theorem 5.8.4 to get out $m = \frac{(k')^2}{8n} = \Omega(k^2/n(\max(\log(n/k), 1))^2)$ bits. ■

Notice that here we only need $k \gg n^{4/5}$ whereas before we required $k \gg n^{1-\eta}$ for some small constant η.

Now we describe how we express the restricted width two source X as a convex combination of previous-bit sources Z_j. This is done recursively on the layers of the branching program for the source. We say we are in a given state at each layer; either "open", "closed at 0", or "closed at 1". Each sequence of states corresponds to a previous-bit source. The way we divide the next layer up depends on the state we are in. The high level picture is that each random bit corresponds to going into the open state, which we are in until we get a fixed bit, which takes us to the corresponding closed state. We stay closed until another random bit occurs. An example is shown in Figure 5.8.2.

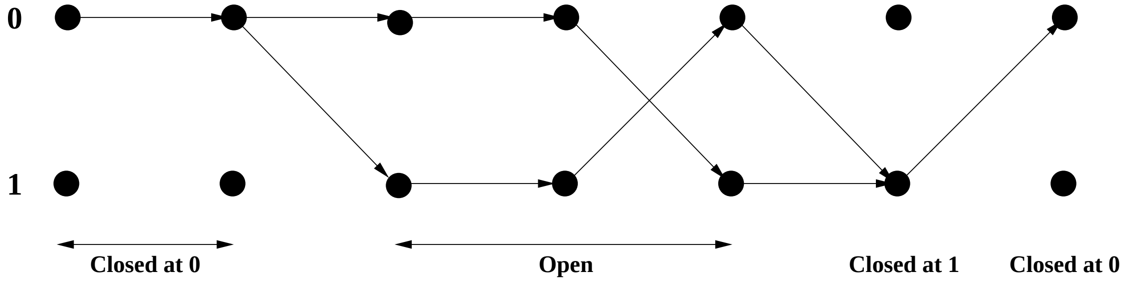

Figure 5.2: A previous-bit source viewed as a restricted width two source. This source consists of the bits $0, 0, r, r, \overline{r}, 1, 0$, where r is a random bit.

More formally, we define the following probabilities, shown in Figure 5.8.2.

$$\begin{aligned} p_{i0} &= \Pr[X_i = 0 | X_{i-1} = 0] \\ p_{i1} &= \Pr[X_i = 1 | X_{i-1} = 0] \\ q_{i0} &= \Pr[X_i = 0 | X_{i-1} = 1] \\ q_{i1} &= \Pr[X_i = 1 | X_{i-1} = 1] \end{aligned}$$

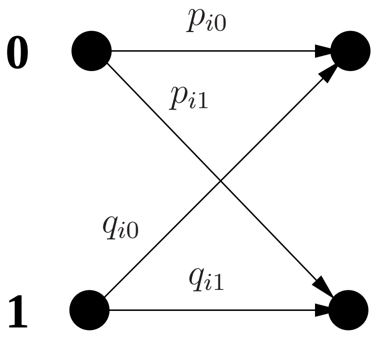

Figure 5.3: The probabilities for a single bit of a restricted width two source.

First, we describe what happens if we are currently in the open state. The next bit is fixed to 0 (resp. 1) and the state becomes closed at 0 (1) with probability $\min(p_{i0}, q_{i0})$ $(\min(p_{i1}, q_{i1}))$. Else we stay in the open state and the next bit is either equal to the previous bit or the negation of the previous bit depending on which edges have the remaining probability.

If we are closed at 0, the next bit is random and we go into the open state with probability $2\min(p_{i0}, p_{i1})$. If $p_{i0} < p_{i1}$, the next bit is fixed to 1 and we go into the closed at 1 state with probability $1 - 2p_{i0}$. Else the next bit is fixed to 0 and we go into the closed at 0 state with probability $1 - 2p_{i1}$.

If we are closed at 1, the next bit is random and we go into the open state with probability $2\min(q_{i0}, q_{i1})$. If $q_{i0} < q_{i1}$, the next bit is fixed to 1 and we go into the closed at 1 state with probability $1 - 2q_{i0}$. Else the next bit is fixed to 0 and we go into the closed at 0 state with probability $1 - 2q_{i1}$.

Now we show that with high probability, the sources in the convex combination have sufficient min-entropy. We do this by looking at the relationships between paths in the original source X and the min-entropy of the Z_j. First, note that each path in the branching program corresponds to an output value of X, so each path has probability at most 2^{-k}. Note that the min-entropy of Z_j is equal to the number of openings in Z_j.

Each path can be divided into edges that are the most probable edge coming out of a node and those that are the least probable. We will show how the number of least probable edges on a path in X relates to the min-entropy of a Z_j that contains this path. First note that every least probable edge corresponds to either an opening, a closing, or what we call a "false closing". A false closing is defined as transitioning from the open state to the open state yet still taking a least probable edge. Let $C(Z_j)$ denote the number of closings in Z_j, $A(Z_j)$ denote the number of openings, and $B(Z_j)$ denote the number of false closings.

If we could ignore the false closings, showing that with high probability we take the least probable edge a large number of times would be enough. Since $C(Z_j) \leq A(Z_j)$, this would imply that with high probability $A(Z_j)$ is large, and thus the Z_j have large min-entropy with high probability. To take account of the false closings, we also have to show that there aren't too many of them, which we will do by a martingale argument.

First, we show that with high probability over all paths in X, we take the least probable edge a large number of times.

Lemma 5.8.7. *For any length n restricted width two source with min-entropy k, the total probability of all paths that have at most $t = \min(k/8\log(n/k), k/16)$ least probable edges is less than $2^{-k/4}$.*

Proof. Since the source has min-entropy k, each path has probability at most 2^{-k}. There are $\binom{n}{i}$ paths that have i least probable edges. Thus the total probability of all paths that have at most t least probable edges is at most

$$2^{-k}\sum_{i=0}^{t}\binom{n}{i} \leq 2^{-k}2^{nH(t/n)} < 2^{-k+2t\log(n/t)}$$

where $H(t/n)$ is the standard Shannon entropy $H(p) = -p\log p - (1-p)\log(1-p)$.

Suppose $k \leq n/4$. Then $s = k/8\log(n/k)$, so

$$2s\log\frac{n}{t} = \frac{k}{4}\left(1 + \frac{\log(8\log\frac{n}{k})}{\log\frac{n}{k}}\right) \leq \frac{3k}{4}.$$

If $k > n/4$, then $t = \frac{k}{16}$, so

$$2t\log\frac{n}{t} = \frac{k}{8}\left(4 + \log\frac{n}{k}\right) \leq \frac{3k}{4}.$$

Thus the probability of taking at most t least probable edges is at most $2^{-k+2t\log(n/t)} \leq 2^{-k/4}$. □

To show that the number of false closings is small, we first define a submartingale that is equal to the number of closings minus the number of false closings after the first i bits. Then we use the following simple variant of Azuma's inequality for submartingales (see [Wor99] for a proof).

Definition 5.8.8. A submartingale with respect to a random process $G_0, G_1, \ldots$, with G_0 fixed, is a sequence $Y_0, Y_1, \ldots$ of random variable defined on the random process such that

$$\mathbb{E}[Y_{i+1}|G_0, G_1, \ldots, G_i] \geq Y_i$$

for all $i \geq 0$.

Lemma 5.8.9. *Let $Y_0, Y_1, \ldots, Y_n$ be a submartingale with respect to $G_0, G_1, \ldots, G_n$ where $Y_0 = 0$ and $|Y_i - Y_{i-1}| \leq 1$ for $i \geq 1$. Then for all $\alpha > 0$,*

$$\Pr[Y_n \leq -\alpha] \leq e^{-\frac{\alpha^2}{2n}}.$$

Now we are ready to prove that with high probability the number of false closings can't be too large.

Lemma 5.8.10. *For all $\alpha > 0$,*

$$\Pr[B(Z_j) \geq C(Z_j) + \alpha] \leq e^{-\frac{\alpha^2}{2n}}.$$

Proof. Let Y_i be the number of closings from $X_1, \ldots, X_i$ minus the number of false closings from $X_1, \ldots, X_i$ and let $Y_0 = 0$. Let $G_0, G_1, \ldots, G_n$ be the random process for dividing X into previous-bit sources, so G_i is the state after the first i bits have been divided.

Now we show that $Y_0, \ldots, Y_n$ is a submartingale with respect to $G_0, G_1, \ldots, G_n$. If we are in a closed state after i bits, then we have no closings or false closings at $i+1$, so $\mathbb{E}[Y_{i+1}|G_0, G_1, \ldots, G_i] = Y_i$. If we are in an open state at i, we show that if we have the possibility of a false closing at $i+1$, then the probability of closing is greater than $1/2$, and in particular is greater than the probability of a false closing. This would imply that $\mathbb{E}[Y_{i+1}|G_0, G_1, \ldots, G_i] \geq Y_i$, as desired. First, note that the probability of closing at $i+1$ is

$$\min(p_{i+1,0}, q_{i+1,0}) + \min(p_{i+1,1}, q_{i+1,1}) = \min(p_{i+1,0} + q_{i+1,1}, q_{i+1,0} + p_{i+1,1}).$$

Suppose without loss of generality that $p_{i+1,0} + q_{i+1,1} \geq q_{i+1,0} + p_{i+1,1}$, so we close with probability $q_{i+1,0} + p_{i+1,1}$. In this case, the edges we would take in a false closing are the 00 and 11 edges. So if we have a false closing, either $p_{i+1,0} \leq 1/2$ or $q_{i+1,1} \leq 1/2$, which implies either $p_{i+1,1} \geq 1/2$ or $q_{i+1,0} \geq 1/2$, and thus the probability of closing is at least $1/2$.

By the definition of Y_i, $|Y_i - Y_{i-1}| \leq 1$, so we can apply Lemma 5.8.9 to get

$$\Pr[Y_n \leq -\alpha] \leq e^{-\frac{\alpha^2}{2n}},$$

which implies the desired result.

□

Now we are finally ready to prove Lemma 5.8.5.

Proof of Lemma 5.8.5. First, express the restricted width two source X as a convex combination of previous-bit sources Z_j as described previously, so $X = \sum_j \alpha_j Z_j$. Now look at a randomly chosen Z_j, chosen with probability α_j. The number of random bits in Z_j is equal to the number of openings $A(Z_j)$. Since the number of closings is either equal to or one less than the number of openings, either $C(Z_j) = A(Z_j)$ or $C(Z_j) = A(Z_j) - 1$. So if we can prove with high probability that $C(Z_j)$ is large, then with high probability the number of random bits in Z_j is also large. For every path in Z_j, every least probable edge on the path corresponds to either an opening, a closing, or a false closing. Thus the probability that $A(Z_j) + B(Z_j) + C(Z_j) \geq s$ is at least the probability over all paths that the path has at least s least probable edges. Thus we can apply Lemma 5.8.7 and get

$$\Pr[B(Z_j) + 2C(Z_j) \geq s - 1] \geq \Pr[A(Z_j) + B(Z_j) + C(Z_j) \geq s] > 1 - 2^{-k/4}$$

for $s = \min(k/8 \log(n/k), k/16)$.

By Lemma 5.8.10,

$$\Pr[B(Z_j) < C(Z_j) + \frac{s}{2}] \geq 1 - e^{-\frac{s^2}{8n}}.$$

With high probability both of these events occur, so

$$\Pr[C(Z_j) \geq \frac{s}{6}] \geq 1 - 2^{-k/4} - e^{-\frac{s^2}{8n}}.$$

□

Chapter 6

Extractors for Low-Weight Affine Sources

Fix a vector space over a finite field $\mathbb{F}^n$. Then an *affine* source is a distribution which is uniform over some affine subspace of $\mathbb{F}^n$. An affine source extractor is an extractor that can extract from any affine source.

Affine sources are thus a generalization of bit-fixing sources, which correspond to the special case of affine subspaces that have a basis consisting only of weight one vectors. A *weight w* affine source is an affine source in which every basis vector has at most w non-zero coordinates. A bit-fixing source is thus just a weight 1 affine source.

One reason why bit-fixing sources are interesting is an application in cryptography, where they can be used to tolerate adversaries that are capable of learning or altering some bits of a long secret key. This application is referred to as *exposure resilient cryptography*. There has been a significant body of work [CFG+85, Riv97, Boy99, CDH+00, Dod00, DSS01] involved in constructing and using such extractors.

6.1 Extractors for Affine Sources via the Probabilistic Method

The number of affine sources with entropy k is determined by the choice of $k/\log(|\mathbb{F}|)$ basis elements and so is at most $\binom{|\mathbb{F}|^n}{k/\log(|\mathbb{F}|)} \leq 2^{nk}$.

Theorem 2.5.1 thus implies that a random function that outputs m bits is an extractor for affine sources with entropy k and error ϵ as long as $\epsilon^2 2^k \geq 6 \log e(2^m + nk)$, which is easily satisfied as long as $k > 2\log n$ and $m < k - \log(1/\epsilon) - O(1)$.

6.2 Previous Work and Our Results

Table 6.1 highlights some previous work for this type of problem. When the field is small (as in the case of $GF(2)$), the best known affine source extractor is due to Bourgain, who gives an extractor for any linear min-entropy with exponentially small error. When the field is sufficiently large, Gabizon and Raz [GR05] show how to extract many random bits, even when the entropy is arbitrarily small. For the case of bit-fixing sources (which as we mentioned above are important for the application to exposure resilient cryptography), the only earlier extractor with negligible

Construction	Min-Entropy	Error	Output	Ref
Extractor for bit-fixing sources over $GF(2)$	any k	$1/\mathrm{poly}(k)$	$\frac{\log k}{4}$	[KZ03]
Extractor for bit-fixing sources over $GF(2)$	$k > \sqrt{n}$	$2^{-\Omega(k^2/n)}$	$\Omega(k^2/n)$	[KZ03]
Extractor for bit-fixing sources over $GF(2)$	$k > \log^c(n)$ for some constant c	$1/\mathrm{poly}(k)$	$k - o(k)$	[GRS04]
Extractor for bit-fixing sources over $GF(2)$	$k > \sqrt{n}$	$2^{-\Omega(k^2/n)}$	$k - o(k)$	[GRS04]
Extractor for affine sources over $GF(2)$	$(0.5 + \alpha)n$, for positive constant α	$2^{-\Omega(n)}$	$\Omega(n)$	[KZ]
Extractor for affine sources over a large field, $\lvert\mathbb{F}\rvert > n^{20}$	Any k	$1/\mathrm{poly}(\lvert\mathbb{F}\rvert)$	$k - 1$ field elements	[GR05]
Disperser for affine sources over $GF(2)$	δn for any constant δ	Any constant	$\Theta(1)$	[BKS+05]
Extractor for affine sources over $GF(2)$	δn for any constant δ	$2^{-\Omega(n)}$	$\Omega(n)$	[Bou07]
Extractor for low-weight affine sources over $GF(2)$	$k > \log^c(n)$ for some constant c	$2^{-k^{\Omega(1)}}$	$k - o(k)$	This chapter

Table 6.1: Performance of extractors for affine and bit-fixing sources

error for $k < \sqrt{n}$ is due to [KZ03] and gives only $O(\log k)$ random bits. Our results in particular imply an improvement to this, giving extractors that output almost all of the bits of entropy with negligible error, as long as k is polylogarithmically large in n.

Our main result is:

Theorem 6.2.1. *There exist constants c, ϵ s.t. for every $k > \log^c n$, there exists a polynomial time computable function $\mathsf{AffExt} : \{0,1\}^n \to \{0,1\}^m$ that is an extractor for weight $w < n^\epsilon$ affine sources over $GF(2)$ with min-entropy k and output length $m = k - o(k)$.*

6.3 Techniques

Our techniques bear a striking resemblance to the techniques used to get our basic extractor for independent sources, discussed in Chapter 4.

We make progress by considering a more restricted class of affine sources, called *somewhere random affine sources*. This is merely an affine source that is also somewhere random. We discussed these sources in Chapter 3. SR-sources were first introduced by by Ta-Shma [TS96].

We will think of the number of rows of an affine SR-sources as a measure of the quality of the source. The fewer the number of rows, the better the quality is. We will iteratively improve the quality (reduce the number of rows) of the SR-sources that we are working with until extracting randomness from them becomes easy. We will need two basic tools:

- Our construction will use *linear strong seeded extractors* as a basic tool. This is simply a strong seeded extractor which is a linear function for every fixing of the seed. A strong seeded extractor can be viewed as a small family of deterministic functions (each function in the family indexed by a unique seed), such that for any fixed adversarially chosen source of randomness, almost all functions from the family are good extractors for that source. Linear strong seeded extractors simply give a family of *linear* functions with the same property. One example of a good linear strong seeded extractor is Trevisan's extractor [Tre01].

- Another basic tool we will use is a good *linear error correcting code*. This a linear subspace $\mathcal{C} \subset \{0,1\}^n$ with the property that every non-zero element of the subspace has a high weight. We will use such a code to get a linear function $H : \{0,1\}^n \to \{0,1\}^t$ (this is simply the parity check matrix of such a code) with the property that $P(c) = 0$ if and only if c is a codeword. Here t will be extremely small (say $\mathrm{polylog}(n)$).

Given these two basic tools, we can describe some basic observations that go into the construction. We will then show how to put these together to get the high level view of our extractor construction.

Idea 1: There is a simple linear condenser for low-weight affine sources. If P is the linear function (the parity check matrix) associated with the code we mentioned above, and X is any weight $d/l-1$ affine source, $P(X)$ is another affine source with entropy at least l. To see this, observe that P is an injective function over any low-weight subspace of X of dimension l, since any such subspace only has vectors of weight a most $(d/l-1)l < d$. Thus the dimension of $P(X)$ must be at least l.

Idea 2: Extraction is easy from *high quality* affine somewhere random sources. It is easy to extract from affine SR-sources when the source has very few rows relative to the length of each of the rows. In the extreme case, when the SR-source has just one row, it is a uniformly random string. When the number of rows is only a constant, we can simply use Bourgain's extractor Theorem 2.6.9 to get random bits. Building on this simple ideas, we will show how to build extractors for affine SR-sources even when the number of rows is superconstant. We will be able to extract from such sources as long as the number of rows is significantly less than the length of each row. These results are discussed in Section 3.3 of Chapter 3.

Idea 3: The quality of affine SR-sources can be transferred, even when they are dependent. A single affine SR-source S with t rows can be used to convert another affine source into an affine SR-source with t rows, even if the two sources are dependent, as long as the number of bits that S gives is less than the entropy of the other source. We simply use the t rows of S as seeds with a linear strong seeded extractor to extract from each of the other affine sources. Although the sources are dependent, we can show that the second affine source can be written as the sum of two affine sources, one of which is independent of S. With high probability, the random row of S is a good seed to extract from this independent affine source. It turns out that the output we obtain in this way is close to a convex combination of affine SR-sources, each with t rows. This observation can be interpreted as a way to *transfer* quality from a single affine SR-source to another dependent affine source.

Given these observations, the high level informal view of our extractor construction is the following:

1. Use **Idea 1** to convert the input affine source into a much shorter affine source which still has entropy.

2. Use a linear strong seeded extractor to convert this short affine source into an affine SR source with few ($\ll k$) rows.

3. Use **Idea 3** to transfer the quality of this affine somewhere random source back to the original affine source to get a new affine source whose rows are much longer than the length of each row.

4. Use **Idea 2** to extract from the new high quality affine somewhere random source.

6.4 Preliminaries

We will need to use some very basic, well studied concepts from the theory of error correcting codes.

Definition 6.4.1. An $[n, k, d]$ linear error correcting code over $\mathbb{F}$ is a linear subspace $\mathcal{C} \subset \mathbb{F}^n$ which is the image of an injective linear function $E : \mathbb{F}^k \to \mathbb{F}^n$, called the *encoding function* with the property that every non-zero element of $\mathcal{C}$ has weight at least d (i.e., the number of non-zero coordinates is at least d). Given such a code, $k = \dim \mathcal{C}$ is called the *message length or dimension* of the code.

Given any such error correcting code, it's easy to see that we can find a linear map $H : \mathbb{F}^n \to \mathbb{F}^{n-k}$ with the property that $H(c) = 0$ if and only if $c \in \mathcal{C}$. An ϵ-Biased distribution is a distribution that is pseudorandom for linear functions.

Definition 6.4.2 (ϵ-Biased Distribution)**.** A distribution X over $\{0, 1\}^n$ is *ϵ-biased* if for every non-zero element $v \in \{0, 1\}^n$, $v \cdot X$ is ϵ-close to uniform.

Another concept we will need is the concept of ϵ-biased distributions for low weight tests.

Definition 6.4.3 (ϵ-Biased for Low-Weight)**.** A distribution X over $\{0, 1\}^n$ is *ϵ-biased for linear tests of size w* if for every non-zero element v of $\{0, 1\}^n$ whose weight is at most w, $v \cdot X$ is ϵ-close to uniform.

Typically, we are interested in finding ϵ-biased distributions which can be generated with a very small seed length. These objects have turned out to be very useful theoretical computer science. We study the notion of being ϵ-biased further in Appendix D.

Constructions have been given in [NN93, AGHP92]. A construction in [AGHP92] gives a distribution that is ϵ-biased for weight w tests with seed length $2 \cdot \lceil \log(1/\epsilon) + \log w + \log \log n \rceil$.

Given any such ϵ-biased distribution with small seed length, we can immediately get a good code. Let $P : \{0, 1\}^n \to \{0, 1\}^t$ be the linear map whose i'th bit is the dot product of the input with the i'th element of the ϵ-biased distribution. Then we see that if $P(x) = 0$, x must have weight larger than w. Thus P is the parity check of some good error correcting code.

6.5 Converting Low-Weight Affine Sources into Affine Somewhere Random Sources

In this section, we show how to use linear error correcting codes to convert any low-weight affine source, into an affine source over fewer bits that still has entropy. We simply apply the parity check matrix of a good linear error correcting code to the sample from the affine source. Suppose we are dealing with an affine source of weight w and entropy k.

Lemma 6.5.1. *Let $P : \{0,1\}^n \to \{0,1\}^t$ be the parity check function for any linear error correcting code of distance greater than wk^α. Let X be any weight w affine source. Then $P(X)$ is an affine source with entropy at least k^α.*

Proof. First note that $P(X)$ is clearly an affine source, since it is obtained by applying a linear function to an affine source. It remains to show that $P(X)$ has the promised entropy. To see this, let $v_1, \ldots, v_k$ be a weight w basis for X. Then we see that every vector in the span of $v_1, \ldots, v_{k^\alpha}$ has weight at most wk^α. Thus, P is injective over this subspace. $P(X)$ is thus an affine source with a support of size at least 2^{k^α}, which means that $P(X)$ has entropy at least k^α. □

As our discussion in Section 6.4 shows, we can find such a function P that outputs

$$t = 2^{2\lceil \log(1/\epsilon) + \log(wk^\alpha) + \log\log n \rceil} \leq O(w^2 k^{2\alpha} \log^2 n)$$

bits for $\epsilon = 1/4$.

Now let $\mathsf{Ext} : \{0,1\}^t \times \{0,1\}^d \to \{0,1\}^m$ be a linear seeded extractor for min-entropy k^α. Then we can apply Ext to X using every possible seed to get a somewhere random source. By Proposition 2.3.6, the source we get in this way has 0 error.

If we use the extractor promised by Corollary 2.6.3, we get a seed length of at most $h \log t$. If $k > \log^2 n$, we can set $w = k^{\Omega(1)}$ and α to be small enough so that this seed length is less than $\log k/2$. Our discussion gives us the following lemma:

Lemma 6.5.2. *There exists a constant $l < 1/2$ and a polynomial time computable function $L : \{0,1\}^n \to \{0,1\}^{k^{1/2+l}}$ such that if X is any weight k^l affine source with entropy k, $L(X)$ is an affine $\sqrt{k} \times k^l$ somewhere random source.*

Unfortunately, this affine somewhere random source is not good enough, since its rows are not long enough for us to apply the extractor from Theorem 3.3.1. Still, we can use this source to turn our original source into a somewhere random source via the following algorithm:

Algorithm 6.5.3 ($\mathsf{AffineCondense}$).

Input: $x \in \{0,1\}^n$.
Output: z a $\sqrt{k} \times m$ matrix with $m = k - o(k)$.

Sub-Routines and Parameters:
Let L, l be as in Lemma 6.5.2, set up to work with entropy k.
Let $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^l \to \{0,1\}^m$ be a linear seeded extractor set up to extract m bits from entropy $k - k^{1/2-l}$.

1. Let z be the matrix whose i'th row is $\mathsf{Ext}(x, L(x)_i)$

We will then prove the following theorem:

Theorem 6.5.4. *Let X be a weight k^l affine source over $\{0,1\}^n$ with entropy k. Then* $\mathsf{AffineCondense}(X)$ *is $2^{-k^{\Omega(1)}}$-close to being a convex combination of affine somewhere random sources.*

Note that $L(X)_i$ is not independent of X, in fact it is completely determined by X! Thus it seems strange that we can prove anything about the distribution of Z. The key point is that $L(X)_i$ is a linear function of X. We can use this to show that even though these two are not independent, we can analyze them as if they are independent.

Proof. We will use Lemma 2.1.27. By the lemma, we can write $X = A+B$ where $H(B) \geq k-k^{1/2+l}$, and B is completely independent of $L(X) = L(A)$.

Note that for any fixing of $L(X) = L(A) = s$, the output of our algorithm is an affine source. Let h be an index such that $L(X)_h$ is uniformly random. Then we see that

$$\Pr_{s \leftarrow_{\mathrm{R}} L(A)}[|\mathsf{Ext}(B,s) - U_m| = 0] < 2^{-k^{\Omega(1)}}$$

But this implies that

$$\Pr_{s \leftarrow_{\mathrm{R}} L(X)}[|\mathsf{Ext}(X,s)|L(X) = s - U_m| = 0] < 2^{-k^{\Omega(1)}}$$

since $\mathsf{Ext}(X,s) = \mathsf{Ext}(A,s) + \mathsf{Ext}(B,s)$ and so is uniform as long as $\mathsf{Ext}(B,s)$ is uniform. Thus for $1-2^{-k^{\Omega(1)}}$ fraction of s, the output is a somewhere random affine source.

□

6.6 The Extractor

To get the final extractor, we simply compose the algorithm from the last section with our extractor for somewhere random sources, which we discussed in Section 3.3.

This gives us the following theorem:

Theorem 6.6.1. *There exist constants $\alpha, l > 0$ and a polynomial time computable function* $\mathsf{AffineExt}$: $\{0,1\}^n \to \{0,1\}^m$ *which is an extractor for affine sources with entropy k, weight k^l, error $2^{-k^{\Omega(1)}}$ and output length $k - k^{1-\alpha}$.*

Chapter 7

Explicit Ramsey Graphs

Definition 7.0.2. A graph on N vertices is called a *K-Ramsey Graph* if it contains no clique or independent set of size K.

In 1947 Erdős published his paper inaugurating the *Probabilistic Method* with a few examples, including a proof that *most* graphs on $N = 2^n$ vertices are $2n$-Ramsey. The quest for constructing such graphs explicitly has existed ever since and led to some beautiful mathematics.

The best record to date was obtained in 1981 by Frankl and Wilson [FW81], who used intersection theorems for set systems to construct N-vertex graphs which are $2^{\sqrt{n \log n}}$-Ramsey. This bound was matched by Alon [Alo98] using the *Polynomial Method*, by Grolmusz [Gro00] using low rank matrices over rings, and also by Barak [Bar06] boosting Abbot's method with almost k-wise independent random variables (a construction that was independently discovered by others as well). Remarkably all of these different approaches got stuck at essentially the same bound. In recent work, Gopalan [Gop06] showed that other than the last construction, all of these can be viewed as coming from low-degree symmetric representations of the OR function. He also shows that any such symmetric representation cannot be used to give a better Ramsey graph, which gives a good indication of why these constructions had similar performance. Indeed, as we will discuss in a later section, the $\sqrt{n}$ min-entropy bound initially looked like a natural obstacle even for our techniques, though eventually we were able to surpass it.

The analogous question for bipartite graphs seemed much harder.

Definition 7.0.3. A bipartite graph on two sets of N vertices is a *K-Ramsey Bipartite Graph* if it has no $K \times K$ complete or empty bipartite subgraph.

While Erdős' result on the abundance of $2n$-Ramsey graphs holds as is for bipartite graphs (we discussed this in Section 4.2), until recently the best explicit construction of bipartite Ramsey graphs was $2^{n/2}$-Ramsey, using the Hadamard matrix. This was improved recently, first to $o(2^{n/2})$ by Pudlak and Rődl [PR04] and then to $2^{o(n)}$ by Barak, Kindler, Shaltiel, Sudakov and Wigderson [BKS$^+$05].

It is convenient to view such graphs as functions $f : (\{0,1\}^n)^2 \to \{0,1\}$. This then gives exactly the definition of a disperser.

Definition 7.0.4. A function $f : (\{0,1\}^n)^2 \to \{0,1\}$ is called a *2-source disperser for min-entropy k* if for any two sets $X, Y \subset \{0,1\}^n$ with $|X| = |Y| = 2^k$, we have that the image $f(X,Y)$ is $\{0,1\}$.

This allows for a more formal definition of *explicitness*: we simply demand that the function f is computable in polynomial time. Most of the constructions mentioned above are explicit in this sense.[1]

Our main result (stated informally) significantly improves the bounds in both the bipartite and non-bipartite settings:

Theorem 7.0.5. *For every N we construct polynomial time computable bipartite graphs which are $2^{2^{\log^{1-\alpha_0} n}}$-Ramsey, for some universal constant $\alpha_0 > 0$.*

A standard transformation[2] of these graphs can be used to convert any bipartite Ramsey-graph into a Ramsey-graph. Thus we obtain the following corollary.

Corollary 7.0.6. *For every N we construct polynomial time computable graphs which are $2^{2^{\log^{1-\alpha_0} n}}$-Ramsey, for some universal constant $\alpha_0 > 0$.*

In this thesis we continue the work initiated in [BKS+05], showing how to use extractors for more than 2 sources to get dispersers for 2 sources. The seminal paper of Bourgain, Katz and Tao [BKT04] proved the so-called "sum-product theorem" in prime fields, a result in arithmetic combinatorics. This result has already found applications in diverse areas of mathematics, including analysis, number theory, group theory and extractor theory. Their work implicitly contained dispersers for $c = O(\log(n/k))$ independent sources of min-entropy k (with output $m = \Omega(k)$). The use of the "sum-product" theorem was then extended by Barak et al. [BIW04] to give extractors with similar parameters. Note that for linear min-entropy $k = \Omega(n)$, the number of sources needed for extraction c is a constant!

Relaxing the independence assumptions via the idea of repeated condensing allowed the reduction of the number of independent sources. This gave extractors for just $c = 3$ independent sources of any linear entropy $k = \Omega(n)$, by Barak et al. [BKS+05] and independently by Raz [Raz05].

For 2 sources Barak et al. [BKS+05] were able to construct dispersers for sources of min-entropy $o(n)$. To do this, they first showed that if the sources have extra block source structure (Definition 2.1.15), even extraction is possible from 2 sources. The notion of block-sources, capturing "semi independence" of parts of the source, was introduced by Chor and Goldreich [CG88]. It has been fundamental in the development of seeded extractors and as we shall see, is essential for us as well.

This definition allowed Barak et al. [BKS+05] to show that their extractor for 4 independent sources, actually performs as well with only 2 independent sources, as long as both are 2-block-sources.

Theorem 7.0.7 ([BKS+05])**.** *There exists a polynomial time computable extractor $f : (\{0,1\}^n)^2 \to \{0,1\}$ for 2 independent 2-block-sources with min-entropy $o(n)$.*

There is no reason to assume that the given sources are block-sources, but it is natural to try and reduce to this case. This approach has been one of the most successful in the extractor

[1] The Abbot's product based Ramsey-graph construction of [BIW04] and the bipartite Ramsey construction of [PR04] only satisfy a weaker notion of explicitness.

[2] The standard transformation is this one: Let the bipartite graph have $2N$ vertices, N on each side, with edge set E. Then define a new graph with vertex set $[N]$ and edge set $\{\{a,b\} : a \geq b \text{ and } (a,b) \in E\}$. It's easy to check that if the bipartite graph was K-Ramsey, the new graph is $\frac{K}{2}$-Ramsey.

literature. Namely try to partition a source X into two blocks $X = X_1, X_2$ such that X_1, X_2 form a 2-block-source. Barak et al. introduced a new technique to do this reduction called the *challenge-response mechanism*, which is crucial for this chapter. This method gives a way to "find" how min-entropy is distributed in a source X, guiding the choice of such a partition. This method succeeds only with small probability, dashing the hope for an extractor, but still yielding a disperser.

Theorem 7.0.8 ([BKS+05]). *There exists a polynomial time computable 2-source disperser* $f : (\{0,1\}^n)^2 \rightarrow \{0,1\}$ *for min-entropy* $o(n)$.

Reducing the min-entropy requirement of the above 2-source disperser, which is what we achieve in this chapter, again needed progress on achieving a similar reduction for extractors with more independent sources, which was achieved in a result discussed in Chapter 4. There we improved all the above results for $c \geq 3$ sources. Our result improves the results of Barak et al. [BIW04] to give $c = O((\log n)/(\log k))$-source extractors for min-entropy k. Note that now the number c of sources needed for extraction is constant, even when the min-entropy is as low as n^δ for any constant δ!

The following theorem, which we proved in Chapter 4, will be crucial to our results.

Theorem 7.0.9 (Chapter 4). *There is a polynomial time computable extractor* $f : (\{0,1\}^n)^2 \rightarrow \{0,1\}^m$ *for a source with min-entropy* k *and an independent* c*-block-source with block min-entropy* k *as long as* $c = O((\log n)/(\log k))$.

7.0.1 Our Results

The results in this chapter are based on work with Boaz Barak, Ronen Shaltiel and Avi Wigderson [BRSW06].

The main result of this chapter is a polynomial time computable disperser for 2 sources of min-entropy $n^{o(1)}$, improving the results of Barak et al. [BKS+05] ($o(n)$ min-entropy). It also improves on Frankl and Wilson [FW81], who built Ramsey Graphs (which in our terms is a disperser for two *identically distributed* sources) for min-entropy $\tilde{O}(\sqrt{n})$.

Theorem 7.0.10 (Main theorem, restated). *There exists a constant* $\alpha_0 > 0$ *and a polynomial time computable* 2*-source disperser* $\mathsf{D} : (\{0,1\}^n)^2 \rightarrow \{0,1\}$ *for min-entropy* $2^{\log^{1-\alpha_0} n}$.

This result is obtained via a variant of the challenge-response mechanism. We generalize and extend this technique so that it can be used in a recursive manner. There are many technical complications that we need to deal with to accomplish this. In the next section, we give a very high level abstract overview of the challenge-response mechanism. A more detailed informal overview of the high level ideas in our construction, followed by a formal description and analysis, are in Section 7.2.

Several constructions of Barak et al. [BKS+05] and Raz [Raz05] worked by first reducing the extractor/disperser problem they were solving to the case of extracting from independent somewhere random sources. Then they used brute force search to construct an extractor for this case. In this thesis we obtain much better explicit extractors for independent somewhere random sources. This allows us to improve some of the results from earlier works. In particular, we obtain a polynomial time computable 2-source disperser that outputs a linear number of bits with exponentially small error when the min-entropy rate of the source is an arbitrarily small constant, by modifying the construction of Barak et al. [BKS+05], which had constant output length.

7.0.2 Techniques

In this section we give an overview of the high level structure of our argument. Most of the concepts below were introduced by Barak et al. [BKS$^+$05]. Here we will not say much about how these techniques are adapted to our application, leaving that for Section 7.2 (where there is also an informal discussion before the formal proof).

A basic concept we will need is that of a *subsource* of a source. This is the analogue of a subset of a set and indeed, given a source X, we will write $\hat{X} \subset X$ to denote that $\hat{X}$ is a subsource of X. Given a source of randomness X, we say that $\hat{X}$ is a subsource of X if $\hat{X}$ is defined as $X | X \in A$, where A is some set. So a subsource is just a new source which gives the same distribution as the original source *conditioned* on giving a point from some set. Throughout this chapter we will need to make sure that the subsources we are working with have a large density in the original source. We measure this density by measuring $\Pr[X \in A]$. If this probability is 2^{-d}, we call d the *deficiency* of the subsource $\hat{X}$. It is immediate that if X has min-entropy k and $\hat{X} \subset X$ has deficiency d, $\hat{X}$ must have min-entropy $k - d$.

It turns out that although it seems hard to build good extractors for two independent sources, we can, with some effort, build a *subsource extractor*. This is an extractor which is only guaranteed to work on subsources of the original source. Barak et al. manage to construct a function SExt with the property that for any linear min-entropy independent sources X, Y, there exist subsources $\hat{X} \subset X, \hat{Y} \subset Y$ such that $\mathsf{SExt}(\hat{X}, \hat{Y})$ is close to uniformly random. Given such a function, we see that it must be a disperser! Indeed, if A, B are the defining sets for the subsources $\hat{X}, \hat{Y}$, for any string c we have:

$$\begin{aligned}
&\Pr[\mathsf{SExt}(X,Y) = c] \\
&\geq \Pr[\mathsf{SExt}(X,Y) = c | X \in A, Y \in B] \Pr[X \in A \wedge Y \in B] \\
&= \Pr[\mathsf{SExt}(\hat{X},\hat{Y}) = c] \Pr[X \in A \wedge Y \in B]
\end{aligned}$$

Thus, if the string c is hit with positive probability in the subsources, then it is hit with positive probability in the original sources.

The point is that while it may be hard to build an extractor for two general sources, by moving to subsources we can weed out problematic points in our sources. For example, although block sources (Definition 2.1.15) are much more restricted than general sources, it can be shown that every general source X with linear min-entropy has a low deficiency subsource $\hat{X} \subset X$ which can be partitioned to get a block source!

More precisely, given any source X with some min-entropy δn, [BKS$^+$05] showed that if X is broken into equally sized parts $X_1, \ldots, X_t$ with $t \gg 1/\delta$, there is a subsource $\hat{X}$ and an index i such that $(X_1, \ldots, X_i), (X_{i+1}, \ldots, X_t)$ is a block source. It is then natural to try and *find* this index i. Indeed this is what Barak et al. manage to do. They design an efficiently computable function $\mathsf{Test}(X, Y, i)$ which outputs a bit and when given two sources X and Y of *linear* min-entropy and an index $1 \leq i \leq t$ distinguishes (this will be made more precise below) between two cases:

1. $X_1, \ldots, X_i$ has no entropy and
2. $X_1, \ldots, X_i$ has a lot of entropy

More formally, Test has the following properties: If X and Y are independent sources of *high* entropy then:

1. If $X_1, \dots, X_i$ is fixed then there exist (low deficiency) subsources $\hat{X} \subset X, \hat{Y} \subset Y$ such that $\mathsf{Test}(\hat{X}, \hat{Y}, i) = 0$.
2. If $X_1, \dots, X_i$ has high entropy then there exist (low deficiency) subsources $X' \subset X, Y' \subset Y$ such that $\mathsf{Test}(X', Y', i) = 1$ with high probability.

Thus, when given two sources X, Y from one of the two cases, there are subsources $(\hat{X}, \hat{Y})$ or (X', Y') which have roughly the same structure as the initial sources and on which Test decides correctly. By applying Test iteratively, we can locate high entropy regions of the initial sources X, Y. We can then use this knowledge to design an extractor that succeeds on the relevant subsources. As explained earlier such an extractor yields a disperser for the initial sources.

Next we give some intuition for how this function Test is constructed.

The Challenge-Response Mechanism

We now describe abstractly a mechanism which is used, both in the work of Barak et al. [BKS+05] and in our work, to detect entropy concentrations in sources. The reader may decide, now or in the middle of this section, to skip ahead to Section 7.2, which describes the construction of the disperser that extensively uses this mechanism. Then this section may seem less abstract.

Using the challenge-response mechanism, one can partition a given source into blocks in a way which make it a block source, or alternatively, focus on a part of the source which has an unusually high concentration of entropy– two cases which may simplify the extraction problem. Though the use of the challenge-response mechanism is more involved in this chapter than in [BKS+05], the basic idea behind the mechanism is the same:

Let Z be a source and Z' a part of Z (Z projected on a subset of the coordinates). We know that Z has entropy k, and want to distinguish two possibilities: Z' has no entropy (it is fixed) or it has at least k' entropy. In the context of the initial 2 sources X, Y we will operate on, think of Z as a part of X, so Y is independent of Z and Z'. To execute the test, we will define two functions:

The Challenge A function $\mathsf{Challenge} : \{0,1\}^{n''} \times \{0,1\}^n \to (\{0,1\}^{\mathsf{clen}})^{\ell}$. The output is viewed as a set of ℓ strings of length clen.

The Response A function $\mathsf{Response} : \{0,1\}^{n'} \times \{0,1\}^n \to (\{0,1\}^{\mathsf{clen}})^{\mathrm{poly}(n)}$. The output is viewed as a set of poly(n) strings of length clen.

We will decide that Z' has low entropy if $\mathsf{Challenge}(Z', Y) \subset \mathsf{Response}(Z, Y)$ and we will decide that it has high entropy otherwise.

The key to the usefulness of this mechanism is the following lemma, which states that what *should* happen *does* happen, at least on some subsources of Z and Y. We state it and then explain how the functions Challenge and Response are chosen to accommodate its proof.

Informal Lemma 7.0.11. Assume Z, Y are sources of entropy k.

1. If Z' has entropy $k' + O(\mathsf{clen})$, then

$$\Pr[\mathsf{Challenge}(Z', Y) \subsetneq \mathsf{Response}(Z, Y)] \geq 1 - \mathrm{poly}(n) 2^{-\mathsf{clen}}$$

2. If Z' is fixed (namely, has zero entropy), then there exist subsources $\hat{Z} \subset Z$ and $\hat{Y} \subset Y$, such that
$$\Pr[\mathsf{Challenge}(\hat{Z}', \hat{Y}) \subseteq \mathsf{Response}(\hat{Z}, \hat{Y})] = 1$$

Once we have such a mechanism, we will design our disperser algorithm assuming that the challenge-response mechanism correctly identifies parts of the source with high or low levels of entropy. Then in the analysis, we will ensure that our algorithm succeeds in making the right decisions, at least on subsources of the original input sources.

Now let us talk about how we obtain such functions $\mathsf{Challenge}$ and $\mathsf{Response}$. The response set $\mathsf{Response}(Z, Y)$ is chosen to be the output of a *somewhere extractor* with a polynomial number of rows. This is a function which on input two sources of low entropy, produces a polynomial sized set of outputs, one of which is guaranteed to be uniformly random. The response function that we use will have two additional properties:

1. Given any sources Z, Y and any fixed index i, (Z, Y) is a convex combination of $O(\mathsf{clen})$ deficiency subsources $\overline{Z} \subset Z, \overline{Y} \subset Y$ s.t. the ith element of $\mathsf{Response}(\overline{Z}, \overline{Y})$ is fixed for every subsource $(\overline{Z}, \overline{Y})$ in the combination.

2. Given any fixed string c and sources Z, Y, there are low deficiency subsources $\overline{Z}, \overline{Y}$ s.t. $\Pr[c \in \mathsf{Response}(\overline{Z}, \overline{Y})] = 1$.

It turns out that it's not too hard to construction an efficiently computable function $\mathsf{Response}$ with these properties.

The challenge set $\mathsf{Challenge}(Z', Y)$ is chosen to be the output of another somewhere extractor with very few, say constant number ℓ of outputs. The challenge function is only required to succeed when the sources it operates on have high entropy (for this case we can build a somewhere extractor with few outputs).

Why does it work? We explain each of the two claims in the lemma in turn.

1. Z' has high entropy. We will point to the output string in $\mathsf{Challenge}(Z', Y')$ which avoids $\mathsf{Response}(Z, Y)$ with high probability as follows. In the analysis we will use the union bound on several events, one associated with each (poly(n) many) string in $\mathsf{Response}(Z, Y)$. We note that by the first property of the response function, we can break (Z, Y) into a convex combination such that for every element of the combination $(\overline{Z}, \overline{Y})$, the first element of the response set is fixed. However, even in this element of the convex combination, we will be able to ensure that one of the outputs of $\mathsf{Challenge}(\overline{Z}, \overline{Y})$ is uniform. The probability that this output will equal any fixed value is thus $2^{-\mathsf{clen}}$. Repeating this for each of the poly(n) elements of the response set completes the argument.

2. Z' has no entropy. We now need to guarantee that in the subsources that we choose $(\hat{Z}, \hat{Y})$, all strings in $\mathsf{Challenge}(\hat{Z}', \hat{Y})$ are in $\mathsf{Response}(\hat{Z}, \hat{Y})$. First notice that since $Z' = z'$ is fixed, $\mathsf{Challenge}(Z', Y)$ is only a function of Y. We set $\tilde{Y}$ to be the subsource of Y that fixes all strings in $\mathsf{Challenge}(z', Y)$ to their most popular values (losing only $\ell \cdot \mathsf{clen}$ entropy from Y). We take care of including these fixed strings in $\mathsf{Response}(Z, \tilde{Y})$ one at a time, by using the second property of the response function. This is repeated successively ℓ times, and results in the final subsources $\hat{Z}, \hat{Y}$ on which $\Pr[\mathsf{Challenge}(\hat{Z}', \hat{Y}) \subset \mathsf{Response}(\hat{Z}, \hat{Y})] = 1$. Note that we keep reducing the entropy of our sources ℓ times, which necessitates that this ℓ be tiny.

Initially it seemed like the challenge-response mechanism as used in [BKS+05] could not be used to handle entropy that is significantly less than $\sqrt{n}$ (which is approximately the bound that many of the previous constructions got stuck at). The techniques of [BKS+05] involved partitioning the sources into t pieces of length n/t each, with the hope that one of those parts would have a significant amount of entropy, yet there'd be enough entropy left over in the rest of the source (so that the source can be partitioned into a block source).

However it is not clear how to do this when the total entropy is less than $\sqrt{n}$. On the one hand we will have to partition our sources into blocks of length significantly more than $\sqrt{n}$ (or the adversary could distribute a negligible fraction of entropy in all blocks). On the other hand, if our blocks are so large, a single block could contain all the entropy. Thus it was not clear how to use the challenge-response mechanism to find a block source. In this chapter, we resolve this issue by finding a way to use it recursively. The details are in Section 7.2.

7.1 A Somewhere Extractor with exponentially small error

A technical tool that we will need is a somewhere extractor which has a polynomial number of output rows, but exponentially small error. This will be used to generate the *responses* throughout our disperser construction. Note that we can get a polynomial number of output rows by using a seeded extractor with just one of the sources, but in this case the error would not be small enough. In addition, in this section we will prove some other technical properties of this construction which will be critical to our construction.

Theorem 7.1.1 (Low Error Somewhere Extractor)**.** *There is a constant γ such that for every $n, k(n) > \log^{10} n, \log^4 n < m < \gamma k$, there is a polynomial time computable function $\mathsf{SE} : (\{0,1\}^n)^2 \rightarrow (\{0,1\}^m)^l$ with the property that for any two (n,k) sources X, Y,*

Few rows *$l = \text{poly}(n)$.*

Small error *$\mathsf{SE}(X,Y)$ is 2^{-10m}-close to a convex combination of somewhere random distributions and this property is strong with respect to both X and Y. Formally:*

$$\Pr_{y \leftarrow_R Y}[\mathsf{SE}(X,y) \textit{ is } 2^{-10m}\textit{-close to being SR }] > 1 - 2^{-10m}$$

Hitting strings *Let c be any fixed m bit string. Then there are deficiency $2m$ subsources $\hat{X} \subset X, \hat{Y} \subset Y$ such that $\Pr[c \in \mathsf{SE}(\hat{X},\hat{Y})] = 1$.*

Fixed rows on low deficiency subsources *Given any particular row index i, there is a $20m$ deficiency subsource $(\hat{X},\hat{Y}) \subset (X,Y)$ such that $\mathsf{SE}(\hat{X},\hat{Y})_i$ is a fixed string. Further, (X,Y) is 2^{-10m}-close to a convex combination of subsources such that for every $(\hat{X},\hat{Y})$ in the combination,*

- *$\hat{X}, \hat{Y}$ are independent.*
- *$\mathsf{SE}(\hat{X},\hat{Y})_i$ is constant.*
- *$\hat{X}, \hat{Y}$ have min-entropy $\geq k - 20m$*

Proof. The algorithm SE is the following:

Algorithm 7.1.2 ($\mathsf{SE}(x,y)$).

Input: $x, y \in \{0,1\}^n$.
Output: A $n^d \times m$ boolean matrix.

Sub-Routines and Parameters:

- A seeded extractor Ext with $O(\log n)$ seed length (for example by Theorem 2.6.1), set up to extract from entropy threshhold $0.9k$, with output length m and error $1/100$.
- Raz's extractor with weak random seeds (Theorem 2.6.6) set up to extract m bits from an (n,k) source using a weak seed of length m bits with entropy $0.9m$. We can get such an extractor with error 2^{-10m}.

```
1. For every seed $i$ to the seeded extractor $\mathsf{Ext}$, output $\mathsf{Raz}(x, \mathsf{Ext}(y,i))$.
2. For every seed $i$ to the seeded extractor $\mathsf{Ext}$, output $\mathsf{Raz}(y, \mathsf{Ext}(x,i))$.
```

We will prove each of the items in turn.

Few rows By construction.

Small error We will argue that the strong error with respect to Y is small. By symmetry the same argument can be used to prove that the strong error for X is strong. Consider the set of bad y's,

$$B = \{y : \forall i \text{ s.t. } |\mathsf{Raz}(X, \mathsf{Ext}(y,i)) - U_m| \geq 2^{-\gamma' k}\}$$

where here γ' is the constant that comes from the error or Raz's extractor.

We would like to show that this set is very small.

Claim 7.1.3. $|B| < 2^{0.9k}$

Suppose not. Let B denote the source obtained by picking an element of this set uniformly at random. Since Ext has an entropy threshhold of $0.9k$, there exists some i such that $\mathsf{Ext}(B,i)$ is $1/100$ close to uniform. In particular, $|\mathsf{Supp}(\mathsf{Ext}(B,i))| \geq 0.99 \cdot 2^m > 2^{0.9m}$. This is a contradiction, since at most $2^{0.9m}$ seeds can be bad for Raz.

Thus we get that

$$\Pr_{y \leftarrow_{\mathrm{R}} Y}[|\mathsf{Raz}(X, \mathsf{Ext}(y,i)) - U_m| < 2^{-\gamma' k}] < 2^{0.9k}/2^k = 2^{-0.1k}$$

Setting $\gamma = \min\{\gamma'/10, 0.1\}$, we get that $10m < 10\gamma k < \gamma' k$ and $10m < 0.1k$.

Hitting strings The proof for this fact follows from the small error property. Let $\tilde{Y} = Y | Y \notin B$, where B is the set of bad y's from the previous item. Then we see that for every $y \in \mathsf{Supp}(\tilde{Y})$, there exists some i such that $|\mathsf{Raz}(X, \mathsf{Ext}(y, i)) - U_m| < 2^{-10m}$. By the pigeonhole principle, there must be some seed s and some index i such that:

$$\Pr_{y \leftarrow_{\mathrm{R}} \tilde{Y}}[\mathsf{Ext}(y, i) = s] \geq \frac{1}{l2^m}$$

Fix such an i and seed s and let $\hat{Y} = \tilde{Y}|\mathsf{Ext}(\tilde{Y}, i) = s$. This subsource has deficiency at most $1 + m + \log l < 2m$ from Y. Thus $\mathsf{Ext}(\hat{Y}, i)$ is fixed and $|\mathsf{Raz}(X, \mathsf{Ext}(y, i)) - U_m| < 2^{-10m}$. Note that the i'th element of the output of $\mathsf{SE}(X, \hat{Y})$ is a function only of X. Thus we can find a subsource $\hat{X} \subset X$ of deficiency at most $2m$ such that this i'th row is equal to c.

Fixed rows on low deficiency subsources We will first make the argument for the first half of the rows (where Y is used to generate the seeds to Raz). The argument for the other rows will follow by symmetry. Let i be any fixed row in the first half of the rows. For any m bit string a, let $Y_a \subset Y$ be defined as $Y|\mathsf{Ext}(Y, i) = a$. By Proposition 2.1.10, for any $\ell > 1$, $\Pr_{a \leftarrow_{\mathrm{R}} \mathsf{Ext}(Y,i)}[Y_a \text{ has deficiency more than } m + \ell\ ] < 2^{-\ell}$.

Let $A = \{a : Y_a \text{ has deficiency more than } m + \ell\}$. Then, by the lemma, we see that Y is $2^{-\ell}$-close to a source $\overline{Y}$, where $\Pr[\mathsf{Ext}(\overline{Y}, i) \notin A] = 1$, and $\overline{Y}$ has min-entropy at least $k - 1$.

We break up $\overline{Y}$ into a convex combination of variables $\hat{Y}_a = \overline{Y}|\mathsf{Ext}(\overline{Y}, i) = a$, each of deficiency at most $m + \ell$.

Similarly we can argue that X is $2^{-\ell}$-close to a random variable $\overline{X}_a$ with min-entropy $k - 1$, where $\overline{X}_a$ is a convex combination of subsources $\hat{X}_{a,b}$ with deficiency at most $m + \ell$ such that $\mathsf{Raz}(\hat{X}_{a,b}, a)$ is constant and equal to b.

Thus we obtain our final convex combination. Each element $\hat{X}_{a,b}, \hat{Y}_b$ of the combination is associated with a pair (a, b) of m bit strings. By construction we see that the i'th row $\mathsf{SE}(\hat{X}_{a,b}, \hat{Y}_b)_i = a$ and that $\hat{X}_{a,b}, \hat{Y}_b$ each have min-entropy $k - m - \ell$.

■

7.2 The Disperser

First we give an informal high level overview of the construction and analysis. The goals of this overview are to highlight the main ideas that go into the construction and analysis and assist the reader in reading the formal proof that appears after it. In order to highlight the main ideas and techniques, we sometimes ignore technical details, but give pointers to the places in the formal proof where the details are explained.

7.2.1 Informal Overview of Construction and Analysis

We are given two input sources X, Y which have some min-entropy and would like to output a non-constant bit.

The idea behind the construction is to try to convert the first source X into a block source or at least find a subsource (Definition 2.1.3) $X^{\mathsf{good}} \subset X$ which is a block source. Once we have such a block source, we can make use of some of the technology we have developed for dealing with block sources (for instance the extractor BExt of Theorem 4.5.1).

One problem with this approach is that there is no deterministic procedure that transforms a source into a block-source, or even to a short (e.g. of length much less than $\frac{n}{k}$) list of block sources. Still, as we will explain shortly, we will manage to use the second source Y to "convert" X into a block source. Loosely speaking, we will show that there exist independent subsources $X^{\mathsf{good}} \subset X$ and $Y^{\mathsf{good}} \subset Y$ such that X^{good} is a block source and our construction "finds" this block-source when applied on $X^{\mathsf{good}}, Y^{\mathsf{good}}$. This task of using one source to find the entropy in the other source while maintaining independence (on subsources) is achieved via the challenge-response mechanism.

We will describe our construction in two phases. As a warmup, we will first discuss how to use the challenge-response mechanism in the case when the two sources have linear min-entropy (this was first done by Barak et al. [BKS$^+$05]). Then we describe how to adapt the challenge-response mechanism for the application in this chapter.

Challenge-Response Mechanism for Linear Min-Entropy

The challenge-response mechanism was introduced in [BKS$^+$05] as a way to use one source of randomness to *find* the entropy in another source. Since they were only shooting for 2 source dispersers that could handle linear min-entropy, they avoided several complications that we will need to deal with here. Still, as an introduction to the challenge-response mechanism, it will be enlightening to revisit how to use the mechanism to get dispersers for linear min-entropy. Below we will give a sketch of how we might get such a disperser using the technology that is available to us at this point. Note that the construction we discuss here is slightly different from the one originally used by Barak et al.

We remind the reader again of the high level scheme of our construction. We will construct a polynomial time computable function Disp with the property that for any independent linear entropy sources X, Y, there exist subsources $X^{\mathsf{good}} \subset X, Y^{\mathsf{good}} \subset Y$ with the property that $\mathsf{Disp}(X^{\mathsf{good}}, Y^{\mathsf{good}})$ is both 0 and 1 with positive probability. Since $X^{\mathsf{good}}, Y^{\mathsf{good}}$ are subsources of the original sources, this implies that Disp is a disperser even for the original sources. Now let us describe the construction.

Let us assume that for linear min-entropy our extractor BExt requires only 2 blocks; so we have at our disposal a function $\mathsf{BExt} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ with the property that if X_1, X_2 is a block source with linear min-entropy, and Y is an independent block source, $\mathsf{BExt}(X_1, X_2, Y)$ is exponentially close to being a uniform bit.

We are given two sources X, Y which are independent sources with min-entropy δn, where δ is some small constant. We would be in great shape if we were given some additional advice in the form of an index $j \in [n]$ such that $X_{[j]}, X$ is a block source with min-entropy say $\delta n/10$ (i.e., the first j bits of X have min-entropy $\delta n/10$ and conditioned on any fixing of these bits the rest of the source still has min-entropy at least $\delta n/10$). In this case we would simply use our block source extractor BExt and be done. Of course we don't have any such advice, on the other hand, the good news is that it can be shown that such an index j *does* exist.

Step 1: Existence of a structured subsource We associate a *tree of parts* with the source X. This a tree of depth 1, with the sample from X at the root of the tree. We break the sample from the source X into a constant $\mathsf{t} \gg 1/\delta$ number of equally sized parts $x = x_1, \ldots, x_{\mathsf{t}}$, each containing n/t bits. These are the children of the root. Our construction will now operate on the bits of the source that are associated with the nodes of this tree.

The first step of the analysis is to show (via applications of Lemma 2.1.18 and Corollary 2.1.19) that

Informal Lemma 7.2.1. If X has min-entropy δn, there is a $j \in [t]$ and a subsource $\hat{X} \subset X$ in which:

- $\hat{X}_i$ is fixed for $i < j$.
- $H_\infty(\hat{X}_j) \geq \delta^2(n/t)$.
- $(\hat{X}_{j+1}, \ldots, \hat{X}_t)$ has conditional min-entropy at least $\delta^2 n$ given any fixing of $\hat{X}_j$.

Given this lemma, our goal is to find this index j (which is the 'advice' that we would like to obtain). We will be able to do so on independent subsources of $\hat{X}$, Y. This is achieved via the challenge-response mechanism.

Step 2: Finding the structure using the challenge-response mechanism Here are the basic pieces we will use to find the index j:

1. A polynomial time computable function $\mathsf{Challenge} : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^{\mathsf{clen}}$. In view of the final construction, we view the output of this function as a matrix with 1 row of length clen. We also require the following properties:

   Output length is much smaller than entropy $\mathsf{clen} \ll \delta^{20} n$.

   Output has high min-entropy Given $\hat{X}, \hat{Y}$ which are independent sources with min-entropy $\delta n/100$ each, $\mathsf{Challenge}(\hat{X}, \hat{Y})$ is statistically close to having min-entropy $\Omega(\mathsf{clen})$.

   In extractor terminology, these conditions simply say that $\mathsf{Challenge}$ is a *condenser* for 2 independent sources. In [BKS⁺05] such a function (in fact a somewhere random extractor) was constructed using results from additive number theory.

2. A polynomial time computable function $\mathsf{Response} : \{0,1\}^n \times \{0,1\}^n \rightarrow (\{0,1\}^{\mathsf{clen}})^\ell$. We interpret the output as a list of ℓ matrices that have the same dimensions as the challenge matrix given by the $\mathsf{Challenge}$ function above. This function must satisfy:

   Few matrices $\ell = \mathrm{poly}(n)$[3].

   Hitting matrices Given $\hat{X}, \hat{Y}$, independent sources with min-entropy $\delta^3 n$ each and any fixed matrix $\alpha \in \{0,1\}^{\mathsf{clen}}$, there there exists i and low deficiency subsources $X' \subset \hat{X}, Y' \subset \hat{Y}$ such that in these subsources $\mathsf{Response}(X', Y')_i = \alpha$ with probability 1.

   Fixed matrices on low deficiency subsources Given any index i and any independent sources $\hat{X}, \hat{Y}$, we can decompose $(\hat{X}, \hat{Y})$ into a convex combination of low deficiency independent sources such that for every element of the combination X', Y', $\mathsf{Response}(X', Y')_i$ is fixed to a constant.

   Note that these are exactly the properties we have proved about our somewhere extractor (Theorem 7.1.1). Indeed this is the function that we will use (both here and later) in our construction to generate responses.

[3] In [BKS⁺05] the component they use for this step has an ℓ which is only constant. We can tolerate a much larger ℓ here because of the better components available to us.

Given the explicit functions $\mathsf{Challenge}$ and $\mathsf{Response}$ satisfying the properties above, we can now discuss how to use them to find the index j given samples $x \leftarrow_{\mathrm{R}} X$ and $y \leftarrow_{\mathrm{R}} Y$.

Definition 7.2.2. Given a challenge matrix and a list of response matrices, we say that the challenge is *responded* by the response if the challenge matrix is equal to one of the matrices in the response.

Note that throughout construction, every row of the challenge matrix will always be of length clen. In every challenge matrix we will encounter, our analysis will maintain the invariant that either the entire matrix is fixed under the subsources that we are currently working with, or one of the rows of the matrix is uniformly random.

To find the index j:

1. Compute the response $\mathsf{Response}(x, y)$.

2. For every $i \in [t]$, compute a challenge $\mathsf{Challenge}(x_i, y)$.

3. Set r to be the smallest i for which $\mathsf{Challenge}(x_i, y)$ was *not responded* by $\mathsf{Response}(x, y)$.

We remind the reader that we will prove that the disperser works by arguing about *subsources* of the original adversarially chosen sources X, Y. Recall that we are currently working with the subsource $\hat{X} \subset X$ which has the properties guaranteed by Informal Lemma 7.2.1. Using the functions $\mathsf{Challenge}$ and $\mathsf{Response}$, we can then prove the following lemma:

Informal Lemma 7.2.3. There exist low deficiency subsources $X^{\mathsf{good}} \subset \hat{X}, Y^{\mathsf{good}} \subset Y$ s.t. in these subsources $r = j$ with high probability.

Proof Sketch: The lemma will follow from two observations.

Informal Claim 7.2.4. There are subsources $X^{\mathsf{good}} \subset \hat{X}, Y^{\mathsf{good}} \subset Y$ in which for every $i < j$, $\mathsf{Challenge}(X_i^{\mathsf{good}}, Y^{\mathsf{good}})$ is responded by $\mathsf{Response}(X^{\mathsf{good}}, Y^{\mathsf{good}})$ with probability 1. Furthermore X^{good} is a block source (with roughly the same entropy as X) and Y^{good} has roughly the same entropy as Y.

Proof Sketch: Note that for $i < j$, $\hat{X}_i$ is fixed to a constant, so $\mathsf{Challenge}(\hat{X}_i, Y)$ is a function only of Y. Since the output length of $\mathsf{Challenge}$ is only clen bits, this implies (by Proposition 2.1.10) that there exists a subsource $\hat{Y} \subset Y$ of deficiency at most $\mathsf{clen} \cdot \mathsf{t}$ such that $\mathsf{Challenge}(\hat{X}_i, \hat{Y})$ is fixed for every $i < j$.

We can then use the **{Hitting matrices}** property of $\mathsf{Response}$ to find smaller subsources $X' \subset \hat{X}, Y' \subset Y$ s.t. there exists an index h_1 for which $\Pr[\mathsf{Challenge}(X'_1, Y') = \mathsf{Response}(X', Y')_{h_i}] = 1$. Repeating this, we eventually get subsources $X^{\mathsf{good}} \subset \hat{X}, Y^{\mathsf{good}} \subset Y$ s.t. for every $i < j$, there exists an index h_i such that s.t. $\Pr[\mathsf{Challenge}(X_i^{\mathsf{good}}, Y^{\mathsf{good}}) = \mathsf{Response}(X^{\mathsf{good}}, Y^{\mathsf{good}})_{h_i}] = 1$, i.e., the challenge of every part of the source before the jth part is responded with probability 1 in these subsources.

The fact that X^{good} remains a block source follows from Corollary 2.1.17.

□

Informal Claim 7.2.5. $\mathsf{Challenge}(X_j^{\mathsf{good}}, Y^{\mathsf{good}})$ is not responded by $\mathsf{Response}(X^{\mathsf{good}}, Y^{\mathsf{good}})$ with high probability.

Proof Sketch: The argument will use the union bound over ℓ events, one for each of the ℓ matrices in the response. We want to ensure that each matrix in the response is avoided by the challenge. Consider the ith matrix in the response $\mathsf{Response}(X^{\mathsf{good}}, Y^{\mathsf{good}})_i$. By the **{Fixed matrices on low deficiency subsources}** property of $\mathsf{Response}$, we know that $X^{\mathsf{good}}, Y^{\mathsf{good}}$ is a convex combination of independent sources in which the ith matrix is fixed to a constant. For every element of this convex combination, the probability that the challenge is equal to the ith response is extremely small by the property that the output of $\mathsf{Challenge}$ has high min-entropy. □

Step 3: Computing the output of the disperser The output of the disperser is then just $\mathsf{BExt}(x_{[r]}, x, y)$. To show that our algorithm outputs a distribution with large support, first note that $\mathsf{BExt}(X^{\mathsf{good}}_{[r]}, X^{\mathsf{good}}, Y^{\mathsf{good}})$ is a subsource of $\mathsf{BExt}(X_{[r]}, X, Y)$. Thus it is sufficient to show that that $\mathsf{BExt}(X^{\mathsf{good}}_{[r]}, X^{\mathsf{good}}, Y^{\mathsf{good}})$ has a large support. However, by our choice of r, $r = j$ with high probability in $X^{\mathsf{good}}, Y^{\mathsf{good}}$. Thus $\mathsf{BExt}(X^{\mathsf{good}}_{[r]}, X^{\mathsf{good}}, Y^{\mathsf{good}})$ is statistically close to $\mathsf{BExt}(X^{\mathsf{good}}_{[j]}, X^{\mathsf{good}}, Y^{\mathsf{good}})$ and hence is statistically close to being uniform.

□

The Challenge-Response Mechanism in Our Application

Let us summarize how the challenge-response mechanism was used for linear min-entropy. The first step is to show that in any general source there is a small deficiency subsource which has some "nice structure". Intuitively, if the additional structure (in the last case the index j) was given to the construction, it would be easy to construct a disperser. The second step is to define a procedure (the challenge-response mechanism) which is able to "find" the additional structure with high probability, at least when run on some subsource of the good structured subsource. Thus, on the small subsource it is easy to construct a disperser. Since the disperser outputs two different values on the small subsource, it definitely does the same on the original source.

Now we discuss our disperser construction. In this discussion we will often be vague about the settings of parameters, but will give pointers into the actual proofs where things have been formalized.

There are several obstacles to adapting the challenge-response mechanism as used above to handle the case of min-entropy $k = n^{o(1)}$, which is what we achieve in this chapter. Even the first step of the previous approach is problematic when the min-entropy k is less than $\sqrt{n}$. There we found a subsource of X which was block source. Then we fixed the leading bits of the source to get a subsource which has a leading part which is fixed (no entropy), followed by a part with significant (medium) entropy, followed by the rest of the source which contains entropy even conditioned on the medium part.

When $k < \sqrt{n}$, on the one hand, to ensure that a single part of the source X_i cannot contain all the entropy of the source (which would make the above approach fail), we will have to make each part be smaller than $\sqrt{n}$ bits. On the other hand, to ensure that some part of the source contains at least one bit of min-entropy, we will have to ensure that there are at most $\sqrt{n}$ parts, otherwise our construction will fail for the situation in which each part of the source contains $k/\sqrt{n}$ bits of entropy. These two constraints clearly cannot be resolved simultaneously. Thus it seems like there is no simple deterministic way to partition the source in a way which nicely splits the entropy of the source.

The fix for this problem is to use recursion. We will consider parts of very large size (say

$n^{0.9}$), so that the parts may contain all the entropy of the source. We will then develop a finer grained challenge-response mechanism that we can use to handle three levels of entropy differently: low, medium or high, for each part of the source. If we encounter a part of the source that has low entropy, as before we can fix it and ensure that our algorithm correctly identifies it as a block with low entropy. If we encounter a part which has a medium level of entropy, we can use the fact that this gives a way to partition the source into a block source to produce a bit which is both 0 and 1 with positive probability. We will explain how we achieve this shortly. We note that here our situation is more complicated than [BKS+05] as we do not have an extractor that can work with a block source with only two blocks for entropy below $\sqrt{n}$. Finally, if we encounter a part of the source which has a high entropy, then this part of the source is *condensed*, i.e., its entropy rate is significantly larger than that of the original source. Following previous works on seeded extractors, in this case we run the construction recursively on that part of the source (and the other source Y). The point is that we cannot continue these recursive calls indefinitely. After a certain number of such recursive calls, the source that we are working with will have to have such a high entropy rate that it *must* contain a part with a medium level of entropy.

Although this recursive description captures the intuition of our construction, to make the analysis of our algorithm cleaner, we open up the recursion to describe the construction and do the analysis.

Now let us give a more concrete description of our algorithm. Let $\mathsf{C}(\delta)$ be the number of blocks the extractor BExt of Theorem 4.5.1 requires for entropy $k = n^{\delta}$ and let t be some parameter to be specified later (think of t as a very small power of k).

We define a degree-t tree with depth $\log n / \log \mathsf{t} < \log n$ tree $\mathcal{T}_{n,\mathsf{t}}$ that we call the n, t *partition tree*. The nodes of $\mathcal{T}_{n,\mathsf{t}}$ are subintervals of $[1, n]$ defined in the following way:

1. The root of the tree is the interval $[1, n]$.
2. If a node v is identified with the interval $[a, b]$ of length greater than $k^{1/3}$, we let $v_1, \ldots, v_{\mathsf{t}}$ denote the t consecutive disjoint length-$|v|/\mathsf{t}$ subintervals of v. That is, $v_i = [a + \frac{b-a}{t}(i-1), a + \frac{b-a}{t} i]$. We let the i^{th} child of v be v_i.

For a string $x \in \{0,1\}^n$ and a set $S \subseteq [1, n]$ we'll denote by x_S the projection of x onto the coordinates of S. If v is a node in $\mathcal{T}_{n,\mathsf{t}}$ then x_v denotes the projection of x onto the interval v.

Step 1 of analysis In analogy with our discussion for the case of linear min-entropy, we can show that any source X with min-entropy k contains a very nice structured low deficiency subsource $\hat{X}$. We will show that there is a vertex v^{b} in the tree s.t.:

- Every bit of $\hat{X}$ that precedes the bits in v^{b} is fixed.
- There are C children $i_1, \ldots, i_{\mathsf{C}}$ of v^{b} s.t. $\hat{X}_{i_1}, \hat{X}_{i_2}, \ldots, \hat{X}_{i_{\mathsf{C}}}$ is a C-block source with entropy at least $\sqrt{k}$ in each block (even conditioned on previous blocks).
- There is an ancestor v^{med} of v^{b} such that $\hat{X}_{v^{\mathsf{med}}}, \hat{X}$ is a block source with $k^{0.9}$ entropy in each block.

These three properties are captured in Figure 7.1.

This is done formally in **Step 1** of the analysis.

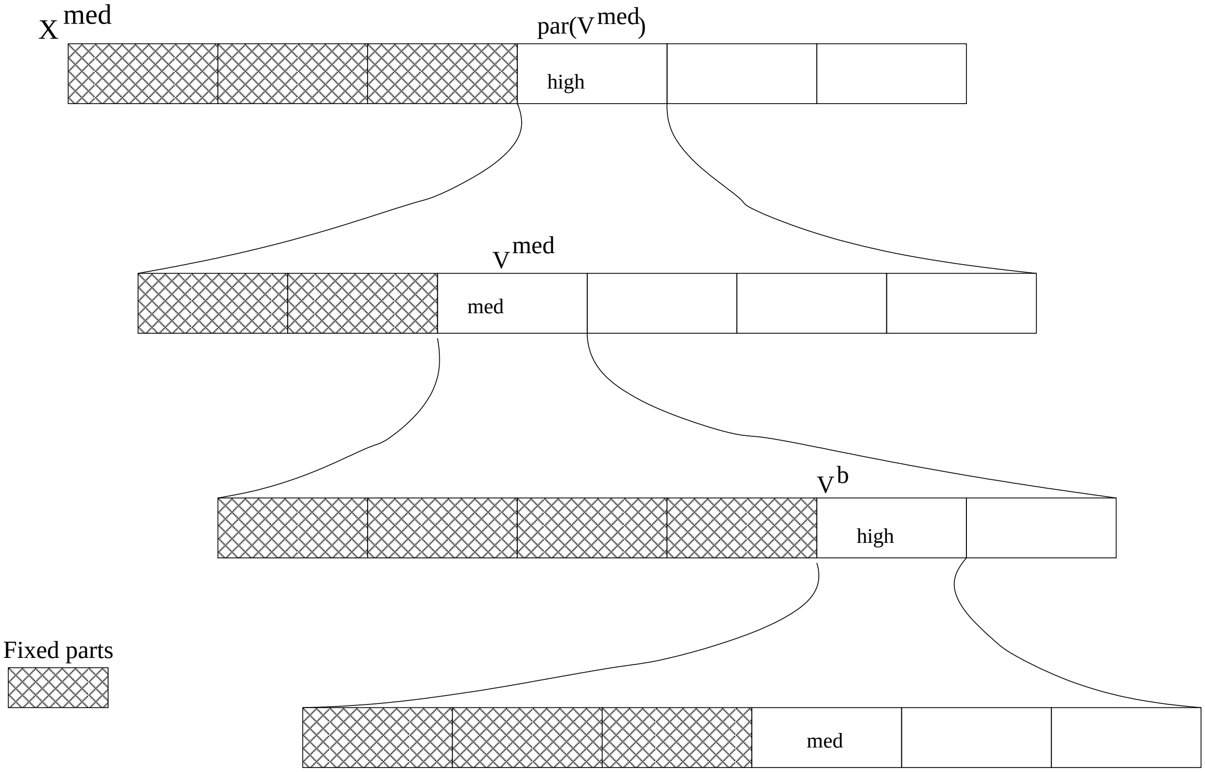

Figure 7.1: Finding two related medium parts in X^{med}

As in the case of linear min-entropy, we would be in great shape if we were given $v^{\mathsf{b}}, v^{\mathsf{med}}, i_1, \dots, i_{\mathsf{C}}$. Of course we don't know these and even worse, this time we will not even be able to identify all of these with high probability in a subsource. Another obstacle to adapting the construction for linear min-entropy to the case of $k = n^{o(1)}$ is that we don't have a simple replacement for the function Challenge that we had for the case of linear min-entropy. However we will be able to use the components that are available to us to compute challenge matrices which are still useful.

The construction will proceed as follows:

1. For every vertex v of the tree, we will compute a small $\mathsf{nrows} \times \mathsf{clen}$ challenge matrix $\mathsf{Challenge}(x_v, y)$ of size $\mathsf{len} = \mathsf{nrows} \cdot \mathsf{clen}$, that is a function only of the bits that correspond to that vertex in x and all of y.
2. For every vertex v of the tree, we will associate a response $\mathsf{Response}(x_v, y)$ which is interpreted as a list of $\mathrm{poly}(n)$ matrices each of size $\mathsf{len} = \mathsf{nrows} \cdot \mathsf{clen}$.

For every vertex v in the tree, we will call the set of vertices whose intervals lie strictly to the left of v (i.e., the interval does not intersect v and lies to the left of v), and whose parent is an ancestor of v, the *left family* of v. In **Step 2** of the formal analysis, we will find low deficiency subsources $X^{\mathsf{good}} \subset \hat{X}, Y^{\mathsf{good}} \subset Y$ s.t. for every vertex v which is in the left family of v^{b}, $\mathsf{Challenge}(X_v^{\mathsf{good}}, Y^{\mathsf{good}})$ is a fixed matrix that occurs in $\mathsf{Response}(X_{\mathsf{par}(v)}^{\mathsf{good}}, Y^{\mathsf{good}})$ with probability 1.

In **Step 3** of the formal analysis, we will show that for every vertex v which lies on the path from v^{b} to the root $\mathsf{Challenge}(X_v^{\mathsf{good}}, Y^{\mathsf{good}})$ is statistically close to being somewhere random. For technical reasons we will actually need a property which is stronger than this. We will actually show that for every vertex v which lies on the path from v^{b} to the root and *all low deficiency subsources* $X' \subset X^{\mathsf{good}}, Y' \subset Y^{\mathsf{good}}$, $\mathsf{Challenge}(X'_v, Y')$ is statistically close to being somewhere random.

At this point we will have made a lot of progress in the construction and analysis. We have found subsources $X^{\mathsf{good}}, Y^{\mathsf{good}}$ s.t. the challenges for all the vertices that occur to the left of the path to v^{b} have been fixed. Moreover the challenges for vertices on this good path have high min-entropy, even if we move to *any* subsources of small deficiency X', Y'. In some sense we will have identified the good path that goes to v^{b} in these subsources, though we still don't know where $v^{\mathsf{b}}, v^{\mathsf{med}}$ are on this path. From here we will need to do only a little more work to compute the output of the disperser.

Now let us describe how we compute the challenges and ensure the properties of $X^{\mathsf{good}}, Y^{\mathsf{good}}$ that we discussed above more concretely. We will need the following components:

1. To generate the challenges, we will need a polynomial time computable function $\mathsf{BExt} : (\{0,1\}^n)^{\mathsf{C}} \times \{0,1\}^n \to \{0,1\}^{\mathsf{clen}}$ that is an extractor for a $(\mathsf{C}, \sqrt{k})$ block source and an independent $\sqrt{k}$ source. Here think of clen as roughly $k^{0.9}$.

2. The second component is exactly the same as the second component from the case of linear min-entropy and will be used to generate the responses. We need a polynomial time computable function $\mathsf{Response} : \{0,1\}^n \times \{0,1\}^n \to (\{0,1\}^{\mathsf{len}})^{\ell}$ (the output is interpreted as a list of ℓ $\mathsf{nrows} \times \mathsf{clen}$ matrices) with the property that:

   Few outputs $\ell = \text{poly}(n)$.

   Hitting matrices Given $\hat{X}, \hat{Y}$, independent sources with min-entropy $\sqrt{k}$ each and any fixed $\mathsf{nrows} \times \mathsf{clen}$ matrix c, there there exists i and low deficiency subsources $X' \subset \hat{X}, Y' \subset \hat{Y}$ such that in these subsources $\mathsf{Response}(X', Y')_i = c$ with probability 1.

   Fixed matrices on low deficiency subsources Given any independent sources $\hat{X}, \hat{Y}$, and an index i, $(\hat{X}, \hat{Y})$ is a convex combination of low deficiency independent sources such that for every element (X', Y') of the combination, $\mathsf{Response}(X', Y')_i$ is fixed to a constant.

   As before, we will use the function SE promised by Theorem 7.1.1 for this component.

We define for every node v of the tree a relatively small challenge matrix $\mathsf{Challenge}(x_v, y)$ with nrows rows of length clen each. We will set up the size of these challenge matrices as roughly $\mathsf{len} = k^{0.9}$.

Let $x_{v_1}, \ldots, x_{v_t}$ be the division of x_v to t sub-parts. Then, we let $\mathsf{Challenge}(x_v, y)$ contain one row that is equal to $\mathsf{BExt}(x_{v_{i_1}}, \ldots, x_{v_{i_{\mathsf{C}}}}, y)$ for every possible C-tuple $1 \le i_1 < i_2 < \cdots < i_{\mathsf{C}} \le t$. If v is a leaf then $\mathsf{Challenge}(x_v, y)$ has no other rows and we will pad the matrix with 0's to make it of size $\mathsf{nrows} \cdot \mathsf{clen}$. If v is a non-leaf then we let $\mathsf{Challenge}(x_{v_1}, y), \ldots, \mathsf{Challenge}(x_{v_t}, y)$ be the challenges of all the children of v in the tree. We will append the rows of $\mathsf{Challenge}(x_{v_i}, y)$ to $\mathsf{Challenge}(x_v, y)$ where i is the smallest index such that $\mathsf{Challenge}(x_{v_i}, y)$ does not equal any of the matrices in $\mathsf{Response}(x_v, y)$. Again, if the matrix we obtain contains fewer than nrows rows, we pad it with 0s to ensure that it is of the right size.

Note that in this way every challenge $\mathsf{Challenge}(x_v, y)$ is indeed only a function of the bits in x_v, y. This will be crucial for our analysis.

Step 2 of analysis: ensuring that challenges are responded in left family The following claim is proved in **Step 2** of the analysis (Claim 7.2.24).

Informal Claim 7.2.6 (Left family challenges are responded)**.** There are subsources $X^{\mathsf{good}} \subset \hat{X}, Y^{\mathsf{good}} \subset Y$ in which for every vertex w to the left of v^{b} whose parent $\mathsf{par}(w)$ lies on the path from v^{b} to the root, $\mathsf{Challenge}(X^{\mathsf{good}}_w, Y^{\mathsf{good}})$ is responded by $\mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(w)}, Y^{\mathsf{good}})$ with probability 1.

Proof Sketch: Note that for w which is to the left of v^{b}, $\hat{X}_w$ is fixed to a constant, so $\mathsf{Challenge}(\hat{X}_w, Y)$ is a function only of Y. Since the output length of $\mathsf{Challenge}$ is only len bits, this implies (by Proposition 2.1.10) that there exists a subsource $\hat{Y} \subset Y$ of deficiency at most $\mathsf{len} \cdot \mathsf{t} \log n$ such that $\mathsf{Challenge}(\hat{X}_w, \hat{Y})$ is fixed for every such w. Then, since $\hat{X}_v, \hat{Y}$ are still high entropy sources for every v on the path from v^{b} to the root, we can repeatedly use the **{Hitting matrices}** property of $\mathsf{Response}$ to find smaller subsources $X^{\mathsf{good}} \subset \hat{X}, Y^{\mathsf{good}} \subset \hat{Y}$ s.t. for every w to the left of v, $\exists l$ s.t. $\Pr[\mathsf{Challenge}(\hat{X}_w, \hat{Y}) = \mathsf{Response}(\hat{X}_{\mathsf{par}(w)}, \hat{Y})_l] = 1$. □

Step 3 of analysis: ensuring that challenges along the good path are somewhere random We argue that the challenges along the good path are statistically close to being somewhere random in $X^{\mathsf{good}}, Y^{\mathsf{good}}$. This is done formally in **Step 3** in Lemma 7.2.25. The intuition for this is that first the challenge associated with the vertex v^{b} is somewhere random since v^{b} has children that form a block source. We will then show that with high probability this challenge of v^{b} appears in the challenge matrix of every ancestor of v^{b}.

Informal Claim 7.2.7 (Challenges along path to v^{b} are somewhere random)**.** For all low deficiency subsources $X' \subset X^{\mathsf{good}}, Y' \subset Y^{\mathsf{good}}$ and any vertex v that's on the path from v^{b} to the root, $\mathsf{Challenge}(X'_v, Y')$ is statistically close to being somewhere random.

Proof Sketch: We will prove this by induction on the distance of the vertex v from v^{b} on the path. When $v = v^{\mathsf{b}}$, note that $\mathsf{Challenge}(X'_{v^{\mathsf{b}}}, Y')$ contains $\mathsf{BExt}(x_{v_{i_1}}, \ldots, x_{v_{i_u}}, y)$ for every C-tuple of children $v_{i_1}, \ldots, v_{i_{\mathsf{C}}}$ of v^{b}. By the guarantee on $\hat{X}$, we know that there exist $i_1, \ldots, i_{\mathsf{C}}$ s.t. $\hat{X}_{v_{i_1}}, \ldots, \hat{X}_{v_{i_{\mathsf{C}}}}$ is a C-block source. Since X' is a low deficiency subsource of $\hat{X}$, $\hat{X}'_{v_{i_1}}, \ldots, \hat{X}'_{v_{i_{\mathsf{C}}}}$ must also be close to a C block source by Corollary 2.1.17. Thus we get that $\mathsf{Challenge}(X'_{v^{\mathsf{b}}}, Y')$ is statistically close to somewhere random.

To do the inductive step we show that $\mathsf{Challenge}(X'_{\mathsf{par}(v)}, Y')$ is close to being somewhere random given that $\mathsf{Challenge}(X''_v, Y'')$ is somewhere random for even smaller subsources $X'' \subset X', Y'' \subset Y'$.

The argument will use the union bound over ℓ events, one for each of the ℓ strings in the response. We want to ensure that each string in the response is avoided by the challenge. Consider the ith string in the response $\mathsf{Response}(X'_{\mathsf{par}(v)}, Y')_i$. By the **{Fixed matrices on low deficiency subsources}** property of $\mathsf{Response}$, we know that X', Y' is a convex combination of independent sources in which the ith string is fixed to a constant.

Now every element of this convex combination X'', Y'' is a subsource of the original sources, the probability that $\mathsf{Challenge}(X''_v, Y'')$ is equal to the ith response is extremely small by the property that the output of $\mathsf{Challenge}(X''_v, Y'')$ has high min-entropy. Thus with high probability $\mathsf{Challenge}(X'_{\mathsf{par}(v)}, Y')$ contains $\mathsf{Challenge}(X'_v, Y')$ as a substring. This implies that $\mathsf{Challenge}(X'_{\mathsf{par}(v)}, Y')$ is statistically close to being somewhere random. □

Step 4 of analysis: ensuring that the disperser outputs both 0 and 1 The output for our disperser is computed in a way that is very different from what was done for the case of linear min-entropy. The analysis above included two kinds of tricks:

- When we encountered a part of the source which had a low amount of entropy, we went to a subsource where the part was fixed and the corresponding challenge was responded with probability 1.
- When we encountered a part of the source which had a high level of entropy, we went to a subsource where the corresponding challenge is not responded with high probability

The intuition for our disperser is that if we encounter a part of source (such as v^{med} above) which both has high min-entropy and such that fixing that part of the source still leaves enough entropy in the rest of the source, we can ensure that the challenge is both responded and not responded with significant probability. We will elaborate on how to do this later on. This is very helpful as it gives us a way to output two different values! By outputting "0" in case the challenge is responded and "1" in case it is not we obtain a disperser. Now let us be more concrete.

Definition 7.2.8. Given two $\mathsf{nrows} \times \mathsf{clen}$ matrices and an integer $1 \le q \le \mathsf{clen}$, we say that one matrix is q-responded by the other if the first q columns of both matrices are equal.

The first observation is the following claim which is proved formally in **Step 4** (Lemma 7.2.26). The claim will be used with $q \ll \mathsf{clen}, \mathsf{len}$.

Below we use the symbol $\lesssim$ to denote an inequality that is only approximate in the sense that in the formal analysis there are small error terms (which may be ignored for the sake of intuition) that show up in the expressions.

Informal Claim 7.2.9. For every vertex v on the path from v^{b} to the root,

$$\Pr[\mathsf{Challenge}(X_v^{\mathsf{good}}, Y^{\mathsf{good}}) \text{ is } q\text{-responded by } \mathsf{Response}(X_{\mathsf{par}(v)}^{\mathsf{good}}, Y^{\mathsf{good}})] \lesssim 2^{-q}$$

Proof Sketch: As before, we will use the **{Fixed matrices on low deficiency subsources}** property of $\mathsf{Response}$ and the fact that $\mathsf{Challenge}(X'_v, Y')$ is somewhere random for any low deficiency subsources $X' \subset X^{\mathsf{good}}, Y' \subset Y^{\mathsf{good}}$ to argue that the probability that for every index q,

$$\Pr[\mathsf{Challenge}(X_v^{\mathsf{good}}, Y^{\mathsf{good}}) \text{ is } q\text{-responded by } \mathsf{Response}(X_{\mathsf{par}(v)}^{\mathsf{good}}, Y^{\mathsf{good}})_q] \lesssim 2^{-q}$$

Then we just apply a union bound over the $\mathrm{poly}(n)$ response strings to get the claim. □

Next we observe that for the vertex v^{med}, its challenge is responded with a probability that behaves very nicely. In particular, note that we get that the challenge is both responded and not responded with noticeable probability. This is Lemma 7.2.28 in the formal analysis.

Informal Claim 7.2.10.

$$2^{-q \cdot \mathsf{nrows}} \lesssim \Pr[\mathsf{Challenge}(X_{v^{\mathsf{med}}}^{\mathsf{good}}, Y^{\mathsf{good}}) \text{ is } q\text{-responded by } \mathsf{Response}(X_{\mathsf{par}(v^{\mathsf{med}})}^{\mathsf{good}}, Y^{\mathsf{good}})] \lesssim 2^{-q}$$

Proof Sketch: The idea is that $X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}$ is a convex combination of sources $X'_{\mathsf{par}(v^{\mathsf{med}})}$ in which $X'_{\mathsf{par}(v^{\mathsf{med}})}$ is fixed, but X' still has a significant amount of entropy. Thus we are in the situation where we proved Claim 7.2.6. We can then show that X', Y^{good} are a convex combination of sources X'', Y'' s.t. $\mathsf{Challenge}(X''_{v^{\mathsf{med}}}, Y'')$ is fixed to a constant. Thus

$$\Pr[\mathsf{Challenge}(X''_{v^{\mathsf{med}}}, Y'') \text{ is } q\text{-responded by } \mathsf{Response}(X''_{\mathsf{par}(v^{\mathsf{med}})}, Y'') \gtrsim 2^{-q \cdot \mathsf{nrows}}$$

This implies that

$$\Pr[\mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) \text{ is } q\text{-responded by } \mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}, Y^{\mathsf{good}}) \gtrsim 2^{-q \cdot \mathsf{nrows}}$$

The upper bound is just a special case of Claim 7.2.9. □

Given these two claims, here is how we define the output of the disperser:

1. We define a sequence of decreasing challenge lengths: $\mathsf{clen} \gg \mathsf{clen}_{1,0} \gg \mathsf{clen}_{1,1} \gg \mathsf{clen}_{1,2} \gg \mathsf{clen}_{2,0} \gg \mathsf{clen}_{2,1} \gg \mathsf{clen}_{2,2} \gg \mathsf{clen}_{3,0} \cdots$.
2. If v is not a leaf, let $v_1, \ldots, v_{\mathsf{t}}$ be v's t children. Let q be the depth of v. If for every i $\mathsf{Challenge}(x_{v_i}, y)$ is $\mathsf{clen}_{q,0}$-responded by $\mathsf{Response}(x_{v_i}, y)$, set $\mathsf{val}(x_v, y) = 0$, else let i_0 be the smallest i for which this doesn't happen. Then,
   (a) If $\mathsf{Challenge}(x_{v_{i_0}}, y)$ is $\mathsf{clen}_{q,1}$-responded by $\mathsf{Response}(x_v, y)$, set $\mathsf{val}(x_v, y) = 1$.
   (b) Else if $\mathsf{Challenge}(x_{v_{i_0}}, y)$ is $\mathsf{clen}_{q,2}$-responded but not $\mathsf{clen}_{q,1}$-responded by $\mathsf{Response}(x_v, y)$, set $\mathsf{val}(x_v, y) = 0$.
   (c) Else set $\mathsf{val}(x_v, y) = \mathsf{val}(x_{v_{i_0}}, y)$.
3. The disperser outputs $\mathsf{val}(x, y)$.

Let h be the depth of v^{med}. The correctness is then proved by proving two more claims:

Informal Claim 7.2.11. The probability that $\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ differs from $\mathsf{val}(X^{\mathsf{good}}, Y^{\mathsf{good}})$ is bounded by $2^{-\mathsf{clen}_{h,0}}$.

Proof Sketch: In fact, we can argue that with high probability,

$$\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) = \mathsf{val}(X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}, Y^{\mathsf{good}}) = \mathsf{val}(X^{\mathsf{good}}_{\mathsf{par}(\mathsf{par}(v^{\mathsf{med}}))}, Y^{\mathsf{good}}) = \cdots = \mathsf{val}(X^{\mathsf{good}}, Y^{\mathsf{good}})$$

The reason is that by Claim 7.2.9, for any vertex v on the path from v^{med} to the root at depth q,

$$\Pr[\mathsf{val}(X^{\mathsf{good}}_{v}, Y^{\mathsf{good}}) \neq \mathsf{val}(X^{\mathsf{good}}_{\mathsf{par}(v)})] \lesssim 2^{-\mathsf{clen}_{q,0}} \ll 2^{-\mathsf{clen}_{h,0}}$$

Thus, by the union bound, we get that with high probability all of these are in fact equal. □

Next, we will argue that $\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ is both 0 and 1 with significant probability. This will complete the proof, since this will show that $\mathsf{val}(X^{\mathsf{good}}, Y^{\mathsf{good}})$ is both 0 and 1 with significant probability.

Informal Claim 7.2.12.

$$\Pr[\mathsf{val}(X^{\mathsf{good}}_{\upsilon_{\mathsf{med}}}, Y^{\mathsf{good}})] = 1] \gtrsim 2^{-\mathsf{clen}_{h,1}}$$

$$\Pr[\mathsf{val}(X^{\mathsf{good}}_{\upsilon_{\mathsf{med}}}, Y^{\mathsf{good}})] = 0] \gtrsim 2^{-\mathsf{clen}_{h,2}}$$

Proof Sketch: This follows from Claim 7.2.10. The probability that $\mathsf{val}(X^{\mathsf{good}}_{\upsilon_{\mathsf{med}}}, Y^{\mathsf{good}}) = 1$ is lower-bounded by the probability that $\mathsf{Challenge}(X^{\mathsf{good}}_{\upsilon_{\mathsf{med}}}, Y^{\mathsf{good}})$ is $\mathsf{clen}_{h,1}$-responded by $\mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(\upsilon_{\mathsf{med}})}, Y^{\mathsf{good}})$ minus the probability that $\mathsf{Challenge}(X^{\mathsf{good}}_{\upsilon_{\mathsf{med}}}, Y^{\mathsf{good}})$ is $\mathsf{clen}_{h,0}$-responded by $\mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(\upsilon_{\mathsf{med}})}, Y^{\mathsf{good}})$. By Claim 7.2.10, we can ensure that this difference is significantly large.

The argument for the 0 output is very similar. □

These two claims then ensure that overall $\Pr[\mathsf{val}(X^{\mathsf{good}}, Y^{\mathsf{good}}) = 1] \gtrsim 2^{-\mathsf{clen}_{h,1}}$ and $\Pr[\mathsf{val}(X^{\mathsf{good}}, Y^{\mathsf{good}}) = 0] \gtrsim 2^{-\mathsf{clen}_{h,2}}$.

Thus $\mathsf{val}(X, Y) = \{0, 1\}$ as required.

7.2.2 Parameters

Setting the parameters We will first list the various parameters involved in the construction and say how we will set them.

- Let n be the length of the samples from the sources.
- Let k be the entropy of the input sources. Set $k = 2^{\log^{0.9} n}$.
- Let c_1 be the error constant from Theorem 4.5.1.
- Let $\mathsf{C} = O\left(\frac{\log n}{\log k}\right)$ be the number of blocks that the extractor BExt of Theorem 4.5.1 requires to extract from $\sqrt{k}$ entropy. (See Corollary 7.2.14 below for the precise parameters we use BExt for.) Without loss of generality, we assume that $c_1 \gg 1/\mathsf{C}$.
- We use t to denote the branching factor of the tree. We set $\mathsf{t} = n^{1/\mathsf{C}^4}$.
- We use $\mathsf{nrows} = \mathsf{t}^{\mathsf{C}} \log n$ to denote the maximum number of rows in any challenge matrix.
- We use clen to denote the length of every row in a challenge matrix. We set $\mathsf{clen} = n^{1/\mathsf{C}^2}$.
- We use $\mathsf{len} = \mathsf{nrows} \cdot \mathsf{clen}$ to denote the total size of the challenge matrices.
- We use $\mathsf{clen}_{q,r}$ to denote smaller challenge lengths and analogously define $\mathsf{len}_{q,r} = \mathsf{nrows}\cdot\mathsf{clen}_{q,r}$. We set $\mathsf{clen}_{q,r} = n^{\frac{1}{(3q+r)\mathsf{C}^2}}$.

Constraints needed in analysis Here are the constraints that the above parameters need to satisfy in the analysis.

- $\mathsf{t}^{1/\mathsf{C}^4} \geq 20\mathsf{C}$, used in the proofs of Lemma 7.2.21 and Lemma 7.2.22.
- $\frac{k}{(10\mathsf{t}^2\cdot\mathsf{C})^2} \geq k^{0.9}$, used at the end of Step 1 in the analysis.
- $\mathsf{clen}^3 = o(k^{0.9})$, use at the end of Step 1 in the analysis and in the proof of Lemma 7.2.25.

Name	Description	Restrictions	Notes
n	Input length		
k	Entropy	k	Assume $k \geq 2^{\log^{0.9} n}$
C	Number of blocks for BExt	$O(\log n/\log k)$	We always invoke BExt with entropy $\geq \sqrt{k}$
t	Degree of partition tree	$\mathsf{t} = n^{1/\mathsf{C}^4}$	
c_1	Error parameter of BExt		Inherited from BExt Corollary 7.2.14.
nrows	No. of rows in challenges and responses	$\mathsf{nrows} \leq (\log n)\mathsf{t}^{\mathsf{C}}$	
clen	Length of each row in challenges and responses	$\mathsf{clen} = n^{1/\mathsf{C}^2}$	
$\mathsf{clen}_{q,r}$	Shorter challenge lengths	$\mathsf{clen}_{q,r} = n^{\frac{1}{(3q+r)\mathsf{C}^2}}$	

Table 7.1: Parameters used in the construction

- $\mathsf{clen} = o(k^{c_1})$, used in the proof of Lemma 7.2.27.
- $\mathsf{t} \cdot \mathsf{len} \cdot \log n = o(\mathsf{clen}^{2.1}) \Leftrightarrow \mathsf{t}^{\mathsf{C}+1} \cdot \log^2 n = o(\mathsf{clen}^{1.1})$, used at the end of Step 2 in the analysis.
- For any positive integers q, r, $\mathsf{nrows} = o(\mathsf{clen}_{q,r}/\mathsf{clen}_{q,r+1})$ and $\mathsf{nrows} = o(\mathsf{clen}_{q,r+2}/\mathsf{clen}_{q+1,r})$, used in the proof of Lemma 7.2.29.

7.2.3 Formal Construction

Definition 7.2.13. Given a challenge string Challenge interpreted as a $d \times \mathsf{len}$ boolean matrix with $d \leq \mathsf{nrows}$, a response string Response interpreted as a $\mathsf{nrows} \times \mathsf{len}$ boolean matrix, and a parameter q, we say that Challenge is *q-responded* by Response, if the $d \times q$ submatrix of Challenge obtained by taking the first q bits from each row is equal to the $d \times q$ submatrix of Response obtained by taking the first q bits each from the first d rows of Response.

Components

Block extractor We'll use the following corollary of Theorem 4.5.1:

> **Corollary 7.2.14** (Block Extractor)**.** *There is a constant c_1 s.t. if the parameters C, n, k are as above, there is a polynomial-time computable function $\mathsf{BExt} : \{0,1\}^{\mathsf{C}n} \times \{0,1\}^n \to \{0,1\}^{\mathsf{out}}$ satisfying:*
>
> *For every every independent sources $X \in \{0,1\}^{\mathsf{C}n}$ and $Y \in \{0,1\}^n$ with $H_\infty(Y) \geq \sqrt{k}$ and $X = X_1, \cdots, X_{\mathsf{C}}$ a $\sqrt{k}$ block source,*[4]
>
> $$\left|\mathsf{BExt}(X, Y) - U_{\mathsf{clen}}\right| < 2^{-k^{c_1}}$$

[4] That is, for every $i < \mathsf{C}$ and $x_1, \ldots, x_i \in \mathsf{Supp}(X_{1,\ldots,i})$, $H_\infty(X_{i+1}|x_1, \ldots, x_i) > 10\mathsf{clen}^5$.

Somewhere extractor with small error We will use the following corollary of Theorem 7.1.1 to generate our responses. We will set up SE to work on strings with entropy $\sqrt{k}$ with output length clen. For every string x of length at most n (if the input is shorter we will pad it to make it long enough), string $y \in \{0,1\}^n$, we define $\mathsf{Response}(x,y)$ to be the list of strings obtained from $\mathsf{SE}(x,y)$, by interpreting each row of the output of $\mathsf{SE}(x,y)$ as an $\mathsf{nrows} \times \mathsf{clen}$ boolean matrix.

Corollary 7.2.15 (Somewhere Extractor to generate Responses)**.** *For every n, k, len that satisfy the constraints above, there is a polynomial time computable function $\mathsf{Response} : (\{0,1\}^n)^2 \to (\{0,1\}^{\mathsf{len}})^\ell$ (here the output is interpreted as a $\mathsf{nrows} \times \mathsf{clen}$ matrix) with the property that for any two $(n, \sqrt{k})$ sources X, Y,*

Few outputs $\ell = \mathrm{poly}(n)$.

Small error $\mathsf{Response}(X,Y)$ *is $2^{-10\mathsf{len}}$-close to a convex combination of somewhere random distributions and this property is strong with respect to both X and Y. Formally:*

$$\Pr_{y \leftarrow_R Y}[\mathsf{Response}(X,y) \textit{ is } 2^{-10\mathsf{len}}\textit{-close to being SR }] > 1 - 2^{-10\mathsf{len}}$$

Hitting matrices *Let c be any fixed $\mathsf{nrows} \times \mathsf{clen}$ matrix. Then there are deficiency $2\mathsf{len}$ subsources $\hat{X} \subset X, \hat{Y} \subset Y$ such that $\Pr[c \in \mathsf{SE}(\hat{X}, \hat{Y})] = 1$.*

Fixed matrices on low deficiency subsources *Given any particular index i, there are $20\mathsf{len}$ deficiency subsources $\hat{X} \subset X, \hat{Y} \subset Y$ such that $\mathsf{Response}(\hat{X}, \hat{Y})_i$ is a fixed matrix. Further, X, Y is $2^{-10\mathsf{len}}$-close to a convex combination of subsources such that for every $\hat{X}, \hat{Y}$ in the combination,*

- *$\hat{X}, \hat{Y}$ are independent.*
- *$\mathsf{Response}(\hat{X}, \hat{Y})_i$ is constant.*
- *$\hat{X}, \hat{Y}$ are of deficiency at most $20\mathsf{len}$.*

The Tree of Parts

We define a degree-t with depth $\log n / \log \mathsf{t} < \log n$ tree $\mathcal{T}_{n,\mathsf{t}}$ that we call the n, t *partition tree*. The nodes of $\mathcal{T}_{n,\mathsf{t}}$ are subintervals of $[1,n]$ defined in the following way:

1. The root of the tree is the interval $[1,n]$.

2. If a node v is identified with the interval $[a,b]$ of length greater than $k^{1/3}$, we let $v_1, \ldots, v_{\mathsf{t}}$ denote the t consecutive disjoint length-$|v|/\mathsf{t}$ subintervals of v. That is, $v_i = [a + \frac{b-a}{t}(i-1), a + \frac{b-a}{t} i]$. We let the i^{th} child of v be v_i.

For a string $x \in \{0,1\}^n$ and a set $S \subseteq [1,n]$ we'll denote by x_S the projection of x onto the coordinates of S. If v is a node in $\mathcal{T}_{n,\mathsf{t}}$ then x_v denotes the projection of x onto the interval v.

Operation of the algorithm Disp

Algorithm 7.2.16 ($\mathsf{Disp}(x, y)$)**.**

Input: $x, y \in \{0,1\}^n$.
Output: 1 bit.

1. On inputs $x, y \in \{0,1\}^n$, the algorithm Disp, working from the leaves upwards, will define for each node v in the tree $\mathcal{T}_{n,\mathsf{t}}$ a boolean challenge matrix ($\mathsf{Challenge}(x_v, y)$) with at most nrows rows, each of length clen in the following way:
    - (a) If v is a leaf then $\mathsf{Challenge}(x_v, y)$ is the matrix with a single all 0s row.
    - (b) If v is not a leaf then $\mathsf{Challenge}(x_v, y)$ is computed as follows:
        - i. For each C-tuple $1 \leq i_1 < i_2 < \cdots < i_{\mathsf{C}} \leq \mathsf{t}$ let $S = v_{i_1} \cup v_{i_2} \cup \cdots \cup v_{i_{\mathsf{C}}}$ and append the row $\mathsf{BExt}(x_S, y)$ to the matrix $\mathsf{Challenge}(x_v, y)$.
        - ii. Let $v_1, \ldots, v_{\mathsf{t}}$ be v's t children. If there exists an i such that $\mathsf{Challenge}(x_{v_i}, y)$ is not clen-responded by $\mathsf{Response}(x_v, y)$, let i_0 be the smallest such i and append all the rows of $\mathsf{Challenge}(x_{v_{i_0}}, y)$ to $\mathsf{Challenge}(x_v, y)$.
2. Next Disp will make a second pass on the tree, again working from the leaves upwards. This time it will define for each node v in the tree $\mathcal{T}_{n,\mathsf{t}}$ a bit $\mathsf{val}(x_v, y)$ in the following way:
    - (a) If v is a leaf then $\mathsf{val}(x_v, y) = 0$.
    - (b) If v is not a leaf, let $v_1, \ldots, v_{\mathsf{t}}$ be v's t children. Let q be the depth of v. If for every i $\mathsf{Challenge}(x_{v_i}, y)$ is $\mathsf{clen}_{q,0}$-responded by $\mathsf{Response}(x_{v_i}, y)$, set $\mathsf{val}(x_v, y) = 0$, else let i_0 be the smallest i for which this doesn't happen. Then,
        - i. If $\mathsf{Challenge}(x_{v_{i_0}}, y)$ is $\mathsf{clen}_{q,1}$-responded by $\mathsf{Response}(x_v, y)$, set $\mathsf{val}(x_v, y) = 1$.
        - ii. Else if $\mathsf{Challenge}(x_{v_{i_0}}, y)$ is $\mathsf{clen}_{q,2}$-responded but not $\mathsf{clen}_{q,1}$-responded by $\mathsf{Response}(x_v, y)$, set $\mathsf{val}(x_v, y) = 0$.
        - iii. Else set $\mathsf{val}(x_v, y) = \mathsf{val}(x_{v_{i_0}}, y)$.
3. The output of Disp is $\mathsf{val}(x_{[1,n]}, y)$.

7.2.4 Formal Analysis

The analysis proceeds in several steps. In each step we make a restriction on one or both of the input sources. When we're done, we'll get the desired subsources X^{good}, Y^{good}.

Definition 7.2.17 (Path to a vertex)**.** Given a partition tree $\mathcal{T}_{n,\mathsf{t}}$ and a vertex v, let $\mathcal{P}_v$ to denote the path from the vertex v to the root in the tree $\mathcal{T}_{n,\mathsf{t}}$. That is, the set of nodes (including v) on the path from v to the root.

Definition 7.2.18 (Parent of a vertex)**.** Given a partition tree $\mathcal{T}_{n,\mathsf{t}}$ and a vertex v, let $\mathsf{par}(v)$ denote the parent of v.

Definition 7.2.19 (Left family of v)**.** Given a partition tree $\mathcal{T}_{n,\mathsf{t}}$ and a vertex v, let $\mathcal{L}_v$ denote the *left family* of v, i.e., if v is the interval $[c, d]$, define $\mathcal{L}_v = \{[a, b] \in \mathcal{T}_{n,\mathsf{t}} : a \leq c \text{ and } \mathsf{par}(w) \in \mathcal{P}_v\}$.

Note that for every vertex v, $|\mathcal{L}_v| = O(\mathsf{t} \log n)$, since the number of vertices in $\mathcal{P}_v$ is at most $\log n$.

Step 1: Preprocess X

The first step involves only the first source X. We'll restrict X to a subsource X^{med} that will have some attractive properties for us: we will ensure that in X^{med} there are a couple of parts which have entropy but do not have all the entropy of the source. We first prove a general lemma — Lemma 7.2.20 — and then use it to prove Lemma 7.2.21 and Lemma 7.2.22 to show that we obtain the desired subsource X^{med}.

Lemma 7.2.20 (Two-types lemma.)**.** *Let X be a general k source over $\{0, 1\}^n$ divided into t parts $X = X_1, \cdots, X_{\mathsf{t}}$. Let C be some positive integer and let $k' < k$ be such that $(\mathsf{C} + 1)k' + 4t^2 \leq k$. Then, there exists a subsource $X' \subseteq X$ of deficiency at most $d = \mathsf{C}k' + 2\mathsf{t}^2$ that satisfies one of the following properties:*
Either

Somewhere high source — one high part *There exists $i \in [\mathsf{t}]$ such that the first $i - 1$ parts of X' (namely $X'_1, \ldots, X'_{i-1}$) are* constant*, and $H_\infty(X'_i) \geq k'$.*

or

Somewhere block-source — C medium parts *There exist $0 < i_1 < i_2 < \cdots < i_{\mathsf{C}} \leq \mathsf{t}$ such that the first $i_1 - 1$ parts of X' are constant for every $j \in [\mathsf{C}]$, and $X'_{i_1}, X'_{i_2}, \ldots, X'_{i_{\mathsf{C}}}$ is a $(C, k'/t)$ block-source.*

Proof. We let $\tau_1 = 0$, $\tau_2 = k'/\mathsf{t}$, $\tau_3 = k'$ and $\tau_4 = n$ and use Lemma 2.1.18 to reduce X to a deficiency $2\mathsf{t}^2$ source X'' such that for every $i \in [\mathsf{t}]$ and every $x_1, \ldots, x_{i-1} \in \mathsf{Supp}(X''_{1,\ldots,i-1})$, the conditional entropy $H_\infty(X''_i | x_1, \ldots, x_{i-1})$ always falls into the same interval of $[0, k'/\mathsf{t}]$, $[k'/\mathsf{t}, k']$ and $[k', n]$ regardless of the choice $x_1, \ldots, x_i$.

We call parts where this conditional entropy falls into the interval $[0, k'/\mathsf{t})$ low, parts where this entropy falls into the interval $[k'/\mathsf{t}, k')$ medium and parts where it is at least k' high. We divide to two cases:

Case 1: if there are at most $\mathsf{C} - 1$ medium parts before the first high part, we let i be the position of the first high part and fix the first $i - 1$ parts to their most typical values. The conditional entropy X_1 given this prefix is still at least k'. Furthermore, since we fixed at most t low parts and at most C medium parts the overall deficiency is at most $(\mathsf{C} - 1)k' + \mathsf{t}k'/\mathsf{t} = \mathsf{C}k'$.

Case 2: If there are at least C medium parts in the source, we let i be the position of the first medium part and fix the first $i - 1$ parts to their most typical value. All medium parts remain medium conditioned on this prefix and the entropy we lose is at most $\mathsf{t}k'/\mathsf{t} \leq k'$.

□

We'll now use Lemma 7.2.20 to show that we can restrict the input source X to a subsource X^{sb} (for "somewhere block") satisfying some attractive properties:

Lemma 7.2.21. *Let X be a source over $\{0,1\}^n$ with min-entropy k. Let C, t be values satisfying $\mathsf{t}^{1/\mathsf{C}^4} \geq 20\mathsf{C}$. Then, there exists a deficiency $k/10 + 4\mathsf{t}^2 \log n$ subsource X^{sb} of X and a vertex v^{med} of $\mathcal{T}_{n,\mathsf{t}}$ with the following properties:*

- *For every $v \in \mathcal{L}_{v^{\mathsf{med}}}$, X^{sb}_v is fixed to a constant.*
- *The source $X^{\mathsf{sb}}_{\mathsf{par}(v^{\mathsf{med}})}$ is a $(\mathsf{C}, \frac{k}{20\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}})$-somewhere block source.*
- *$X^{\mathsf{sb}}_{v^{\mathsf{med}}}$ is the first block of the block source in $X^{\mathsf{sb}}_{\mathsf{par}(v^{\mathsf{med}})}$.*

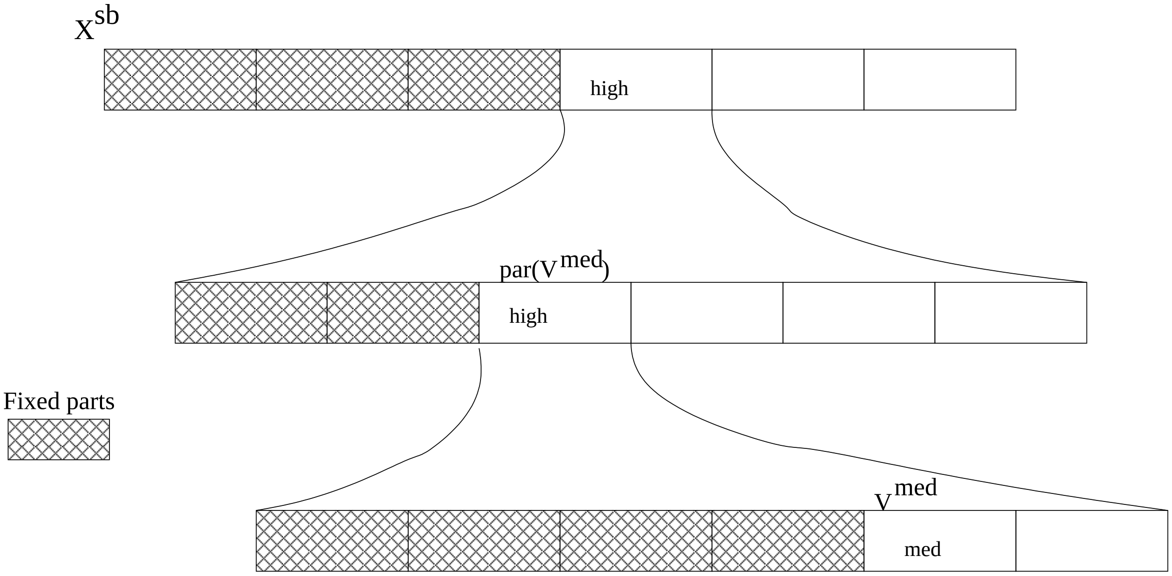

Figure 7.2: Finding a medium part in X^{sb}

Proof. We prove the lemma by induction on $\lceil \log(n/k) \rceil = \lceil \log n - \log k \rceil$. If $n = k$ then this is the uniform distribution and everything is trivial. We invoke Lemma 7.2.20 with parameter $k' = k/(20\mathsf{C})$ to obtain a deficiency $k/20 + 4\mathsf{t}^2$ subsource X' that is either k'-somewhere high or $(\mathsf{C}, k'/\mathsf{t})$-somewhere block source.

If X' is $(\mathsf{C}, k'/\mathsf{t})$-somewhere block source then set $X^{\mathsf{sb}} = X'$, $\mathsf{par}(v^{\mathsf{med}}) = [1, n]$ and v^{med} corresponding to the first part of the block source given by Lemma 7.2.20. Since $k'/\mathsf{t} = k/(20\mathsf{t}\mathsf{C})$ we have that $X^{\mathsf{sb}}, v^{\mathsf{med}}$ satisfy the properties in the conclusion of the lemma.

The second possibility is that X' is a k'-somewhere high source. We let i be the index of the high block of entropy k', and let v_i be the corresponding interval. Note that X'_{v_j} attains some fixed value with probability 1, for all $j < i$. Let $n' = |v_i| = n/\mathsf{t}$. Since $\frac{n'}{k'} = \frac{n}{k}\frac{20\mathsf{C}}{\mathsf{t}} < \frac{n}{4k}$ we have that $\log(n'/k') < \log(n/k) - 2$ and so can assume by the induction hypothesis that the statement holds for the source $Z = X'_{v_i}$. This means that we have a subsource $Z' \subset Z$ of deficiency $k'/10 + 4\mathsf{t}^2 \log n'$ of Z and a node $\mathsf{par}(v^{\mathsf{med}})$ in the tree $\mathcal{T}_{n',\mathsf{t}}$ such that (below we use that $\mathsf{t}^{1/\mathsf{C}^4} \geq 20\mathsf{C}$):

- For every $v \in \mathcal{L}_{v^{\mathsf{med}}}$, Z'_v is fixed to a constant.
- The source $Z'_{\mathsf{par}(v^{\mathsf{med}})}$ is a $(\mathsf{C}, \frac{k'}{20\mathsf{t}\mathsf{C}n'^{1/\mathsf{C}^4}} = \frac{k}{20\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}} \cdot \frac{\mathsf{t}^{1/\mathsf{C}^4}}{20\mathsf{C}} \geq \frac{k}{20\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}})$-somewhere block source.

- $Z'_{v^{\mathsf{med}}}$ is the first block of the block source in $Z'_{\mathsf{par}(v^{\mathsf{med}})}$.

We define X^{sb} to be the natural extension of the subsource Z' to a subsource of X'. Then we see that $X^{\mathsf{sb}} \subset X'$ is of deficiency at most $k'/10 + 4\mathsf{t}^2 \log n'$. Since $\log n' \leq \log n - 1$ and $k'/10 < k/20$, $k'/10 + 4\mathsf{t}^2 \log n' \leq k/20 + 4\mathsf{t}^2(\log n - 1)$. Hence $X^{\mathsf{sb}} \subset X$ is a source of deficiency at most $k/10 + 4\mathsf{t}^2 \log n$. It is clear that X^{sb} and $\mathsf{par}(v^{\mathsf{med}})$ satisfy our requirements. □

Note that by our setting of parameters, the entropy of the medium part promised by the above lemma is actually $\frac{k}{20\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}} = \frac{k}{20\mathsf{t}^2\mathsf{C}}$.

Next we show that by invoking the above lemma twice, we can move to a subsource X^{med} that has even more structure.

Lemma 7.2.22. *Let X be a source over $\{0,1\}^n$ with min-entropy k. Let C, t be as above. Then, there exists a deficiency $k/5 + 8\mathsf{t}^2 \log n$ subsource X^{med} of X and three vertices $\mathsf{par}(v^{\mathsf{med}})$, v^{med} and $v^{\mathsf{b}} = [a, b]$ of $\mathcal{T}_{n,\mathsf{t}}$ with the following properties:*

- *v^{med} is an ancestor of v^{b}.*
- *The source $X^{\mathsf{med}}_{\mathsf{par}(v^{\mathsf{med}})}$ is a $(\mathsf{C}, \frac{k}{40\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}})$-somewhere block source, and $X^{\mathsf{med}}_{v^{\mathsf{med}}}$ is the first medium block in this source.*
- *The source $X^{\mathsf{med}}_{v^{\mathsf{b}}}$ is a $(\mathsf{C}, \frac{k}{(20\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4})^2})$-somewhere block source.*
- *There is a value $x \in \{0,1\}^{a-1}$ such that $X^{\mathsf{med}}_{[1,a-1]} = x$ with probability 1.*

Proof. We prove this lemma by invoking Lemma 7.2.21 twice. We start with our source X and invoke Lemma 7.2.21 to find a subsource X^{sb} and vertices $\mathsf{par}(v^{\mathsf{med}}), v^{\mathsf{med}}$ as in the conclusion of the lemma. Next we apply the lemma again to $X^{\mathsf{sb}}_{v^{\mathsf{med}}}$.

Since $X^{\mathsf{sb}}_{v^{\mathsf{med}}}$ is a source on $n' < n$ bits with min-entropy $\frac{k}{20\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}}$, we get that there is a subsource $X^{\mathsf{med}} \subset X^{\mathsf{sb}}$ with deficiency at most $\frac{k}{400\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}} + 4\mathsf{t}^2 \log n$ and a vertex v^{b} which is a somewhere block source. Since $X^{\mathsf{sb}} \subset X$ was of deficiency at most $k/10 + 4\mathsf{t}^2 \log n$, we get that $X^{\mathsf{med}} \subset X$ is a subsource of X with deficiency at most $k/5 + 8\mathsf{t}^2 \log n$. Further note that $H_\infty(X^{\mathsf{med}}_{v^{\mathsf{med}}}) \geq \frac{k}{20\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}} - \frac{k}{400\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}} - 4\mathsf{t}^2 \log n \geq \frac{k}{30\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}} - 4\mathsf{t}^2 \log n \geq \frac{k}{40\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}}$ by our choice of parameters. □

We apply Lemma 7.2.22 to the input source X with our parameters k, t as chosen in Section 7.2.2. We obtain a deficiency $k/4$ subsource (since $4\mathsf{t}^2 = o(k)$) X^{med} of X, and three nodes $\mathsf{par}(v^{\mathsf{med}}), v^{\mathsf{med}}, v^{\mathsf{b}} = [a, b]$ in the tree $\mathcal{T}_{n,\mathsf{t}}$ satisfying (by our choice of parameters):

Result of Step 1: A deficiency $k/4$ subsource $X^{\mathsf{med}} \subset X$ satisfying:

v^{med} is the leading block in a block source: $X^{\mathsf{med}}_{\mathsf{par}(v^{\mathsf{med}})}$ is a $(\mathsf{C}, \frac{k}{40\mathsf{t}\mathsf{C}n^{1/\mathsf{C}^4}} \geq k^{0.9})$-somewhere block source, with a sub-block $X^{\mathsf{med}}_{v^{\mathsf{med}}}$ which is the first non-constant “good” sub-block.

$X^{\mathsf{med}}_{v^{\mathsf{b}}}$ has a block source: The source $X^{\mathsf{med}}_{v^{\mathsf{b}}}$ is a $(\mathsf{C}, \frac{k}{(10\mathsf{t}^2\mathsf{C})^2} \geq k^{0.9})$-somewhere block source.

Fixed left family: For every $w \in \mathcal{L}_{v^{\mathsf{b}}}$ (Definition 7.2.19), X^{med}_w is fixed.

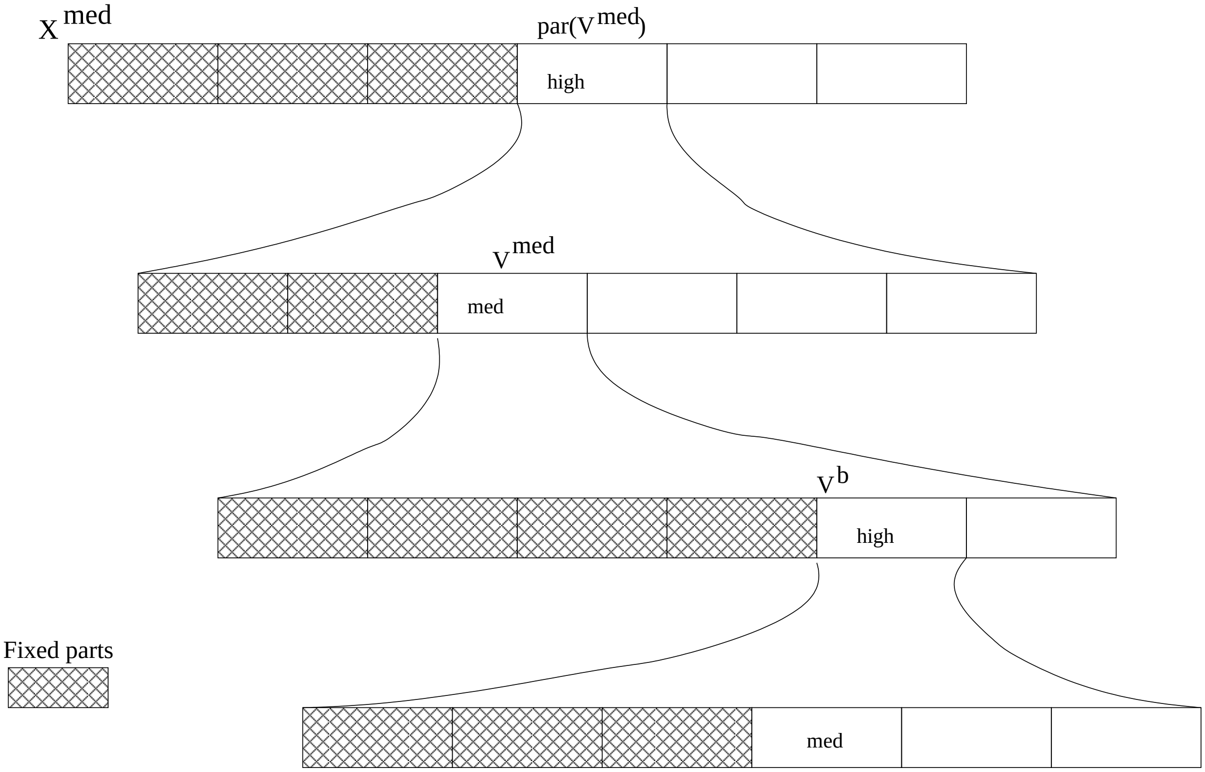

Figure 7.3: Finding two related medium parts in X^{med}

Step 2: Ensuring that challenges from the left family are properly responded.

Our desired good subsources X^{good} and Y^{good} will be deficiency clen^3 subsources of X^{med} and Y. We will ensure that in the final subsources, for every element $w \in \mathcal{L}_{v^{\mathsf{b}}}$, $\mathsf{Challenge}(X_w^{\mathsf{good}}, Y^{\mathsf{good}})$ is clen-responded by the response $\mathsf{Response}(X_{\mathsf{par}(w)}^{\mathsf{good}}, Y^{\mathsf{good}})$ with probability 1.

First we will show that we can move to a subsource where the relevant challenges are fixed.

Claim 7.2.23. *There is a subsource $Y' \subset Y$ of deficiency at most $\mathsf{t} \cdot \mathsf{len} \cdot \log n$ s.t. every challenge $\mathsf{Challenge}(X_w^{\mathsf{med}}, Y')$ for $w \in \mathcal{L}_{v^{\mathsf{b}}}$ is fixed to a constant string in the subsources X^{med}, Y'.*

Proof. By the **{Fixed left family}** property after Step 1, we have that for every $w \in \mathcal{L}_{v^{\mathsf{b}}}$, X_w^{med} is fixed. Note that $\mathsf{Challenge}(X_w^{\mathsf{med}}, Y)$ is a function only of X_w^{med} and Y. Thus, for every $w \in \mathcal{L}_{v^{\mathsf{b}}}$, $\mathsf{Challenge}(X_w^{\mathsf{med}}, Y)$ is a function only of Y.

There are at most $|\mathcal{L}_{v^{\mathsf{b}}}| \leq \mathsf{t} \log n$ challenges to consider, each of length len bits. Thus by Proposition 2.1.10, we can ensure that there is a deficiency $\mathsf{t} \cdot \mathsf{len} \cdot \log n$ subsource $Y' \subset Y$ in which all the challenges are also fixed. □

Next we will prove that there are even smaller subsources in which each of these challenges is responded with probability 1.

Claim 7.2.24. *There are subsources $X^{\mathsf{good}} \subset X^{\mathsf{med}}$ and $Y^{\mathsf{good}} \subset Y'$ of deficiency at most $O(\mathsf{t} \cdot \mathsf{len} \cdot \log n)$ in which every challenge $\mathsf{Challenge}(X_w^{\mathsf{good}}, Y^{\mathsf{good}}), w \in \mathcal{L}_{v^{\mathsf{b}}}$ is clen-responded with probability 1 by the response $\mathsf{Response}(X_{\mathsf{par}(w)}^{\mathsf{good}}, Y^{\mathsf{good}})$.*

Proof. Let $\mathcal{L}_{v^{\mathsf{b}}} = \{w_1, w_2, \ldots, w_d\}$. We will prove the stronger statement that for every i with $1 \leq i \leq d$, there are subsources $X'' \subset X^{\mathsf{med}}, Y'' \subset Y'$ of deficiency at most $2\mathsf{len}i$ in which each $\mathsf{Challenge}(X''_{w_j}, Y'')$ is clen-responded by $\mathsf{Response}(X''_{\mathsf{par}(w_j)}, Y'')$ for $1 \leq j \leq i$. We prove this by induction on i.

For the base case of $i = 1$, note that $\mathsf{Challenge}(X^{\mathsf{med}}_{w_1}, Y')$ is fixed to a constant in the source X^{med}. Since $H_\infty(X^{\mathsf{med}}_{\mathsf{par}(w_1)}) \geq H_\infty(X^{\mathsf{med}}_{v^{\mathsf{b}}}) \geq k^{0.9}$ and $H_\infty(Y') \geq k - \mathsf{t} \cdot \mathsf{len} \cdot \log n \geq k^{0.9}$, we get that $X^{\mathsf{med}}_{\mathsf{par}(w)}, Y'$ are sources that have enough entropy for our somewhere extractor SE to succeed. By the **{Hitting matrices}** property of Corollary 7.2.15, we can then ensure that there are deficiency $2\mathsf{len}$ subsources $X'' \subset X^{\mathsf{med}}, Y'' \subset Y'$ in which $\mathsf{Challenge}(X''_{w_1}, Y'')$ is clen-responded by the $\mathsf{Response}(X''_{\mathsf{par}(w_1), Y''}$ with probability 1.

For $i > 1$, we use the inductive hypothesis to find subsources $\hat{X} \subset X^{\mathsf{med}}, \hat{Y} \subset Y'$ of deficiency at most $2\mathsf{len}(i-1)$ on which all the previous challenges are clen-responded. Then, since $H_\infty(\hat{X}_{\mathsf{par}(w_i)}) \geq H_\infty(X^{\mathsf{med}}_{v^{\mathsf{b}}}) - 2\mathsf{len}(i-1) \geq k^{0.9}$ and $H_\infty(\hat{Y}) \geq k - \mathsf{t} \cdot \mathsf{len} \cdot \log n - 2\mathsf{len}(i-1) \geq k^{0.9}$, we get that $\hat{X}_{\mathsf{par}(w)}, \hat{Y}$ are sources that have enough entropy for our somewhere extractor SE to succeed. Thus we can find deficiency $2\mathsf{len} \cdot i$ subsources $X'' \subset X^{\mathsf{med}}, Y'' \subset Y'$ in which even $\mathsf{Challenge}(X''_{w_i}, Y'')$ is clen-responded by $\mathsf{Response}(X''_{\mathsf{par}(w_i)}, Y'')$. □

Together the claims give that $X^{\mathsf{good}} \subset X^{\mathsf{med}}, Y^{\mathsf{good}} \subset Y$ are subsources in which all the challenges of the left family are responded with probability 1 and are of deficiency at most $O(\mathsf{t} \cdot \mathsf{len} \cdot \log n) < \mathsf{clen}^{2.1}$ by our choice of parameters.

Since we only went down to a $\mathsf{clen}^{2.1}$ deficiency subsource of X^{med} in all of these steps, by Corollary 2.1.17, we still retain the block source structure of $X^{\mathsf{med}}_{v^{\mathsf{b}}}$. In particular, the corollary implies that $X^{\mathsf{good}}_{v^{\mathsf{b}}}$ is $2^{-19\mathsf{clen}^3}$ close to being a $(\mathsf{C}, k^{0.9} - 20\mathsf{clen}^3 \geq k^{0.8})$-somewhere block source.

Similarly $H_\infty(X^{\mathsf{good}}_{v^{\mathsf{med}}}) \geq H_\infty(X^{\mathsf{med}}_{v^{\mathsf{med}}}) - \mathsf{clen}^3 \geq k^{0.9} - \mathsf{clen}^3 \geq k^{0.8}$ and conditioned on any fixing of $X^{\mathsf{good}}_{v^{\mathsf{med}}}$, $H_\infty(X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}) \geq k^{0.9}$, since $X^{\mathsf{med}}_{\mathsf{par}(v^{\mathsf{med}})}$ was shown to be a block source with min-entropy $k^{0.9}$.

Result of Step 2: At this point we have X^{good} and Y^{good}, which are deficiency $k/4 + \mathsf{clen}^3$ subsources of the sources X and Y satisfying:

$X^{\mathsf{good}}_{v^{\mathsf{med}}}, X^{\mathsf{good}}$ **is a block source:** $H_\infty(X^{\mathsf{good}}_{v^{\mathsf{med}}}) \geq k^{0.8}$ and $X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}$ has entropy greater than $k^{0.9}$ even conditioned on any fixing of $X^{\mathsf{good}}_{v^{\mathsf{med}}}$.

$X^{\mathsf{good}}_{v^{\mathsf{b}}}$ **has a block source:** The source $X^{\mathsf{good}}_{v^{\mathsf{b}}}$ is $2^{-19\mathsf{clen}^3}$ close to being a $(\mathsf{C}, k^{0.8})$-somewhere block source.

Low blocks are correctly identified: For every $w \in \mathcal{L}_{v^{\mathsf{b}}}$ $\mathsf{Challenge}(X^{\mathsf{good}}_w, Y^{\mathsf{good}})$ is clen-responded with probability 1 by $\mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(w)}, Y^{\mathsf{good}})$.

Step 3: Ensuring that challenges along the path are somewhere random

We argue that in $X^{\mathsf{good}}, Y^{\mathsf{good}}$, for every $w \in \mathcal{P}_{v^{\mathsf{b}}}$, $\mathsf{Challenge}(X^{\mathsf{good}}_w, Y^{\mathsf{good}})$ is $2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$-close to having min-entropy clen. In fact something even stronger is true:

Lemma 7.2.25 (The challenges along the good path are somewhere random)**.** *Let $X' \subset X^{\mathsf{good}}, Y' \subset Y^{\mathsf{good}}$ be any deficiency $20\mathsf{len}$ subsources. Then in these subsources, if $w \in \mathcal{P}_{v^{\mathsf{b}}}$ is an ancestor of v^{b}, $\mathsf{Challenge}(X'_w, Y')$ is $2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$-close to being somewhere random.*

Proof. We will prove the lemma by induction on the vertices in $\mathcal{P}_{v^{\mathsf{b}}}$, starting from v^{b} and moving up the path.

Let h be the depth of v^{b} in the tree (note that $h = O(\log n)$). Let ℓ be the number of matrices in the output of $\mathsf{Response}$ (note that $\ell = \mathrm{poly}(n)$ by Corollary 7.2.15). For $w \in \mathcal{P}_{v^{\mathsf{b}}}$ at a distance of i from v^{b}, we will prove that as long as $X' \subset X^{\mathsf{good}}, Y' \subset Y^{\mathsf{good}}$ are of deficiency at most $(h-i-1)20\mathsf{len}$, $\mathsf{Challenge}(X'_w, Y')$ is $(2\ell)^i(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$-close to being somewhere random.

For the base case note that by Corollary 2.1.17, $X'_{v^{\mathsf{b}}}$ is $2^{-19\mathsf{clen}^3} + 2^{-20\mathsf{clen}^3} < 2^{-18\mathsf{clen}^3}$-close to being a $(\mathsf{C}, k^{0.8} - (h-1)20\mathsf{len} - 20\mathsf{clen}^3 > \sqrt{k})$ somewhere block source and Y' is an independent source with min-entropy $k - (k/4 + \mathsf{clen}^3 + (h-1)20\mathsf{len}) > \sqrt{k}$. Thus, in the subsources X', Y', $\mathsf{Challenge}(X'_{v^{\mathsf{b}}}, Y')$ is $2^{-18\mathsf{clen}^3} + 2^{-k^{c_1}} < (2^{-\mathsf{clen}} + 2^{-k^{c_1}})$-close to being somewhere random by Corollary 7.2.14.

Now let w be an ancestor of v^{b} and let w' be its child on the path to v^{b}. We want to show that the challenge has entropy even on deficiency $(h-i-1)20\mathsf{len}$ subsources $X' \subset X^{\mathsf{good}}, Y' \subset Y^{\mathsf{good}}$.

We will show that with high probability $\mathsf{Challenge}(X'_w, Y')$ contains $\mathsf{Challenge}(X'_{w'}, Y')$ as a substring. By the induction hypothesis we will then get that $\mathsf{Challenge}(X'_w, Y')$ must be statistically close to being somewhere random also. By our construction, to ensure that this happens we merely need to ensure that $\mathsf{Challenge}(X'_{w'}, Y')$ is clen unresponded by $\mathsf{Response}(X'_w, Y')$. We will argue this using the union bound. Fix an index j and consider the j'th response string $\mathsf{Response}(X'_w, Y')_j$.

By the **{Fixed matrices on low deficiency subsources}** property of Corollary 7.2.15, we get that X', Y' is $2^{-10\mathsf{len}}$ close to a convex combination of independent sources $\hat{X}, \hat{Y}$, where each element of the convex combination is of deficiency at most $20\mathsf{len}$ and the j'th response string $\mathsf{Response}(\hat{X}_w, \hat{Y})_j$ is fixed to a constant on these subsources. Each element of this convex combination then has a deficiency of at most $(h-i-1)20\mathsf{len} + 20\mathsf{len} = (h-(i-1)-1)20\mathsf{len}$ from $X^{\mathsf{good}}, Y^{\mathsf{good}}$.

By the induction hypothesis, we get that $\mathsf{Challenge}(\hat{X}_{w'}, \hat{Y})$ is $(2\ell)^{i-1}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$-close to being somewhere random. Thus, the probability that $\mathsf{Challenge}(X'_{w'}, Y')$ i responded by $\mathsf{Response}(X'_w, Y')$ is at most $2^{-\mathsf{clen}} + (2\ell)^{i-1}(2^{-k^{c_1}} + 2^{-\mathsf{clen}}) < 2 \cdot (2\ell)^{i-1}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$. Thus by the union bound over the ℓ response strings, we get that the probability that the challenge is responded is at most $(2\ell)^i(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$.

Note that the length of the path to v^{b} from the root is $o(\log(n))$, so we will need to repeat the induction only $\log(n)$ times. We get that the challenge is $(2\ell)^h(2^{-k^{c_1}} + 2^{-\mathsf{clen}}) < 2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$-close to being somewhere random.

□

Result of Step 3: At this point we have X^{good} and Y^{good}, which are deficiency $k/4 + \mathsf{clen}^3$ subsources of the sources X and Y satisfying:

Challenges along the path are somewhere random, even on subsources If $X' \subset X^{\mathsf{good}}, Y' \subset Y^{\mathsf{good}}$ are deficiency $20\mathsf{clen}$ subsources, $\mathsf{Challenge}(X'_w, Y')$ is $2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$ close to being somewhere random in X', Y', for every vertex $w \in \mathcal{P}_{v^{\mathsf{med}}}$.

Step 4: Ensuring that Disp outputs both 0 and 1

We will ensure that our disperser outputs both 1 and 0 with significant probability. There are two remaining steps:

- We will ensure that in our good subsources $X^{\mathsf{good}}, Y^{\mathsf{good}}$, with high probability (say $1-\gamma$) $\mathsf{val}(X^{\mathsf{good}}_{[1,n]}, Y^{\mathsf{good}}) = \mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$.
- We will ensure that in our good subsources $X^{\mathsf{good}}, Y^{\mathsf{good}}$, $\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ is both 0 and 1 with significant probability (say $\gamma^{1/10}$).

By the union bound these two facts imply that the disperser outputs both 0 and 1 with positive probability.

Lemma 7.2.26. *For every vertex v on the path from v^{med} to the root and for any $1 \le q \le \mathsf{clen}$,*

$$\Pr[\mathsf{Challenge}(X^{\mathsf{good}}_v, Y^{\mathsf{good}}) \textit{ is } q\textit{-responded by } \mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(v)}, Y^{\mathsf{good}})] \le 2^{-q} + 2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$$

Proof. By the **{Fixed matrices on low deficiency subsources}** property of Corollary 7.2.15, we get that $X^{\mathsf{good}}, Y^{\mathsf{good}}$ is $2^{-10\mathsf{len}}$-close to a convex combination of independent sources, where each element X', Y' of the convex combination is of deficiency at most $20\mathsf{len}$ and the j'th response string $\mathsf{Response}(X'_{\mathsf{par}(v)}, Y')_j$ is fixed to a constant on these subsources. Thus by Lemma 7.2.25,

$$\Pr[\mathsf{Challenge}(X'_v, Y') \text{ is } q\text{-responded by } \mathsf{Response}(X'_{\mathsf{par}(v)}, Y')] < 2^{-q} + 2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$$

□

Lemma 7.2.27 ($\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ propagates to the root)**.** *Let h be the depth of v^{med} in the tree. Then*

$$Pr_{X^{\mathsf{good}}, Y^{\mathsf{good}}}[\mathsf{val}(x_{v^{\mathsf{med}}}, y) \ne \mathsf{val}(x_{[1,n]}, y)] < 2^{-\mathsf{clen}_{h,0}}$$

Proof. We will show that for every $w \in \mathcal{P}_{v^{\mathsf{med}}}, w \ne [1,n]$, $\Pr[\mathsf{val}(X^{\mathsf{good}}_w, Y^{\mathsf{good}}) \ne \mathsf{val}(X^{\mathsf{good}}_{\mathsf{par}(w)}, Y^{\mathsf{good}})] < 2^{-\mathsf{clen}_{h,0}}/\log^2 n$. Then we will apply a union bound over all the edges in the path from the root to v^{med} to get the bound for the lemma.

Let h' be the depth of w in the tree. Now note that by our construction

$$\begin{aligned}
&\Pr[\mathsf{val}(X^{\mathsf{good}}_w, Y^{\mathsf{good}}) \ne \mathsf{val}(X^{\mathsf{good}}_{\mathsf{par}(w)}, Y^{\mathsf{good}})] \\
&< \Pr[\mathsf{Challenge}(X^{\mathsf{good}}_w, Y^{\mathsf{good}}) \text{ is } \mathsf{clen}_{h',2}\text{-responded by } \mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(w)}, Y^{\mathsf{good}})] \\
&\le 2^{-\mathsf{clen}_{h',2}} + 2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})
\end{aligned}$$

Where the last inequality is by Lemma 7.2.26. Using the union bound over all poly(n) response strings, we then get that the probability that the challenge is responded is at most $\mathrm{poly}(n)(2^{-\mathsf{clen}_{h',2}} + 2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}}) + 2^{-10\mathsf{len}}) < (1/\log^2 n)2^{-\mathsf{clen}_{h,0}}$ by our choice of parameters. Applying a union bound over the path from the root of the tree to v^{med}, we get the bound claimed by the lemma.

□

Finally we argue that the probability that $\mathsf{val}(x_{v^{\mathsf{med}}}, y)$ is 0 or 1 is significantly higher than $2^{-\mathsf{clen}_{h,0}}$. We do this by showing that for any q, the probability that $\mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ is q-responded by
$\mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}, Y^{\mathsf{good}})$ can be bounded from above and below:

Lemma 7.2.28. *Let* $p = \Pr[\mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ *is* q*-responded by* $\mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}, Y^{\mathsf{good}})]$. *Then,*

$$2^{-q\cdot\mathsf{nrows}} - 2^{-10\mathsf{len}} - 2^{-20\mathsf{len}} \le p \le 2^{-q} + 2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})$$

Proof. In Step 2 of the analysis we showed that $X^{\mathsf{good}}_{v^{\mathsf{med}}}, X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}$ is block source with block entropy $k^{0.9}$. Thus X^{good} is a convex combination of sources where for every element of the combination $\hat{X}$,

- $\hat{X}_{v^{\mathsf{med}}}$ is fixed
- $\hat{X}_{\mathsf{par}(v^{\mathsf{med}})}$ has min-entropy $k^{0.8}$

For every such subsource $\hat{X}$, $\mathsf{Challenge}(\hat{X}^{\mathsf{med}}_v, Y^{\mathsf{good}})$ is a function only of Y^{good}. Thus by Proposition 2.1.10, for every such subsource $\hat{X}$, Y^{good} is $2^{-20\mathsf{len}}$ close to a convex combination of sources where for each element of the combination $\hat{Y}$ is of deficiency at most $21\mathsf{len}$ and $\mathsf{Challenge}(\hat{X}^{\mathsf{med}}_v, \hat{Y})$ is fixed to a constant. Thus overall we get a convex combination of sources where for each element of the convex combination:

- In $\hat{X}, \hat{Y}$, $\mathsf{Challenge}(\hat{X}^{\mathsf{med}}_v, \hat{Y})$ is fixed.
- $\hat{X}_{\mathsf{par}(v^{\mathsf{med}})}, \hat{Y}$ are independent sources with min-entropy $k^{0.8}$ each.

By Corollary 7.2.15 we get that $\mathsf{Response}(\hat{X}_{\mathsf{par}(v^{\mathsf{med}})}, \hat{Y})$ is $2^{-10\mathsf{len}}$-close to being somewhere random, implying that the challenge is q-responded with probability at least $2^{-q\cdot\mathsf{nrows}} - 2^{-10\mathsf{len}}$ in these subsources. Thus we get that $\Pr \mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ is q-responded by $\mathsf{Response}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})] \ge 2^{-q\cdot\mathsf{nrows}} - 2^{-10\mathsf{len}} - 2^{-20\mathsf{len}}$.

The upper bound follows from Lemma 7.2.26.

□

This lemma then implies that $\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ takes on both values with significant probability:

Lemma 7.2.29 ($\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})$ is both 0 and 1 with significant probability)**.**

$$\Pr[\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) = 1] > (0.5)2^{-\mathsf{len}_{h,1}}$$

$$\Pr[\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) = 0] > (0.5)2^{-\mathsf{len}_{h,2}}$$

Proof. Note that

$$\begin{aligned}
&\Pr[\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) = 1]\\
&\ge \Pr[\mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) \text{ is } \mathsf{clen}_{h,1}\text{-responded by } \mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}, Y^{\mathsf{good}})]\\
&- \Pr[\mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) \text{ is } \mathsf{clen}_{h,0}\text{-responded by } \mathsf{Response}(X^{\mathsf{good}}_{\mathsf{par}(v^{\mathsf{med}})}, Y^{\mathsf{good}})]\\
&\ge 2^{-\mathsf{clen}_{h,1}\cdot\mathsf{nrows}} - 2^{-10\mathsf{len}} - 2^{-20\mathsf{len}}\\
&- 2^{-\mathsf{clen}_{h,0}} + 2^{\log^2 n}(2^{-k^{c_1}} + 2^{-\mathsf{clen}})\\
&\ge 2^{-\mathsf{len}_{h,1}} - 2^{-10\mathsf{len}} - 2^{-20\mathsf{len}} - 2\cdot 2^{-\mathsf{clen}_{h,0}}\\
&\ge (0.5)2^{-\mathsf{len}_{h,1}}
\end{aligned}$$

Similarly,

$$
\begin{aligned}
&\Pr[\mathsf{val}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) = 0] \\
&\geq \Pr[\mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) \text{ is } \mathsf{clen}_{h,2}\text{-responded by } \mathsf{Response}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})] \\
&- \Pr[\mathsf{Challenge}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}}) \text{ is } \mathsf{clen}_{h,1}\text{-responded by } \mathsf{Response}(X^{\mathsf{good}}_{v^{\mathsf{med}}}, Y^{\mathsf{good}})] \\
&\geq 2^{-\mathsf{len}_{h,2}} - 2^{-10\mathsf{len}} - 2^{-20\mathsf{len}} - 2 \cdot 2^{-\mathsf{clen}_{h,1}} \\
&> (0.5)2^{-\mathsf{len}_{h,2}}
\end{aligned}
$$

□

Chapter 8

Distributed Computing with Weak Randomness

Distributed computing is rich with examples of problems that have efficient solutions under the assumption that we have access to truly random bits, yet are impossible to solve deterministically.

The object of our work in this chapter is to find the weakest assumption on the source of randomness that would still be sufficient to design efficient distributed computing protocols. This question was first considered by Goldwasser, Sudan and Vaikuntanathan [GSV05]. They showed that Byzantine agreement, a fundamental problem in distributed computing, can be solved even if each player only has defective sources of randomness available to them. However, the sources they considered are fairly restrictive, and their main open question was whether similar results can be obtained for general weak sources. In this chapter, we show that, indeed, strong results can be obtained for general weak random sources.

Note that any extractor already gives a *generic* way to weaken the assumption that we have access to truly random bits — we can simply apply an extractor to the defective sources of randomness to get true random bits, and then run the protocols that work under the stronger assumption. For instance, if we have an efficient construction of a 2 source extractor, we can use it get distributed computing protocols as long as each player has access to 2 independent sources with sufficient entropy.

It turns out that we can do much better than this. We will use our extractor constructions to get protocols that operate under much weaker assumptions. For instance, we will be able to give protocols that operate even under the assumption that each player only has access to a single weak source with extremely small entropy, even though it is impossible to deterministically extract randomness from a single such source. We will be more precise about the exact parameters that we achieve soon.

First we discuss the models of distributed computing that we consider. We assume p total players communicate with each other in order to attain some (common or individual) computational goal with the aid of the other players. We assume that an unknown t of the players are *faulty*. We allow Byzantine faults: faulty players may behave arbitrarily and even adversarially[1]. Not only do

[1] It turns out that the protocols in this chapter can withstand a stronger form of adversarial action — we can even withstand *bounded bandwidth adversaries* at every player. It is easy to show that even if the faulty players have access to a small number of bits of information about the sources of randomness of the non-faulty players, they cannot disrupt these protocols. This follows from the fact that if X is any source with min-entropy k and $f : \{0,1\}^n \to \{0,1\}^t$

the players seek random bits, but they want their randomness to be private.

We focus on the *full information* model: communication between the players is by broadcast. Most of our results are for *synchronous* networks: communication between players 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 players may wait for all good players to transmit their messages for a particular round, before transmitting their own messages. We also have results on *asynchronous* networks: the only guarantee is that every message will eventually be received.

8.0.5 Network Extractors

Following the strategy outlined in [GSV05], we obtain our results by considering the problem of extracting private randomness for players in a network. We design protocols that allow players to communicate with each other in order to obtain private truly random bits, even if they only have access to defective sources of randomness to start with.

We will be interested in designing protocols which guarantee that at least g of the non-faulty players will end up with distributions that look close to uniform, even to a player who observes all of the transmissions that are made during the running of the protocol (i.e., the distributions are private). We call such a protocol a *network extractor*[2].

Before defining this, we fix some notation. Player i begins with an input weakly-random sample $x_i \in \{0,1\}^n$ and ends in possession of a hopefully-random sample $z_i \in \{0,1\}^m$. Let b be the concatenation of all publicly broadcasted strings in the protocol. Capital letters such as Z_i and B denote these strings viewed as random variables.

Definition 8.0.30 (Network Extractor)**.** A protocol is a (t, g, ϵ) *network extractor* for min-entropy k if for any min-entropy k independent sources $X_1, \ldots, X_n$ over $\{0,1\}^n$ and any choice of t faulty players, after running the protocol, the number of players 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 say that a protocol is a *network extractor for block sources* if each player is assumed to have access to a block source. We say that a protocol is *polynomial time computable* if the protocol can be run by players who can only evaluate polynomial time computable functions.

Once we have designed a good network extractor, we can use it to ensure that a large number of non-faulty players in the network end up with private randomness. We can then run any of the distributed computing protocols that were designed to work under the assumption that players have access to private randomness. Our gains in our paper over Goldwasser et al. are obtained by building better network extractors than they manage to build.

is any function, conditioning on the value of $f(X)$ essentially reduces the entropy of X by t. If X is a block source, a similar fact can be shown.

[2] This concept was introduced by Goldwasser et al. [GSV05], though they didn't assign it a name.

We focus on two basic distributed computing problems: leader election/collective coin-flipping and Byzantine agreement.

8.0.6 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 players. In the presence of faulty players, we would like bound the probability that one of the faulty players gets selected as the leader.

Leader election is known to be roughly equivalent to the problem of *collective coin-flipping*. The goal of a protocol for collective coin flipping is to simulate access to a trusted public coin — the outcome of the protocol is a bit (or bits) that is supposed to be uniform, even if many of the participants in the protocol are faulty or even malicious.

Collective coin-flipping was first considered in the full information model by Ben-Or and Linial [BOL78]. A result of Kahn, Kalai and Linial [KKL88] shows that if among p players there are $\omega(p/\log p)$ faulty players, there is no one round protocol for this problem in this model. A long sequence of work [Sak89, AN93, BN00, CL95, ORV94, Zuc97, RZ01, Fei99] has resulted in a protocol which requires only $\log^*(p) + O(1)$ rounds to elect a leader (and hence perform a collective coin flip) under the assumption that each player has access to private truly random bits [RZ01, Fei99] in a synchronous full information network.

8.0.7 Byzantine Agreement

In the *Byzantine agreement* problem, introduced by Pease, Shostak and Lamport [PSL80], the goal is to design a protocol that would allow the n players to agree on the result of some computation, even if some t of them are faulty. As in the case of leader election, Byzantine agreement has been studied under several models for the distributed computing environment.

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 players (for instance [Rab83]) while others do not require secret communication (for instance [CC85]).

In the full information model, there are protocols that tolerate $t < p/3$ faulty players using $O(\log p)$ rounds [GPV06] in synchronous networks, while the best asynchronous protocols require an exponential number of rounds [BO83].

8.0.8 Previous Work and Our Results

The results in this chapter are based on work with Xin Li and David Zuckerman [LRZ07].

First we summarize the results of Goldwasser et al. [GSV05].

Restricted Sources Under the assumption that the players have access to sources that are more restricted than block sources[3] of length n with min-entropy that is larger than $n/2$, they give private channel protocols that operate in $O(1)$ expected rounds and tolerate $t < p/3$ faulty players in synchronous networks. In the full information model, they give an $O(t/\log p)$ expected round protocol tolerating $t < p/3$ faulty players and an $O(2^p)$ expected round protocol tolerating $t < p/3$ faulty players in asynchronous networks.

[3] 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.

General Sources Under the assumption that the players have access to general sources, they give results only for the case of private channels. They give an $O(1)$ expected round protocol that tolerates $t < p/3$ faulty players in synchronous networks and a $O(1)$ expected round protocol that tolerates $t < p/5$ faulty players for asynchronous networks.

They posed the open question of whether protocols can be designed in the full information model assuming only that each player has access to general weak sources of randomness. We answer this question in this paper. We have many new results. We highlight some of them below. All of these results are in the full-information model, assuming that each player only has access to weak sources.

Using Weak Random Sources Without Loss As long as the min-entropy rate of the sources is greater than 1/2, we give protocols for Leader Election and Byzantine Agreement that match the best protocols for these problems under the assumption that true randomness is available. In fact, as long as the fraction of faulty players t is bounded by a constant less than 1, we show how to build a 1 round synchronous network extractor which leaves almost every non-faulty player with private randomness. We prove the following theorem:

Theorem 8.0.31 (High Entropy Network Extractor)**.** *There exists a constant $c > 0$ such that for every $\gamma > \delta > 0$, constant $\beta > 0$ and p large enough, there exists a polynomial time computable 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.*

This theorem allows us to simulate *any* distributed computing protocol that operates under the assumption that each player has access to truly random bits with the aid of general weak sources.

In particular, using the Byzantine agreement protocol of Goldwasser et al. [GPV06], we obtain the following theorem:

Theorem 8.0.32 (Byzantine Agreement Without Loss)**.** *Let $\alpha, \beta > 0$ be any constants. Then there exists a constant $\gamma > 0$ such that if each player has access to a $(n, (1/2 + \beta)n)$ source and $\gamma \log n > \log\log p$ and p is large enough, there exists a polynomial time computable synchronous $O(\log p)$ round protocol for Byzantine Agreement tolerating $(1/3 - \alpha)p$ faulty players in the full information model.*

For Leader Election, using the protocol of Feige [Fei99], gives us the following theorem:

Theorem 8.0.33 (Leader Election Without Loss)**.** *Let $\alpha, \beta > 0$ be any constants. Then there exists a constant $\gamma > 0$ such that if each player has access to a $(n, (1/2 + \beta)n)$ source and $\gamma \log n > \log\log p$ and p is large enough, there exists a polynomial time computable synchronous $\log^* p + O(1)$ round protocol for Leader Election tolerating $(1/2 - \alpha)p$ faulty players in the full information model.*

Results for Low Entropy in Synchronous Networks In the case that the min-entropy of the general sources is much lower, we can still design a network extractor, though we suffer a loss in terms of the number of non-faulty players that end up with private truly random bits. We can show that if t players are faulty to start off with, roughly $p - 2t$ of the good players end up with private randomness. We get the following theorem:

Theorem 8.0.34 (Low Entropy Network Extractor)**.** *There exists a constant $c > 0$ such that for every $\gamma > \delta > 0$, $\beta > 0$ and p large enough, there exists a polynomial time computable 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.*

Note that the number of players that end up with private randomness is much smaller that the number of non-faulty players. Luckily, it turns out that some of the protocols for Byzantine Agreement only require that a few of the non-faulty players have access to good random bits. The rest need to just follow the instructions of the protocol.

Using the best available results for Byzantine Agreement and Leader Election when players have access to truly random bits gives us the following theorems:

Theorem 8.0.35 (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.*

Theorem 8.0.36 (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.*

Results for General Sources in Asynchronous Networks In the case of asynchronous networks, we can use our new extractor construction to build a network extractor that works as long as the number of faulty players is bounded by $p/6$ and one third of the sources are significantly shorter than the others.

Theorem 8.0.37 (Asynchronous Extractor)**.** *There exist constants $\beta, c > 0$ such that for every $\alpha, \gamma > 0$, if our network has $2p/3$ players with access to (n, n^γ) sources and the remaining players have access to a $(n^{\gamma\beta}, k = n^{\alpha\gamma\beta})$ source, then as long as p is large enough and $t < p/6$, there exists a polynomial time computable 1 round $(t, p/6 - t, 2^{-k^c})$ asynchronous extractor in the full-information model.*

We use this extractor to obtain the following theorem:

Theorem 8.0.38 (Asynchronous Byzantine Agreement)**.** *There exists a constant $\beta > 0$ such that for every $\alpha, \gamma > 0$, if our network has p players of which $2p/3$ players with access to (n, n^γ) sources and the remaining players have access to a $(n^{\gamma\beta}, k = n^{\alpha\beta\gamma})$ source, then as long as p is large enough and the number of faulty players t satisfies $18t + 3 < p$, there exists a polynomial time computable $O(2^p)$ expected round protocol for Byzantine Agreement in the full-information model.*

8.0.9 Techniques

Given a C-source extractor, we can immediately get some kind of network extractor. We could designate p/C of the players as receivers, and each receives the weak random strings from a distinct set of $\mathsf{C}-1$ other players. The player can then apply a C-source extractor to these $\mathsf{C}-1$ strings plus her own. If an honest player receives strings from only honest players, then the extractor output will be close to uniform. This is the basis for our asynchronous extractor.

For synchronous networks, we can do much better. The idea is to allow a player to receive several C-tuples. We then apply a C-source extractor to each C-tuple (for now the received player doesn't use her own string). If at least one of these C-tuples contain strings only from honest players, then the output Y will be a somewhere random (Chapter 3).

In Chapter 4 we constructed an extractor for two independent sources as long as one of the sources is a somewhere random source with a small number of rows. By applying this extractor to the above string Y plus the player's own string, the output will be close to uniform.

Now the issue is how to choose the C-tuples. To choose them, we use two ingredients. The first ingredient is what we call an AND-disperser. This is a bipartite graph with low left degree such that for any large enough set S on the right, there are several vertices on the left all of whose neighbors lie in S. Intuitively, such vertices on the left correspond to good C-tuples.

However, the AND-disperser by itself doesn't give good results. This is because it doesn't use somewhere random sources. So our second ingredient is a disperser. The disperser will allow us to pick several C-tuples and obtain a somewhere-random source. Since we only need a constant-degree disperser, it's enough to use a constant-degree expander.

8.1 Synchronous Network Extractors

In this section we discuss how to build network extractors (Definition 8.0.30) in a full-information synchronous network. Our best results in this section will work even in the case that a broadcast channel is not available. If we do need a broadcast channel, we explicitly state this.

We assume that each of the p players has access to an (n, k) source, and that each of these sources is independent of each other. Our constructions will use constructions of extractors for independent sources and other intermediate objects. Although we have already seen several constructions of such extractors in Chapter 4, we abstract out our dependence on these objects in this chapter.

Assumption 8.1.1. We assume we have access to a C-Source extractor $\mathsf{IndepExt} : (\{0,1\}^n)^{\mathsf{C}} \to \{0,1\}^m$ with error ϵ for (n, k) sources. Throughout this section, we reserve C for the number of sources that $\mathsf{IndepExt}$ needs to function, which is assumed to be a constant. We further assume that $\mathsf{IndepExt}$ is strong in the sense that for any (n, k) sources $X_1, Y_2, \ldots, Y_{\mathsf{C}}$, we have that

$$|(\mathsf{IndepExt}(X_1, Y_2, \ldots, Y_{\mathsf{C}}), Y_2, \ldots, Y_{\mathsf{C}}) - (U_m, Y_2, \ldots, Y_{\mathsf{C}})| < \epsilon$$

where here U_m is independent of $Y_2, \ldots, Y_{\mathsf{C}}$.

Another object that turns out to be relevant to this problem is an extractor for a somewhere random source (Definition 2.1.21) and an independent general source, which we constructed in Chapter 4 (Theorem 4.5.7).

Assumption 8.1.2. We assume we have access to a function $\mathsf{SRExt} : \{0,1\}^n \times \{0,1\}^{k^{1.9}} \to \{0,1\}^m$ such that if X is an (n, k) source and Y is a $k^{0.9} \times k^{0.1}$ somewhere random source,

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

where U_m is independent of Y.

To get concrete results, we shall simply plug in one of the extractors that we have constructed in Chapter 4.

8.1.1 General Weak Sources

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 8.1.3. 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 $\mathsf{IndepExt}, m, \epsilon, k$ be a $\mathsf{C} = 2$ source extractor for $(1.1tn, \min\{0.1tk, k\})$ sources as in Assumption 8.1.1.

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

Round 1 :

1. Every player $i \in A$ announces their string x_i.
2. Let $y^j = x_1, \ldots, x_{1.1t}$ denote the concatenation of these strings received by j.
3. For every $j \in B$, j'th player computes $z_j = \mathsf{IndepExt}(y^j, x_j)$.

We see that if $X_1, \ldots, X_p$ are independent (n, k) sources, and the network contains t faulty players. 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 $\mathsf{IndepExt}$. This means that every non-faulty player in the set B ends up with private randomness. We get that the above protocol is a $(t, p - 2t, \epsilon)$ synchronous network extractor in the full-information model, as long as $t < p/2$.

There are two problems with the above protocol. First, the best known polynomial time 2-source extractor construction at the time of this writing is Bourgain's extractor (Appendix C), which can only extract from sources when the min-entropy rate is close to half. This means we only get explicit protocols for such high entropy. Secondly, the above network extractor only guarantees that $p - 2t$ players get private randomness, while we can hope that as many as $p - t$ players can get private randomness. We shall improve our results on both these fronts. First, we show to use Assumption 8.1.2 to get results for low entropy.

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

Protocol 8.1.4. 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 $\mathsf{SRExt}, n_1, m_1, \epsilon_1, k_1$ be an extractor with parameters as in Assumption 8.1.2. Let $\mathsf{IndepExt}$ be a C source extractor with parameters $n_2, k_2, m_2, \epsilon_2$ as in Assumption 8.1.1. 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.

Round 1 :

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

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 problem with this approach is that our extractor SRExt from Theorem 4.5.7 only works when the somewhere random source has much fewer rows than the length of each row. Thus, this protocol only succeeds in the case that the entropy of the sources is much larger than $\binom{t+\mathsf{C}}{\mathsf{C}}$. On the other hand, this protocol does work for polynomially small entropy, since Theorem 4.5.7 and Theorem 4.1.1.

Protocol for Low Entropy

Now we describe our best result for the case of low entropy sources. Our protocol will be a variation on Protocol 8.1.4. 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 8.1.5 (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 $(l, r, \mathsf{C}, \delta)$ 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 8.1.6. *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 8.1.7 (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 8.1.8 (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 easily obtained just by deleting vertices from the above graph. We get the following corollary:

Corollary 8.1.9. *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 8.1.10. We can show that Protocol 8.1.10 is a network extractor for entropy k.

Theorem 8.1.11 (Low Entropy Network 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.*

Proof. Let SRExt be as in Theorem 4.5.7, 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 $\mathsf{IndepExt}, \mathsf{C}$ be as in Theorem 4.1.1, 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. 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 8.1.6.

Proof of Lemma 8.1.6. We break up $[r]$ into equally sized disjoint sets $S_1, \ldots, S_{\frac{\delta r}{2\mathsf{C}}}$, 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.

Protocol 8.1.10 (Network Extractor for Low Entropy)**.** 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 Assumption 8.1.2. Let $\mathsf{IndepExt}$ be a C source extractor with parameters $n, k, m_2 = k, \epsilon_2$ as in Assumption 8.1.1.
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 8.1.6.
Set $\lambda = \min\{\gamma', \frac{\gamma-\delta}{1-\gamma}\}$.
Let G_2 denote the $(p-r, hr, d, \lambda)$ bipartite expander given by Corollary 8.1.9.

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.

Round 1 :

1. Every player $i \in A$ announces his string x_i.
2. For every vertex g in the left vertex set of G_1, every remaining player j computes the string $y^j_g = \mathsf{IndepExt}(x^j_{g1}, x^j_{g2}, \dots, x^j_{g\mathsf{C}})$, where here $x^j_{g1}, x^j_{g2}, \dots, x^j_{g\mathsf{C}}$ are the strings announced by the C neighbors of g.
3. Every player $j \in B$ computes the $d \times k$ matrix s^j whose w'th row is $y^j_{j_w}$, where here j_w is the w'th neighbor of j in G_2.
4. Every player $j \in B$ computes the private string $\mathsf{SRExt}(x_j, s^j)$.

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 8.1.10 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.

Protocol for Block Sources

Next we show that in the case that each player has access to a block source with just 2 blocks (Definition 2.1.15), 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 8.1.12 (Network Extractor for Block Sources)**.** 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 Assumption 8.1.2. Let $\mathsf{IndepExt}$ be a C source extractor with parameters $n, k, m_2 = k, \epsilon_2$ as in Assumption 8.1.1.
Set $\alpha = (1-\delta)/2$. Set $r = \alpha p$.
Let G_1, γ', h be such that there is a $(hr, r, \mathsf{C}, \frac{1-\delta}{1+\delta}, \gamma')$-$\mathsf{AND}$-disperser promised by Lemma 8.1.6.
Set $\lambda = \min\{\gamma', \gamma - \delta\}$.
Let G_2 denote the $(p-r, hr, d, \lambda)$ bipartite expander given by Corollary 8.1.9.

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

Round 1 :

1. Every player i announces x_i.
2. For $i = 1, 2, \ldots, 1/\alpha$,
   (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 $y^j_g = \mathsf{IndepExt}(x^j_{g1}, x^j_{g2}, \ldots, x^j_{g\mathsf{C}})$, where here $x^j_{g1}, x^j_{g2}, \ldots, x^j_{g\mathsf{C}}$ 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 $y^j_{j_w}$, 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 $\mathsf{SRExt}(x^2_j, s^j)$.

Then, we can prove the following theorem.

Theorem 8.1.13 (Low Entropy Network 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. The analysis is only slightly more complicated than before.

Let SRExt be as in Theorem 4.5.7, 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 $\mathsf{IndepExt}, \mathsf{C}$ be as in Theorem 4.1.1, 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 8.1.14 (High Entropy Network 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 8.1.15. *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 2.1.11 (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 2.1.20, setting $\ell = \beta n/10$, we get that X_1, X_2 is $2(2^{-\beta n/10} + 2^{-betan/10+1})$-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 8.1.12 to get random bits. ■

8.2 Asynchronous Network Extractors

We turn to the case of asynchronous networks in the full-information model. The first protocol we consider is a generalization of one due to Goldwasser et al. [GSV05].

We can prove the following theorem about Protocol 8.2.1.

Theorem 8.2.2. *For every constant $\beta > 0$, there exist a constants c, C such that for every $t < p/2\mathsf{C}$, Protocol 8.2.1 is a 2 round $(t, (p/\mathsf{C} - 2t), 2^{-k^c})$ asynchronous network extractor for min-entropy k in the full-information model.*

Protocol 8.2.1 (Asynchronous Extractor)**.** For an asynchronous 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 $\mathsf{IndepExt}$ be a C source extractor with parameters n, k, m, ϵ as in Assumption 8.1.1.

Partition the p players into p/C sets $S_1, ...S_{p/\mathsf{C}}$, each of size C. In each set S_i we set a special player i.

1. For every $i = 1, 2, \ldots, p/\mathsf{C}$, each player $j \in S_i$ sends x_j to i.
2. Each special player i waits to receive $\mathsf{C} - 1$ strings from other players in her set S_i.
3. Every special player i that receives $\mathsf{C}-1$ strings computes $z_i = \mathsf{IndepExt}(x_i, \ldots, x_{i_\mathsf{C}})$, where $i, \ldots, i_\mathsf{C}$ are the players in S_i. She then sends the message "complete" to all other special players.
4. If any special player receives $p/\mathsf{C} - t$ "complete" messages before she receives the $\mathsf{C} - 1$ messages from players in her set, she aborts.

Proof. First note that at most t of the sets S_i can contain a faulty player. In every set that doesn't contain a faulty player, the special player will eventually receive $\mathsf{C} - 1$ messages from other in his set, and then send out a "complete" message. Since there are at least $p/\mathsf{C} - t$ sets which contain no faulty players, every special player eventually receives $p/\mathsf{C} - t$ "complete" messages and terminates.

To see that this protocol leaves at least $p/\mathsf{C} - 2t$ players with private randomness, consider the first special player that receives $p/\mathsf{C} - t$ "complete" messages. Since at most t of the players are faulty, this means that at least $p/\mathsf{C} - t - t = p/\mathsf{C} - 2t$ of the complete messages were sent to this special player from sets S_i that contain no faulty players. The special players in each of these sets received strings only from non-faulty players, thus they each succeed in extracting randomness. ■

8.3 A Lower Bound

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 over synchronous networks in the full information model.

Theorem 8.3.1. *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.

■

8.4 Applications to 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.

8.4.1 Collective Coin-Flipping and Leader Election

We use the following theorem as a black box.

Theorem 8.4.1 ([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 8.1.11) 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 8.4.2 (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 8.1.10 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 $p2^{-k^{\Omega(1)}}$ close to being truly random. Since we can choose β in the theorem, we can make this error an arbitrarily small constant.

Since $\gamma < 1/3$, 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 8.4.1 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.

8.4.2 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 8.4.3 ([GPV06])**.** *For every $\beta < 1/3$, 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 8.4.4 ([BO83])**.** *For every $t < p/3$, there exists a $O(2^p)$ round protocol for Byzantine agreement in an asynchronous network tolerating t faulty players in the full information model.*

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.

Theorem 8.4.5 (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 8.1.10 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 $p2^{-k^{\Omega(1)}}$ 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 8.4.3 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'rd 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.

Finally, we discuss the case of asynchronous networks.

Theorem 8.4.6 (Asynchronous Byzantine Agreement)**.** *There exists a constant $\beta > 0$ such that for every $\alpha, \gamma > 0$, if our network has p players of which $2p/3$ players with access to (n, n^{γ}) sources and the remaining players have access to a $(n^{\gamma\beta}, k = n^{\alpha\beta\gamma})$ source, then as long as $\beta\gamma \log n \geq \log\log p$ and p is large enough and the number of faulty players t satisfies $18t < p$, there exists a polynomial time computable $O(2^p)$ expected round protocol for Byzantine Agreement in the full-information model.*

Proof Sketch. We use the independent source extractor promised by Theorem 4.1.3. We first run Protocol 8.2.1 on the players, using this extractor as a subroutine. This leaves us with a set of $p/3$ players, of which at most $p/3 - 2t$ do not have access to bits which are close to uniform. Since $\frac{2t}{p/3} = 6t/p < 1/3$, we can then run the protocol of Ben Or et al. (Theorem 8.4.4) to achieve consensus on this set. We can then have one more round to take a majority vote on this set to the rest of the players. ■

Chapter 9

Conclusions and Future Directions

As we mentioned in the introduction, this thesis is part of the larger project in derandomization of finding the weakest assumption on the physical resources available to us that will still allow us to use randomized solutions to problems in computer science. Building extractors is an attempt at finding a generic way to weaken the assumptions about the availability of randomness for *all* applications in computer science by using extractors in a black box fashion.

Although some of the types of sources of randomness that we considered (such as independent sources and small space sources) are plausible as models for distributions coming from nature, it is not at all clear if these are the models that best capture physical reality. One future research direction would be to come up with other models for sources that are general enough to capture physical processes, but still permit the design of randomness extractors for them.

Another motivation for designing extractor algorithms is that they give efficient constructions of objects with strong combinatorial properties that may be useful tools to solve other problems. While past research has given many such applications for seeded extractors, most of the extractors in this thesis haven't yet found many good applications in this sense. Of course, a two source extractor is a strictly stronger object than a seeded extractor, so every application of seeded extractors can also be seen as an application of two source extractors, but most of the applications seem to gain nothing useful from using two source extractors instead of seeded extractors[1].

In Chapter 8 of this thesis, we gave protocols for distributed computing problems that can successfully operate when each player has access to a single weak source of randomness, even though no extractor for such a source of randomness can be designed. These protocols specifically exploited the fact that we can build very good extractors in the case that one or more of the sources involved are somewhere random sources. A future research direction might be to find other important problems in computer science where the dependence on truly random bits can be reduced in non-black box ways, perhaps using one of the many intermediate objects that have been built on the path to constructing extractors.

An area where an application may be hiding is in derandomizing randomized small space algorithms. Nisan and Zuckerman [NZ96] used constructions of seeded extractors to prove that any randomized small space algorithm that uses a polynomial number of random bits can be simulated by one that uses a linear number of random bits. This seems to be a natural place where our extractors for small space sources may be used to get improvements, but we have not yet been

[1] The network extractors discussed in Chapter 8 give one example of an applications of independent source extractors, though this application is pretty direct

successful at improving older results.

9.0.3 Potential Improvements to Our Constructions

In addition to new research problems that may be lurking in this area, even the questions we have addressed in this thesis are far from resolved.

Independent Sources The best known construction for independent sources (in terms of the tradeoff between entropy requirements and the number of sources) at the time of the writing of this thesis is our 3 source extractor for polynomially small entropy from Chapter 4. This extractor is not very satisfying since it places a strange unnatural condition on the lengths of the sources. In addition, there is no reason to believe that 2-source extractors for logarithmically small entropy cannot be efficiently constructed, so there is a big gap between what we know how to do and what can potentially be done.

Small Space Sources Our best construction only gives an extractor for a slightly sublinear entropy rate $n^{-\xi_0}$ for an absolute constant ξ_0. Again, there is a big gap, since the probabilistic method gives extractors even for logarithmic entropy.

Affine Sources We were unable to give a new construction for this model in its full generality. Our results worked for a very restricted case (though this case is still a generalization of the well studied model of bit-fixing sources). The best known extractor for true affine sources again requires linear entropy [Bou07], though a random function would extract from logarithmic entropy.

Ramsey Graphs While our disperser is polynomial time computable, it is not as explicit as one might have hoped. For instance the Ramsey Graph construction of Frankl-Wilson is extremely simple: For a prime p, let the vertices of the graph be all subsets of $[p^3]$ of size $p^2 - 1$. Two vertices S, T are adjacent if and only if $|S \cap T| \equiv -1 \bmod p$. It would be nice to find a good disperser that beats the Frankl-Wilson construction, yet is comparable in simplicity. As usual, there is also the issue of improving the parameters of our construction.

Distributed Computing Many of our network extractors were obtained by reducing to the case of extractors for somewhere random sources. It would be interesting to see if we can construct network extractors by considering some weaker model of source.

9.0.4 New Techniques

Many of the results in this thesis relied on the technique of *condensing somewhere random sources* (Chapter 3). The idea of incrementally improving the quality of a distribution by increasing the min-entropy rate was well established [SZ99, NT99, ISW99, ISW00, RSW00] in the extractor literature. We were able to achieve our constructions by changing our measure of *quality*. Instead of measuring the min-entropy rate, we measured the somewhere randomness, or alternatively, the number of advice bits needed to extract from the source[2].

This raises the possibility that there are other measures of quality that might be even more productive for constructing extractors. Finding such a measure is another research direction.

[2] See the introductory comments in Chapter 3.

Appendix A

A 3-Source Extractor for slightly sublinear entropy

One way to compose our techniques with previous work to get something new was noticed by Avi Wigderson. The results in this appendix are due to him. He observed that recent constructions of randomness efficient condensers [BKS+05, Raz05] immediately imply the following theorem:

Theorem A.0.7. *For every sufficiently small constant $\gamma > 0$ there exist constants $\alpha = \alpha(\gamma) > 0$, $\beta(\gamma) > 2\gamma$ and a polynomial time computable function* $\mathsf{Cond} : \{0,1\}^n \to (\{0,1\}^{n^\beta})^{n^\gamma}$ *s.t. for any $(n, n^{1-\alpha})$ source X,* $\mathsf{Cond}(X)$ *is $2^{-n^{\Omega(1)}}$-close to a source with somewhere min-entropy rate 0.9.*

Once we have this condenser, we can compose it with Theorem 2.6.6 to get the following theorem:

Theorem A.0.8. *There exists a polynomial time computable function* $\mathsf{Ext} : (\{0,1\}^n)^3 \to \{0,1\}^{\Omega(n^\delta)}$ *s.t. for any sufficiently small constant $\delta > 0$ there exists a constant $\alpha = \alpha(\delta) > 0$ so that if X^1 is an $(n, n^{1-\alpha})$ source and X^2, X^3 are (n, n^δ) sources, with all sources independent of each other,* $\mathsf{Ext}(X^1, X^2, X^3)$ *is ϵ-close to the uniform distribution with with $\epsilon < 2^{-n^{\Omega(1)}}$.*

Proof Sketch: Set $\gamma = \delta/2$. Let $\alpha(\gamma)$ be as in Theorem A.0.7. We first apply the function Cond promised by Theorem A.0.7 to convert the first source to a source with n^γ rows, so that the source has somewhere-min-entropy rate 0.9. We now interpret this source as n^γ candidate 0.9-min-entropy rate seeds. We use these seeds with Raz's strong extractor from Theorem 2.6.6 and one of the other sources to obtain two sources which, conditioned on the seeds, are statistically close to independent aligned $(n^\gamma \times n^\delta)$ somewhere random sources. Since $\delta > \gamma = \delta/2$, we can then use our extractor from Theorem 4.5.7 to get $\Omega(n^\delta)$ bits which are exponentially close to uniformly distributed. ■

In this way we obtain an extractor that can extract from just 3 sources which need have only polynomial min-entropy (the polynomial cannot be arbitrarily small).

Remark A.0.9. Note that the above construction would even work if we had two sources, where one of them is a block source with 2 blocks.

Appendix B

Basic Fourier Analysis

In this appendix, we introduce basic concepts from Fourier Analysis that will be needed to understand the rest of the appendices.

B.1 Notation

We reserve the variable p to denote primes.

$\mathbb{F}_p$ will denote the field of size p.

$\mathbb{C}$ will denote the complex numbers.

U_m will denote the uniform distribution on m bits.

G will denote a finite abelian group.

We use the convention that $N = 2^n, M = 2^m$.

For two elements of a vector space x, y, we will use $x \cdot y$ to denote the dot product $\sum_i x_i y_i$.

For a complex number x, we will use $\overline{x}$ to represent its complex conjugate.

In this section we set up some basic background. We state several facts without proof though all of them can be worked out easily.

B.2 Inner product and Norms

Let $f : G \to \mathbb{C}$ and $g : G \to \mathbb{C}$ be two functions from a finite abelian group G to the complex numbers.

We define the inner product $\langle f, g \rangle \overset{def}{=} (1/|G|) \sum_{x \in G} f(x)\overline{g(x)}$.

The ℓ^p norm of f is defined to be $\|f\|_{\ell^p} \overset{def}{=} (\sum_{x \in G} |f(x)|^p)^{1/p}$.

The L^p norm of f is defined to be $\|f\|_{L^p} \overset{def}{=} (\frac{\sum_{x \in G} |f(x)|^p}{|G|})^{1/p} = |G|^{-1/p} \|f\|_{\ell^p}$.

The ℓ^∞ norm is defined to be $\|f\|_{\ell^\infty} \overset{def}{=} \max_x |f(x)|$.

We have the following basic relations between the norms:

Fact B.2.1. $\|f\|_{\ell^\infty} \geq (1/\sqrt{|G|})\|f\|_{\ell^2}$.

Fact B.2.2. $\|f\|_{\ell^2} \geq (1/\sqrt{|G|})\|f\|_{\ell^1}$.

Fact B.2.3 (Triangle Inequality)**.** $|\langle f, g \rangle| \leq \|f\|_{L^1} \|g\|_{\ell^\infty}$.

B.3 The Cauchy Schwartz Inequality

The Cauchy Schwartz inequality will play a central role in the proof.

Proposition B.3.1 (Cauchy Schwartz)**.** *For any two functions f, g as above, $|\langle f, g\rangle| \leq \|f\|_{L^2}\|g\|_{L^2}$.*

B.4 Characters and Discrete Fourier Basis

Let $\mathbb{F}$ be any field. Let $\psi : G \to \mathbb{F}^*$ be a group homomorphism. Then we call ψ a *character*. We call ψ non-trivial if $\psi \neq 1$. Unless we explicitly state otherwise, in this chapter all characters will map into the multiplicative group of $\mathbb{C}$.

Definition B.4.1 (Bilinear maps)**.** We say a map $e : G\times G \to \mathbb{C}$ is *bilinear* if it is a homomorphism in each variable (for every ξ, both $e(\cdot, \xi)$ and $e(\xi, \cdot)$ are homomorphisms). We say that it is *non-degenerate* if for every ξ, $e(\xi, \cdot)$ and $e(\cdot, \xi)$ are both non-trivial. We say that it is *symmetric* if $e(x, y) = e(y, x)$ for every $x, y \in G$.

Let $\mathbb{Z}_r$ denote the ring $\mathbb{Z}/(r)$. It is easy to check that if we let e be the map that maps $(x, y) \mapsto \exp(2\pi xy\imath/r)$, then e is a symmetric non-degenerate bilinear map. Let $G = H_1 \oplus H_2$ be the direct sum of two finite abelian groups. Let $e_1 : H_1 \times H_1 \to \mathbb{C}$ and $\epsilon_2 : H_2 \times H_2 \to \mathbb{C}$ be symmetric non-degenerate bilinear maps. Then it is easy to see that the map $(x_1 \oplus y_1, x_2 \oplus y_2) \mapsto e_1(x_1, x_2)e_2(y_1, y_2)$ is a symmetric non-degenerate bilinear map.

By the fundamental theorem of finitely generated abelian groups, every finitely generated abelian group is isomorphic to a direct sum of cyclic groups. Thus the previous discussion gives that:

Fact B.4.2. *For every finite abelian group G, there exists a symmetric non-degenerate bilinear $e : G \times G \to \mathbb{C}$.*

It can be shown that the characters of a finite abelian group G themselves form a finite abelian group $G^\wedge$ (called the *dual* group of G), where the group operation is point-wise multiplication. Now fix *any* symmetric, non-degenerate, bilinear map e. For every $\xi \in G$, let e_ξ denote the character $e(\xi, \cdot)$. The map $\xi \mapsto e_\xi$ can then be shown to be an isomorphism from G to $G^\wedge$.

Fact B.4.3 (Orthogonality)**.** *For any two characters e_x, e_y, we have that $\langle e_x, e_y\rangle = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$.*

We define the fourier transform of f (with respect to the above e) to be the function $\hat{f} : G \to \mathbb{C}$ to be: $\hat{f}(\xi) = \langle f, e_\xi\rangle$. Then it is easy to check that this is a linear, invertible operation on the space of all such functions. We get that:

Fact B.4.4 (Parseval)**.** $\|f\|_{L^2} = \|\hat{f}\|_{\ell^2}$.

Proposition B.4.5. $\|f\|_{\ell^1} \leq |G|^{3/2}\|\hat{f}\|_{\ell^\infty}$.

Proof.

$$
\begin{aligned}
&\|f\|_{\ell^1}\\
&\leq \sqrt{|G|}\|f\|_{\ell^2}\\
&= |G|\|f\|_{L^2}\\
&= |G|\|\hat{f}\|_{\ell^2} && \text{by Parseval(Fact B.4.4)}\\
&\leq |G|^{3/2}\|\hat{f}\|_{\ell^\infty}
\end{aligned}
$$

□

Fact B.4.6 (Fourier Inversion)**.** $f(x) = |G|\hat{\hat{f}}(-x) = \sum_{\xi \in G} \hat{f}(\xi)e_\xi(x)$.

Fact B.4.7 (Preservation of Inner Product)**.** $\langle f, g\rangle = |G|\langle \hat{f}, \hat{g}\rangle$.

By the *additive characters* of a vector space over a finite field, we mean the characters of the additive group of the vector space. In our applications for 2-source extractors, the characters will always be additive characters of some such vector space. The following proposition is easy to check:

Proposition B.4.8. *Let $\mathbb{F}^l$ be a vector space over a finite field $\mathbb{F}$. Let ψ be any non-trivial additive character of $\mathbb{F}$. Then the map $e(x,y) = \psi(x \cdot y) = \psi(\sum_i x_i y_i)$ is symmetric, non-degenerate and bilinear.*

B.5 Distributions as Functions

Note that we can view every distribution on the group G as a function that maps every group element to the probability that the element shows up. Thus we will often view distributions as real valued functions in the natural way: $X(x) = \Pr[X = x]$.

Fact B.5.1. *Let X be any random variable over G. Then $H_\infty(X) \geq k$ simply means that $\|X\|_{\ell^\infty} \leq 2^{-k}$ and implies that $\|X\|_{\ell^2} \leq 2^{-k/2}$.*

Fact B.5.2. *Let X be any random variable over G, then $\mathbb{E}_X(f(X)) = |G|\langle f, X\rangle$.*

Fact B.5.3. *If X is a distribution, $\hat{X}(0) = 1/|G|$.*

Let U denote the uniform distribution. Then note that $|G|U$ is simply the trivial character e_0. Thus:

Fact B.5.4. $\hat{U}(\xi) = \begin{cases} 1/|G| & \xi = 0 \\ 0 & \xi \neq 0 \end{cases}$.

Appendix C

Bourgain's 2-Source Extractor

In this appendix, we describe Bourgain's 2-source extractor [Bou05], which is crucial to some of our results.

C.1 Line Point Incidences

Let $\mathbb{F}$ be a finite field.

We will call a subset $\ell \subset \mathbb{F} \times \mathbb{F}$ a line if there exist two elements $a, b \in \mathbb{F}$ s.t. the elements of ℓ are exactly the elements of the form $(x, ax + b)$ for all $x \in \mathbb{F}$.

Let $P \subseteq \mathbb{F} \times \mathbb{F}$ be a set of points and L be a set of lines. We say that a point (x, y) has an incidence with a line ℓ if $(x, y) \in \ell$. A natural question to ask is how many incidences can we generate with just K lines and K points. Bourgain, Katz and Tao [BKT04] proved a bound on the number of incidences for special fields when the number of lines and points is high enough. Konyagin [Kon03] improved the bound to eliminate the need for K to be large.

Theorem C.1.1 (Line Point Incidences)**.** *[BKT04, Kon03] There exists universal constants $\beta, \alpha > 0$ such that for any prime field $\mathbb{F}_p$, if L, P are sets of K lines and K points respectively, with $K \leq p^{2-\beta_0}$, the number of incidences $I(L, P)$ is at most $O(K^{3/2-\alpha})$.*

An interesting thing to note is that the theorem above does not hold for pseudolines (sets with small pairwise intersections) over finite fields, though a similar theorem does hold over the reals.

When the field is of size 2^p for a prime p a weaker version of the line point incidences theorem holds.

Theorem C.1.2 (Line Point Incidences)**.** *[BKT04, Kon03] There exists a universal constant $\beta > 0$ such that for any field $\mathbb{F}_{2^p}$ of size 2^p for prime p, if L, P are sets of K lines and K points respectively with $2^{(1-\beta)p} \leq K \leq 2^{(1-\beta)p}$, the number of incidences $I(L, P)$ is at most $O(K^{3/2-\alpha})$.*

C.2 Bourgain's Extractor

In this section we describe Bourgain's construction. We start by revisiting the argument for why the hadamard matrix gives a good 2 source extractor for higher min-entropy.

C.2.1 Review: The Hadamard Extractor

First let us recall how to extract from two sources when the min-entropy is high. For a finite field $\mathbb{F}$, let $\mathsf{Had} : \mathbb{F}^l \times \mathbb{F}^l \to \mathbb{F}$ be the dot product function, $\mathsf{Had}(x, y) = x \cdot y$.

Let us review the following theorem.

Theorem C.2.1 ([CG88, Vaz85])**.** *For every constant $\delta > 0$, there exists a polynomial time algorithm $\mathsf{Had} : (\{0,1\}^n)^2 \to \{0,1\}^m$ s.t. if X, Y are independent $(n, (1/2 + \delta)n)$ sources, $\mathbb{E}_Y[\|\mathsf{Had}(X, Y) - U_m\|_{\ell^1}|] < \epsilon$ with $m = \Omega(n)$ and $\epsilon = 2^{-\Omega(n)}$.*

Proof. For a convenient l, we treat both inputs as elements of $\mathbb{F}^l$ (so $|\mathbb{F}|^l = N$) and then use the dot product function as described above.

We can view the random variable X as a function $X : \mathbb{F}^l \to [0, 1]$, which for each element of $\mathbb{F}^l$ assigns the probability of taking on that element. We will prove the theorem by using the XOR lemma (Appendix D). To use the lemma, we need to bound $\mathsf{bias}_\psi(X, Y) = |\mathbb{E}[\psi(\mathsf{Had}(X, Y))]|$ for every non-trivial character ψ.

Fix such a character ψ and let $e(x, y)$ be the symmetric non-degenerate bilinear map $e(x, y) = \psi(x \cdot y)$ (Proposition B.4.8). Recall that e_x denotes the character $e(x, \cdot)$. Below we will use Fourier analysis according to e.

Note that

$$\mathsf{bias}_\psi(X, Y) = \left| \sum_{y \in \mathbb{F}^l} Y(y) \sum_{x \in \mathbb{F}^l} X(x)\psi(x \cdot y) \right| \tag{C.1}$$

Now observe that $\sum_{x \in \mathbb{F}^l} X(x)\psi(x \cdot y) = |\mathbb{F}|^l \langle e_y, X\rangle = |\mathbb{F}|^l \overline{\hat{X}(y)}$. Thus we get that

$$\mathsf{bias}_\psi(X, Y) = |\mathbb{F}|^l \left| \sum_{y \in \mathbb{F}^l} Y(y)\overline{\hat{X}(y)} \right| = |\mathbb{F}|^{2l} |\langle Y, \hat{X}\rangle|$$

Using the Cauchy Schwartz inequality and the fact that $\|f\|_{\ell^2}^2 = |\mathbb{F}|^l \|f\|_{L^2}^2$ for every $f : \mathbb{F}^l \to \mathbb{C}$, we obtain the bound:

$$\begin{aligned}
\mathsf{bias}_\psi(X, Y)^2 &\leq |\mathbb{F}|^{4l} \|Y\|_{L^2}^2 \|\hat{X}\|_{L^2}^2 \\
&= |\mathbb{F}|^{2l} \|Y\|_{\ell^2}^2 \|\hat{X}\|_{\ell^2}^2 \\
&= |\mathbb{F}|^{2l} \|Y\|_{\ell^2}^2 \|X\|_{L^2}^2 && \text{by Parseval(Fact B.4.4)} \\
&= |\mathbb{F}|^{l} \|Y\|_{\ell^2}^2 \|X\|_{\ell^2}^2 \\
&\leq 2^n 2^{-k_1} 2^{-k_2}
\end{aligned}$$

Where the last inequality is obtained by Fact B.5.1, assuming X, Y have min-entropy k_1, k_2. Thus, as long as $k_1 + k_2 > n$, the bias is less than 1.

Set l so that $N^{1/l} = M = |\mathbb{F}|$. By the XOR lemma Lemma D.0.11 we get m bits which are $2^{(n-k_1-k_2+m)/2}$ close to uniform. The fact that the extractor is strong follows from Theorem E.0.15. ■

Remark C.2.2. If we use the more general XOR lemma Lemma D.0.10, we can even afford to have $l = 1$. The final extractor function would then be $\sigma(\mathsf{Had}(X, Y))$.

One question we might ask is: is this error bound just an artifact of the proof? Does the Hadamard extractor actually perform better than this bound suggests? If $l = 1$, the answer is clearly no, since the output must have at least n bits of entropy to generate a uniformly random point of $\mathbb{F}$. If l is large the answer is still no; there exist sources X, Y with entropy exactly $n/2$ for which the above extractor does badly. For example let X be the source which picks the first half of its field elements at random and sets the rest to 0. Let Y be the source that picks the second half of its field elements at random and sets the rest to 0. Then each source has entropy rate exactly 1/2, but the dot product function always outputs 0.

C.2.2 Bourgain's Extractor

A key observation of Bourgain's is that the counterexample that we exhibited for the Hadamard extractor is just a pathological case. He shows that although the Hadamard function doesn't extract from any sources with lower entropy, there are essentially very few counterexamples for which it fails. He then demonstrates how to encode any general source in a way that ensures that it is not a counterexample for the Hadamard function. Thus his extractor is obtained by first encoding each source in some way and then applying the Hadamard function.

For instance, consider our counterexamples from the last subsection. The counterexamples were essentially subspaces of the original space. In particular, each source was *closed under addition*, i.e., the entropy of the source $X+X$ obtained by taking two independent samples of X and summing them is exactly the same as the entropy of X. We will argue that when the source *grows with addition* (we will define exactly what we mean by this), the Hadamard extractor does not fail.

Our proof of Bourgain's theorem will be obtained in the following steps:

- First we will argue that for sources which grow with addition, the Hadamard extractor succeeds.

- Then we will show how to encode any source with sufficiently high entropy in a way that makes it grow with addition.

Hadamard succeeds when the sources grow with addition

To show that the Hadamard extractor succeeds, we were trying to bound the bias of the output distribution of the extractor $\mathsf{bias}_\psi(X, Y)$ Equation C.1:

$$\mathsf{bias}_\psi(X, Y) = \left| \sum_{y \in \mathbb{F}^l} Y(y) \sum_{x \in \mathbb{F}^l} X(x)\psi(x \cdot y) \right| \tag{C.2}$$

Now for any source X, let $X - X$ be the source that samples a point by sampling two points independently according to X and subtracting them.

Lemma C.2.3. $\mathsf{bias}_\psi(X, Y)^2 \leq \mathsf{bias}_\psi(X - X, Y)$

Proof.

$$\begin{aligned}\mathsf{bias}_\psi(X,Y) &= \left|\sum_{y\in\mathbb{F}^l} Y(y)\sum_{x\in\mathbb{F}^l} X(x)\psi(x\cdot y)\right| \\ &\leq \sum_{y\in\mathbb{F}^l} Y(y)\left|\sum_{x\in\mathbb{F}^l} X(x)\psi(x\cdot y)\right|\end{aligned}$$

Then by convexity,

$$\begin{aligned}\mathsf{bias}_\psi(X,Y)^2 &\leq \sum_{y\in\mathbb{F}^l} Y(y)\left|\sum_{x\in\mathbb{F}^l} X(x)\psi(x\cdot y)\right|^2 \\ &= \left|\sum_{y\in\mathbb{F}^l} Y(y)\sum_{x_1,x_2\in\mathbb{F}^l} X(x_1)X(x_2)\psi(x_1\cdot y)\psi(-x_2\cdot y)\right| \\ &= \left|\sum_{y\in\mathbb{F}^l} Y(y)\sum_{x_1,x_2\in\mathbb{F}^l} X(x_1)X(x_2)\psi((x_1-x_2)\cdot y)\right|\end{aligned}$$

Now let X' denote the source $X-X$. Then by grouping terms, we see that the last expression is simply:

$$\begin{aligned}\mathsf{bias}_\psi(X,Y)^2 &\leq \left|\sum_{y\in\mathbb{F}^l} Y(y)\sum_{x\in\mathbb{F}^l} X'(x)\psi(x\cdot y)\right| \\ &= \mathsf{bias}(X-X,Y)\end{aligned}$$

□

Notice the magic of this "squaring the sum" trick. By squaring the sum for the expectation via Cauchy Schwartz, starting with our original bound for the error of the extractor, we obtained a bound that behaves as if our original source was $X' = X - X$ instead of X! If X' has much higher entropy than X, we have made progress; we can follow the rest of the proof of Theorem C.2.1 in the same way and obtain an error bound that is a bit worse (because we had to square the bias), but now assuming that our input source was X' instead of X.

Let us explore how else we might use this trick. For one thing, we see that we can easily compose this trick with itself. Applying the lemma again we obtain $\mathsf{bias}_\psi(X,Y)^4 \leq \mathsf{bias}_\psi(X-X,Y)^2 \leq \mathsf{bias}_\psi(X-X-X+X,Y) = \mathsf{bias}_\psi(2X-2X,Y)$.

Applying the lemma with respect to Y (by symmetry), we obtain $\mathsf{bias}_\psi(X,Y)^8 \leq \mathsf{bias}_\psi(2X-2X,Y-Y)$.

In general, we obtain the following lemma:

Lemma C.2.4. *There exists a polynomial time computable function* $\mathsf{Had} : \mathbb{F}^l \times \mathbb{F}^l \to \{0,1\}^m$ *s.t. given two independent sources* X, Y *taking values in* $\mathbb{F}^l$ *and constants* c_1, c_2 *with the property that the sources* $2^{c_1}X - 2^{c_1}X$ *and* $2^{c_2}Y - 2^{c_2}Y$ *have min-entropy* k_1, k_2, *then* $|\mathbb{E}[\psi(\mathsf{Had}(X,Y))]| \leq (|\mathbb{F}^l|2^{-(k_1+k_2)})^{1/2^{c_1+c_2+2}}$ *for every non-trivial character* ψ.

Note that $X - X$ has at least as high min-entropy as X, thus if it is convenient we may simply ignore the subtraction part of the hypothesis; it is sufficient to have that $2^{c_1}X, 2^{c_2}Y$ have high min-entropy to apply the above lemma.

Encoding sources to give sources that grow with addition

Given Lemma C.2.4 our goal will be to find a way to encode X, Y in such a way that the resulting sources grow with addition. Then we can apply the dot product function and use the lemma to prove that our extractor works. How can we encode a source in a way that guarantees that it grows with addition? Our main weapon to do this will be bounds on the number of line point incidences (Theorem C.1.1 or Theorem C.1.2). We will force the adversary to pick a distribution on lines and a distribution on points with high entropy. Then we will argue that if our encoding produces a source which does not grow with addition, the adversary must have picked a set of points and a set of lines that violates the line point incidences theorem.

We will use the following corollary of Theorem C.1.1, which is slightly stronger than a theorem due to Zuckerman [Zuc06]. We will follow his proof closely.

Corollary C.2.5. *Let $\mathbb{F}$ and $K = 2^{(2+\alpha)k}$ be such that a line point incidences theorem holds for $\mathbb{F}, K$, with α the constant from Theorem C.1.1. Suppose L, X are two independent sources, with min-entropy $2k, k$ with L picking an element of $\mathbb{F}^2$ and X picking an element of $\mathbb{F}$ independently. Then the distribution $(X, L(X))$[1] where $L(X)$ represents the evaluation of the L'th line at X is $2^{-\Omega(k)}$-close to a source with min-entropy $(1 + \alpha/2)2k$.*

Proof. Every source with min-entropy k is a convex combination of sources with min-entropy k and support of size exactly 2^k. So without loss of generality we assume that $\mathsf{supp}(L)$ is of size 2^{2k} and that $\mathsf{supp}(X)$ has size 2^k.

Suppose $(X, L(X))$ is ϵ-far from any source with min-entropy $(1 + \alpha/2)2k$ in terms of statistical distance. Then there must exist some set H of size at most $2^{(1+\alpha/2)2k}$ s.t. $\Pr[(X, L(X)) \in H] \geq \epsilon$.

Then we have

- A set of points H : $2^{2k+k\alpha}$ points
- A set of lines $\mathsf{supp}(L)$: 2^{2k} lines.

Now we get an incidence whenever $(X, L(X)) \in H$. Thus the number of incidences is at least

$$\Pr[(X, L(X)) \in H] |\mathsf{supp}(L)| |\mathsf{supp}(X)| \geq \epsilon 2^{3k}$$

However, by the line point incidences theorem (Theorem C.1.1), the number of incidences is at most $2^{(3/2-\alpha)(2k+k\alpha)} = 2^{3k+3k\alpha/2-2k\alpha-k\alpha^2} < 2^{3k(1-\alpha/2)} = 2^{-(3k\alpha/2)}2^{3k}$.

These two inequalities imply that $\epsilon < 2^{-(3k\alpha/2)}$.

□

Remark C.2.6. The above proof would work even if L, X is a blockwise source with the appropriate min-entropy.

[1] Zuckerman [Zuc06] proved the slightly weaker fact that the distribution $L(X)$ has higher entropy.

Given this corollary, we now describe several ways to encode a source so that it grows with addition. It suffices to understand any one of these encodings to complete the proof for the extractor.

Encoding 1: $x \mapsto (x, g^x)$ We treat the input x from the source as an element of $\mathbb{F}^*$ for a field in which a version of the line point incidences theorem holds. Then we encode it into an element of $\mathbb{F}^2$ as (x, g^x) where g is a generator of the multiplicative group $\mathbb{F}^*$. Now fix an adversarially chosen source X. Consider the source $\overline{X}$ obtained by performing the above encoding.

$\overline{X}$ is a distribution on points of the form (x, g^x) where $x \neq 0$. By doing a change of variables, we think of every such point as $(\log_g \overline{x}, \overline{x})$.

First consider the distribution of $2\overline{X}$. An element of $\mathsf{supp}(2\overline{X})$ is of the form $(\log_g(\overline{x}_1\overline{x}_2), \overline{x}_1 + \overline{x}_2)$ for some $\overline{x}_1, \overline{x}_2$ in the support of $\overline{X}$. Notice that for each a, b with $a = \overline{x}_1\overline{x}_2$ and $b = \overline{x}_1 + \overline{x}_2$, there are at most two possible values for $(\overline{x}_1, \overline{x}_2)$, since for the solutions for $\overline{x}_1$ must satisfy some quadratic equation in a, b. This means that the min-entropy of $2\overline{X}$ is at least $2k - 1$ since the probability of getting a particular (a, b) is at most twice the probability of getting a single pair from $\overline{X}, \overline{X}$. By changing k, in the rest of this discussion we assume that the min-entropy of $2\overline{X}$ is $2k$.

Now for each $a, b \in \mathbb{F}$ with $a, b \neq 0$ define the line

$$\begin{aligned}\ell_{a,b} &= \{(ax, b + x) \in \mathbb{F}^2 | x \in \mathbb{F}\} \\ &= \{(x, x/a + b) \in \mathbb{F}^2 | x \in \mathbb{F}\}\end{aligned}$$

Every (a, b) in our encoding then determines the line $\ell_{a,b}$. Let $L = 2\overline{X}$ be a random variable that picks a line according to $2\overline{X}$.

Every element of $\mathsf{supp}(3\overline{X})$ is of the form $(\log_g(\overline{x}_1\overline{x}_2\overline{x}_3), \overline{x}_1 + \overline{x}_2 + \overline{x}_3)$ and determines the point $(\overline{x}_1\overline{x}_2\overline{x}_3, \overline{x}_1 + \overline{x}_2 + \overline{x}_3) \in \mathbb{F}^2$.

Now think of the distribution of $3\overline{X}$ as obtained by first sampling a line according to $2\overline{X}$ and then evaluating that line at an independent sample from $\overline{X}$ and outputting the resulting point. Then we see that we are in a position to apply Corollary C.2.5 to get that the encoding does grow with addition.

Encoding 2: $x \mapsto (x, x^2)$ Again we treat x as an element of the multiplicative group of a field $\mathbb{F}^*$ with characteristic not equal to 2 in which a version of the line point incidences theorem holds. Now fix an adversarially chosen source X. Let $\overline{X}$ denote the source obtained by encoding X in the above way.

First consider the distribution of $2\overline{X}$. An element of $\mathsf{supp}(2\overline{X})$ is of the form $(\overline{x}_1 + \overline{x}_2, \overline{x}_1^2 + \overline{x}_2^2)$ for some $\overline{x}_1, \overline{x}_2$ in the support of $\overline{X}$. Notice that for each a, b with $a = \overline{x}_1 + \overline{x}_2$ and $b = \overline{x}_1^2 + \overline{x}_2^2$, there are at most two possible values for $(\overline{x}_1, \overline{x}_2)$. This means that the min-entropy of $2\overline{X}$ is at least $2k - 1$ since the probability of getting a particular (a, b) is at most twice the probability of getting a single pair from $\overline{X}, \overline{X}$. By changing k, in the rest of this discussion we assume that the min-entropy of $2\overline{X}$ is $2k$.

Now for each $a, b \in \mathbb{F}$ with $a, b \neq 0$ define the line

$$\begin{aligned}\ell_{a,b} &= \{(2ax + a^2 - b, a + x) \in \mathbb{F}^2 | x \in \mathbb{F}\} \\ &= \{(x, x/(2a) + (a^2 + b)/(2a)) \in \mathbb{F}^2 | x \in \mathbb{F}\}\end{aligned}$$

Every (a,b) in our encoding then determines a unique line $\ell_{a,b}$. Let $L = 2\overline{X}$ be a random variable that picks a line according to $2\overline{X}$.

Every element of $\mathsf{supp}(3\overline{X})$ is then of the form $(\overline{x}_1 + \overline{x}_2 + \overline{x}_3, \overline{x}_1^2 + \overline{x}_2^2 + \overline{x}_3^2)$ and determines the point

$$\begin{aligned}&((\overline{x}_1 + \overline{x}_2 + \overline{x}_3)^2 - (\overline{x}_1^2 + \overline{x}_2^2 + \overline{x}_3^2), \overline{x}_1 + \overline{x}_2 + \overline{x}_3)\\ &= (2(\overline{x}_1 + \overline{x}_2)\overline{x}_3 + (\overline{x}_1 + \overline{x}_2)^2 - (\overline{x}_1^2 + \overline{x}_2^2), (\overline{x}_1 + \overline{x}_2) + \overline{x}_3)\\ &= (2a\overline{x}_3 + a^2 - b, a + \overline{x}_3)\end{aligned}$$

Now think of the distribution of $3\overline{X}$ as obtained by first sampling a line according to $2\overline{X}$ and then evaluating that line at an independent sample from $\overline{X}$ and outputting the resulting point. Then we see that we can apply Corollary C.2.5 to get that the encoding does grow with addition.

Conclusion By picking an appropriate constant γ, we obtain the following lemma:

Lemma C.2.7. *There is a universal constant γ s.t. if X is any source that picks an element of $\mathbb{F}$ with min-entropy $(1/2-\gamma)\log|\mathbb{F}|$, $3\overline{X}$ is $|\mathbb{F}|^{-\Omega(1)}$-close to a source with min-entropy $(1/2+\gamma)\log|\mathbb{F}^2|$.*

Putting things together

Putting together the results from the two previous sections and applying Lemma D.0.10, we obtain the theorem for Bourgain's extractor.

Theorem C.2.8 ([Bou05])**.** *There exists a universal constant $\gamma > 0$ and a polynomial time computable function $\mathsf{Bou} : (\{0,1\}^n)^2 \to \{0,1\}^m$ s.t. if X, Y are two independent n-bit sources with min-entropies k_1, k_2 and $k_1 + k_2 \geq 2n(1/2-\gamma)$, $\mathbb{E}_Y[\|\mathsf{Bou}(X,Y) - U_m\|_{\ell^1}] < \epsilon$, with $\epsilon = 2^{-\Omega(n)}, m = \Omega(n)$.*

Another nice theorem that we get as a consequence of the structure of Bourgain's proof is the following:

Theorem C.2.9. *There exists a universal constant $\gamma > 0$ and a polynomial time computable function $\mathsf{Bou} : (\{0,1\}^n)^2 \to \{0,1\}^m$ s.t. if $X_1, X_2, \ldots, X_t$ are t independent n-bit sources with min-entropies $k_1, k_2, \ldots, k_t$ and $k_i + k_j \geq 2n(1/2-\gamma)$ for some $i, j \in [t]$,*

$$\mathbb{E}_{X_i}[\|\mathsf{Bou}(X_1, \ldots, X_t) - U_m\|_{\ell^1}] < \epsilon$$

$$\mathbb{E}_{X_j}[\|\mathsf{Bou}(X_1, \ldots, X_t) - U_m\|_{\ell^1}] < \epsilon$$

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

Proof Sketch: Treating each x_i as non-zero elements of $GF(2^p)$ for prime p, define $f(x_1, \ldots, x_t) = \prod_i x_i + \prod_i x_i^2$.

Without loss of generality, assume we know that $k_1 + k_2 \geq 2n(1/2-\gamma)$. Then for every nontrivial character ψ of $GF(2^p)$, we can bound $|\mathbb{E}[\psi(f(x_1, \ldots, x_t))]| \leq \mathbb{E}_{X_3,\ldots,X_t}|E_{X_1,X_2}[\psi(f(X_1, \ldots, X_t))]|$. For every fixing of $X_3, \ldots, X_t$, the inner expectation is small by our analysis of Bourgain's extractor. Thus the overall expectation is just as small.

Then, just as before, we can use the XOR lemma to get random bits.

□

Appendix D

XOR lemma for abelian groups

In this subsection we will prove a generalization of Vazirani's XOR lemma. This lemma appears to be folklore, but we were unable to find a proof written down anywhere.

Throughout this section we reserve G for a finite abelian group.

The lemma we will prove is the following:

Lemma D.0.10 (XOR lemma for cyclic groups)**.** *For every cyclic group $G = \mathbb{Z}_N$ and every integer $M \le N$, there is an efficiently computable function $\sigma : \mathbb{Z}_N \to \mathbb{Z}_M = H$ with the following property: Let X be any random variable taking values in $\mathbb{Z}_N$ s.t. for every non-trivial character $\psi : \mathbb{Z}_N \to \mathbb{C}^*$, we have $|\mathbb{E}[\psi(X)]| < \epsilon$, then $\sigma(X)$ is $O(\epsilon \log N \sqrt{M}) + O(M/N)$ close to the uniform distribution.*

It turns out that it is easy to extend this result to work for *any* abelian group G, though it's hard to state the result for general abelian groups in a clean way. In this section we will discuss the proof of the above lemma and just make a few remarks about how to extend it to general abelian groups.

Before we move on to prove Lemma D.0.10, let us first prove a special case of this lemma which is a generalization of Vazirani's XOR lemma. For the proof of this case below, we essentially follow the proof as in Goldreich's survey [Gol95].

Lemma D.0.11. *X be a distribution on a finite abelian group G s.t. $|\mathbb{E}[\psi(X)]| \le \epsilon$ for every non-trivial character ψ. Then X is $\epsilon\sqrt{|G|}$ close to the uniform distribution: $\|X - U\|_{\ell^1} \le \epsilon\sqrt{|G|}$.*

Proof. By the hypothesis, for every non-trivial character ψ of G, $|\langle \psi, X\rangle| = (1/|G|)|\mathbb{E}_X[\psi(X)]| \le \epsilon/|G|$.

Then note that if $\psi \ne 1$, $|\langle \psi, X - U\rangle| = |\langle \psi, X\rangle - \langle \psi, U\rangle| = |\langle \psi, X\rangle| \le \epsilon/|G|$. Also, since X, U are distributions, $\langle 1, X - U\rangle = \langle 1, X\rangle - \langle 1, U\rangle = 0$.

Thus we have shown that $\|\widehat{X - U}\|_{\ell^\infty} \le \epsilon/|G|$. Proposition B.4.5 then implies that $\|X - U\|_{\ell^1} \le \epsilon\sqrt{|G|}$. □

In Lemma D.0.11, given a bound of ϵ on the biases, the statistical distance blows up by a factor of $\sqrt{|G|}$. This is too much if ϵ is not small enough. Lemma D.0.10 gives us the flexibility to tradeoff this blowup factor with the number of bits that we can claim are statistically close to uniform. As M is made smaller, the blowup factor is reduced, but we get "less" randomness. Our proof for the general case will work (more or less) by reducing to the case of Lemma D.0.11.

Note that if σ is an onto homomorphism, for every non-trivial character ϕ of H, $\phi \circ \sigma$ is a non-trivial character of G. Thus the bounds on the biases of X give bounds on the biases of

$\sigma(X)$ and we can reduce to the case of Lemma D.0.11. The problem is that we cannot hope to find such a homomorphism σ for every M. For instance, if $G = \mathbb{Z}_p$ for p a large prime, G contains no non-trivial subgroup and so σ cannot be a homomorphism for $M = \lceil p/2 \rceil$. Instead, we will show that we can find a σ which *approximates* a homomorphism in the sense:

- For every non-trivial character ϕ of H, $\phi \circ \sigma$ is approximated by a *few* characters of G. Formally, this is captured by bounding $\|\widehat{\phi \circ \sigma}\|_{L^1}$ (observe that if σ is a homomorphism, this quantity is $1/|G|$).
- We'll ensure that $\sigma(U)$ is the close to the uniform distribution on H.

Then we will be able to use the bounds on the biases of X to give bounds on the biases of $\sigma(X) - \sigma(U)$, where U is the uniform distribution. This will allow us to apply Proposition B.4.5 to conclude that X is a pseudorandom generator for σ, i.e., $\|\sigma(X) - \sigma(U)\|_{\ell^1}$ is small, which implies that $\sigma(X)$ is close to uniform, since $\sigma(U)$ is close to uniform.

The following lemma asserts that every ϵ-biased distribution is pseudorandom for any function σ that satisfies the first condition above.

Lemma D.0.12. *Let G, H be finite abelian groups. Let X be a distribution on G with $|\mathbb{E}_X[\psi(X)]| \leq \epsilon$ for every non-trivial character ψ of G and let U be the uniform distribution on G. Let $\sigma : G \to H$ be a function such that for every character ϕ of H, we have that*

$$\|\widehat{\phi \circ \sigma}\|_{L^1} \leq \tau/|G|$$

Then $\|\sigma(X) - \sigma(U)\|_{\ell^1} < \tau\epsilon\sqrt{|H|}$.

Proof. First note that the assumption on X is equivalent to $\|\widehat{X - U}\|_{\ell^\infty} \leq \epsilon/|G|$. Let ϕ be any non-trivial character of H. Then

$$\begin{aligned}
&|\langle \phi, \sigma(X) - \sigma(U) \rangle| = |\langle \phi, \sigma(X)\rangle - \langle \phi, \sigma(U)\rangle| \\
&= \frac{|\mathbb{E}_{\sigma(X)}[\phi(\sigma(X))] - \mathbb{E}_{\sigma(U)}[\phi(\sigma(U))]|}{|H|} && \text{by Fact B.5.2 applied to } \sigma(X) \text{ and } \sigma(U) \\
&= \frac{|\mathbb{E}_X[\phi(\sigma(X))] - \mathbb{E}_U[\phi(\sigma(U))]|}{|H|} \\
&= \frac{|G|}{|H|}|\langle \phi \circ \sigma, X\rangle - \langle \phi \circ \sigma, U\rangle| && \text{by Fact B.5.2 applied to } X \text{ and } U \\
&= \frac{|G|}{|H|}|\langle \phi \circ \sigma, X - U\rangle| \\
&= \frac{|G|^2}{|H|}|\langle \widehat{\phi \circ \sigma}, \widehat{X - U}\rangle| && \text{by preservation of inner product (Fact B.4.7)} \\
&\leq \frac{|G|^2}{|H|}\|\widehat{\phi \circ \sigma}\|_{L^1}\|\widehat{X - U}\|_{\ell^\infty} && \text{by the triangle inequality (Fact B.2.3)} \\
&\leq \tau\epsilon/|H| && \text{since } \|\widehat{\phi \circ \sigma}\|_{L^1} \leq \tau/|G| \text{ and } \|\widehat{X - U}\|_{\ell^\infty} \leq \epsilon/|G|
\end{aligned}$$

On the other hand, $\langle 1, \sigma(X) - \sigma(U)\rangle = 0$, since $\sigma(X)$ and $\sigma(U)$ are distributions. Thus, we have shown that $\|\widehat{\sigma(X) - \sigma(U)}\|_{\ell^\infty} \leq \tau\epsilon/|H|$, which by Proposition B.4.5 implies that $\|\sigma(X) - \sigma(U)\|_{\ell^1} \leq \tau\epsilon\sqrt{|H|}$. □

Note that when σ is the identity function (or any surjective homomorphism onto a group H), $\tau = 1$. Thus Vazirani's XOR lemma corresponds exactly to the case of σ being the identity function.

Next we show that in the special when G is a cyclic group, we can find a σ which satisfies the hypothesis of Lemma D.0.12 with small τ.

Lemma D.0.13. *Let M, N be integers satisfying $N > M$. Let $\sigma : \mathbb{Z}_N \to \mathbb{Z}_M$ be the function $\sigma(x) = x \bmod M$. Then for every character ϕ of $\mathbb{Z}_M$, $\|\widehat{\phi \circ \sigma}\|_{L^1} \leq O(\log N)/N$*

Proof. Note that if M divides N, the statement is trivial, since σ is a homomorphism. Below we show that even in the general case, this expectation is small. Define the function $\rho(x) = \exp(2\pi\imath x)$. Then note that $\rho(a+b) = \rho(a)\rho(b)$.

First let ϕ be any character of $\mathbb{Z}_M$. Then $\phi(y) = \rho(wy/M)$ for some $w \in \mathbb{Z}_M$. Clearly, $\phi(\sigma(x)) = \rho(wx/M)$.

$$\begin{aligned}
&\|\widehat{\phi \circ \sigma}\|_{L^1} \\
&= (1/N^2) \sum_{t \in \mathbb{Z}_N} \left| \sum_{x \in \mathbb{Z}_N} \rho(tx/N)\rho(-wx/M) \right| \\
&= (1/N^2) \sum_{t \in \mathbb{Z}_N} \left| \sum_{x \in \mathbb{Z}_N} \rho\left(\frac{x(tM - wN)}{NM}\right) \right|
\end{aligned}$$

Recall that for any geometric sum $\sum_{i=0}^{N} br^i = \frac{br^N - b}{r-1}$, as long as $r \neq 1$. The inner sum in this expression is exactly such a geometric sum. Thus we get:

$$\begin{aligned}
&\|\widehat{\phi \circ \sigma}\|_{L^1} \\
&\leq (1/N^2) \sum_{t \in \mathbb{Z}_N, t \neq wN/M} \left| \sum_{x \in \mathbb{Z}_N} \rho\left(\frac{x(tM - wN)}{NM}\right) \right| + 1/N \\
&= (1/N^2) \sum_{t \in \mathbb{Z}_N, t \neq wN/M} \left| \frac{\rho(\frac{N(tM - wN)}{NM}) - 1}{\rho(\frac{tM - wN}{NM}) - 1} \right| + 1/N && \text{by simplifying the geometric sum} \\
&\leq (1/N^2) \sum_{t \in \mathbb{Z}_N, t \neq wN/M} \left| \frac{2}{\rho(\frac{tM - wN}{NM}) - 1} \right| + 1/N && \text{since } \left| \rho\left(\frac{N(tM - wN)}{NM}\right) - 1 \right| \leq 2 \\
&\leq (1/N^2) \sum_{t \in \mathbb{Z}_N, t \neq wN/M} \left| \frac{2}{\rho(\frac{t - (wN/M)}{N}) - 1} \right| + 1/N
\end{aligned}$$

Now write $wN/M = c + d$, where c is an integer, and $d \in [0, 1]$. Then, by doing a change of variable from t to $t - c$, we get that the above sum is

$$(1/N^2) \sum_{t \in \mathbb{Z}_N, t \neq d} \left| \frac{2}{\rho(\frac{t-d}{N}) - 1} \right| + 1/N$$

We will bound two parts of this sum separately. Let r be a constant with $0 < r < 1/4$. Now note that $|\rho(\frac{t-d}{N}) - 1| \geq \Omega(1)$ when $rN < t < (1-r)N$, since in this situation the quantity is the distance between two points on the unit circle which have an angle of at least $2\pi r$ between them.

When t is not in this region, $|\rho(\frac{t-d}{N}) - 1| \geq |\sin(2\pi(t-d)/N)|$, since the sin function gives the vertical distance between the two points. This is at least $(t-d)/100N$ for r small enough, since we have that $|\sin x| > |x|$ for $-\pi/2 < x < \pi/2$. Thus, choosing r appropriately, we can bound the sum:

$$\begin{aligned}
&(1/N^2) \sum_{t \in \mathbb{Z}_N, t \neq d} \left| \frac{2}{\rho(\frac{t-d}{N}) - 1} \right| + 1/N \\
&= (1/N^2) \left(\sum_{t \neq d, t \in [rN,(1-r)N]} \left| \frac{2}{\rho(\frac{t-d}{N}) - 1} \right| + \sum_{t \neq d, t \notin [rN,(1-r)N]} \left| \frac{2}{\rho(\frac{t-d}{N}) - 1} \right| \right) + 1/N \\
&\leq (1/N^2) \left(\sum_{t \neq d, t \in [0,rN]} \frac{800N}{t-d} + \sum_{t \neq d, t \notin [rN,(1-r)N]} O(1) \right) + 1/N \\
&\leq (1/N^2)(O(N \log N) + O(N)) + 1/N
\end{aligned}$$

Here the last inequality used the fact that $\sum_{i=1}^{n} 1/i = O(\log n)$. Overall this gives us a bound of $\tau \leq O(\log N/N)$.

□

On uniform input the distribution $\sigma(U)$ is quite close to uniform. Specifically, if $N = qM + r$, with q, r the quotient and remainder of N on dividing by N, we have that $\sigma(U)$ is $2r((q+1)/N - 1/M) = (2r/M)(M(q+1)/N - 1) = (2r/M)(M-r)/N = 2M/N$ close to the uniform distribution. Thus, overall we get that this σ turns any distribution which fools characters with bias at most ϵ into one that is $\epsilon \log N\sqrt{M} + O(M/N)$ close to uniform.

Now we discuss the situation for general abelian groups. The basic observation is that approximate homomorphisms can be combined to give a new approximate homomorphism:

Lemma D.0.14. *Let $G = G_1 \oplus G_2$ and $H = H_1 \oplus H_2$ be finite abelian groups. Let $\sigma_1 : G_1 \to H_1$ and $\sigma_2 : G_2 \to H_2$ be two functions that satisfy the hypotheses of Lemma D.0.12 with constants τ_1 and τ_2 respectively. Then the function $\sigma : G \to H$ defined as $\sigma(x \oplus y) \overset{def}{=} \sigma_1(x) \oplus \sigma_2(y)$ satisfies the hypotheses of the lemma with parameters $\tau_1\tau_2$.*

Given this lemma, it is clear how to get an xor lemma for every abelian group. Simply write the abelian group as a direct sum of cyclic groups. Then depending on how much randomness is needed, we can compose several homomorphisms with approximate homomorphisms to get a function σ that does the job.

Appendix E

Every 2-Source Extractor is Strong

In this section we give an argument due to Boaz Barak showing that every 2 source extractor which has sufficiently small error is in fact strong.

Theorem E.0.15. *Let* $\mathsf{IndepExt} : (\{0,1\}^n)^2 \to \{0,1\}^m$ *be any two source extractor for min-entropy* k *with error* ϵ*. Then* $\mathsf{IndepExt}$ *is a strong two source extractor for min-entropy* k' *(strong with respect to both sources) with error* $2^m(\epsilon + 2^{k-k'})$.

Proof. Without loss of generality, we assume that X, Y have supports of size k'. Then we need to bound:

$$\sum_{y \in \mathsf{supp}(Y)} 2^{-k'} \|\mathsf{IndepExt}(X,y) - U_m\|_{\ell^1}$$

For any $z \in \{0,1\}^m$, define the set of bad y's for z

$$B_z = \{y : |\Pr[\mathsf{IndepExt}(X,y) = z] - 2^{-m}| \geq \epsilon\}$$

Claim E.0.16. *For every* z, $|B_z| < 2^k$

Suppose not, then the flat distributions on B_z, X are two independent sources for which the extractor $\mathsf{IndepExt}$ fails. Now let $B = \cup_z B_z$. We see that $|B| < 2^k 2^m$. Thus,

$$\begin{aligned}
&\sum_{y \in \mathsf{supp}(Y)} 2^{-k'} \|\mathsf{IndepExt}(X,y) - U_m\|_{\ell^1} \\
&= \sum_{y \in \mathsf{supp}(Y) \cap B} 2^{-k'} \|\mathsf{IndepExt}(X,y)q - U_m\|_{\ell^1} + \sum_{y \in \mathsf{supp}(Y) \setminus B} 2^{-k'} \|\mathsf{IndepExt}(X,y) - U_m\|_{\ell^1} \\
&\leq 2^{-k'} 2^{k+m} + \epsilon 2^m \\
&= 2^m (2^{k-k'} + \epsilon)
\end{aligned}$$

■

Bibliography

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160, 2004.

[AS92] N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley–Interscience Series, John Wiley & Sons, Inc., New York, 1992.

[Alo98] Noga Alon. The Shannon capacity of a union. *Combinatorica*, 18, 1998.

[AGHP92] Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple construction of almost k-wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992.

[AN93] Noga Alon and Moni Naor. Coin-flipping games immune against linear-sized coalitions. *SIAM Journal on Computing*, 22(2):403–417, April 1993.

[ACRT99] Alexander E. Andreev, Andrea E. F. Clementi, José D. P. Rolim, and Luca Trevisan. Weak random sources, hitting sets, and BPP simulations. *SIAM Journal on Computing*, 28(6):2103–2116, 1999.

[Bar06] Boaz Barak. A simple explicit construction of an $n^{\tilde{o}(\log n)}$-ramsey graph. Technical report, Arxiv, 2006. `http://arxiv.org/abs/math.CO/0601651`.

[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.

[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.

[BR94] Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, pages 276–287, 1994.

[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.

[Blu84] M. Blum. Independent unbiased coin flips from a correlated biased source: a finite state Markov chain. In *Proceedings of the 25th Annual IEEE Symposium on Foundations of Computer Science*, pages 425–433. IEEE, 1984.

[BN00] Ravi B. Boppana and Babu O. Narayanan. Perfect-information leader election with optimal resilience. *SIAM J. Comput*, 29(4):1304–1320, 2000.

[BGK06] J. Bourgain, A. Glibichuk, and S. Konyagin. Estimates for the number of sums and products and for exponential sums in fields of prime order. *J. London Math. Soc.*, 73(2):380–398, 2006.

[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*, 1:33–57, 2007.

[BKT04] Jean Bourgain, Nets Katz, and Terence Tao. A sum-product estimate in finite fields, and applications. *Geometric and Functional Analysis*, 14:27–57, 2004.

[Boy99] Victor Boyko. On the security properties of OAEP as an all-or-nothing transform. *Lecture Notes in Computer Science*, 1666:503–518, 1999.

[CDH$^+$00] Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 453–469. Springer-Verlag, May 2000.

[CRVW02] M. Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 659–668, 2002.

[CW79] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.

[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.

[CFG$^+$85] Benny Chor, Joel Friedman, Oded Goldreich, Johan Håstad, Steven Rudich, and Roman Smolensky. The bit extraction problem or t–resilient functions. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 396–407, 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.

[CV06] Kai-Min Chung and Salil Vadhan. Personal communication. 2006.

[CW89] Aviad Cohen and Avi Wigderson. Dispersers, deterministic amplification, and weak random sources. In *Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science*, pages 14–19, 1989.

[CL95] Jason Cooper and Nathan Linial. Fast perfect-information leader-election protocols with linear immunity. *Combinatorica*, 15(3):319–332, 1995.

[Cra37] Harald Cramer. On the order of magnitude of the difference between consecutive prime numbers. *Acta Arithmetica*, pages 23–46, 1937.

[Dod00] Yevgeniy Dodis. *Exposure-resilient cryptography.* PhD thesis, Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 2000.

[DSS01] Yevgeniy Dodis, Amit Sahai, and Adam Smith. On perfect and adaptive security in exposure-resilient cryptography. *Lecture Notes in Computer Science*, 2045, 2001.

[DR05] Zeev Dvir and Ran Raz. Analyzing linear mergers. Technical Report TR05-25, ECCC: Electronic Colloquium on Computational Complexity, 2005.

[Fei99] Uriel Feige. Noncryptographic selection protocols. In IEEE, editor, *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 142–152, pub-IEEE:adr, 1999. IEEE Computer Society Press.

[FW81] Peter Frankl and R. M. Wilson. Intersection theorems with geometric consequences. *Combinatorica*, 1(4):357–368, 1981.

[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.

[Gol95] Oded Goldreich. Three XOR-lemmas - an exposition. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(56), 1995.

[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.

[GSV05] Shafi Goldwasser, Madhu Sudan, and Vinod Vaikuntanathan. Distributed computing with imperfect randomness. In Pierre Fraigniaud, editor, *DISC*, volume 3724 of *Lecture Notes in Computer Science*, pages 288–302. Springer, 2005.

[Gop06] Parkshit Gopalan. Constructing Ramsey graphs from boolean function representations. In *Proceedings of the 21th Annual IEEE Conference on Computational Complexity*, 2006.

[Gro00] Vince Grolmusz. Low rank co-diagonal matrices and ramsey graphs. *Electr. J. Comb*, 7, 2000.

[Gur03] Venkatesan Guruswami. Better extractors for better codes? *Electronic Colloquium on Computational Complexity (ECCC)*, (080), 2003.

[GR06] Venkatesan Guruswami and Atri Rudra. Explicit capacity-achieving list-decodable codes. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, 2006.

[GUV07] Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. In *Proceedings of the 22nd Annual IEEE Conference on Computational Complexity*, 2007.

[ISW00] Russel Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Extractors and pseudorandom generators with optimal seed length. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 1–10, 2000.

[ISW99] Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Near-optimal conversion of hardness into pseudo-randomness. In *FOCS*, pages 181–190, 1999.

[KKL88] Jeff Kahn, Gil Kalai, and Nathan Linial. The influence of variables on Boolean functions (extended abstract). In *29th Annual Symposium on Foundations of Computer Science*, pages 68–80, White Plains, New York, 24–26 October 1988. IEEE.

[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.

[KZ] Jesse Kamp and David Zuckerman. Deterministic extractors for affine sources from bent functions. *Manuscript*.

[KZ03] Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 92–101, 2003.

[KM04] Robert Koenig and Ueli Maurer. Extracting randomness from generalized symbol-fixing and markov sources. In *Proceedings of 2004 IEEE International Symposium on Information Theory*, page 232, June 2004.

[KM05] Robert Koenig and Ueli Maurer. Generalized strong extractors and deterministic privacy amplification. In Nigel Smart, editor, *Cryptography and Coding 2005*, volume 3796 of *Lecture Notes in Computer Science*, pages 322–339. Springer-Verlag, December 2005.

[Kon03] S. Konyagin. A sum-product estimate in fields of prime order. Technical report, Arxiv, 2003. `http://arxiv.org/abs/math.NT/0304217`.

[LRZ07] Xin Li, Anup Rao, and David Zuckerman. Network extractor protocols and three-source extractors. *Manuscript*, 2007.

[Lov96] László Lovász. Random walks on graphs: A survey. In D. Miklós, V. T. Sós, and T. Szőnyi, editors, *Combinatorics, Paul Erdős is Eighty, Vol. 2*, pages 353–398. J. Bolyai Math. Soc., Budapest, 1996.

[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.

[Lu04] Chi-Jen Lu. Encryption against storage-bounded adversaries from on-line strong extractors. *J. Cryptology*, 17(1):27–42, 2004.

[LPS88] Alexander Lubotzky, R. Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[MW97] Ueli Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In Burton S. Kaliski Jr., editor, *Advances in Cryptology — CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 307–321. Springer-Verlag, August 1997.

[MNSW98] Peter Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. *Journal of Computer and System Sciences*, 57:37–49, 1 1998.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[NN93] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, August 1993.

[Nis96] Noam Nisan. Extracting randomness: How and why – a survey. In *Proceedings of the 11th Annual IEEE Conference on Computational Complexity*, pages 44–58, 1996.

[NT99] Noam Nisan and Amnon Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 58:148–173, 1999.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, October 1994.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

[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.

[PV05] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the guruswami-sudan radius in polynomial time. In *FOCS '05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 285–294, Washington, DC, USA, 2005. IEEE Computer Society.

[PSL80] Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.

[Pip87] Nicholas Pippenger. Sorting and selecting in rounds. *SIAM Journal on Computing*, 16(6):1032–1038, 1987.

[PR04] Pavel Pudlak and Vojtech Rodl. Pseudorandom sets and explicit constructions of ramsey graphs. *Submitted for publication*, 2004.

[Rab83] Michael O. Rabin. Randomized Byzantine generals. In *24th Annual Symposium on Foundations of Computer Science*, pages 403–409, Tucson, Arizona, 7–9 November 1983. IEEE.

[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.

[Raz05] Ran Raz. Extractors with weak random seeds. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 11–20, 2005.

[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.

[RSW00] Omer Reingold, Ronen Shaltiel, and Avi Wigderson. Extracting randomness via repeated condensing. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 22–31, 2000.

[Riv97] Ronald Rivest. All-or-nothing encryption and the package transform. *Lecture Notes in Computer Science*, 1267:210–??, 1997.

[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.

[Sak89] Michael Saks. A robust noncryptographic protocol for collective coin flipping. *SIAM Journal on Discrete Mathematics*, 2(2):240–244, May 1989.

[SSZ98] Michael Saks, Aravind Srinivasan, and Shiyu Zhou. Explicit OR-dispersers with polylog degree. *Journal of the ACM*, 45:123–154, 1998.

[SV86] Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semirandom sources. *Journal of Computer and System Sciences*, 33:75–87, 1986.

[Sha02] Ronen Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the European Association for Theoretical Computer Science*, 77:67–95, 2002.

[Sha06] Ronen Shaltiel. How to get more mileage from randomness extractors. pages 49–60, 2006.

[SZ99] Aravind Srinivasan and David Zuckerman. Computing with very weak random sources. *SIAM Journal on Computing*, 28:1433–1459, 1999.

[TS96] Amnon Ta-Shma. Refining randomness. In *ECCCTH: Electronic Colloquium on Computational Complexity, theses*, 1996.

[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.

[TZ04] Amnon Ta-Shma and David Zuckerman. Extractor codes. *IEEE Transactions on Information Theory*, 50, 2004.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, pages 860–879, 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*, pages 32–42, 2000.

[Vad04] Salil P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *J. Cryptology*, 17(1):43–77, 2004.

[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*, pages 366–378, 1985.

[Vaz86] Umesh V. Vazirani. *Randomness, Adversaries and Computation.* PhD thesis, EECS, University of California at Berkeley, 1986.

[VV85] Umesh V. Vazirani and Vijay V. Vazirani. Random polynomial time is equal to slightly-random polynomial time. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 417–428, 1985.

[vN51] John von Neumann. Various techniques used in connection with random digits. *Applied Math Series*, 12:36–38, 1951. Notes by G.E. Forsythe, National Bureau of Standards. Reprinted in *Von Neumann's Collected Works*, 5:768-770, 1963.

[WZ99] Avi Wigderson and David Zuckerman. Expanders that beat the eigenvalue bound: Explicit construction and applications. *Combinatorica*, 19(1):125–138, 1999.

[Wor99] N. C. Wormald. *The differential equation method for random graph processes and greedy algorithms*, pages 73–155. PWN, Warsaw, 1999.

[Zuc96] David Zuckerman. Simulating BPP using a general weak random source. *Algorithmica*, 16:367–391, 1996.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, 11:345–367, 1997.

[Zuc06] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, 2006.

Vita

Anup Rao was born in Ibadan, Nigeria on July 29th, 1980, the son of Shyamala Rao and Aravinda Rao. After completing high school at FAIPS high school, Kuwait in 1998, he attended Georgia Institute of Technology. He received the degrees of Bachelor of Science in Mathematics and Bachelor of Science in Computer Science in 2002 from Georgia Tech. In September, 2003 he entered the Graduate School of the University of Texas at Austin.

Permanent Address: 2813 1/2 Rio Grande, Apt 7
Austin, TX 78705