Disperser Graphs, Deterministic Amplification and Imperfect Random Sources.

Thesis Submitted for the Degree
"Doctor of Philosophy"
by
Aviad Cohen

Acknowledgements

This has been a long and tortuous road, but there have always been people who made it worthwhile. Thanks to Ilan and Yosi for the friendship and support that far transcends a professional acquaintance. Thanks to Assaf and Yuri for the same reasons. And to Evelin and Mauricio who have been here in the first years.

The results of this thesis are a joint work with my advisor Prof. Avi Wigderson. I thank him for his time, patience and invaluable professional support. His remarkable insights have played an important role in this work.

Special thanks to Noam Nisan for his generosity and kindness, and to Mario Szegedi with whom we shared many activities (notably the sailing sessions), while he was visiting Jerusalem.

The theory group has provided a cordial research environment. I wish to thank especially Nati Linial and Eli Shamir.

ABSTRACT

We use a certain type of expanding bipartite graphs, called **disperser graphs**, to design procedures for sampling a finite number of elements from a finite set, with the property that the probability of hitting any sufficiently large subset is high. The advantages of these procedures are:

1. They require a relatively small number of random bits.

2. They are robust with respect to the quality of the random bits used in the sampling.

Using these sampling procedures to sample random inputs of polynomial time probabilistic algorithms, we can simulate the performance of some probabilistic algorithms with less random bits or with low quality random bits. We obtain the following results:

1. The error probability of an RP or BPP algorithm that operates with a constant error bound and requires n random bits, can be made exponentially small (i.e. 2^{-n}), with only $(3 + \epsilon)n$ random bits, as opposed to standard amplification techniques that require $\Omega(n^2)$ random bits for the same task. This result is nearly optimal, since the information theoretic lower bound on the number of bits required for such an amplification is $2n$.

2. It is shown that the output of any random source whose Renyi entropy rate exceeds $\frac{1}{2}$ $\left(\frac{3}{4}\right)$, can be used to simulate RP (BPP) algorithms. This is far from the information theoretic lower bound which is $m^{\mu-1}$, where m is the number of bits and $0 < \mu < 1$ is any constant. We show that the lower bound can be achieved for a specific class of random sources called oblivious bit fixing sources.

Contents

Chapter 1

Introduction

1.1 The problem of Reducing Randomness in Probabilistic Algorithms.

Randomness is an important computational resource. In some computational problems (i.e. routing in degree bounded networks, decision trees, communication complexity, finite automatons), randomized algorithms are provably more efficient than their deterministic counterparts. As far as the main computational model the Turing machine is concerned, the power of randomization is not known. However, there are many problems for which the known randomized algorithms perform better (in time or space) than the known deterministic ones.

Since randomness is a computational resource, it is natural to try to reduce its amount and in general to investigate the extent to which it can be traded with the deterministic resources. An extreme version of this question is the following: can one eliminate randomness completely with an extra cost of deterministic resources, allowed by the complexity class in question. To put it differently: does randomization increase the power of a given computational model within a given complexity class?

There are two natural ways in which one might set about reducing the amount of randomness available to a probabilistic algorithm:

1. Reduce the number of random bits.

2. Reduce the quality of the random bits (i.e. allow nonuniform distributions and dependence among the random bits).

The problem of reducing randomness (and perhaps eliminating it altogether), appears to be easier in the case of nonuniform computational models. In fact Newman [New] showed that the number of random bits used by a nonuniform probabilistic algorithm, acting on an input of length n, can be reduced to $O(\log n)$, while maintaining the error probability arbitrarily close to its original value. The question of whether randomness can be completely eliminated has been answered for most nonuniform computational models. Roughly speaking, in computational models whose maximal complexity is linear (i.e. decision trees, oblivious routing on bounded

degree networks, communication complexity), randomization cannot be eliminated [SW86, KPU88, Lov90]. On the other hand, in computational models where a polynomial increase of resources within a complexity class makes sense, (i.e. polynomial sized circuits, constant depth polynomial sized circuits), standard amplification techniques can be used to eliminate randomness completely [Adl78, ABO84].

This work is concerned with the more difficult problem of reducing randomness in uniform computational models. The current situation in this field is:

 Reducing the number of random bits.
Various techniques have been developed to reduce the number of random bits. Among them:

1. **Pseudorandom generators**. These are uniform devices that stretch a small seed (random string), into a longer string that "looks" random to a given computational model. The only models for which pseudorandom generators have been given, are AC^0 circuits [NW88], and LOGSPACE machines [BNS88, Nis90]). For the more powerful models such as TMs and PRAMs, the constructions of pseudorandom generators that have been given, are subject to complexity-theoretic assumptions whose proof appears beyond the scope of current theoretical computer science.

2. **Derandomization techniques** i.e. techniques for eliminating random bits altogether have been devised for certain situations where strong dependence among the random bits is allowed [Lub85]. However the applicability of these techniques is rather limited.

3. **Deterministic amplification**. Various methods have been developed to reduce the error probability of a probabilistic algorithms, with a relatively small extra cost of random bits [KPS85, CG86].

Reducing the quality of the random bits.
It has been established that polynomial time probabilistic algorithms can be be simulated with the outputs of certain types of imperfect random sources i.e. sources that output random bits of low quality ([Blu84],[VV85] and [CG85]).

1.2 The Approach of This Thesis.

In this work we develop general procedures for sampling a finite number of elements from a finite set, with the property that the probability of hitting any sufficiently large subset is high. These procedures have two advantages which make them a useful tool in reducing the randomness of probabilistic algorithms:

1. They require a relatively small number of random bits.

2. They are robust with respect to the quality of the random bits used in the sampling.

The sampling procedures developed here rely on a certain type of expanding bipartite graphs which we call **disperser** graphs. The more expanding the graphs are, the better are the statistical properties of the induced sampling procedure. We give explicit constructions of families of disperser graphs with better expansion properties than achieved so far, using a method of [AKS87] of random walks on expander digraphs.

Since the above graphs are to be used by polynomial time algorithms, we emphasize the notion of an **efficiently constructible** family of graphs, that is a family of graphs that can be constructed by a polynomial time Turing machine. It should be stressed that not all the explicit definitions of graph families given in literature, are efficient. We give a precise definition of efficient constructibilty, and make sure that all algorithms use efficiently constructible graphs.

To get optimal results in the sampling procedures mentioned above, one requires degree bounded digraphs for which λ_0 and $\bar{\lambda}$, the largest and second largest eigenvalues, are optimally separated i.e. digraphs for which $\bar{\lambda} = O\left(\sqrt{\lambda_0}\right)$. [1] The simplest solution would be to use the explicitly defined, degree bounded digraphs given by [LPS86], but these constructions are not known to be efficient. Therefore we use degree unbounded family digraphs which achieve the above eigenvalue separation. Various efficient constructions of such digraphs do exist.

Using sampling procedures induced by efficiently constructible families of disperser graphs, we obtain a uniform method of simulating certain polynomial time probabilistic algorithms, with a small number of random bits, or with random bits of low quality. This enables us to give new results for two problems that have been studied in the context of reducing the randomness of polynomial time probabilistic algorithms:

- **The deterministic amplification problem**: amplify the success probability of a probabilistic algorithm, adding as few random bits as possible.

- **Simulation of probabilistic algorithms with imperfect random sources**: maintain the performance of a probabilistic algorithm, when the supply of random bits comes from some imperfect source of randomness.

We will now describe previous work concerning these problems and our contributions. All the results achieved in this work are a joint work with Avi Wigderson and have appeared in [CW89].

1.3 Deterministic Amplification.

Let A be an RP (BPP) algorithm [2] which decides a language $\mathcal{L}$ with n random bits and an error bounded by $\frac{1}{2}$. The standard way of amplifying the success of A

---

[1]It can be shown that any family of digraphs satisfies: $\bar{\lambda} = \Omega\left(\sqrt{\lambda_0}\right)$. See theorem 2.9.2.

[2]An RP algorithm, is a polynomial time probabilistic algorithm with a one sided error bounded away from 0. A BPP algorithm, is a polynomial time probabilistic algorithm with a two sided error bounded away from $\frac{1}{2}$. See section 3.2 for more detailed definitions.

from $\frac{1}{2}$ to $1 - 2^{-k}$, is to run it k independent times and accept if at least one of the computations accepts. This requires kn random bits. In particular, $\Omega(n \log n)$ random bits are required to make the error probability polynomially small (i.e $\frac{1}{P(n)}$ for some polynomial P), and n^2 random bits are required to make the error probability exponentially small (i.e. 2^{-n}). The deterministic amplification problem is to achieve these error bounds with less random bits.

The first result on deterministic amplification was given by Karp, Pippenger and Sipser [KPS85]. They showed that the error probability of an RP algorithm can be made polynomially small, with no additional random bits. Chor and Goldreich [CG86], showed that the same amplification can be achieved for BPP algorithms, at the cost of doubling the amount of random bits. Bach [Bac87] showed that an exponentially small error can be attained for specific algorithms (e.g. primality testing) , by increasing the number of random bits by only a constant factor.

Using the sampling technique based on random walks on expander graphs we show that one can amplify a constant success probability of an RP or BPP algorithm, to $1 - 2^{-k}$, using $n + O(k)$ random bits. In particular, we show that for any $\epsilon > 0$:

1. If the error probability of a BPP algorithm is bounded by $\frac{1}{8}$, then it can be made polynomially small with no further cost in random bits.

2. The error probability of an RP algorithm can be made less than 2^{-n}, with $(3 + \epsilon)n$ random bits.

3. The error probability of a BPP algorithm can be made less than 2^{-n}, with $(6 + \epsilon)n$ random bits.

4. If the error probability of a BPP algorithm is bounded by $\frac{1}{8}$, it can be made less than 2^{-n}, with $(3 + \epsilon)n$ random bits.

These results are nearly optimal, since the information theoretic lower bound is $2n$ random bits.

Impagliazzo and Zuckerman [IZ], have obtained independently the result that the error probability of an RP (BPP) algorithm can be made exponentially small at a linear cost of random bits.

1.4   Imperfect Random Sources.

Blum [Blu84] initiated the study of simulating probabilistic algorithms with imperfect random sources. He generalized an idea of Von Neuman and showed that a linear amount of random bits can be extracted from the output of a Markov source. This implies immediately that RP and BPP algorithms can be simulated with the output of a Markov source. [SV84, VV85, Vaz86, CG85] studied imperfect sources from the output of which, not a single random bit can be extracted. They showed however, that RP and BPP can be simulated with these sources, using sophisticated methods to sample the random inputs for the algorithms.

4

The imperfect sources studied in [SV84, VV85, Vaz86, CG85] have a constant lower bound on their entropy rate. However, the methods used by these authors do not work for other classes of imperfect sources with a constant bound on their entropy rate. One such class are the constant rate bit fixing sources of [CGH$^+$85, BOL85, LLS87]. Here the adversary can choose a linear subset of positions (the values of the rest are set by independent coin flips) and fix their values before, during, or after the coins are flipped. For this class of sources no simulation results are known. These facts raise the question whether a constant lower bound on the entropy rate or some other global information-theoretic quantity, is a sufficient condition for the simulation, and if so, whether there exists a uniform simulation algorithm, that applies to all random sources satisfying the lower bound.

Using the sampling procedure developed in this work, we show that for any $\nu > 1$, any random source with with Renyi ν-entropy rate [3] greater than $\frac{1}{2}$ ($\frac{3}{4}$) can simulate RP (BPP). This yields a "black box" simulation method for any source satisfying the above information theoretic condition regardless of the specific details of its probability distribution. In particular we obtain new simulation results for constant rate bit fixing sources: RP and BPP can be simulated with bit fixing sources, if the fraction of the bits fixed, does not exceed $\frac{1}{2} - \epsilon$ ($\frac{1}{4} - \epsilon$). It also gives a new simulation method for the sources of [SV84, VV85, Vaz86, CG85], but only for a subrange of parameters values. Recently, Zuckerman [Zuc], has improved our results, showing that BPP can be simulated with any source that has a constant lower bound on its Renyi entropy rate.

A simple counting argument shows that a simulation of an RP or BPP algorithm using m random bits, requires $m^{\Omega(1)}$ Renyi entropy. This a leaves a wide gap between the lower bound and the upper bounds stated above. We show that for the weakest bit fixing source the lower bound can be achieved: An $m^{\Omega(1)}$ entropy (i.e. $m - m^{\Omega(1)}$ positions are fixed) suffices to simulate BPP.

1.5 The Structure of The Thesis.

This work can be divided into three parts, of two chapters each:

1. The first part consists of chapters 2 and 3. It is concerned with background and mathematical tools. Chapter 2 discusses the connection between the spectrum of a digraph and its expansion. Chapter 3 reviews some basic notions from the realm of probabilistic algorithms.

2. The second part consists of chapters 4 and 5. It develops the sampling procedures referred to in section 1.2. Chapter 4 deals with the definitions of the basic notions of sampling procedures and disperser graphs. It shows that disperser graphs yield sampling procedures with desirable statistical properties. Chapter 5 shows how to efficiently construct highly expanding disperser graphs that give rise to sampling procedures of a high performance. The algebraic methods of chapter 2 are used heavily here.

---

[3]See section 7.2 for the definition of the Renyi generalized entropies.

3. The third part (chapters 6 and 7), deals with applications of the sampling procedures. In chapter 6 they are applied to the deterministic amplification problem, and in chapter 7, they are applied to the imperfect random sources problem.

In the conclusion (chapter 8), we mention some relevant problems that remain open.

Chapter 2

Tools from Algebraic Digraph Theory.

In this chapter we assemble together the various parts of the mathematical machinery to be applied in later chapters. The chapter deals with some aspects of the algebraic theory of digraphs (directed graphs), and in particular with the connection between the spectrum of a digraph and its expansion properties. In chapters 4 and 5 the methods of this chapter are used to construct sampling procedures for random inputs of probabilistic algorithms, from expanding digraphs.

In order to gain maximum flexibility and generality, we work with directed graphs rather than undirected graphs. Standard Perron-Frobenius theory (which can be found in any text on nonnegative matrices, e.x. [Gan59, Min88]), is used. All the results given, are direct generalizations of the connection between the spectrum of an undirected graph and its expansion properties which was established in a sequence of results on expander graphs [Tan84, AM85, JM85, Alo86b, Alo86a, Chu].

The chapter can be divided into two parts. The first deals with the connection between the expansion of a digraph and its spectrum. Its logical structure is:

1. Section 2.1 recalls basic concepts from digraph theory.

2. Section 2.2 reviews some facts about the spectrum of a square matrix.

3. Sections 2.3 and 2.4 review some results of the Perron-Frobenius theory (which studies the spectrum of nonnegative matrices) and its application to digraphs.

4. Section 2.5 deals with the spectrum of regular digraphs which is particularly convenient to work with. The quantity $\bar{\lambda}$ is defined. Its importance lies in the fact, that estimates for the expansion of a digraph can be given in terms of the separation between this quantity and the degree of the digraph. To measure this separation we introduce a parameter called the **separation exponent**. This parameter measures the expansion of the digraph: the smaller it is, the more expanding the graph is.

5. Section 2.6 discusses the connection between $\bar{\lambda}$ and the spectrum of a digraph. This connection enables one in many cases, to estimate $\bar{\lambda}$ and its separation

from the degree (largest eigenvalue) of the digraph.

6. In section 2.7 we stress the importance of **primitive** digraphs, and show that one can always construct primitive digraphs from nonprimitive digraphs without increasing the value of the separation exponent.

7. Section 2.8 discusses the relation between the expansion of a digraph and the separation exponent.

The second part deals with the explicit constructions of digraphs with as small a separation exponent as possible. The emphasis is on constructions that can be implemented by polynomial time Turing machines (**efficient constructions**), since these digraphs are to be used by polynomial time algorithms in subsequent chapters. The structure of this part:

1. Section 2.9 shows that the separation exponent of a family of digraphs, is asymptotically bounded below by $\frac{1}{2}$. Thus a family of digraphs with a separation exponent $\frac{1}{2}$ is optimal.

2. In section 2.10 we recall the notion of a Cayley digraph of a finite group. A beautiful theorem of Diaconis determines the eigenvalues and eigenvectors of a Cayley digraph in terms of the irreducible representations of the group.

3. Section 2.11 describes constructions of Cayley digraphs with an optimal separation exponent. Optimality is established using Diaconis theorem and character sums upper bounds.

4. Section 2.12 describes an alternative construction of digraphs with an optimal separation exponent, based on symmetric block designs.

2.1 Digraphs.

For the sake of completeness, some basic notions and definitions from graph theory are listed below:

- **Digraph**. A directed graph. Multiple edges and self loops are allowed. Thus formally, a digraph G is an ordered pair $G = (V, E)$. V is a finite nonempty set, and E is a finite multiset of $V \times V$. The elements of V are called vertices and the elements of E are called edges. The edge (v, w) is said to be directed from v to w.

- A **graph** G is a digraph which satisfies the following property: for every pair of vertices v,w there are as many edges directed from v to w, as there are edges directed from w to v. In such a case one may forget about the direction of the edges and treat them as unordered pairs of vertices. [1]

---

[1]Note again that multiple edges and self loops are allowed. In some places graphs with multiple edges and self loops are called - multigraphs.

- The **indegree** of a vertex is the number of edges directed into it. The **outdegree** of a vertex is the number of edges directed out of it. For a graph both degrees are the same and are referred to as the **degree** of the vertex. A digraph G is **d-regular** if each vertex has an indegree and outdegree d.

- A **walk** on a digraph is a sequence of edges $(v_0, w_0), \ldots (v_{k-1}, w_{k-1})$ such that $w_i = v_{i+1}$ for $0 \leq i < k$. We say that the walk is of length k and connects vertex v_0 to vertex w_{k-1}. A walk is **closed** if $v_0 = w_{k-1}$.

- The **adjacency matrix** A of a digraph G is given by:

$$A_{ij} = \text{number of edges directed from vertex i to vertex j}$$

  Note that the adjacency matrix of a **graph** is symmetric.

- The **product** of two digraphs on N vertices, is the digraph on N vertices whose adjacency matrix is the product of the adjacency matrices. Thus the **k-th power** of a digraph G with adjacency matrix A, is the digraph whose adjacency matrix is A^k. It can also be interpreted as the digraph obtained from G by putting an edge directed from vertex v to vertex w for each walk in G of length k, that connects v to w.

- The **transpose** of a digraph G is the digraph obtained by reversing the direction of each edge in G. It will be denoted by G^T. Note that the adjacency matrix of G^T is the transpose of the adjacency matrix of G.

- A **family of digraphs** is a sequence of digraphs $\{G_i\}_{i=1}^{\infty}$ such that $\{|G_i|\}_{i=1}^{\infty}$ is an increasing sequence.

- **Strong connectivity**. Define a binary relation R on the vertex set of a digraph G as follows: $(v, w) \in R$ if there is a walk on G connecting v to w, and there is a walk connecting w to v. R is an equivalence relation and induces a partition of the vertex set into equivalence classes. If there is only one equivalence class G is said to be **strongly connected**. If G is a strongly connected *graph* we refer to it merely as being **connected**.

- A digraph G on N vertices is **k-partite** if its vertex set can be partitioned into k nonempty subsets $V_0, \ldots, V_{k-1}$ such that the edges coming out of vertices of V_i go into vertices of $V_{(i+1) \bmod k}$. A 2-partite digraph G is also called a **bipartite** digraph. Note that if G is strongly connected, this partition is unique. Also if G is d-regular then $|V_i| = \frac{N}{k}$ for $0 \leq i \leq k - 1$.

- Let G be a d-regular connected bipartite graph G on 2N vertices with adjacency matrix A. The **reduced** digraph G_r, is defined to be the digraph on N vertices, whose adjacency matrix is the matrix A_r:

$$(A_r)_{ij} = \text{number of edges connecting vertex i of } V_0(G) \text{ to vertex j of } V_1(G)$$

9

- The **double cover** of a digraph G, is a bipartite **graph** D such that $|V_0(D)| = |V_1(D)| = |V(G)|$ and $i \in V_1(D)$ is connected to $j \in V_0(D)$ with one edge for each edge (i, j) in G.

Strongly connected bipartite graphs, occur frequently in this work. If G is such a graph, the following notation is used throughout: $V_0(G)$ and $V_1(G)$ denote the two vertex sets of the 2-partition and N and M denote the respective cardinalities. It is always assumed that $M \geq N$. Also it will be assumed that all vertices of $V_1(G)$ have the same degree which is denoted by $\bar{d}$.

2.2 The Spectrum of a Square Matrice.

Throughout this section it is assumed that A is a square matrix of order $n \times n$. The **spectrum** of A is the set of its eigenvalues (with multiplicities). A has n eigenvalues (including multiplicities). We use $\lambda_0, \ldots, \lambda_{n-1}$ to denote the eigenvalues in a nonincreasing modulus [2] order. With each eigenvalue we associate an algebraic multiplicity (its multiplicity as a root of the characteristic polynomial), and a geometric multiplicity (the dimension the subspace generated by the eigenvectors corresponding to that eigenvalue). We note in passing, that the geometric multiplicity cannot exceed the algebraic multiplicity and the two are equal for all eigenvalues, if and only if A is diagonalizable.

It easy to verify that the spectrum of A^k is $\quad \lambda_0^k, \ldots, \lambda_{n-1}^k$

Matrices which possess an orthogonal basis of eigenvectors are particularly convenient to deal with. It is worthwhile to bear in mind the following facts: A complex matrix has an orthogonal basis of eigenvectors if and only if it is normal (i.e. commutes with its conjugate transpose). A real matrix has an orthogonal basis of eigenvectors over the field of real numbers, if and only if it is symmetric.

Finally, recall that the L_2 norm of A has a spectral interpretation: it is equal to the square root of the largest eigenvalue of A^*A (A^* being the complex conjugate of A).

2.3 The Perron-Frobenius Theory.

Spectra of adjacency matrices of digraphs play a central role in this work. Since adjacency matrices are nonnegative (i.e. matrices with nonnegative entries), one can invoke the Perron-Frobenius theory (which studies the spectrum of nonnegative matrices) to glean useful information. In this section the relevant results of the theory are briefly reviewed. All matrices dealt with, are assumed square and nonnegative.

First, some definitions. A matrix A is cogredient to a matrix B if there exists a permutation matrix P such that $B = P^T A P$. A matrix A is reducible if it is

---

[2]By the modulus of a number we mean its absolute value.

cogredient to a matrix of the form:

$$\begin{pmatrix} C & D \\ 0 & E \end{pmatrix}$$

where C and E are square matrices. Otherwise it is irreducible. Irreducibility of a matrix A can be given a graph theoretic interpretation: Let G be the digraph obtained by replacing each positive entry of A by 1, and viewing the resulting matrix as the adjacency matrix of G. Then the matrix A is irreducible if and only if G is strongly connected.

The main result of the Perron-Frobenius theory was proved by Perron for positive matrices (i.e. matrices with positive entries) and extended by Frobenius to irreducible matrices:

Theorem 2.3.1 *(Perron-Frobenius) An irreducible matrix A has a real positive eigenvalue r such that:*

1. *The geometric and algebraic multiplicity of r is 1.*

2. *There is a positive eigenvector corresponding to r. Any eigenvector associated with an eigenvalue different from r, cannot be nonnegative.*

3. *Every eigenvalue λ satisfies $|\lambda| \leq r$*

The eigenvalue r will henceforth be referred to as the **maximal eigenvalue** of A.

Definition 2.3.1 *Let A be an irreducible matrix with a maximal eigenvalue r. Assume that A has k distinct eigenvalues of modulus r. The number k is called the **index of imprimitivity** of A. If $k = 1$ the matrix A is said to be **primitive**. A digraph G is defined to be **primitive** if its adjacency matrix is primitive. Similarly if the adjacency matrix of G is imprimitive with index of imprimitivity k, G is imprimitive with index of imprimitivity k.*

A useful criterion for primitivity is the following: An irreducible matrix A is primitive if and only if some positive power of A is positive. In particular this implies that a strongly connected digraph with self loops is primitive. The index of imprimitivity of a matrix A can be given a graph theoretic interpretation: Let G be the digraph obtained by replacing each positive entry of A by 1, and viewing the resulting matrix as the adjacency matrix of G. The index of imprimitivity of A is the gcd of the set of lengths of closed walks on G.

Frobenius discovered some useful facts about the spectrum of imprimitive matrices. These can be deduced from the next two results:

Theorem 2.3.2 *(Wielandt) Let A be an irreducible matrix with maximal eigenvalue r. If $\lambda = re^{i\phi}$ is an eigenvalue of A, then the matrix $Ae^{i\phi}$ is similar to A.*

11

Theorem 2.3.3 *(Frobenius) Let A be an irreducible matrix with index of imprimitivity $k \geq 2$. Then A is cogredient to a matrix of a "superdiagonal" block form:*

$$\begin{pmatrix} 0 & A_0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & A_1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 & A_{k-2} \\ A_{k-1} & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}$$

where the block A_i has order $n_i \times n_{(i+1) \bmod k}$ for $0 \leq i \leq k-1$ (so that the diagonal blocks are square).

Theorem 2.3.3 has the following graph theoretic interpretation: Let A be an irreducible matrix with index of imprimitivity k. Let G be the digraph obtained by replacing each positive entry of A by 1, and viewing the resulting matrix as the adjacency matrix of G. Then G is k-partite.

Let A be an irreducible matrix with index of imprimitivity $k \geq 2$. Taken together, theorems 2.3.3 and 2.3.2 reveal a certain "cyclic" relationship between eigenvalues of A of the the same modulus, and a similar relationship between the corresponding eigenvectors. The following facts are consequences of the above theorems and were first obtained by Frobenius:

Eigenvalues.
The spectrum of A is invariant under a rotation by $\frac{2\pi}{k}$ in the complex plane. It is not invariant under a rotation by a smaller angle. The k eigenvalues whose absolute value is equal to r, are precisely the set of roots of the equation:

$$x^k = r^k$$

These eigenvalues have algebraic (and hence geometric) multiplicity 1. In general, the set of n eigenvalues can be partitioned into "cycles" of the form:

$$\mu, \mu e^{\frac{2\pi i}{k}}, \mu e^{\frac{2\pi i 2}{k}}, \ldots, \mu e^{\frac{2\pi i(k-1)}{k}}$$

where all the eigenvalues of a given cycle have the same algebraic multiplicity.

eigenvectors.
Assume (by theorem 2.3.3) that A is in a superdiagonal form. If X is an eigenvector corresponding to the eigenvalue λ, it can be written as:

$$X = \left(x_1^0, \ldots, x_{n_0}^0, x_1^1, \ldots, x_{n_1}^1, \ \ldots \ , x_1^{k-1}, \ldots, x_{n_{k-1}}^{k-1} \right)$$

where the coordinates have been partitioned according to the diagonal blocks of A: the upper index indicates the block and lower index indicates position within the block. Let θ be a root of unity of order k. Then an eigenvector corresponding to the eigenvalue $\lambda\theta$ is given by:

$$X = \left(x_1^0, \ldots, x_{n_0}^0, \ \ldots \ , x_1^i \theta^i, \ldots, x_{n_i}^i \theta^i, \ \ldots \ , x_1^{k-1} \theta^{k-1}, \ldots, x_{n_{k-1}}^{k-1} \theta^{k-1} \right)$$

This proves in particular, that the geometric multiplicities of λ and $\lambda\theta$ are also the same.

Throughout this work all digraphs are assumed to be strongly connected. There is no much loss of generality in this assumption because of the following:

Theorem 2.3.4 *Let G be a digraph with the property that the indegree of each vertex is equal to its outdegree. Then the adjacency matrix of G is cogredient to a direct sum of irreducible matrices.*

Proof Since the indegree of each vertex is equal to its outdegree, the edge set of G can be partitioned into a set of edge disjoint closed walks. Therefore if there is a directed walk from vertex **v** to vertex **w**, there is also a directed walk from from **w** to **v**. It follows that there are no edges of G connecting two distinct strongly connected components of G. This means that the adjacency matrix of G is cogredient to the the direct sum of the adjacency matrices of the components. ■

From theorems 2.3.1, 2.3.4 and the fact that the maximal eigenvalue of a d-regular strongly connected digraph is d, one gets:

Corollary 2.3.5 *Let G be d-regular digraph. The number of strongly connected components of G is equal to the multiplicity of the maximal eigenvalue of G.*

2.4 Graphs.

If G is a connected graph then its a adjacency matrix A is a symmetric matrix and has two important properties:

1. A has an orthonormal basis of eigenvectors.

2. the spectrum of A is real and therefore its index of imprimitivity does not exceed 2.

If the index of imprimitivity of G is 2, then by theorem 2.3.3 G is bipartite, and by theorem 2.3.2 the spectrum of A is invariant under a rotation through π. This implies that if λ is an eigenvalue of A, so is $-\lambda$ and with the same multiplicity. In fact as was seen in 2.3, if x is the eigenvector corresponding to the eigenvalue λ then the eigenvector corresponding to the eigenvalue $-\lambda$ is obtained by inverting the sign of the coordinates of x corresponding to $V_1(G)$.

2.5 Regular Digraphs.

The Perron-Frobenius theory yields some information about the spectra of adjacency matrices of digraphs. If the digraphs are assumed to be regular, further information can be deduced, and this makes regular digraphs particularly convenient to work with.

Let G be a strongly connected d-regular digraph on N vertices, with an adjacency matrix A. First observe that the maximal eigenvalue is d, and the eigenspace corresponding to it, is spanned by the vector all of whose coordinates are 1. Suppose that the index of imprimitivity of A is k. By theorem 2.3.3 it may assumed that A is in a superdiagonal form:

$$\begin{pmatrix} 0 & A_0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & A_1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 & A_{k-2} \\ A_{k-1} & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}$$

where the block A_i has order $\frac{N}{k} \times \frac{N}{k}$. The vertex set V of G can be partitioned into k subsets $V_0, \ldots, V_{k-1}$, each with cardinality $\frac{N}{k}$, in such a way that the edges going out of vertices of V_i, come into vertices of $V_{(i+1) \bmod k}$.

Let W be the the subspace spanned by the eigenvectors corresponding to the eigenvalues of maximal modulus. This subspace is k dimensional and one can easily display a basis of eigenvectors for it. If $\{\theta_i\}_{i=0}^{k-1}$ are the roots of unity of order k, then the discussion following theorem 2.3.2 implies that the vectors:

$$v_i = \frac{1}{\sqrt{N}}(\overbrace{\theta_i^0, \cdots, \theta_i^0}^{\frac{N}{k}}, \ldots, (\overbrace{\theta_i^{k-1}, \cdots, \theta_i^{k-1}}^{\frac{N}{k}})$$

are eigenvectors corresponding to the eigenvalues $d\theta_i$ and constitute an orthonormal basis for W. Now the vectors v_i are also eigenvectors of A^T since they satisfy:

$$A^T v_i = d\theta_i^{-1} v_i$$

It follows that W is invariant under the action of both A and A^T (viewed as linear operators). Therefore $W^\perp$ the orthogonal complement of W, is also invariant under the action of A and A^T. This permits the following definition:

Definition 2.5.1 *Let G be a connected regular digraph with adjacency matrix A. $\bar{\lambda}(G)$ is the L_2 norm of the restriction of A to $W^\perp$.*

The quantity $\bar{\lambda}(G)$, plays a central role in this work. Its importance lies in the fact , that estimates for the expansion as well as other parameters of a digraph G, can be given in terms of the separation between λ_0 and $\bar{\lambda}$. (see section 2.8). Thus it is natural to introduce a parameter of a regular digraph, that quantifies this separation. For some purposes, the ratio $\frac{\bar{\lambda}}{\lambda_0}$ may suffice, but for our goals it is convenient to adopt the following measure:

Definition 2.5.2 *Let G be a strongly connected regular digraph. The **separation exponent** of G is defined by:*

$$\delta(G) = \frac{\log \bar{\lambda}}{\log \lambda_0}$$

*G is said to be $\delta(G)$ **separable**. A strongly connected d-regular digraph for which $\bar{\lambda} < d$ is called separable.*

2.6 On The Connection Between the Separability of a Digraph and its Spectrum.

The importance of the separation between λ_0 and $\bar{\lambda}$ of a regular strongly connected digraph G has already noted. In order to estimate this separation, one can sometimes resort to the spectrum of G. The basic relation between $\bar{\lambda}$ and the spectrum is given by:

Theorem 2.6.1 *Let G be a connected d-regular digraph with index of imprimitivity k. Then:*

$$|\lambda_k(G)| \leq \bar{\lambda}(G) = \sqrt{\lambda_k\left(GG^T\right)}$$

Proof Let B be the linear operator associated with the adjacency matrix A. $\bar{B}$, the Hilbert adjoint operator of B, is represented by the matrix A^*, where $*$ denotes complex conjugation. Let θ be a primitive root of unity of order k. Then A and A^* have a common set of k orthonormal eigenvectors $\{v_i\}_{i=0}^{k-1}$, where v_i corresponds to the eigenvalues $d\theta^i$ and $d\theta^{k-i}$ respectively. Let $A_{W^\perp}$ be the matrix representing the restriction of B to $W^\perp$, with respect to some orthonormal basis. Note that $(A_{W^\perp})^*$, represents the corresponding restriction of $\bar{B}$. There exists a unitary matrix U such that:

$$UAU^{-1} = \begin{pmatrix} d & 0 & 0 & \ldots & 0 \\ 0 & d\theta & 0 & \ldots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & d\theta^{k-1} & \vdots \\ 0 & \cdots & \cdots & \cdots & A_{W^\perp} \end{pmatrix} \quad UA^*U^{-1} = \begin{pmatrix} d & 0 & 0 & \ldots & 0 \\ 0 & d\theta^{k-1} & 0 & \ldots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & d\theta & \vdots \\ 0 & \cdots & \cdots & \cdots & (A_{W^\perp})^* \end{pmatrix}$$

It follows that $\lambda_0\left(A_{W^\perp}\right) = \lambda_k(G)$ and $\lambda_0\left(A_{W^\perp}(A_{W^\perp})^*\right) = \lambda_k\left(GG^T\right)$. If v_k is an eigenvector of $A_{W^\perp}$ corresponding to its maximal eigenvalue, then left side of the inequality is implied by:

$$(\lambda_k(G))^2 = \frac{\langle A_{W^\perp}v_k, A_{W^\perp}v_k \rangle}{\langle v_k, v_k \rangle} \leq sup_{v \in W^\perp} \frac{\langle A_{W^\perp}v, A_{W^\perp}v \rangle}{\langle v, v \rangle} = (\bar{\lambda}(G))^2$$

The right side of the inequality follows from:

$$(\bar{\lambda}(G))^2 = sup_{v \in W^\perp} \frac{\langle A_{W^\perp}(A_{W^\perp})^*v, v \rangle}{\langle v, v \rangle} = \lambda_0\left(A_{W^\perp}(A_{W^\perp})^*\right) = \lambda_k\left(GG^T\right)$$

∎

Corollary 2.6.2 *If the adjacency matrix of G has an orthonormal basis of eigenvectors then*

$$\bar{\lambda}(G) = |\lambda_k(G)|$$

Proof This follows from theorem 2.6.1 and the fact that if the matrix A has an orthonormal basis of eigenvectors then $\lambda_k (AA^*) = |\lambda_k (A)|^2$ ∎

Corollary 2.6.2 applies in particular to the case of graphs (the adjacency matrix of a graph is symmetric and hence has an orthonormal basis of eigenvectors), and Cayley digraphs (see theorem 2.10.1).

By Corollary 2.6.2 every graph is separable. We will be interested however, in separable primitive digraphs (see section 2.7 for motivation). The next theorem show that if a graph is not primitive, one can extract from it a primitive digraph of the same separation exponent:

Theorem 2.6.3 *Let G be a d-regular connected bipartite graph. Then the reduced digraph* [3] *of G is primitive and has the same values for λ_0 and $\bar{\lambda}$ as those of G.*

Proof The adjacency matrix of G can be assumed to be in the superdiagonal form so that:

$$A = \begin{pmatrix} 0 & C \\ C^T & 0 \end{pmatrix} \qquad A^2 = \begin{pmatrix} CC^T & 0 \\ 0 & C^TC \end{pmatrix}$$

A^2 has maximal eigenvalue d^2 with multiplicity 2. Since CC^T and C^TC have the same spectrum (theorem 2.7.1), CC^T has maximal eigenvalue d^2 with multiplicity 1. The reduced graph G_r has adjacency matrix C and is d-regular (so that $\lambda_0 (G_r) = d$). If G_r is not strongly connected then by corollary 2.3.5, CC^T has maximal eigenvalue d^2 with multiplicity > 1. If G_r is strongly connected with index of imprimtivity $k > 1$ then the same conclusion follows since C and C^T have k common eigenvectors belonging to eigenvalues of absolute value d. Thus G_r is primitive.

Since CC^T and C^TC have the same spectrum, $\lambda_1 \left(CC^T \right) = \lambda_2(G^2)$ and so:

$$\bar{\lambda}(G_r) = \sqrt{\lambda_1 \left(CC^T \right)} = \sqrt{\lambda_2(G^2)} = \bar{\lambda} (G)$$

∎

Let G be a d-regular strongly connected digraph on N vertices, with index of imprimitivity $k > 1$. Under what conditions is G separable ? Without loss of generality we may assume A the adjacency matrix of G to be in a superdiagonal form:

$$\begin{pmatrix} 0 & A_0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & A_1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 & A_{k-2} \\ A_{k-1} & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}$$

where the block A_i has order $\frac{N}{k} \times \frac{N}{k}$. We can associate with G k bipartite graphs $H_0, \ldots, H_{k-1}$, on $\frac{2N}{k}$ vertices each, as follows: H_i is obtained by restricting G to the vertices $V_i \bigcup V_{i+1}$ and to edges going out of V_i, forgetting about the direction of the edges. In other words H_i has the adjacency matrix:

$$\begin{pmatrix} 0 & A_i \\ A_i^T & 0 \end{pmatrix}$$

---

[3] See section 2.1 for a definition of a reduced digraph.

Also we let G_i denote the reduced digraph of H_i i.e. the digraph whose adjacency matrix is A_i.

Theorem 2.6.4 G *is separable if and only if* $H_0, \ldots, H_{k-1}$ *are connected, in which case:*

$$\bar{\lambda}(G) = \max_i \bar{\lambda}(H_i)$$

Proof G is separable iff GG^T has the eigenvalue d^2 with multiplicity k. GG^T is the direct sum of $G_i G_i^T$ $(0 \le i \le k-1)$. Therefore G is separable iff $G_i G_i^T$ $(0 \le i \le k-1)$ has the eigenvalue d^2 with multiplicity 1. If H_i is connected then by theorem 2.6.3 G_i is primitive and separable with $\bar{\lambda}(G_i) = \bar{\lambda}(H_i)$. Hence $G_i G_i^T$ has the eigenvalue d^2 with multiplicity 1. Conversely, if H_i is not connected for some i, then $G_i G_i^T$ has the eigenvalue d^2 with multiplicity at least 2, since each component contributes 1 to the multiplicity of the maximal eigenvalue of $G_i G_i^T$.
If G is separable then since GG^T is the direct sum of $G_i G_i^T$ $(0 \le i \le k - 1)$:

$$\bar{\lambda}(G) = \sqrt{\lambda_k(GG^T)} = \max_i \sqrt{\lambda_1(G_i G_i^T)} = \max_i \bar{\lambda}(G_i) = \max_i \bar{\lambda}(H_i)$$

∎

2.7   Primitive Digraphs.

In the next chapters we will work primarily with primitive digraphs. The advantages of primitive digraphs are:

1. In some cases primitive adjacency matrices yield better expansion estimates for the digraphs. This is intuitively clear: an imprimitive digraph of index k is k-partite and each of the k sets of vertices is connected with outgoing edges to only one of the other sets.

2. Powers of imprimitive digraphs may not be strongly connected. For example if G is a bipartite graph, then G^{2k} has two components. Thus powers of G may lose expansion if G is not primitive. On the other hand powers of primitive graphs remain primitive (and hence strongly connected).

In this section it is shown that given a separable digraph, one can extract from it a primitive digraph with no larger separation exponent. Note that a specific instance of this fact has already appeared in theorem 2.6.3. Now in this work the quality of the digraph is measured by the smallness of the separation exponent. (see section 2.8 for a motivation of this approach). Thus the results of this section say that whenever one manages to construct a digraph with a small separation exponent one can assume that it is primitive.

Since an irreducible matrix with a positive trace is primitive, the brute force way of making a strongly connected digraph primitive, is to add self loops to some of the vertices. To maintain the desirable property of regularity, the same number of self loops should be added to every vertex. However, this transformation may affect

the quality the the digraph: adding c self loops to every vertex, translates the whole spectrum of G by c. Therefore, the separation exponent of the digraph may increase in the process.

A better way of producing primitive digraphs from a digraph G, is to take G^k where $k \geq 2$, is the index of imprimitivity of G. The following theorem is useful in the study of the spectrum of G^k.

Theorem 2.7.1 *(Sylvester [Syl83]) Let $A_0, \ldots, A_{k-1}$ be matrices such that A_i has order $n_i \times n_{(i+1) \bmod k}$ for $0 \leq i \leq k-1$. Define $B_i = A_i A_{i+1} \cdots A_{i-1}$ for $0 \leq i \leq k-1$ (subscripts taken modulo k). Then the matrices B_i and B_j have the same nonzero eigenvalues (with the same multiplicity) while the multiplicities of the 0 eigenvalue differ by $n_i - n_j$.*

Proof Assume first that $k = 2$. Define the matrices:

$$M = \begin{pmatrix} xI_{n_0} & -A_0 \\ 0 & I_{n_1} \end{pmatrix} \quad N = \begin{pmatrix} I_{n_0} & A_0 \\ A_1 & xI_{n_1} \end{pmatrix}$$

Then one has:

$$MN = \begin{pmatrix} xI_{n_0} - A_0 A_1 & 0 \\ A_1 & xI_{n_1} \end{pmatrix} \quad NM = \begin{pmatrix} xI_{n_0} & 0 \\ xA_1 & xI_{n_1} - A_1 A_0 \end{pmatrix}$$

And therefore:

$$x^{n_1} det(xI_{n_0} - A_0 A_1) = x^{n_0} det(xI_{n_1} - A_1 A_0)$$

and the claim follows for $k = 2$. For $k > 2$ the claim follows immediately for B_i and $B_{(i+1) \bmod k}$ from the case $k = 2$, since $B_i = A_i(A_{i+1} \cdots A_{i-1})$ and $B_{i+1} = (A_{i+1} \cdots A_{i-1})A_i$. By repeated applications of this fact to consecutive pairs of indices it follows that the claim is true for any pair of indices. ∎

Corollary 2.7.2 *If the matrices $A_0, \ldots, A_{k-1}$ are square, then the matrices $B_0, \ldots, B_{k-1}$ have the same spectrum.*

Now let G be a strongly connected regular digraph on N vertices, with index of imprimitivity $k \geq 2$. Let A be the adjacency matrix of G. We may assume by theorem 2.3.3 that A is in a superdiagonal form:

$$\begin{pmatrix} 0 & A_0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & A_1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 & A_{k-2} \\ A_{k-1} & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}$$

where the block A_i has order $\frac{N}{k} \times \frac{N}{k}$ for $0 \leq i \leq k-1$. Thus one has:

$$A^k = B_0 \bigoplus, \ldots, \bigoplus B_{k-1}$$

where $B_i = A_i A_{i+1} \cdots A_{i-1}$ (subscripts taken modulo k). Since A^k has the eigenvalue r^k with multiplicity k, it follows by corollary 2.7.2 that each B_i has maximal eigenvalue r^k with multiplicity 1 and $\lambda_1(B_i) = (\lambda_{k+1}(G))^k < r^k$. Therefore by corollary 2.3.5 each B_i is primitive and its ratio $\frac{\log|\lambda_1|}{\log\lambda_0}$ is the same as the ratio $\frac{\log|\lambda_k|}{\log\lambda_0}$ of G.

Thus G^k has precisely k primitive components $C_0, \ldots, C_{k-1}$. We claim that the separation exponent of each C_j is no larger than that of G. To see this, assume without loss of generality that G is separable, and note that $C_j = G_j G_{j+1} \cdots G_{j-1}$ (subscripts taken modulo k), where $G_0, \ldots, G_{k-1}$ are the digraphs with adjacency matrices $A_0, \ldots, A_{k-1}$. The proof of theorem 2.6.4 shows that each G_i is primitive so that $\bar{\lambda}(G_i)$ is the norm of the restriction of A_i to the subspace perpendicular to the vector $(1, 1, \ldots, 1)$. Hence by theorem 2.6.4:

$$\bar{\lambda}(C_j) \leq \prod_{i=0}^{k-1} \bar{\lambda}(G_i) \leq \left(\max_i \bar{\lambda}(G_i) \right)^k = \left(\bar{\lambda}(G) \right)^k$$

And since C_j has degree d^k it satisfies:

$$\delta(C_j) \leq \frac{k \log \bar{\lambda}(G)}{k \log d} = \delta(G)$$

Thus each C_j is primitive with a separation exponent no larger than that of G. Note that another alternative is to take the digraphs G_i which also satisfy this property.

2.8 The Eigenvalue Method and Expansion.

Let G be a strongly connected regular digraph G on N vertices with adjacency matrix A. Tanner [Tan84] has observed that the spectrum of a digraph contains information about its expansion properties. Similar connections were later established with other quantities of the digraphs such its diameter [AM85, Chu] and its its independence number [Sar]. Since the connection between the spectrum of digraphs and their expansion properties plays an important role in this work, it will be described here in some detail.

Expansion is usually formulated in terms of lower bounds on the number of edges connecting one subset of vertices to another. Let G be a primitive d-regular digraph on N vertices and S a subset of vertices. There are $|S|d$ edges emanating out of the vertices of S. Consider a partition of the vertices of G into subsets $T_1, \ldots, T_k$ each of size $\frac{N}{k}$. There must be one subset T_i such that the number of edges from S to T_i does not exceed $\frac{d|S||T_i|}{N}$. Thus given two subsets of vertices S and T the number of edges connecting S to T cannot be guaranteed to exceed $\frac{d|S||T|}{N}$. On the other hand $\frac{d|S||T|}{N}$ is the average number of edges connecting S to T if the neighbours of S are picked at random. It turns out that the size of $\bar{\lambda}$ provides an upper bound on the extent to which G deviates from a random behaviour. To see this let $\chi_S, \chi_T \in R^n$ be the characteristic vectors of S and T respectively, let $\vec{1}$ denote the vector all of whose coordinates are 1, and let A be the adjacency matrix of G. Note that $\lambda_0 = d$ with the

eigenvector $\vec{1}$. If E_{ST} denotes the set of edges connecting S to T, then one has:

$$|E_{ST}| = \langle \chi_S A \chi_T \rangle$$

Set $W = SPAN\left\{\vec{1}\right\}$ and let P be the orthogonal projection of R^n onto W. Writing $\chi_S, \chi_T \in R^n$ as sums of components along W and $W^\perp$ one gets:

$$\begin{aligned} \chi_S &= \frac{|S|}{\sqrt{n}}\vec{1} + z_s & z_s \in W^\perp \\ \chi_T &= \frac{|T|}{\sqrt{n}}\vec{1} + z_t & z_t \in W^\perp \end{aligned}$$

Therefore:

$$\left| |E_{ST}| - \frac{d|S||T|}{n} \right| \leq \|A_{W^\perp}\|_2 \|(1-P)\chi_S\|_2 \|(1-P)\chi_T\|_2$$

where $A_{W^\perp}$ is the restriction of A to $W^\perp$. Since:

$$\|A_{W^\perp}\|_2 = \bar{\lambda}$$

$$\|(1-P)\chi_S\|_2 = \sqrt{|S|\left(1 - \frac{|S|}{N}\right)}$$

$$\|(1-P)\chi_T\|_2 = \sqrt{|T|\left(1 - \frac{|T|}{N}\right)}$$

one arrives at:

Lemma 2.8.1 *[FP87, AC88] Let G be a primitive d-regular digraph on N vertices and S,T subsets of $V(G)$. Then:*

$$\left| E_{ST} - \frac{d|S||T|}{N} \right| \leq \bar{\lambda}\sqrt{|S||T|\left(1 - \frac{|S|}{N}\right)\left(1 - \frac{|T|}{N}\right)}$$

Corollary 2.8.2 *Let G be a primitive d-regular digraph on N vertices, and Z a subset of vertices s.t. $|Z| = N\alpha$. Then:*

$$\begin{aligned} ||E_{ZZ}| - dN\alpha^2| &\leq \bar{\lambda}N\alpha(1-\alpha) \\ ||E_{ZZ^c}| - dN\alpha(1-\alpha)| &\leq \bar{\lambda}N\alpha(1-\alpha) \end{aligned}$$

The following lemma can be derived by similar considerations:

Lemma 2.8.3 *[Pip85] Let G be a primitive d-regular digraph on N vertices with adjacency matrix A. Let $S \subset V(G)$ with $|S| = N\alpha$. Then the number of vertices, at least half of whose neighbours lie in S is bounded by:*

$$4N\alpha^2 + 4\left(\frac{\bar{\lambda}(A)}{\lambda_0(A)}\right)^2 N\alpha(1-\alpha)$$

While we are at it, we remark that the eigenvalue method provides information about other parameters of a digraph. Two examples are :

Theorem 2.8.4 *[Chu] Let G be a d-regular strongly connected digraph with index of imprimitivity k. Then:*

$$diam(G) \leq k \frac{\log \frac{|G|}{k}}{\log \frac{d}{\lambda}} + k - 1$$

Theorem 2.8.5 *[Sar] Let G be a d-regular connected graph and let $i(G)$ denote its independence number. Then:*

$$i(G) \leq \frac{\bar{\lambda}}{d}|G|$$

2.9 Explicit Definitions of Expanding Digraphs.

The eigenvalue method provides a guideline for constructing expanding digraphs. In section 2.8 lower bounds for the expansion of a digraph G, are given in terms of the separation between λ_0 and $\bar{\lambda}$. We have quantified this separation by the separation exponent $\delta(G)$ (see 2.5 for a definition). The eigenvalue method indicates that to get optimal lower bounds, one should strive to make the separation exponent as small as possible.

In dealing with **families** of expanding digraphs, one has to secure a lower bound on the expansion of every member of the family. This leads to the following definition:

Definition 2.9.1 *Let $0 < \eta < 1$. A family of digraphs $\{G_i\}$ is η-separable if $\delta(G_i) \leq \eta$ for each i.*

The smaller η is, the better the lower bound on the expansion of each member of the family.

As will be seen in chapters 6 and 7, there are various algorithms that use digraphs **of size exponential in the input length**, and require separability to ensure a good performance. For these algorithms one needs explicit definitions of separable graphs, with the property that certain graph operations can be performed in a time polylogarithmic in the size of the graphs. Such definitions are called **efficient constructions** and are defined in section 4.4. Our programme for the rest of the section is to study the following questions:

1. Can one give efficient constructions of separable digraph families?

2. If so, for what values of η ?

The answer to the first question turns out to be positive. In fact a separable family of regular bipartite graphs with a constant bound on their degree, is precisely a family of expander graphs, and various explicit constructions of expander families are known. To see this recall the definition of expander graphs:

Definition 2.9.2 *Let $G = (E, V)$ be a bipartite graph on the sets of vertices $V_0(G)$ and $V_1(G)$ where $|V_0(G)| = |V_1(G)| = N$. For a subset X of V put:*

$$\Gamma(X) = \{v \in V : vx \in e \ for \ some \ x \in X\}$$

G is an (N, d, c) expander if its maximal degree is bounded by d, and for every $X \subset V_1(G)$:

$$|\Gamma(X)| \geq \left(1 + c(1 - \frac{|X|}{N})\right)|X|$$

A family $\{G_i\}_{i=1}^{\infty}$ of bipartite graphs, is a family of (d,c) expander graphs if each member G_i is an $\left(\frac{|G_i|}{2}, d, c\right)$ expander.

The following theorem shows that (constant degree) separable families and expander families are qualitatively the same:

Theorem 2.9.1 *Assume G is a d-regular bipartite graph with $|V_0(G)| = |V_1(G)| = N$. then:*

1. *[Alo86a] If G is an (N, d, c) expander then:*

$$\bar{\lambda} \leq d - \frac{c^2}{1024 + 2c^2}$$

2. *[Tan84, AM85, Alo86a] If $\bar{\lambda} \leq d - \epsilon$ then G is an $\left(N, d, \frac{2d\epsilon - \epsilon^2}{d^2}\right)$ expander.*

Therefore one gets explicit constructions of separable primitive digraphs by using explicit constructions of expander families and theorem 2.6.3. Various explicit constructions of expander families have been given [Mar75, GG81, Buc86, AM85, JM85]. In subsequent chapters we we shall use the family of Gabber and Galil expander graphs [GG81] defined as follows: For each $n = m^2$, G_n is a bipartite graph on $2n$ vertices. with the vertex sets $V_0 = Z_m \times Z_m$ and $V_1 = Z_m \times Z_m$. The edges between V_0 and V_1 are given by the permutations:

$$\sigma_0(x, y) = (x, y)$$
$$\sigma_1(x, y) = (x, x + y) mod \ m$$
$$\sigma_2(x, y) = (x, x + y + 1) mod \ m$$
$$\sigma_3(x, y) = (x + y, y) mod \ m$$
$$\sigma_4(x, y) = (x + y + 1, y) mod \ m$$

Although the proofs of expansion of these constructions are fairly complicated, the implementations of the constructions by polynomial time algorithms are simple i.e. these constructions are efficient.

Now to the second question. For what values of η can one efficiently construct η-separable families? Unfortunately, one runs into a lower bound. Alon and Boppana showed that for a constant degree family of d regular graphs:

$$\liminf_{i \to \infty} \bar{\lambda}(G_i) \geq 2(\sqrt{d - 1})$$

This can be extended to the case of unbounded degree:

Theorem 2.9.2 *Let $\{G_i\}_{i=1}^{\infty}$ be a family of primitive d_i regular digraphs with adjacency matrices $\{A_i\}_{i=1}^{\infty}$ and $|G_i| = N_i$. Suppose:*

$$d_i = o\left(N_i\right)$$

then:

$$\frac{\bar{\lambda}(G_i)}{\sqrt{\lambda_0(G_i)}} \geq 1 - o(1)$$

Proof We will show that:

$$\frac{|\lambda_1(G_i)|}{\sqrt{\lambda_0(G_i)}} \geq 1 - o(1)$$

The theorem then follows since $|\lambda_1| \leq \bar{\lambda}$ for any primitive regular digraph.

As usual let $\lambda_0, \ldots, \lambda_{n-1}$ denote the eigenvalues of A_i in nonincreasing moduli order. Since A_i is similar to a triangular matrix and the trace is invariant under a similarity transformation:

$$\sum_{k=0}^{N_i-1} \lambda_k^2 = \sum_{k=0}^{N_i-1} (A_i^2)_{kk}$$

Using:

$$\begin{aligned}
\lambda_0^2 &= d_i^2 \\
(A_i^2)_{kk} &= d_i
\end{aligned}$$

one gets:

$$(N_i - 1)\lambda_1^2 \geq \sum_{k=0}^{N_i-1} \lambda_k^2 - \lambda_0^2 = N_i d_i - d_i^2$$

Therefore:

$$\lambda_1^2 \geq \left(\frac{N_i - d_i}{N_i - 1}\right) d_i$$

And:

$$\frac{|\lambda_1(G_i)|}{\sqrt{d_i}} \geq \sqrt{\left(\frac{N_i - d_i}{N_i - 1}\right)}$$

The statement of the theorem follows, by taking the limit $N_i \mapsto \infty$. ∎

Thus in an infinite family of digraphs the separation exponent is ultimately bounded below by $\frac{1}{2} - o(1)$. Can one realize the lower bound in efficient constructions? In the case of a bounded degree, a subtle difficulty crops up. [LPS86] have given explicit definitions of $\frac{1}{2}$-separable constant degree families, using nontrivial group-theoretic notions, and with cardinalities defined in terms of primes. It not clear that these construction are efficient. On the other hand, the constructions of [Mar75, GG81, Buc86, AM85, JM85] are far from optimal, yielding separation exponents close to 1.

In the following sections it is shown that the situation is substantially simpler if the degree is not required to be bounded: various constructions that can be implemented by algorithms do exist.

2.10 Cayley Digraphs.

There is a well known method of constructing digraphs from finite groups. From our standpoint these constructions have two important advantages: First, the adjacency matrix of the digraph has an orthonormal basis of eigenvectors. Second, explicit formulae can be given for all the eigenvalues in terms of the characters of the irreducible representations of the underlying group. Using corollary 2.6.2 $\bar{\lambda}$ can in some instances be computed exactly, and in other instances be bounded from above. These tools will be used in 2.11 to show that certain simple constructions of Cayley digraphs are $\frac{1}{2}$-separable.

Let H be a finite group with $|H| = N$ and suppose $S \subset H$.

Definition 2.10.1 *The Cayley digraph of H with respect to S is the digraph $G = (V, E)$ where $V = H$ and:*

$$E = \left\{ (xy) : x^{-1}y \in S \right\}$$

Under general conditions, the spectrum of the adjacency matrix of a Cayley digraph can be determined in terms of the irreducible representations of the group H. Let $\rho_1 \ldots \rho_r$ denote the irreducible representations of H, and let $\chi_1 \ldots \chi_r$ denote the corresponding characters. Also let ρ_l^{kj} be the kj entry of the irreducible representation ρ_l. Then:

Theorem 2.10.1 *[Dia] Let H be a finite group, and S be a union of conjugacy classes of H. If A is the adjacency matrix of the Cayley digraph of H with respect to S, then the ρ_i^{kj} form an orthogonal basis of eigenvectors for A. For a fixed $1 \leq l \leq r$ the eigenvectors ρ_l^{kj} belong to the eigenvalue λ_l:*

$$\lambda_l = \frac{1}{dim(\rho_l)} \sum_{s \in S} \chi_l(s)$$

2.11 $\frac{1}{2}$-separable Cayley Digraphs.

Because of theorem 2.9.2, $\frac{1}{2}$-separable digraph families are optimal from the standpoint of the eigenvalue method. At the end of section 2.9, the difficulties associated with efficient constructions of **constant degree** $\frac{1}{2}$-separable digraphs were pointed out. The situation is markedly different, if the degree is not required to be bounded. Various simple efficient constructions of $\frac{1}{2}$-separable digraphs with unbounded degrees do exist. In this section some constructions based on character sums estimates are described. In 2.12 a different method based on symmetric block designs is discussed. The constructions can be classified according to the degrees of the digraphs. Those of 2.11.2 produce degrees $\simeq |G|^{\frac{1}{k}}$, while those of 2.11.3 and 2.12, yield degrees $\simeq |G|^{\frac{k-1}{k}}$ ($k > 1$ is a fixed integer). All of them can be employed in the algorithms of chapters 6 and 7, although only those of 2.11.2 are actually used.

2.11.1 Separability and Characters Sums.

Let G be a Cayley digraph of the finite abelian group H with respect to the subset S. Since the group H is abelian, the irreducible representations are one-dimensional, and (by theorem 2.10.1) the eigenvalues of G are ordinary character sums. Also by corollary 2.6.2, one has $\bar{\lambda}(G) = |\lambda_k(G)|$ where k is the index of imprimitivity of G.

The digraph G, is $|S|$ regular, and it can be easily seen that the trivial character yields the maximal eigenvalue $\lambda_0 = |S|$. If one wishes to construct a primitive digraph for which the eigenvalue of second largest modulus (and hence $\bar{\lambda}$), is well separated from the maximal eigenvalue, one should find a set S such that:

$$\sum_{s \in S} \chi(s) \ll |S|$$

for every nontrivial character χ. Thus the construction should rely on some character sums estimates. This approach was introduced by Chung [Chu]. Note that one should like to make the set S as small as possible , in order to minimize the degree of G.

One useful estimate is the following:

Theorem 2.11.1 *(Katz - see [Chu]) Let Ψ be a nontrivial complex-valued multiplicative character defined on an extension field E over a field F with dimension t. Then for any $x \in E$ which generates E over F, we have:*

$$|\sum_{a \in F} \Psi(x + a)| \leq (t - 1)\sqrt{|F|}$$

Estimates on **Jacoby sums** also prove useful. Let $\phi_1, \ldots, \phi_k$ be k multiplicative characters of F_q. It is convenient to extend the definition of the ϕ_i to F_q by setting $\phi(0) = 1$ if ϕ is the trivial character and $\phi(0) = 0$ otherwise. The Jacobi sums are defined by:

$$J_a(\phi_1, \ldots, \phi_k) = \sum_{c_1 + \cdots + c_k = a} \phi_1(c_1) \cdots \phi_k(c_k)$$

The following theorem gives information about Jacobi sums:

Theorem 2.11.2 *(see [LN83])*

1. *If the multiplicative characters $\phi_1, \ldots, \phi_k$ are trivial then:*

$$J_1(\phi_1, \ldots, \phi_k) = q^{k-1}$$

2. *If some of but not all of the multiplicative characters $\phi_1, \ldots, \phi_k$ are trivial then:*

$$J_1(\phi_1, \ldots, \phi_k) = 0$$

3. *If the multiplicative characters $\phi_1, \ldots, \phi_k$ are nontrivial and $\phi_1 \cdots \phi_k$ is trivial, then:*

$$J_1(\phi_1, \ldots, \phi_k) = q^{\frac{k-2}{2}}$$

4. *If the multiplicative characters $\phi_1, \ldots, \phi_k$ are nontrivial and $\phi_1 \cdots \phi_k$ is nontrivial, then:*

$$J_1(\phi_1, \ldots, \phi_k) = q^{\frac{k-1}{2}}$$

2.11.2 Chung's Digraphs.

The construction of Chung is obtained as follows: Let F be a finite field say of characteristic 2, and f an irreducible polynomial of degree t. Let E be the extension field of F formed by adjoining a root w of f to F, and g a generator for E^* (the multiplicative group of invertible elements of E). We shall use the notation $Ind(x)$ to denote the discrete log of x with respect to g. Let S be the set of natural numbers in $[0..|E^*| - 1]$ given by:

$$S = \{Ind(w + i)\}_{i \in F}$$

We may now define G the Cayley digraph on $Z_{|E^*|}$ with respect to S. Then from theorems 2.10.1 and 2.11.1 it follows that the digraph G is primitive and satisfies:

$$\bar{\lambda} \leq (t - 1)\sqrt{\lambda_0}$$

So that if t is fixed, G is a $\frac{1}{2}$-separable digraph, of degree $\sim |G|^{\frac{1}{t}}$.

We shall think of G as a digraph on E^* defined by the rule: there is a directed edge from x to y, if :

$$(Ind(x) - Ind(y)) \in S$$

that is, there is a directed edge from x to y if $(xy^{-1} - w) \in F$. Note that this definition is generator independent.

2.11.3 Jacobi digraphs.

Jacoby digraphs are efficiently constructible $\frac{1}{2}$-separable digraphs, with degrees $\simeq$ $|G|^{\frac{k-1}{k}}$ ($k > 1$ is a fixed integer). A Jacoby digraph is the following Cayley digraph G: The group H is the direct product of k copies of F_q^*, so that an element of H is a k-tuple $(x_1, \ldots, x_k)$ of elements of F_q^*. The set S is the set:

$$\left\{ (x_1, \ldots, x_k) : x_i \in F_q^* \ \& \ \sum_{i=1}^{k} x_i = 1 \right\}$$

Since G is the direct product of F_q^*, the characters of G are just the products of k characters of F_q^*. Therefore theorem 2.10.1 implies that the spectrum of G is given by:

$$\lambda_i = \bar{J}_1(\phi_1^i, \ldots, \phi_k^i) = \sum_{\substack{x_l \in F_q^* \\ \sum x_l = 1}} \phi_1^i(x_1) \cdots \phi_k^i(x_k)$$

Where ϕ_j^i is multiplicative character of F_q^*. If all the ϕ^i are trivial, λ_i is equal to $(q - 1)^k$ and is the maximal eigenvalue. Otherwise, we observe that the quantity $\bar{J}_1$ resembles a Jacoby sum, except that the summation extends over tuples of elements from F_q^* rather than F_q. Using the inclusion exclusion principle and theorem 2.11.2, it can be verified that:

$$|\bar{J}_1| \leq q^{\frac{k-1}{2}}$$

Hence the digraph G is $\frac{1}{2}$-separable.

2.12 BIBDs and $\frac{1}{2}$-separable digraphs.

There is another method of for achieving $\frac{1}{2}$-separability, which is conceptually simpler, in that it does not rely on the deep results of character sums. This method uses symmetric BIBDs (balanced incomplete block designs). As in 2.11.3, the degrees of these digraphs are $\simeq |G|^{\frac{k-1}{k}}$.

A BIBD consists of v elements and b subsets of these elements (blocks), such that:

1. Each block contains k elements

2. Each element is contained in r blocks

3. Each pair of elements is contained in γ blocks

Two basic relations are:
$$\begin{aligned} vr &= bk \\ \gamma(v-1) &= r(k-1) \end{aligned}$$

A BIBD is symmetric if $r = k$. In such a case one has $v = b$. The incidence matrix of the BIBD is a $v \times b$ matrix defined by:

$$(C)_{ij} = \begin{cases} 1 & \text{if element i} \in \text{block j} \\ 0 & \text{otherwise} \end{cases}$$

It satisfies the equation:
$$CC^T = (r - \gamma)\mathrm{I} + \gamma\mathrm{J}$$

where $\boldsymbol{J}$ is the $v \times v$ matrix with 1 at every place. The spectrum of of CC^T can be readily calculated. The largest eigenvalue is k, and it has multiplicity 1. All other eigenvalues have the value $\sqrt{k - \gamma}$. Let G be the digraph whose adjacency matrix is C. Then G is a k-regular digraph. From theorem 2.6.1 it follows that:

$$\bar{\lambda}(G) < \sqrt{\lambda_0(G)}$$

and thus G is $\frac{1}{2}$ separable.

One common example of symmetric block designs are projective geometries [4]. Let PG(n,q) be the projective geometry of dimension n over the field of q elements. Then we have Singer theorem [Hal86]:

Theorem 2.12.1 *The hyperplanes of $PG(n, p^r)$ as blocks, and the points as objects, form a symmetric BIBD with ($q = p^r$):*

$$v = \frac{q^{n+1} - 1}{q - 1} \quad k = \frac{q^n - 1}{q - 1} \quad \gamma = \frac{q^{n-1} - 1}{q - 1}$$

Observe that the graphs we obtain have degree $d \sim |V(G)|^{\frac{n-1}{n}}$.

---

[4]These were first pointed out to be expanders by Alon [Alo86b]

Chapter 3

Probabilistic Algorithms.

This chapter reviews briefly, the basic notions of probabilistic algorithms, needed in this work:

1. Probabilistic Turing machines.

2. The RP and BPP complexity classes.

3. The performance of a probabilistic algorithm.

4. Random sources.

3.1 The Computational model.

The notion of a probabilistic algorithm depends of course, on the computational model in question. Although the methods to be developed, apply to a wide variety of computational models, attention is focused on **probabilistic Turing machines** (PTMs). A PTM is a nondeterministic Turing machine with the property that there are precisely two choices for each step of the computation. The computation is thought to proceed in the following way: at each stage the machine tosses a coin and chooses one of the two possible computation steps, according to the outcome of the coin toss.

If there are no space limitations on the PTM, it may record the result of each coin toss and consult it later. In such a case, the sequence of the outcomes of the coin tosses may be regarded as an additional input, and a PTM may alternatively be viewed as a deterministic Turing machine T with two inputs: the **actual input** and the **random input**. The actual input is the input whose membership in some language $\mathcal{L}$ is to be decided. The random input plays the role of the sequence of coin tosses. The computation of the PTM consists of two phases:

1. The **sampling phase** - the PTM samples a random input according to some probability distribution. It is convenient to think of the sampling as being implemented by a **random source**. The notion of a random source is discussed in section 3.4.

2. The **computation phase** - the PTM runs T on the actual input and the sampled random input, interpreting the random input as a sequence of outcomes of coin flips, and using it to make the nondeterministic choices.

Given a PTM and an actual input x, the set of random inputs that produce the correct answer for x is called the **witness set** of x. Its complement is called the **nonwitness** set of x. The error probability for the actual input x, is just the probability given to the nonwitness set of x by the probability distribution on the random inputs.

3.2   Probabilistic Complexity Classes.

Efficient computation on PTMs is captured by the RP and BPP complexity classes, representing fast probabilistic algorithms with a one sided and two sided error respectively:

Definition 3.2.1   *1. A polynomial time PTM is an RP PTM for the language $\mathcal{L}$ if for some constant $0 < \delta < 1$:*

(a) if $x \in \mathcal{L}$ then the fraction of the accepting computations exceeds δ.

(b) if $x \notin \mathcal{L}$ then all the computations reject.

2. A polynomial time PTM is a BPP PTM for the language $\mathcal{L}$ if for some constant $\delta \in (0, \frac{1}{2})$:

(a) if $x \in \mathcal{L}$ then the fraction of the accepting computations exceeds $\frac{1}{2} + \delta$.

(b) if $x \notin \mathcal{L}$ then the fraction of the rejecting computations exceeds $\frac{1}{2} + \delta$.

Definition 3.2.2   *1. The RP complexity class is the class of languages that have an RP PTM.*

2. The BPP complexity class is the class of languages that have a BPP PTM.

Thus a language is in the RP (BPP) complexity class iff there exists a polynomial time probabilistic algorithm, that decides it with a small error probability **provided the random inputs are sampled with a uniform probability distribution**. The problem of maintaining a small error probability when the random inputs are sampled with a nonuniform probability distribution is a deep one and has attracted a considerable amount of attention. It constitutes the subject matter of chapter 7.

3.3   Performance of a Probabilistic Algorithm.

Since every probabilistic computational model can essentially be viewed as a deterministic model with a probability distribution on one of its inputs, the "performance" of a probabilistic algorithm consists of two parts:

The deterministic part:
The ordinary computational resources of the model such as space, running time etc. Since since this work focuses attention on RP and BPP algorithms, the only ingredient of the deterministic performance to be considered , is the running time. As is customary in complexity theory, we will content ourselves with the polynomiality (in the input length) of the running time, without bothering to determine the smallest polynomial bound.

The probabilistic part:

- **The number of random bits used**. This must of course be polynomial in the input length, but in some cases we shall attempt to achieve a polynomial running time and a given error bound, with as few random bits as possible (see chapter 6).

- **The error probability**. The error probability is in general a function of the input length and is desired to decrease with the input length as quickly as possible. This may not be evident from the definitions for the RP and BPP algorithms as formulated above, since only a constant upper bound for the error probability is required. However, given a polynomial time probabilistic algorithm A which decides a language $\mathcal{L}$ with n random bits and a constant bound on its error probability, one can design a new polynomial time probabilistic algorithm B which decides $\mathcal{L}$ with an error probability that decreases exponentially with input size: B runs A n independent times and accepts if there is at least one accepting computation (RP), or if a majority of the computations accept (BPP). Note that one must increase the running time in order to achieve a smaller error, but the extra cost is polynomial in the input length. Also the amount of random bits used, increases by a factor of n.

3.4   Random Sources.

The coin tosses of the PTM represent the randomness of the computation. It is convenient to think of them as being implemented by a **random source**. The notion of a random source provides a convenient conceptual framework within which, the randomness can be isolated from the computational details of the algorithm.

In information theory a random source is a device that outputs a sequence of symbols from a finite alphabet Σ (we always assume $\Sigma = 0, 1$), according to some probability distribution. [1] In theoretical computer science, one would like a random source to model the more general situation, where one has some uncertainty or only a partial information about the probability distribution of the output. Such information is given as a set of conditions on the probability distribution and these determine a class of allowable distributions. Accordingly we adopt the following definition:

---

[1] In view of the correspondence between infinite binary sequences and the interval $(0, 1]$ a random source may equivalently be described as a probability measure on the Borel sets of the interval $(0, 1]$.

Definition 3.4.1 *A random source is sequence $\{\Pi_m\}_{m=1}^{\infty}$, where Π_m is a family of probability distributions on $\{0,1\}^m$. A **strategy** of the random source is a sequence $\{\pi_m\}_{m=1}^{\infty}$, where $\pi_m \in \Pi_m$ (i.e. a strategy is a sequence of allowable probability distribution - one distribution for each binary string length).*

Note that the definition generalizes the concept of a random source as it is usually formulated in information theory, in three respects:

1. More than one probability distribution is allowed for $\{0,1\}^m$.

2. No consistency conditions are required between probability distributions on $\{0,1\}^n$ and $\{0,1\}^m$ for $m \neq n$.

3. The random source is not necessarily "sequential": the value of the i-th output symbol may depend on all other symbols not just the first i-1.

Given a strategy of the random source, the random variable associated with the i-th output symbol, is called the i-th **random bit**. A source which has only one strategy consisting of independent and unbiased (uniformly distributed) random bits, is called a **perfect random source**.

Chapter 4

Sampling Procedures and Disperser Graphs.

This chapter introduces the basic notions of sampling procedures, and disperser graphs, and studies the relation between them. The logical structure of the chapter is as follows:

1. The notion of a sampling procedure is defined (section 4.1).

2. We define certain statistical properties that measure the quality of sampling procedures. These properties are called **amplification** properties and will prove useful in the simulation of probabilistic algorithms (section 4.1).

3. It is shown that families of bipartite graphs give rise to sampling procedures (section 4.2).

4. It is shown that bipartite graphs with certain expansion properties (**disperser** graphs), give rise to sampling procedures with good amplification properties (section 4.2).

5. It is claimed that sampling procedures that are to be used in fast probabilistic algorithm must satisfy certain efficiency requirements. Such procedures are called **efficient** (section 4.3).

6. It is claimed that if a family of bipartite graphs is **efficiently constructible** (precise definition given), then the induced sampling procedure is efficient (section 4.4).

The main conclusion of this chapter, is the need for efficient constructions of disperser graphs. This subject is taken up in chapter 5 where efficient constructions of disperser graphs with better expansion properties than known so far are given. These constructions are applied in later chapters in simulations of probabilistic algorithms with weak randomness.

At this point we should perhaps remind the reader of the conventions adopted in this work concerning a connected bipartite graph G. The two subsets of vertices

which define the partition of G are denoted by $V_0(G)$ and $V_1(G)$ and their respective cardinalities are denoted by N and M. It is always assumed that $M \geq N$. Also, it is always assumed that vertices of $V_1(G)$ have the same degree denoted by $\bar{d}$.

4.1   Sampling Procedures.

Let $\Psi = \{\psi_k\}_{k=1}^{\infty}$ be a family of finite sets with increasing cardinalities. A **sampling procedure** for Ψ is an algorithm which for each k samples a finite multiset of elements of ψ_k in the following way:

1. It gets an output of $f(|\psi_k|)$ random bits from a random source (f is an increasing computable function).

2. It computes deterministically a finite multiset of elements of ψ_k of size d_k, from the output of the random source.

In this work sampling procedures are used to distinguish with a high confidence between sets whose size are sufficiently separated. More precisely, let S be a finite set and let $0 < \kappa < t < \eta < 1$ be numbers. We are interested in sampling procedures that sample a multiset of d elements from S ($d \ll |S|$), in such a way that the probability of hitting any set of size at least $|S|\eta$ (regardless of its structure) with dt elements is large, while the probability of hitting any set of size at most $|S|\kappa$ with dt elements is small.

A sampling procedure with the above property is useful in the following situation: Suppose $T \subset S$, with the property that given $x \in S$ it is easy to check whether $x \in T$ (say, in $DTIME(polylog|S|)$). Suppose further, that it is known that either $|T| \geq |S|\eta$ or $|T| \leq |S|\kappa$ and it is desired to determine which case occurs. The sampling procedure provides a fast probabilistic algorithm for this task: use the sampling procedure to pick d elements, check the membership of each element in T, and decide that $|T| \geq |S|\eta$, iff at least dt elements belong to T. Note that since $d \ll |S|$, this is much faster than the obvious deterministic algorithm that checks membership in T for each $x \in S$.

RP and BPP algorithms are one important instance, in which the above situation occurs. Here S is the set of random inputs, and T is the set of random inputs that lead to acceptance (for a given actual input x). If e is the error probability of the algorithm, then it is known that that either $|T| \geq |S|(1-e)$, or $|T| \leq |S|e$, according to the membership or nonmembership of x in the language.

There is of course a standard sampling procedure: get an output of $d_k \log |\psi_k|$ random bits (assume for simplicity that $|\psi_k|$ is a power of 2), and treat it as d_k blocks, each containing the encoding of one element from ψ_k. If $0 < \kappa < t < \eta < 1$ and the random bits come from a perfect random source, then by the Chernoff inequality the probability that more than $d_k t$ elements belong to a set of size $|\psi_k|\kappa$, or that less than $d_k t$ elements belong to a set of size $|\psi_k|\eta$, is $2^{-\Omega(d_k)}$. Thus if the size of a set is known to be either greater than $|\psi_k|\eta$, or less than $|\psi_k|\kappa$, one can determine which is the case with an error probability which is exponentially small in the size of the sample. This sampling procedure has however two drawbacks:

1. It is wasteful of random bits: $d_k \log |\psi_k|$ random bits are required, but we would like to achieve the same error probability, with significantly less random bits.

2. It is not guaranteed to provide the correct answer with a high probability, if the random bits do not come from a perfect random source.

Our goal in this work is in fact, to design sampling procedures that are less wasteful of random bits, and whose performance is as robust as possible with respect to the quality of the random bits.

The following **amplification** property will prove useful in designing sampling procedures that achieve the above behaviour:

Definition 4.1.1 *Let γ, δ, t be numbers in the interval $(0, 1)$. A sampling procedure for Ψ satisfies the (γ, δ, t) amplification property, if for any $W \subset \psi_k$ with $|W| \geq |\psi_k|\gamma$, the fraction of the multisets that intersect W with more than $d_k t$ elements, is at least δ.*

4.2 Disperser Graphs and Sampling Procedures.

Again let $\Psi = \{\psi_k\}_{k=1}^{\infty}$ be a family of finite sets with increasing cardinalities, and let $\{G_k\}$ be a sequence of bipartite graphs. Assume that:

1. $|V_0(G_k)| = |\psi_k|$ and $\phi : V_0(G_k) \mapsto \psi_k$ is a computable bijection.

2. $|V_1(G_k)| = 2^{m_k}$ for some positive integer m_k and $\varphi : \{0, 1\}^{m_k} \mapsto V_1(G_k)$ is a computable bijection.

Definition 4.2.1 *The sampling procedure for Ψ induced by the family of bipartite graphs $\{G_k\}$ is defined by:*

1. A vertex $x \in V_1(G_k)$ is sampled by obtaining a string of m_k random bits (from the output of a random source), and using the mapping φ.

2. A multiset of elements of ψ_k is obtained by computing the neighbours of x and applying to each of them the mapping ϕ

The advantage of working with sampling procedures induced by bipartite graphs, is that the amplification properties of the procedures can be linked to certain expansion properties of the graphs, and these can in turn, be studied by the algebraic methods described in chapter 2. What are then, the properties of a bipartite graph necessary to guarantee a good amplification in the induced procedure? They are given by the following definitions which generalize the approach of [Pip85], [Sip86] and [San]:

Definition 4.2.2 *Let G be a bipartite graph, and $0 < \alpha, \beta, < 1$ and $0 < t \leq 1$ be fixed numbers. G is an (α, β, t) (threshold) disperser, if it satisfies the following: if X is a subset of $V_0(G)$, of cardinality $N\alpha$, the cardinality of the set of vertices of $V_1(G)$ connected with at least $\bar{d}t$ of their edges to X, is at most $M\beta$. A family $\{G_n\}$ is a family of (α_n, β_n, t) dispersers if there exists n_0, such that for every $n \geq n_0$ G_n is an (α_n, β_n, t) disperser.*

An $(\alpha, \beta, 1)$ disperser will sometimes be referred to as an (α, β) OR disperser, while $(\alpha, \beta, \frac{1}{2})$ disperser will be called an (α, β) MAJORITY disperser. These specific types of dispersers occur in this work more often than arbitrary threshold dispersers.

There is a close relationship between the dispersion properties of a bipartite graph and the amplification properties of the induced sampling procedures. The formal statement of this relationship is deferred to chapter 6 (lemma 6.1.2), but in general the more dispersing the graph is, the better is the amplification of the induced sampling procedure.

4.3 Efficient Sampling Procedures.

The ultimate goal of this work, is to employ sampling procedures (constructed from disperser graphs), in simulations of polynomial time probabilistic algorithms, having access only to weak randomness. The simulations all have the following pattern:

1. A specific sampling procedure is used to pick a multiset of random inputs for the probabilistic polynomial time algorithm A.

2. A is run on the actual input, with each member of the multiset as a random input, and the decision to accept or reject is made on the basis of the results of all the runs.

What properties should one require from a sampling procedure if it is to be used in simulations of the above form? Clearly all the computations entailed by the sampling should be performable in a time polynomial in the input size. To see what these computations are, recall that if Ψ is a family of finite sets, a sampling procedure for Ψ is a procedure which for each k, gets an output of $f(|\psi_k|)$ random bits (from a random source S), and computes a finite multiset of elements of ψ_k from the output of the random source. Now in the above simulations, ψ_k is a set of random inputs of a polynomial time probabilistic algorithm so $\log |\psi_k|$ is polynomial in the input length of the algorithm. Therefore a sampling procedure can be used a in probabilistic polynomial time algorithms if it is **efficient** in the following sense:

Definition 4.3.1 *A sampling procedure for Ψ is **efficient** if:*

1. *$f(|\psi_k|) = O\left(polylog(|\psi_k|)\right).$*

2. *$f \in DTIME\left(polylog(|\psi_k|)\right).$*

3. *The computation of the multiset is in $DTIME\left(polylog(|\psi_k|)\right)$. In particular the size of the multiset is $O\left(polylog(|\psi_k|)\right)$.*

4.4 Efficient Constructions of Bipartite Graphs.

In section 4.2 it was seen how bipartite graphs give rise to sampling procedures. However in section 4.3 it was maintained, that simulations of probabilistic algorithms

required **efficient** sampling procedures. Thus the next natural question is: what properties should a family of bipartite graphs satisfy if it is to induce an **efficient** sampling procedure?

There are two basic requirements:

Density of cardinalities.

Our intention is to identify vertices of digraphs with random inputs of probabilistic algorithms, i.e. with elements of sets whose size is a power of 2. However, the cardinalities of the members of an explicitly constructible digraph family usually constitute a proper subset of the natural numbers, and are not necessarily powers of 2. Such a disparity can be handled as long as the gaps between the cardinalities of the graphs, and the sets of random inputs are not too large. Specifically, it will be required that there exists a natural number k, such that for every natural number N there is an index i satisfying:

$$N \leq |V_0(G_i)| \leq N(\log N)^k.$$

For the need for this specific density requirement, see the discussion concerning definition 6.1.3.

Polynomiality of relevant graph operations.

Since $V_0(G)$ is interpreted as the set of random inputs of a probabilistic algorithm A, $\log |V_0(G)|$ is polynomial in the input length of A, and it suffices to require that all sampling operations be performed in a time polynomial in $\log |V_0(G)|$. The sampling operations consist of obtaining $\lceil \log |V_1(G)| \rceil$ random bits from a random source to pick an element of $V_1(G)$, and computing the $\bar{d}$ neighbours of the picked vertex. Thus $\log |V_1(G)|$ should be polynomial in $\log |V_0(G)|$, each neighbour should be computed in a time polynomial in $\log |V_0(G)|$, and $\bar{d}$ should be polynomial in $\log |V_0(G)|$.

Construction of bipartite graph families, satisfying the above requirements, will be termed **efficient constructions**. Their properties are summarized in the following definition:

Definition 4.4.1 *Let $G = \{G_i\}_{i=1}^{\infty}$ be a family of bipartite graph. Then G is efficiently constructible if:*

1. *There exists a natural number k, such that for every natural number N there is an efficiently computable index i satisfying:*

$$N \leq |V_0(G_i)| \leq N(\log N)^k.$$

2. *Given an encoding of a vertex v of $V_1(G_i)$, its multiset of neighbours can be computed in $DTIME\,(polylog(|V_0(G_i)|))$. In particular we must have:*

   (a) *$\bar{d}$ is bounded by $polylog(|V_0(G_i)|)$.*

   (b) *$\log |V_1(G_i)| = O(polylog\,(|V_0(G_i)|))$.*

4.5 Efficient Constructions and Finite field Arithmetics.

In various instances the vertices of digraphs employed in algorithms, are elements of a finite field $GF(p^n)$. In such cases finite field arithmetic must be performed efficiently. To this end one can use the following theorem:

Theorem 4.5.1 *[Sho88] Let F be a finite field of characteristic p. Given an extension K of F of degree d, we can construct an irreducible polynomial over K of degree n, deterministically with*

$$O\left(p^{\frac{1}{2}+\epsilon}(nd)^{3+\epsilon} + (\log p)^2(nd)^{4+\epsilon}\right)$$

operations in F.

Consider the case where the digraph G is defined on $GF(p^n)$. Since in the applications we have in mind, the size of the digraphs is exponential in the input size of the algorithm, n is polynomial in the input size if p does not change with input length. An algorithm using G, typically has to perform arithmetic operations such as addition, subtraction multiplication, and division in a time polynomial in n. To see what is involved, let r be an irreducible polynomial over $GF(p)$ of degree n. Then the finite field $F = GF(p^n)$ can be viewed as the pair consisting of the vector space of polynomials over $GF(p)$ of degree bounded by $n-1$, and the irreducible polynomial r, and the arithmetic operations are implemented in the following way:

1. Each element of $GF(p^n)$ is a polynomial over $GF(p)$ of a degree $\leq n-1$.

2. Addition of two elements is just the ordinary polynomial addition.

3. Multiplication is the ordinary polynomial multiplication taken modulo r.

4. The inverse of a non zero polynomial f is found by applying the euclidean algorithm to f and r.

By theorem 4.5.1, if one adheres to a fixed characteristic p or does not let the characteristic grow too fast (as a function of input size), one can find an irreducible polynomial of degree n over $GF(p)$ and hence perform arithmetic operations, in a time polynomial in the input length.

Chapter 5

Efficient Constructions of Disperser Graphs.

The main goal of this chapter is to provide efficient constructions of disperser graphs with amplification properties that have not been achieved before. The constructions use a method of [AKS87] based on random walks on expander graphs, and dense Cayley digraphs with an almost optimal separation between their first and second eigenvalue.

From now on we adopt the following point of view: if G is an (α, β, t) disperser, α and β are thought of as probabilities, and G is said to **amplify** a probability $1 - \alpha$ to $1 - \beta$. Also because of the intended applications, $N = |V_0(G)|$ is considered exponential in the input length. Therefore the following terminology is used:

- If $x \in [0, 1]$ is a number, and $p(N)$ is a probability such that $|p - x| = O\left(\frac{1}{polylog(N)}\right)$ then $p(N)$ is said to be **polynomially close** to x. Similarly if $|p - x| = O\left(\frac{1}{N^c}\right)$ for some positive constant c, then $p(N)$ is said to be **exponentially close** to x.

- If $x \in [0, 1]$ is a number, and $p(N)$ is a probability such that $|p - x| = \Omega\left(\frac{1}{polylog(N)}\right)$ then $p(N)$ is said to be **polynomially apart** from x.

- An amplification of a probability bounded by a constant, to a probability polynomially close to 1, is called a **polynomial** amplification.

- An amplification of a probability bounded by a constant, to a probability exponentially close to 1, is called an **exponential** amplification.

The logical structure of the chapter is as follows:

1. It is shown that disperser graphs with strong amplification properties do exist (section 5.1).

2. In section 5.2 it is shown that results of [KPS85] and [CG86] can be interpreted as efficient constructions of dispersers which can achieve a polynomial amplification. These dispersers can amplify very small probabilities up to constant

probabilities which can then be further amplified by the stronger dispersers constructed in 5.3.

3. A method is given for constructing exponentially amplifying disperser graphs (section 5.3).

4. Efficient constructions of exponentially amplifying dispersers is given on the basis of the method of section 5.3 and chapter 2 (section 5.4).

It should be pointed out that the gap between the best explicit constructions of disperser graphs and the probabilistic constructions, is still very large. This is a typical situation in the field of expanding digraphs: the existence of strong expansion can be easily demonstrated by probabilistic considerations, but efficient constructions, are far more difficult to come by. Efficient constructions of dispersers with a better amplification than those constructed in this chapter, would imply improvements of all the results of chapters 6 and 7, and may have far reaching implications on other problems in theoretical computer science (see for instance [Sip86, UW87]).

5.1 The Existence of Disperser Graphs.

The first issue that needs to be addressed, is the range of parameter values for which (α, β, t) disperser graphs do exist. For the applications of this work, it is convenient to set $\alpha = N^{\theta-1}$ and $\beta = M^{\mu-1}$ and enquire about the values of the exponents θ and μ for which disperser graphs exist. Clearly the larger the degree of the graph, the better the parameters that can be achieved. The next theorem shows that the condition:

$$\bar{d} \approx \frac{(1-\mu)\log M}{(1-\theta)\log N}$$

suffices to ensure the existence of dispersers with parameters μ, θ. The proof uses a counting sieve argument similar to those given in [Sip86, San]:

Theorem 5.1.1 Let $0 < \theta, \mu < 1$. An $\left(N^{\theta-1}, M^{\mu-1}, t\right)$ disperser exists if:

$$\bar{d} > \frac{(1-\mu)\log M}{(1-\theta)\log Nt - H(t)} + \frac{N^{\theta}}{M^{\mu}}\left(\frac{1}{t - \frac{H(t)}{(1-\theta)\log N}}\right) + \frac{\log e}{(1-\theta)\log Nt - H(t)}\left(1 + \frac{N^{\theta}}{M^{\mu}}\right)$$

Where $H(t)$ is the binary entropy function.

Proof For each vertex of $V_1(G)$ pick its $\bar{d}$ neighbours at random from $V_0(G)$. For any $S \subset V_0(G)$ and $T \subset V_1(G)$ with $|S| = N^{\theta}$ and $|T| = M^{\mu}$, let $A_{S,T}$ denote the event that each vertex of T has at least $\bar{d}t$ neighbours in S. The event that G is not an $\left(N^{\theta-1}, M^{\mu-1}, t\right)$ disperser, is contained in the event $\bigcup_{S,T} A_{S,T}$ whose probability is bounded by:

$$\binom{M}{M^{\mu}}\binom{N}{N^{\theta}}\left(\frac{\bar{d}}{\bar{d}t}\right)^{M^{\mu}}\left(N^{\theta-1}\right)^{\bar{d}tM^{\mu}}$$

Hence the desired graph exists if the above quantity is less than 1. Let $H(x)$ be the binary entropy function. Since $\binom{n}{n\alpha} < 2^{nH(\alpha)}$ it suffices to require that:

$$2^{MH(M^{\mu-1})}2^{NH(N^{\theta-1})}2^{\bar{d}H(t)M^{\mu}}2^{(\theta-1)\bar{d}t\log NM^{\mu}} < 1$$

Therefore the disperser graph exists if:

$$\left[\frac{H(x)}{x}\right]_{x=M^{\mu-1}} + \frac{N^{\theta}}{M^{\mu}}\left[\frac{H(x)}{x}\right]_{x=N^{\theta-1}} < \bar{d}\left(t(1-\theta)\log N - H(t)\right)$$

and since:

$$\frac{H(x)}{x} \le \log\frac{1}{x} + \log e$$

it suffices to require that:

$$((1-\mu)\log M + \log e) + \frac{N^{\theta}}{M^{\mu}}\left((1-\theta)\log N + \log e\right) < \bar{d}\left(t(1-\theta)\log N - H(t)\right)$$

And the conclusion of the theorem follows. ∎

Theorem 5.1.1 establishes the existence of graphs that play an important role in chapter 7. The next corollary is discussed after theorem 7.3.1:

Corollary 5.1.2 *Let k be a positive integer and let α, t be fixed real numbers in the interval $(0,1)$. For every N and $M = N^m$ with $m = \lceil(\log N)^k\rceil$, there exist an $(\alpha, M^{\mu-1}, t)$ disperser, with $\mu = (\log M)^{-\frac{k}{k+1}}$, and a degree $\bar{d} = O\left((\log N)^k\right)$.*

Theorem 5.1.1 gives the sufficient upper bound $\approx \frac{(1-\mu)\log M}{(1-\theta)\log N}$ on $\bar{d}$ for the existence of disperser graphs with parameter values μ, θ. The next theorem shows that these bounds are virtually tight:

Theorem 5.1.3 *Let $0 < \theta, \mu < 1$. Let $\{N_i\}$ $\{M_i\}$ $\{\bar{d}_i\}$ be increasing sequences of natural numbers such that $\bar{d}_i = O\left(\text{polylog}N_i\right)$ and $M_i < N_i^{\frac{\bar{d}_i}{1-\mu}}$. Then for a sufficiently large i an $\left(N_i^{\theta-1}, M_i^{\mu-1}\right)$ OR disperser with degree $\bar{d}_i$ exists only if:*

$$\bar{d}_i > \left(\frac{1-\mu}{1-\theta + \frac{\log(\bar{d}_i+1)}{\log N_i}}\right)\frac{\log M_i}{\log N_i}$$

Proof Let G be an $\left(N_i^{\theta-1}, M_i^{\mu-1}\right)$ OR disperser with degree $\bar{d}_i$. We may view each vertex of $V_0(G)$ as an element of $[1..N_i]$, and each vertex of $V_1(G)$ as a $\bar{d}_i$ tuple, of elements from $[1..N_i]$ (i.e. the $\bar{d}_i$ tuple of vertices to which it is connected). Thus each vertex of $V_1(G)$ can be represented by a point in a $\bar{d}_i$ dimensional lattice of side length N_i. Define γ by the equality:

$$\frac{N_i^{\bar{d}_i}}{N_i^{\gamma\bar{d}_i}} = M_i^{1-\mu}$$

41

so that:

$$\bar{d}_i = \left(\frac{1-\mu}{1-\gamma}\right)\frac{\log M_i}{\log N_i} \tag{5.1}$$

Partition the lattice into cubes of side length N_i^γ. Since there are $M_i^{1-\mu}$ such cubes, one of them must contain a set S of at least M_i^μ elements of $V_1(G)$. S is connected in G to at most $\bar{d}_i(N_i^\gamma + 1)$ elements of $V_0(G)$. Since $\bar{d}_i < N_i^\gamma$ for a sufficiently large i, we have:

$$N_i^\theta \le (\bar{d}_i + 1)N_i^\gamma$$

Hence:

$$\gamma \ge \theta - \frac{\log(\bar{d}_i + 1)}{\log N_i} \tag{5.2}$$

Combining 5.1 and 5.2 we get the statement of the theorem. ∎

5.2 Polynomially Amplifying Dispersers.

At this stage we begin to create our supply of efficiently constructible dispersers. Our first step is to show that certain results of [KPS85] and [CG86] can be interpreted as efficient constructions of dispersers that achieve a polynomial amplification.

The main advantage of these constructions, is that they can amplify very small probabilities (polynomially apart from 0 in the OR case, and from $\frac{1}{2}$ in the MAJOR-ITY case) for which the more powerful dispersers to be constructed in section 5.3 do not work. Thus a typical exponential amplification of a small probability should proceed in two stages:

1. Pre-amplification: the dispersers of this section are used to boost the probability up to a value which is either constant or polynomially close to 1.

2. Exponential amplification: the dispersers of 5.3 are used to amplify the boosted probability further.

5.2.1 Polynomially Amplifying OR Dispersers.

The basis for the construction is the following theorem:

Theorem 5.2.1 *Let $\{G_i\}_{i=1}^\infty$ be a d-regular, η-separable, family of primitive digraphs on N_i vertices, where $\eta < 1$. Let P be a polynomial. Suppose $\alpha(n) < 1$ is function on the natural numbers which is polynomially apart from 1. Then there is a positive constant c such that W_i, the double cover [1] of $G_i^{c\log\log N_i}$ is a*

$$\left(\alpha(N_i), \frac{1}{P(\log N_i)}, 1\right)$$

disperser, with $V_0(W_i) = V_1(W_i) = N_i$.

---

[1] see 2.1 for the definition of a double cover of a graph.

Proof

We construct a graph H_i, on N_i vertices, such that every subset of $\frac{N_i}{P(\log N_i)}$ vertices has a positive number of edges into any set of $N_i(1-\alpha)$ vertices. Then by definition 4.2.2, the double cover of H_i is the desired disperser graph. By lemma 2.8.1 this condition is met if:

$$\frac{\bar{\lambda}}{\lambda_0} < \sqrt{\frac{1-\alpha}{P(\log N_i)}} \tag{5.3}$$

Now since:

$$
\begin{aligned}
\lambda_0((G_i)^k) &= (\lambda_0(G_i))^k \\
\bar{\lambda}((G_i)^k) &\leq (\bar{\lambda}(G_i))^k
\end{aligned}
$$

$H_i = G_i^{c \log \log N_i}$ satisfies:

$$\frac{\bar{\lambda}(H_i)}{\lambda_0(H_i)} \leq \left(\frac{\bar{\lambda}(G_i)}{\lambda_0(G_i)}\right)^{c \log \log N_i}$$

and since $\frac{\bar{\lambda}(G_i)}{\lambda_0(G_i)} \leq \lambda_0^{\eta-1} < 1$, inequality 5.3 can be satisfied asymptotically, by taking a large enough positive c. ∎

Corollary 5.2.2 *Let P be a polynomial. Suppose $\alpha(n) < 1$ is function on the natural numbers which is polynomially apart from 1. Then there exists an efficiently constructible family of*

$$\left(\alpha(N_i), \frac{1}{P(\log N_i)}\right)$$

OR disperser with $N_i = M_i = 2^i$.

Proof Use theorem 5.2.1 with the family $\{G_i\}_{i=1}^{\infty}$ constructed from the Gabber-Galil expander graphs (see section 2.9). These are defined for cardinalities of the form $2n^2$. For each i define a digraph G_i on 2^i vertices as follows:

1. $i = 2k + 1$ is **odd** - G_i is a [GG81] expander on $2(2^k)^2$ vertices, with a self loop for each vertex to make it primitive.

2. $i = 2k$ is **even** - G_i is the **reduced** digraph [2] of a [GG81] expander on $2(2^k)^2$ vertices.

The digraphs G_i are of a bounded degree. Corollary 2.6.2, theorem 2.6.3, and theorem 2.9.1 show that they are primitive and η-separable for some $\eta < 1$. Therefore theorem 5.2.1 applies. Since each neighbour of a vertex in G_i can be computed in a time polynomial in i and the degree of H_i is polynomial in i, efficient constructibility is ensured. ∎

---

[2]see 2.1 for the definition of a reduced digraph.

5.2.2   Polynomially Amplifying MAJORITY dispersers.

MAJORITY dispersers that achieve polynomial amplification of probabilities polynomially apart from $\frac{1}{2}$, are implicit in a result of Chor and Goldreich [CG86] (see also [ABI86]).

Assume $\bar{d} = \bar{d}(n)$ is an increasing function from the set of natural numbers into itself. Let $F = GF(2^n)$ be a finite field, S a fixed subset of $\bar{d}$ elements of F and $l \geq 2$ a fixed natural number. Define the bipartite graph family $\{G_n\}$ as follows:

- $V_0(G_n)$ consists of the elements of F.

- $V_1(G_n)$ is the set of polynomials over F of degree $\leq l - 1$.

- $f \in V_1(G_n)$ is connected to the $\bar{d}$-elements multiset $\{f(i) : i \in S\}$.

In order to analyze the expansion properties of G_n it is convenient to use probabilistic arguments. Consider the probability space formed by imposing a uniform probability distribution on $V_1(G_n)$. For each $i \in S$ define the random variable $F_i(f) = f(i)$. It can be easily verified that the random variables $\{F_i\}_{i \in S}$ are uniformly distributed and l-wise independent. Therefore one can estimate their deviation from full independence by the following Chebychev-type inequality:

Lemma 5.2.3 *[KRS88, MNN89] Let $l \geq 2$ be a positive integer and $X_1 \cdots X_{\bar{d}}$ be l-wise independent random variables. Assume $EX_i = 0$ and $E|X|^l \leq C$ for $1 \leq i \leq \bar{d}$. Then:*

$$Prob\left[\left|\sum_{i=1}^{\bar{d}} X_i\right| \geq \epsilon\right] \leq \frac{\binom{\bar{d}}{\lfloor \frac{l}{2} \rfloor} A(l, C)}{\epsilon^l}$$

Where: $A(l, C) = \binom{l + \lfloor \frac{l}{2} \rfloor - 1}{\lfloor \frac{l}{2} \rfloor - 1} \frac{l!}{2^{\frac{l}{2}}} C$

Thus one readily has:

Theorem 5.2.4 *For any $\alpha < t < 1$, G_n is a*

$$\left(\alpha, \frac{\binom{\bar{d}}{\lfloor \frac{l}{2} \rfloor} A(l, 1)}{\left(\bar{d}(t - \alpha)\right)^l}, t\right)$$

disperser.

Proof Suppose $T \subset V_0(G_n)$ with $|T| = 2^n \alpha$ and let χ_T be its characteristic function. Define the random variables:

$$\xi_i(f) = \chi_T(f(i)) - \alpha \qquad 1 \leq i \leq \bar{d}$$

44

Clearly the r.v. $\{\xi_i\}$ are l-wise independent with $E(\xi_i) = 0$ and $E|\xi_i|^l \leq 1$. If we pick a $v \in V_1(G)$ with uniform probability, the probability that at least $\bar{d}t$ of its edges will hit T is given by:

$$Prob\left[\sum_{i=1}^{d} \xi_i \geq \bar{d}(t-\alpha)\right]$$

So the statement of the theorem follows from lemma 5.2.3 and the definition of a disperser. ∎

Corollary 5.2.5 *Let P be a polynomial. Suppose $\alpha(n) < \frac{1}{2}$ is function on the natural numbers which is polynomially apart from $\frac{1}{2}$. Then there exists an efficiently constructible family $\{G_i\}$ of*

$$\left(\alpha(N_i), \frac{1}{P(\log N_i)}, \frac{1}{2}\right)$$

dispersers with $N_i = 2^i$ and $M_i = 2^{2i}$.

Proof Use theorem 5.2.4 with:

1. $F = GF(2^i)$.

2. $l = 2$.

3. $\bar{d}(i)$ satisfies: $\frac{P(i)A(2,1)}{(\frac{1}{2}-\alpha)^2} \leq \bar{d}(i) \leq O(poly(i))$.

   Each neighbour of a vertex of $V_1(G_i)$ can be computed with finite field arithmetic which can be performed efficiently (see the discussion in section 4.5). Their number $\bar{d}(i)$ is at most polynomial in i. Therefore efficient constructibility is ensured. ∎

5.3 Exponentially Amplifying Dispersers.

This section contains the main result of the chapter: constructions of dispersers graphs that amplify constant probabilities to probabilities exponentially close to 1. These constructions are based on a method of [AKS87] which uses random walks on expander digraphs. We begin with a definition:

Definition 5.3.1 *Let G be a digraph. The bipartite graph W_G^l has the vertex sets:*

$$\begin{aligned} V_0(W_G^l) &= & V(G) \\ V_1(W_G^l) &= & \text{the set of all walks on } G \text{ of length } l-1. \end{aligned}$$

A walk is connected to all its vertices (one edge for each occurrence).

   The main tool in the construction to follow, is generalization of a lemma of [AKS87]:

Lemma 5.3.1 *Let A be the adjacency matrix of a d-regular connected digraph on N vertices G, with index of imprimitivity k. Suppose S is a subset of the vertices with $|S| = N\alpha$ and let P_S be the matrix of the projection onto the coordinates in R^N corresponding to S, that is:*

$$(P_S)_{ij} = \begin{cases} \delta_{ij} & if \ v_i \in S \\ 0 & otherwise \end{cases}$$

then:

$$\|P_S A\|_2 \leq \sqrt{k\alpha\lambda_0^2 + \bar{\lambda}^2}$$

Proof

Recall the following facts (see 2.5):

1. A may be assumed without loss of generality, to be in a superdiagonal form. This means that the vertex set V of G can be partitioned into k subsets $V_0, \ldots, V_{k-1}$ each with cardinality $\frac{N}{k}$, in such a way that the edges going out of vertices of V_i, come into vertices of $V_{(i+1) mod\ k}$.

2. W, the subspace spanned by the eigenvectors corresponding to the eigenvalues of maximal modulus, is k-dimensional. The vectors:

$$v_i = \frac{1}{\sqrt{N}}(\overbrace{\theta_i^0, \cdots, \theta_i^0,}^{\frac{N}{k}}, \ldots, (\overbrace{\theta_i^{k-1}, \cdots, \theta_i^{k-1},}^{\frac{N}{k}})$$

are eigenvectors corresponding to the eigenvalues $d\theta_i$, and constitute an orthonormal basis for W. Here θ_i is the i-th root of unity of order k.

3. Both W and $W^\perp$ (the orthogonal complement of W), are invariant under A and A^T.

Now let v be a vector in W of unit length. Then $v = \sum_{j=0}^{k-1} a_j v_j$ with $\sum_{j=0}^{k-1} |a_j|^2 = 1$. The action of the operator $P_S A$ on v annihilates $N - |S|$ coordinates, and produces $|S|$ coordinates with values:

$$\lambda_0 \sum_{j=0}^{k-1} \frac{a_j}{\sqrt{N}} \theta_j^l$$

where $0 \leq l \leq k - 1$. Thus by the Cauchy-Schwartz inequality, the absolute value of each nonzero coordinate is bounded by $\lambda_0 \sqrt{\frac{k}{N}}$. We conclude that:

$$sup_{v \in W} \frac{\|P_S Av\|}{\|v\|} \leq \lambda_0 \sqrt{\frac{|S|k}{N}}$$

where $\|\ \|$ denote the L_2 norm. Furthermore, by definition, the norm of the restriction of A to $W^\perp$ is $\bar{\lambda}$, and the norm of P (being a projection operator) is 1, so that one has:

$$sup_{v \in W^\perp} \frac{\|P_S Av\|}{\|||v\|} \leq \bar{\lambda}$$

46

One can now study the action of P_S on $\vec{a}$ - an arbitrary vector of R^N, by representing $\vec{a}$ as a sum:

$$\vec{a} = \vec{v} + \vec{w} \qquad \vec{v} \in W^{\perp} \qquad \vec{w} \in W$$

and looking at the action of P_S on each of the components. By the subadditivity of the norm, and the Cauchy-Schwartz inequality, one gets:

$$\begin{aligned}
\|P_S A(\vec{a})\| &= \|(P_S A)\vec{v} + (P_S A)\vec{w}\| \\[2mm]
&\leq \|(P_S A)\vec{v}\| + \|(P_S A)\vec{w}\| \\[2mm]
&\leq \bar{\lambda}\|\vec{v}\| + \sqrt{k\alpha}\lambda_0\|\vec{w}\| \\[2mm]
&\leq \sqrt{k\alpha\lambda_0^2 + \bar{\lambda}^2}\sqrt{\|\vec{v}\|^2 + \|\vec{w}\|^2} \\[2mm]
&\leq \sqrt{k\alpha\lambda_0^2 + \bar{\lambda}^2}\|\vec{a}\|
\end{aligned}$$

and the conclusion of the lemma follows. ∎

With lemma 5.3.1 one readily obtains the following:

Theorem 5.3.2 *Let G be a d-regular strongly connected digraph on N vertices, with index of imprimitivity k, and let $0 < t < 1$. Then for any $l > 0$, W_G^l is an*

$$\left(\alpha, 2^{lH(t)}d\left(k\alpha + \frac{\bar{\lambda}^2}{d^2}\right)^{\frac{lt}{2}}, t\right)$$

disperser, where $H(x)$ is the binary entropy function.

Proof Note that:

$$|V_0\left(W_G^l\right)| = N \qquad |V_1\left(W_G^l\right)| = Nd^{l-1} \qquad \bar{d} = l$$

Let $S \subset V_0\left(W_G^l\right)$ be a set of vertices such that $|S| = N\alpha$ and let $S_1 \subset V_1\left(W_G^l\right)$ be the set of vertices with at least lt neighbours in S. We have to show that

$$|S_1| \leq Nd^{l-1}2^{lH(t)}d\left(k\alpha + \frac{\bar{\lambda}^2}{d^2}\right)^{\frac{lt}{2}}$$

Let A be the adjacency matrix of G, and $W \subseteq [1..l]$. Define the sequence of matrices $\{A_i\}_1^l$ by:

$$A_i^W = \begin{cases} P_S & i \in W \;\; i = 1 \\ P_S A^T & i \in W \;\; i \neq 1 \\ A^T & i \notin W \end{cases}$$

47

With this notation, the number of walks of length $l-1$ on G, that have vertices of S in the positions indexed by W is $\|(\prod_i A_i^W)\vec{1}\|_1$, where $\vec{1}$ denotes the vector all of whose coordinates are 1. Therefore:

$$|S_1| \quad \leq \quad \sum_{|W|=lt} \|(\prod_i A_i^W)\vec{1}\|_1 \quad \leq \quad \sum_{|W|=lt} \sqrt{N} \left\|\prod_i A_i^W\right\|_2 \|\vec{1}\|_2$$

$$\leq \quad \sum_{|W|=lt} \sqrt{N} \left(\prod_i \left\|A_i^W\right\|_2\right) \|\vec{1}\|_2 \quad \leq \quad \binom{l}{lt} \sqrt{N} \left\|(P_S A^T)\right\|_2^{lt} \left\|A^T\right\|_2^{l(1-t)} \left\|\vec{1}\right\|_2$$

By lemma 5.3.1 $\|(P_S A^T)\| \leq \sqrt{k\alpha\lambda_0^2 + \bar{\lambda}^2}$. Since $\binom{l}{lt} \leq 2^{lH(t)}$, $\left\|A^T\right\|_2 = d$, and $\left\|\vec{1}\right\|_2 = \sqrt{N}$, the desired result follows. ∎

Theorem 5.3.2 has two useful corollaries. The first is used in chapter 6 (see theorem 6.3.1):

Corollary 5.3.3 *Let G be a primitive, d-regular, δ-separable digraph on N vertices. Assume:*

$$\alpha \leq \left(\frac{\bar{\lambda}}{\lambda_0}\right)^2$$

Then for any $c > 0$, $W_G^{c\log N}$ is an

$$(\alpha, N^e, t)$$

disperser, with $M = N^{1+c\log d}$ and:

$$e = c\left(H(t) + \frac{t}{2}\right) + (\delta - 1)ct\log d + \frac{\log d}{\log N}$$

The second corollary is used in chapter 7 (see theorem 7.3.1 and corollary 7.3.2):

Corollary 5.3.4 *Let G be a primitive, d-regular, δ-separable digraph on N vertices. Assume:*

$$\alpha \leq \left(\frac{\bar{\lambda}}{\lambda_0}\right)^2$$

Then for any $c > 0$, $W_G^{c\log N}$ is an

$$\left(\alpha, M^{\mu-1}, t\right)$$

disperser with $M = N^{1+c\log d}$ and:

$$\mu = \frac{c\log d}{1 + c\log d}(t\delta + (1-t)) + \frac{c(H(t) + \frac{t}{2})}{1 + c\log d} + \frac{1}{c\log N} > t\delta + (1-t)$$

48

5.4 Efficient Constructions of Exponentially Amplifying Disperser Graphs.

5.4.1 Strong Constructibility.

Let $\{G_i\}_{i=1}^{\infty}$ be a δ_i-separable family of d_i-regular digraphs and Let l_i be a sequence of natural numbers. Theorem 5.3.2 provides **explicit** definitions of families of disperser graphs with a certain amplification, namely $H_i = W_{G_i}^{l_i}$. But we are interested in **efficient** constructibility. What should one require from G_i in order to ensure that $H_i = W_{G_i}^{l_i}$ is efficiently constructible ?

Definition 5.4.1 *A family $G = \{G_i\}$ of d_i regular digraphs is* **strongly constructible** *if*

1. *G is "dense enough" - There exists a natural number k, such that for every natural number N there is an index i computable in $DTIME\,(polylog(N))$ such that:*
$$N \le |G_i| \le N(\log N)^k.$$

2. *There is a numbering of the neighbours of each vertex such that given as input a vertex v of G_i, and a number $1 \le k \le d_i$, the k-th neighbour of v can be computed in $DTIME\,(polylog(|G_i|))$.*

Strong constructibility implies efficient constructibility of the associated bipartite graphs:

Theorem 5.4.1 *Let $G = \{G_i\}$ be a family of* **strongly constructible** *d_i-regular digraphs. Let l_i be a sequence of natural numbers such that $l_i = O\,(polylog(|G_i|))$, and can be computed in $DTIME\,(polylog(|G_i|))$. Then the family $H_i = W_{G_i}^{l_i}$ is efficiently constructible.*

Proof One has to verify that the conditions of definition 4.4.1 are met.
Density of cardinalities. $|V_0(H_i)|$ is dense since $|V_0(H_i)| = |G_i|$ and $|G_i|$ is dense by the definition of strong constructibility.
Polynomiality of relevant graph operations. Since the degree of G_i is d_i, a vertex v of $V_1(H_i)$ is encoded as a string $< y, n_1, \ldots, n_{l_i-1} >$, where y is a vertex of G_i, and n_j is a number in the interval $[1, d_i]$. The multiset of neighbours of x is computed by computing w - the n_1 neighbour of v in G_i, then the n_2 neighbour of w and so on. Each computation is in $DTIME\,(polylog(|G_i|))$, by the definition of strong constructibility and hence in $DTIME\,(polylog(|V_0(H_i)|))$. By the premises of the theorem, the number of computations (l_i) is bounded by a polynomial in $\log(|V_0(G_i)|)$. Hence the computation of the multiset of neighbours of x is in $DTIME\,(polylog(|V_0(G_i)|))$. $\blacksquare$

5.4.2 Strongly Constructible, Highly Separable, Digraphs.

The discussion of 5.4.1 implies that the requirement of efficient constructibility on the bipartite graphs of theorem 5.3.2, translates to the requirement of strong con-

structibility on the separable digraphs from which they are built. From corollaries 5.3.3 and 5.3.4 it follows that there are two parameters that one should attempt to optimize:

1. The smaller the separation exponent δ, the greater the amplification.

2. The smallest probability that can be amplified is $1 - d^{2(\delta-1)}$, hence the smaller the degree of the digraph, the wider the range of probabilities that can be amplified.

Clearly $\frac{1}{2}$-separable digraphs with a constant degree are optimal. In [LPS86] explicit definitions of such graphs are given but as has already been remarked, these graphs are not known to be strongly constructible. On the other hand the digraphs described in 2.11 and 2.12, are strongly constructible, have an optimal separation exponent, and are in fact adequate for our needs. However since the degrees of these digraphs are large ($|G|^c$ for some constant c), the resulting dispersers will amplify only probabilities exponentially close to 1, so a pre-amplification stage is be needed. To make the pre-amplification stage as simple as possible, we shall use instead a construction of digraphs with an almost optimal separation exponent, but a degree polylogarithmic in the size of the digraph. These graphs can amplify probabilities polynomially close to 1. We need the following simple observation:

Lemma 5.4.2 *Let $\epsilon > 0$. There exists a natural number i_0 such that for any natural $i > i_0$ one can find two natural number k, m satisfying:*

$$\begin{array}{rcl}
i & < km \leq & i + O(\log i) \\
\epsilon k & < \log m \leq & (k+3)\epsilon
\end{array}$$

Proof For a given i let k be the largest natural number satisfying:

$$k \lceil 2^{k\epsilon} \rceil \leq i$$

Now if i is large enough then:

$$k \lceil 2^{(k+2)\epsilon} \rceil > (k+1) \lceil 2^{(k+1)\epsilon} \rceil > i$$

so that there exists a natural number $\lceil 2^{k\epsilon} \rceil < m \leq \lceil 2^{(k+2)\epsilon} \rceil$ such that $k(m-1) \leq i < km$. Thus k and m satisfy the conclusion of lemma. ∎

Theorem 5.4.3 *For every $0 < \epsilon < \frac{1}{2}$, there exists a strongly constructible family G_i of d_i-regular, primitive, $\frac{1}{2} + \epsilon$ separable digraphs, with:*

$$d_i \leq (\log |G_i|)^{\frac{1}{\epsilon}}$$

Proof
We construct G_i as follows: Choose numbers k, m such that:

$$\begin{array}{rcl}
i & < km \leq & i + O(\log i) \\
\epsilon k & < \log m \leq & (k+3)\epsilon
\end{array}$$

50

The possibility of choosing these numbers is guaranteed by lemma 5.4.2. Let $F = GF(2^k)$, E an extension field of F of dimension $t = m$, and G_i the resulting Chung's digraph (subsection 2.11.2). Note that since:

$$\lambda_0 = |F| = 2^k \qquad t = m \leq |F|^{\frac{k+3}{k}\epsilon}$$

one has by the discussion in 2.11.2:

$$\bar{\lambda}(G_i) < t\sqrt{\lambda_0} \leq \lambda_0^{\frac{1}{2}+\frac{k+3}{k}\epsilon}$$

Also the cardinality of G_i is $|E| - 1$, and the degree of G_i is $|F|$ so that:

$$deg(G_i) = |F| = 2^k < m^{\frac{1}{\epsilon}} < (km-1)^{\frac{1}{\epsilon}} = (\log(|G_i|)^{\frac{1}{\epsilon}}$$

Note that this construction is efficient in the sense of definition 5.4.1. The density condition is satisfied since the number k, m chosen above ensure that for every $N_i = 2^i$:

$$N_i < |G_i| \leq N_i(\log N_i)^k$$

for any $k > \frac{1}{\epsilon}$. Also all graph operations are performable in polynomial time, since one is doing finite field arithmetic with a constant characteristic (see section 4.5). ∎

Chapter 6

The Deterministic Amplification Problem.

In this chapter the first application of the sampling procedures developed in chapters 4 and 5, is given. The problem to be addressed is the **deterministic amplification problem**: Given an RP(BPP) algorithm A, which requires n random bits and operates with a constant error bound, reduce the error probability (amplify the success probability) of the algorithm, adding as few random bits as possible.

There is a standard amplification method that serves as a reference point for the deterministic amplification problem. Let A be an RP (BPP) algorithm which decides a language $\mathcal{L}$ with n random bits, and an error bound q ($\frac{1}{2} - q$) where q is a constant. Suppose A is run k independent times, and the actual input is accepted iff there is at least one accepting computation (RP case), or iff at least a majority of the computations accept (BPP case). The new algorithm decides $\mathcal{L}$, with an error bound q^k (($\sqrt{1 - 4q^2})^k$). If k is bounded by a polynomial in n, the extra cost in deterministic computation is in $DTIME(poly(n))$. The number of random bits used is kn. For example, the error probability can be made **exponentially** small (i.e. $O(2^{-n})$), with $\Theta(n)$ runs, so that $\Theta(n^2)$ random bits are required. Similarly, $\Theta(n \log n)$ random bits are required to make the error probability **polynomially** small (i.e. $O(n^{-c})$) for some positive constant c). In the deterministic amplification problem, one strives to achieve the same amplification with significantly less random bits.

The results that have been achieved so far for the deterministic amplification problem:

1. [KPS85] showed how to achieve a polynomial amplification of RP with no further cost of random bits.

2. [CG86] showed how to achieve a polynomial amplification of BPP with twice the original amount of random bits.

Using disperser graphs, we give the following results: Let A be an RP (BPP) algorithm using n random bits, and operating with error bound $\alpha(n)$. Let $0 < \epsilon < 1$ be a number, and P be a polynomial.

1. <u>BPP</u>: if $\alpha(n) < \frac{1}{8}$, then α can be made less than $\frac{1}{P(n)}$, with no increase in the amount of random bits.

2. <u>RP</u>: if $\alpha(n) < 1 - \frac{1}{P(n)}$, α can be made less than 2^{-n}, with $(3 + \epsilon)n$ random bits.

3. <u>BPP</u>: if $\alpha(n) < \frac{1}{2} - \frac{1}{P(n)}$, α can be made less than 2^{-n}, with $(6 + \epsilon)n$ random bits.

4. <u>BPP</u>: if $\alpha(n) < \frac{1}{8}$, α can be made less than 2^{-n}, with $(3 + \epsilon)n$ random bits.

Impagliazzo and Zuckerman [IZ], have obtained independently the result that the error probability of an RP (BPP) algorithm can be made exponentially small at a linear cost of random bits.

Finally we should like to point out that the deterministic amplification problem is an instance of the following general problem: Can one simulate a probabilistic algorithm, with a limited amount of random bits? This problem in its entirety, belongs to the realm of pseudo-random generators. Our results show that the specific algorithms formed by applying the standard amplification method to RP (BPP) algorithms, can be simulated with less random bits. Thus they can be interpreted as the construction of a (rather weak) pseudo-random generator for a specific subclass of RP and BPP algorithms.

6.1 Disperser Graphs and Deterministic Amplification.

Let A be an RP (BPP) algorithm which requires n random bits and decides a language $\mathcal{L}$ with an error probability $\alpha(n)$, where $\alpha(n)$ is assumed to be **polynomially apart** from 1 ($\frac{1}{2}$). Our goal is to achieve a large amplification at a small additional cost of random bits.

Our approach is to simulate A, using a sampling procedure which requires a "small" number of random bits but has strong amplification properties:

Definition 6.1.1 *Let Γ be an efficient sampling procedure for $\{0,1\}^n$, and let A be as above. The simulation of A induced by Γ is the probabilistic algorithm B operating in two phases:*

1. **Sampling phase**. *B uses Γ to pick multiset of binary strings of length n.*

2. **Deterministic phase**. *B runs A on the actual input, with each binary string obtained in the sampling phase as a random input, and accepts if at least one computation accepts (RP case), or if a majority of the computations accept (BPP case).*

Note that since A is a polynomial time algorithm and the sampling procedure Γ is efficient (see definition 4.3.1), B is a polynomial time probabilistic algorithm. Now let $G = \{G_n\}_{n=1}^{\infty}$ be an efficiently constructible family of bipartite graphs satisfying:

1. $|V_0(G_n)| = 2^n$ and there exists a bijection $\phi : V_0(G_n) \mapsto \{0,1\}^n$, such that $\phi \in DTIME\,(poly(n))$.

2. $|V_1(G_n)| = 2^{m_n}$ for some positive integer $m_n \geq n$ and there exists a bijection $\varphi : \{0,1\}^{m_n} \mapsto V_1(G_n)$, such that $\varphi \in DTIME\,(poly(n))$.

Since G_n induces a sampling procedure for $\{0,1\}^n$ (see definition 4.2.1), one can define:

Definition 6.1.2 *The simulation of A induced by G_n is the simulation of A induced by the sampling procedure corresponding to G_n.*

By definitions 4.2.1 and 6.1.1, the simulation of A induced by G_n has the following form:

1. It gets a string of $\log |V_1(G_n)|$ random bits.

2. It uses φ to map the random input to a vertex $v \in V_1(G_n)$.

3. It computes the multiset of neighbours of v (which are vertices of $V_0(G_n)$).

4. It uses ϕ to map the multiset of vertices of $V_0(G_n)$ to a multiset of binary strings of length n.

5. It runs A with each of the binary strings obtained above, as a random input.

6. It accepts iff there is at least one accepting computation (RP case), or iff a majority of the computations accept (BPP case).

In order to provide results in the deterministic amplification problem, a simulation algorithm induced by a sampling procedure, should satisfy two properties: First, it should decide $\mathcal{L}$ with an error probability bounded by $\beta < \alpha$ (amplification). Second, the amount of random bits used by B, should be significantly smaller than that required by the standard amplification method to achieve the same amplification.

Amplification can be achieved, if the sampling procedure satisfies the appropriate amplification properties:

Lemma 6.1.1 *Let A be an RP (BPP) algorithm which decides $\mathcal{L}$, and operates with n random bits, and an error bound $\alpha(n)$. Suppose B is a simulation of A, induced by a sampling procedure that satisfies the $(1 - \alpha(n), 1 - \beta(n), t)$ property, $t = 0$ $(t = \frac{1}{2})$. Then B is an RP (BPP) algorithm which decides $\mathcal{L}$ with an error bound $\beta(n)$.*

Proof We deal with the BPP case, the RP case being similar. Since B accepts iff a majority of the computations accept, B outputs the correct answer if a majority of the sampled random inputs of A are witnesses. The error probability of A is bounded by α so the size of the witness set is at least $2^n(1-\alpha)$. Since the sampling procedure satisfies the $(1 - \alpha(n), 1 - \beta(n), \frac{1}{2})$ property, definition 4.1.1 implies that the probability that a majority of the sampled random inputs are witnesses, is at least $1 - \beta$ ∎

Thus one has to construct efficient, highly amplifying sampling procedures. To this end we use disperser graphs. The key observation is:

Lemma 6.1.2 *The sampling procedure induced by a family of (α, β, t) disperser graphs, satisfies the $(1 - \alpha, 1 - \beta, 1 - t)$ amplification property.*

Proof This an immediate consequence of definitions 4.1.1 and 4.2.2 ∎

Lemmas 6.1.1 and 6.1.2 imply the basic:

Theorem 6.1.3 *Let A be an RP (BPP) algorithm which decides $\mathcal{L}$, and operates with n random bits, and an error bound $\alpha(n)$. Suppose B is a simulation of A, induced by a family of (α, β, t) disperser graphs with $t = 1$ (($t = \frac{1}{2}$). Then B is an RP (BPP) algorithm which decides $\mathcal{L}$ with an error probability bounded by $\beta(n)$ and requires $\log |V_1(G_n)|$ random bits.*

Thus to decrease the error probability of A from $\alpha(n)$ to $\beta(n)$, one should use simulation algorithms induced by an efficiently constructible family of $(\alpha(n), \beta(n))$ dispersers. What about the additional cost of random bits? The simulation algorithm, requires $\log |V_1(G_n)|$ random bits, while the standard amplification method requires $n\frac{\log \beta}{\log \alpha}$ random bits for the same amplification. Therefore, a simulation of A, will improve upon the standard amplification method if it is induced by $(\alpha(n), \beta(n))$ dispersers, that satisfy:

$$\log |V_1(G_n)| \ll n\frac{\log \beta}{\log \alpha}$$

This is the basis for our approach to the deterministic amplification problem.

For example if $|V_1(G_n)| = |V_0(G_n)|$ the disperser amplification entails no more random bits than the original algorithm. This property is satisfied by the dispersers of corollary 5.2.2 and leads to the result of [KPS85]:

Theorem 6.1.4 *Let A be an RP algorithm which decides $\mathcal{L}$ with n random bits and an error bound polynomially apart from 1. Let P be a polynomial. Then there exists an RP algorithm which decides L with n random bits and an error bound $\frac{1}{P(n)}$.*

If $|V_1(G_n)| = |V_0(G_n)|^c$ (c a constant), the amplification requires cn random bits. This property is satisfied by the dispersers of corollary 5.2.5 and leads to the result of [CG86]:

Theorem 6.1.5 *Let A be a BPP algorithm which decides $\mathcal{L}$ with n random bits and an error bound polynomially apart from $\frac{1}{2}$. Let P be a polynomial. Then there exists a BPP algorithm which decides L with $2n$ random bits and an error bound $\frac{1}{P(2n)}$.*

Note that these results improve significantly upon the standard amplification method, which would require $\Omega(n \log n)$ random bits for the same amplification.

Note also, that the dispersers of corollary 5.3.3, also satisfy $|V_1(G_n)| = |V_0(G_n)|^{O(1)}$ if one sets $c \log d = O(1)$. This fact will be used in section 6.3, to achieve exponential amplification, at a linear cost of random bits.

Before we move on, there is one important technicality to be dispensed with. In definition 6.1.2 it is assumed that $|V_0(G_n)|$ and $|V_0(G_n)|$ are powers of 2. This is indeed the case for the dispersers of theorems 5.2.1 and 5.2.4, which are used for a polynomial amplification. However to in order to achieve exponential amplification we will use dispersers that do not satisfy this property. In this case the family G_m still induces a sampling procedure for $\{0, 1\}^n$ as follows:

Definition 6.1.3 *Let $G = \{G_m\}$ be an efficiently constructible family of bipartite graphs. The sampling procedure induced by G on $\{0,1\}^n$, samples a multiset of n-bit strings as follows:*

1. *It computes an index i that satisfies $2^n \leq |V_0(G_i)| \leq 2^n n^k$ for some fixed positive integer k. The fact that such an index can be found, is ensured by the definition of efficient constructibility. We will suppose that the vertices of $V_0(G_i)$ are indexed by the numbers $0, \ldots, |V_0(G_i)| - 1$, and the vertices of $V_1(G_i)$ are indexed by the numbers $0, \ldots, |V_1(G_i)| - 1$.*

2. *The algorithm obtains a string of $\lceil \log |V_1(G_i)| \rceil$ random bits. Let x be the number encoded by the random string. The algorithm computes the vertex $v \in V_1(G_i)$ whose index is $x \bmod |V_1(G_i)|$.*

3. *The algorithm computes the multiset of neighbours of v, and converts each of them to an n-bit string, by taking the binary representation of its index mod 2^n.*

What was actually done here, is to construct from a family G_m of (α, β) dispersers, a new family of bipartite graph H_n such that $|V_0(H_n)| = 2^n$, and $|V_1(H_n)| = 2^{m(n)}$, so that H_n induces a sampling procedure on binary strings. Now if α is polynomially small, and β is exponentially small, then amplification properties of H_n remain essentially the same. α can change by a polynomial factor, but this can be taken into account in the initial polynomial amplification. β can increase by a factor of at most 2.

6.2 Polynomial Amplification of BPP.

In this section we give a result which is in one sense an improvement of theorem 6.1.5. It is shown that the error probability of a BPP algorithm can be made polynomially small, with no further cost in random bits. However, the initial error must be bounded by a sufficiently small constant.

The following lemma is needed:

Lemma 6.2.1 *Let G be a connected d regular primitive digraph on $N = 2^n$ vertices with adjacency matrix A such that:*

$$\left(\frac{\bar{\lambda}(A)}{\lambda_0(A)} \right)^2 < \alpha$$

Then the double cover [1] of G is an $\left(\alpha, 8\alpha^2, \frac{1}{2} \right)$ disperser.

Proof This is a direct consequence of lemma 2.8.3 ∎

Theorem 6.2.2 *Let A be a BPP decision algorithm for a language $\mathcal{L}$, which uses n random bits, with error probability $\alpha < \frac{1}{8}$. Then for any polynomial P, there exists a BPP algorithm that decides $\mathcal{L}$, with n random bits, and error probability bounded by $\frac{1}{P(n)}$.*

---

[1]See 2.1 for a definition of double cover.

Proof Our intention is to perform a sequence of amplifications which achieve the following sequence of bounds on the error probability:

$$\begin{aligned}\alpha_0 &= \tfrac{1}{8} \\ \alpha_{l+1} &= 8\left(\alpha_l\right)^2\end{aligned}$$

Let $\{G_n\}$ a family of primitive, constant degree, η-separable digraphs with $\eta < 1$, and $|G_n| = 2^n$. Such digraphs can be obtained for example, from the Gabber-Galil expander graphs (see section 2.9). By raising G_n to a sufficiently large (but fixed) power, one gets a digraph $\mathcal{G}$ with an adjacency matrix A such that $\frac{\bar{\lambda}^2}{\lambda_0^2} < \frac{1}{8}$.

Now define the following sequence of matrices $\{\Lambda_k\}_{l=0}^{i}$:

$$\begin{aligned}\Lambda_0 &= A \\ \Lambda_{l+1} &= \Lambda_l^2\end{aligned}$$

Let H_l be the double cover of the digraph with adjacency matrix Λ_l. Since:

$$\frac{\bar{\lambda}(\Lambda_{l+1})}{\lambda_0(\Lambda_{l+1})} \leq \left(\frac{\bar{\lambda}(\Lambda_l)}{\lambda_0(\Lambda_l)}\right)^2$$

We have for each l:

$$\left(\frac{\bar{\lambda}(\Lambda_l)}{\lambda_0(\Lambda_l)}\right)^2 < \alpha_l$$

Therefore lemma 6.2.1 implies that H_l is an $\left(\alpha_l, \alpha_{l+1}, \frac{1}{2}\right)$ disperser.

Now we define the sequence of probabilistic algorithms $\left\{A^l\right\}_{l=0}^{i}$ as follows:

1. A_0 is the original algorithm.

2. A_l is the simulation of A_{l-1} induced by H_{l-1}.

By theorem 6.1.3 A_l decides $\mathcal{L}$ with error probability bounded by α_l. Now since $\alpha_i = \frac{1}{8}\left(8\alpha\right)^{2^i}$ $l = \log\log n + c$ iterations make α_l polynomially small:

$$\alpha_l = \frac{1}{8\left(n^{\log\left(\frac{1}{8\alpha}\right)}\right)^{2^c}}$$

It remains to verify that A_l is polynomial time. Observe that there are d^{2^k} recursive calls to the algorithm A_{k-1} in the algorithm A_k where d is the degree of $\mathcal{G}$. Hence there are at most $d^{2^{l+1}}$ recursive calls in A_l. For $l = \log\log n + c$ this number is bounded by a polynomial in n. Since the graphs $\{H_i\}_{i=1}^{l}$ are efficiently constructible, A_l is a polynomial time algorithm. ∎

6.3 Exponential Amplification.

The main result of this chapter is:

Theorem 6.3.1 *Let A be an RP (BPP) algorithm, which decides the language $\mathcal{L}$ with n random bits, and an error bound $\alpha(n) < 1 - \frac{1}{P(n)}$ $(\alpha(n) < \frac{1}{2} - \frac{1}{P(n)})$ where P is a polynomial. Let $\epsilon > 0$ be a constant.*

1. *RP: There exists an RP algorithm B, which decides $\mathcal{L}$, with an error probability bounded by 2^{-n} and uses $(3 + \epsilon)n$ random bits.*

2. *BPP: There exists a BPP algorithm B, which decides $\mathcal{L}$, with an error probability bounded by 2^{-n}, and uses $(6 + \epsilon)n$ random bits.*

3. *BPP: If $\alpha(n) < \frac{1}{8}$, there exists a BPP algorithm B, which decides $\mathcal{L}$ with an error probability bounded by 2^{-n}, and uses $(3 + \epsilon)n$ random bits.*

Proof Let $\eta < \epsilon$ be a positive constant whose value will be specified later, and let κ be any positive constant smaller than η.

1. By theorem 6.1.4 there exists an RP algorithm C which decides $\mathcal{L}$ with n random bits, and an error bound $n^{-\frac{1}{\kappa}}$.

   By theorem 5.4.3 there exists a strongly constructible family G_n of d_n-regular, primitive, $\frac{1}{2} + \eta$ separable digraphs, with:

   $$d_n \le (\log |G_n|)^{\frac{1}{\eta}}$$

   Let $W_n = W_{G_n}^{c_n \log |G_n|}$ where c_n satisfies:

   $$c_n \log d_n = 2 + \epsilon$$

   By definition 6.1.3, and the efficient constructibility of G_n, we may assume:

   $$2^n = |V_0(W_n)| \le |G_n| \le 2^n n^k$$

   for some fixed natural k. By theorem 2.9.2 one has for a sufficiently large n:

   $$\left(\frac{\bar{\lambda}(G_n)}{\lambda_0(G_n)}\right)^2 \ge \quad (1 - o(1))\left(\frac{1}{d_n}\right) \quad \ge \quad (1 - o(1))\left((\log |G_n|)^{-\frac{1}{\eta}}\right)$$
   $$\ge \quad (1 - o(1))(n + k \log n)^{-\frac{1}{\eta}} \quad \ge \quad n^{-\frac{1}{\kappa}}$$

   Hence by corollary 5.3.3, W_n is for a sufficiently large n a:

   $$\left(n^{-\frac{1}{\kappa}}, n^e, 1\right)$$

   disperser, with $|V_1(W_n)| = 2^{(1 + c_n \log d_n)n}$ and:

   $$e = c_n \left(H(1) + \frac{1}{2}\right) + \left(\eta - \frac{1}{2}\right) c_n \log d_n + \frac{\log d_n}{n}$$

Now η is chosen so that:

$$\left(\eta - \frac{1}{2}\right) c_n \log d_n = \left(\eta - \frac{1}{2}\right)(2 + \epsilon) < -1$$

Since $c_n \mapsto 0$, $e < -1$ for a sufficiently large n, and W_n is (for a sufficiently large n) a:

$$\left(n^{-\frac{1}{\kappa}}, 2^{-n}, 1\right)$$

disperser with $|V_1(W_n)| = 2^{(1+c_n \log d_n)n}$.

Let B be the simulation induced on C by W_n. By theorem 6.1.3, B decides $\mathcal{L}$ with error bound 2^{-n}, and requires $\log |V_1(W_n)|$ random bits, i.e $(1 + c_n \log d_n)n$ or $(3 + \epsilon)n$ random bits.

2. By theorem 6.1.5 there exists a BPP algorithm C which decides $\mathcal{L}$ with $2n$ random bits, and an error bound $(2n)^{-\frac{1}{\kappa}}$.

By theorem 5.4.3 there exists a strongly constructible family G_n of d_n-regular, primitive, $\frac{1}{2} + \eta$ separable digraphs, with:

$$d_n \leq (\log |G_n|)^{\frac{1}{\eta}}$$

Let $W_n = W_{G_n}^{c_n \log |G_n|}$ where c_n satisfies:

$$c_n \log d_n = 2 + \epsilon$$

By definition 6.1.3, and the efficient constructibility of G_n, we may assume:

$$2^{2n} = |V_0(W_n)| \leq |G_n| \leq 2^{2n}(2n)^k$$

for some fixed natural k. By theorem 2.9.2 one has for a sufficiently large n:

$$\left(\frac{\bar{\lambda}(G_n)}{\lambda_0(G_n)}\right)^2 \geq (1 - o(1))\left(\frac{1}{d_n}\right) \geq (1 - o(1))\left((\log |G_n|)^{-\frac{1}{\eta}}\right)$$
$$\geq (1 - o(1))(2n + k \log 2n)^{-\frac{1}{\eta}} \geq n^{-\frac{1}{\kappa}}$$

Hence by corollary 5.3.3, W_n is for a sufficiently large n a:

$$\left(n^{-\frac{1}{\kappa}}, n^e, \frac{1}{2}\right)$$

disperser, with $|V_1(W_n)| = 2^{(1+c_n \log d_n)2n}$ and:

$$e = c_n \left(H(\frac{1}{2}) + \frac{1}{4}\right) + \left(\eta - \frac{1}{2}\right)\frac{1}{2}c_n \log d_n + \frac{\log d_n}{n}$$

Now η is chosen so that:

$$\left(\eta - \frac{1}{2}\right) c_n \log d_n = \left(\eta - \frac{1}{2}\right)(2 + \epsilon) < -1$$

Since $c_n \mapsto 0$, $e < -1$ for a sufficiently large n, and W_n is (for a sufficiently large n) a:

$$\left(n^{-\frac{1}{\kappa}}, 2^{-n}, 1\right)$$

disperser with $|V_1(W_n)| = 2^{(1+c_n \log d_n)2n}$.

Let B be the simulation induced on C by W_n. By theorem 6.1.3, B decides $\mathcal{L}$ with error bound 2^{-n}, and requires $\log |V_1(W_n)|$ random bits, i.e $(1+c_n \log d_n)2n$ or $(6+2\epsilon)n$ random bits.

3. This is proved as in the preceding part of the theorem, except that the initial polynomial amplification (algorithm C) is obtained with no increase in the number of random bits, using theorem 6.2.2.

$\blacksquare$

Chapter 7

Imperfect Random Sources.

Let A be an RP (BPP) algorithm, which decides a language $\mathcal{L}$. A is guaranteed to operate with a specified bound on its error probability only if its random bits are independent and unbiased. But what happens if one has at its disposal only low quality random bits (output by some imperfect random source)? Can one still perform the task with a small error probability?

Definition 7.0.1 *Let S be a random source. We say that RP (BPP) can be simulated by S, if for every $\mathcal{L} \in RP\ (BPP)$ there exists a probabilistic polynomial algorithm which decides $\mathcal{L}$ with an error probability that decreases to 0 with input length, when its random input comes from S.*

Note that it does not suffice to require a constant error probability in the above definition, as we did in the definition of RP (BPP), since the standard amplification method need not work here. Suppose A is say, an RP algorithm which requires n random bits and operates with an error probability $\frac{1}{2}$ when its random input comes from the random source S. The standard amplification method proceeds now as follows: A is run k times by getting an output of kn random bits from the source, breaking it into k blocks of n bits each, and running A with each block as a random input. The input is accepted if there is at least one accepting computation. Note however, that the error probability need not decrease to 2^{-k} for two reasons: First, the k blocks need not be independent. Second, it is not necessarily true that each block inherits the statistical properties of source. For example if the source is defined by the condition that its entropy rate exceeds a certain value, this is not necessarily the case for each block. Therefore the error probability of each run is not guaranteed to be bounded by $\frac{1}{2}$.

In this chapter we use sampling procedures induced by disperser graphs to simulate RP and BPP with imperfect random sources. The logical structure of the chapter is:

1. The simulation problem has been investigated for various models of imperfect random sources. Section 7.1 reviews the main models and the status of the simulation problem, with respect to them.

2. In section 7.2 a new model of an imperfect random source is proposed. A (γ, ν)-Renyi source is defined as the collection of all probability distributions for

which, R_ν the Renyi ν-entropy rate, is bounded below by γ. This random source includes strictly all sources studied so far. Simulations of RP and BPP with Renyi sources would imply uniform simulations for a wide variety of random sources of interest.

3. In section 7.3 we show that the simulation problem for a Renyi source reduces to the problem of the efficient construction of highly amplifying disperser graphs. We use the dispersers constructed in this work, to give a simulation of RP (BPP), with any (γ, ν)-Renyi source with $\gamma > \frac{1}{2}$ ($\gamma > \frac{3}{4}$) and $\nu > 1$. This implies a simulation of RP and BPP with various random sources for which no simulation was known.

4. In section 7.4 it is shown any simulation of RP and BPP with m random bits, requires $m^{\Omega(1)}$ Renyi entropy.

5. There is a wide gap between the lower bound of section 7.4 ($m^{\Omega(1)}$ Renyi entropy), and the upper bound of section 7.3($\frac{m}{2}$ Renyi entropy). In section 7.5 we close the gap for a specific random source, the oblivious bit fixing source.

Recently, Zuckerman [Zuc], has improved the results of section 7.3, showing that BPP can be simulated with any source that has a constant lower bound on its Renyi entropy rate.

7.1 Some Models of Imperfect Random Sources.

In section 3.4 an imperfect random source was defined as follows:

Definition 7.1.1 *A random source is a sequence $\{\Pi_m\}_{m=1}^\infty$, where Π_m is a family of probability distributions on $\{0,1\}^m$. A* **strategy** *of the random source is a sequence $\{\pi_m\}_{m=1}^\infty$, where $\pi_m \in \Pi_m$ (i.e. a strategy is a sequence of allowable probability distribution - one distribution for each binary string length).*

Since our ultimate goal is to design probabilistic algorithms that perform well with any member of a wide class of allowable probability distributions, it is often convenient to view the random source as an **adversary**. The adversary has a collection of allowable strategies (probability distributions). It attempts to minimize the probability of the witness set, and may use to this end any of the strategies at his disposal. The choice of the strategy may be adaptive: a probability distribution on a subset of random bits may be picked according to the values of other random bits. The algorithm on the other hand, knows the class of allowable strategies but has no advance knowledge which strategy will actually be picked by the adversary. It attempts to hit the witness set with high probability regardless of the strategy used.

Let us present some examples of imperfect random sources that have been studied in theoretical computer science:

1. The perfect random source. Here Π_m has just one member - the uniform probability distribution on $\{0,1\}^m$. Conceptually such a distribution is obtained by m independent tosses of an unbiased coin.

2. The $\omega(m)$-Von Neuman source ($\omega \in [0, \frac{1}{2}]$). Π_m consists of all p.d. π_m satisfying:

$$\pi_m(a_1, a_2, \ldots, a_m) = p^{\Sigma_{i=1}^m a_i}(1-p)^{m-\Sigma_{i=1}^m a_i}$$

where $a_i \in \{0, 1\}$ and $\omega(m) \leq p \leq 1 - \omega(m)$. Conceptually such a distribution is obtained by m independent tosses of a biased coin.

3. $\omega(m)$-Markov sources [Blu84] ($\omega(m) \in [0, \frac{1}{2}]$). A $\omega(m)$-markov chain is a Markov chain with precisely two allowed transitions from each state (labeled by $0, 1$), and with the condition that each transition probability is bounded by $1 - \omega(n)$. To each sequence of m transitions (starting from some initial state), there corresponds a binary string of length m (the sequence of the labels of the transitions), so the Markov chain induces a probability distribution on $\{0, 1\}^m$. A $\omega(m)$-Markov source is a sequence $\{\Pi_m\}$ where Π_m is the collection of all probability distributions induced on $\{0, 1\}^m$, by $\omega(m)$-Markov chains whose number of states is bounded by some fixed polynomial in m.

4. The $\omega(m)$-semi random source (henceforth abbreviated $\omega(n)$-SR source) was introduced by [SV84] ($\omega \in [0, \frac{1}{2}]$). Π_m consists of all the probability distributions on $\{0, 1\}^m$ satisfying for each $1 \leq i \leq m$:

$$\omega \leq Pr\{X_i = 1 | X_1 = a_1, \ldots, X_{i-1} = a_{i-1}\} \leq 1 - \omega$$

Here X_i denotes the i-th random bit and $a_k \in \{0, 1\}$.

5. The (l, b)-probability bounded source (henceforth abbreviated (l, b)-PRB source) introduced by [CG85]) is a generalization of the SR source. Here l is a positive integer and $b \leq l$ is a positive real number. Π_m consists of of all probability distributions on $\{0, 1\}^m$ with the following property: If the output of the source is divided into blocks of l bits each, then the probability of a block being assigned any value, given any assignment to the previous blocks, is bounded from above by 2^{-b}.

6. The $\omega(m)$-bit fixing sources. Here the adversary may fix the values of up to $(1-\omega)m$ bits. The other bits are obtained by tossing independent unbiased coins. We distinguish four types of bit fixing sources:

   (a) The oblivious bit fixing source. The adversary picks a set of up to $(1-\omega)m$ bit locations and fixes the bits **before** the coins for the other positions are tossed. This model was investigated in [CGH+85].

   (b) The nonoblivious bit fixing source. The adversary picks a set of up to $(1-\omega)m$ bit locations and sets the bits **after** the coin tosses for the other positions. This model was investigated in [BOL85, KKL88].

   (c) The adaptive bit fixing source. Here the bits $b_1, \ldots, b_m$ are ordered by their indices. Depending on the values of the bits $b_1, \ldots, b_{i-1}$ the adversary may toss a fair coin for bit b_i or fix its value provided less than $(1 - \omega)m$ bits have hitherto been fixed. These sources were studied in [LLS87].

(d) The strong bit fixing source. The adversary tosses a coin for each bit location, and then picks a set of up to $(1 - \omega)m$ bit locations and sets them.

Two method have been used the attack the simulation problem:

Bit extraction:
One attempts to extract n independent unbiased random bits from an output of $m = P(n)$ random bits of the imperfect source (P is a polynomial). More precisely one looks for a sequence of **efficiently computable** extraction functions:

$$f_n : \{0,1\}^m \mapsto \{0,1\}^n$$

s.t the p.d of the source on $\{0,1\}^m$ induces a probability distribution on $\{0,1\}^n$ which is either uniform or "almost" uniform. Clearly if such extraction functions exist, RP (BPP) can be simulated by extracting perfect random bits from the output of the random source and using them as a random input for the original algorithm. Simulations based on bit extraction were given for:

1. $\omega(m)$-Von Neuman sources for $\omega(m) \geq m^{\mu-1}$ for any $0 < \mu < 1$ (by Von Neuman).

2. $\omega(m)$-Markov sources for $\omega(m) \geq m^{\mu-1}$ for any $0 < \mu < 1$ [Blu84].

3. $\omega(m)$-oblivious bit fixing sources for $\omega(m) > c$ where $\frac{1}{2} < c < 1$ is some (unspecified) constant [CGH+85].

There are however two basic difficulties which make this method of limited applicability. First, even in the cases where extraction functions can be shown to exist for a given source strategy, they need not be efficiently computable. Second, since one is dealing with a collection of strategies, one has to find one sequence of extraction functions common to all strategies. Again this may not be possible. Indeed the impossibility of bit extraction has been proved for $\frac{1}{3}$-oblivious bit fixing sources, nonoblivious bit fixing sources, and for SR and PRB sources [CGH+85, BOL85, KKL88, LLS87, Vaz86, CG85].

Simulation algorithms induced by sampling procedures:
[VV85] devised the following sophisticated procedure for sampling n^c binary strings of length n: obtain an output of $cn \log n$ random bits from the random source, and partition it into n blocks $p_1, \ldots, p_n$ of size $c \log n$ each. From each binary string of length $c \log n$, produce n bits $b_1, \ldots, b_n$, where b_i is obtained by taking the inner product of the string with p_i. This sampling procedure was used to show that:

1. BPP can be simulated by $\omega(m)$-SR sources for $\omega(m) = \Omega(1)$ [VV85, Vaz86].

2. BPP can be simulated with (l, b)-PRB sources, if $b = O(\log m)$ and $\frac{b}{l} = \Omega(1)$ [CG85].

Thus sampling procedures can bypass the impossibility of bit extraction.

7.2 The Renyi Random Source.

Let P be a probability distribution on $\{0, 1\}^m$. Recall that the entropy of P $I(P)$, and the entropy rate of P $R(P)$, are defined by:

$$I(P) = \sum_{i=0}^{2^m-1} p_i \log \frac{1}{p_i} \qquad R(P) = \frac{I(P)}{m}$$

Note that all the random sources described in section 7.1 have lower bounds on their entropies (and rates). $\omega(m)$-Von Neuman sources, have at least $H(\omega(m))$ entropy rate (H is the binary entropy function). $\omega(m)$ Markov sources, or SR sources, have at least $\log \frac{1}{1-\omega}$ entropy rate. (l, b)-PRB sources has at least $\frac{b}{l}$ entropy rate. $\omega(m)$-bit fixing sources, has at least ω entropy rate.

On the other hand, the simulation technique of [VV85, Vaz86, CG85], used to simulate BPP with $\omega(n)$-SR, and (l, b)-PRB random sources, of entropy rate $\Omega(1)$, does not apply to all random sources which have a constant lower bound on the entropy rate. for example, it can be verified that the technique does not work in the case of constant rate bit fixing sources. The reason for this is that the simulation relies on the fact that if the output of the source is broken into contiguous blocks, each block can be regarded as being itself an output of the source. In particular each block has $\Omega(1)$ entropy rate. Clearly this property is not satisfied by bit fixing sources.

The above discussion leads naturally to the following questions:

- Is a constant lower bound on the entropy rate a sufficient condition for the simulation of RP and BPP?

- If not, is there another general information-theoretic quantity such that a constant lower bound on it, is a sufficient condition for the simulation?

- If so, does there exist one simulation which works for every source satisfying the lower bound?

- Can a $o(1)$ lower bound also suffice for the simulation?

In the rest of the section, the first two questions are discussed. The last two are dealt with, in later sections.

The first question is answered in the negative. A constant lower bound on the entropy rate is too general a condition: a source of rate ω can concentrate a probability of almost $1 - \omega$ on a single string so the error probability cannot be decreased below $1 - \omega$.

Turning to the second question we note that the inadequacy of the entropy rate is due to the fact that "large" probabilities are allowed for individual strings even if the rate is large. Therefore we look for an information-theoretic quantity, a constant lower bound on which, prohibits large probabilities for individual strings. Such quantities are supplied by the Renyi generalized entropies [Ren70]. [1] Let S be a source with a

---

[1] We are grateful to Mauricio Karchmer and Noam Nisan for drawing our attention to the Renyi entropies.

probability distribution $p = (p_0 \ldots p_{2^m-1})$ on $\{0,1\}^m$. Let $\nu \neq 1$ be a real number. The entropy of order ν, I_ν, and the rate R_ν, are defined by:

$$I_\nu = \frac{1}{1-\nu} \log \sum_{i=0}^{2^m-1} p_i^\nu \qquad R_\nu = \frac{I_\nu}{m}$$

It can be easily verified that:

$$I_\infty = \lim_{\nu \to \infty} I_\nu = \log \frac{1}{p_{max}} \qquad I_1 = \lim_{\nu \to 1} I_r = \sum_{i=0}^{2^m-1} p_i \log \frac{1}{p_i}$$

Note also, that $I_\nu = \log \frac{1}{\|p\|_{\nu-1}}$ if $\nu > 1$, and $I_\nu = \log \|\frac{1}{p}\|_{1-\nu}$ if $\nu < 1$. [2] Therefore I_ν is a decreasing function of ν.

The quantity I_1 has already proved inadequate for our needs. Because of the nonincreasing monotonicity, I_ν for $\nu < 1$ is ruled out as well. On the other hand the quantities I_ν $\nu > 1$ turn out to be appropriate since:

1. I_ν actually measures the L_ν norm of the probability distribution vector. A high value for I_ν means that the probability distribution is "close" to the uniform distribution, in the L_ν norm, and this precludes "large" probabilities for individual strings.

2. The family of probability distributions with a constant lower bound on R_ν, is very general and includes $\omega(m)$-Von Newman, Markov, and SR random sources, with $\omega(m) = \Omega(1)$, as well as (l,b) PRB sources with $\frac{b}{l} = \Omega(1)$, and $\omega(m)$-bit fixing sources, with $\omega(m) = \Omega(1)$.

Therefore we choose to define:

Definition 7.2.1 *An $(\gamma(m), \nu)$-Renyi random source is the sequence $\{\Pi_m\}_{m=1}^\infty$, where Π_m is the family of probability distributions on $\{0,1\}^m$ which satisfy:*

$$R_\nu \geq \gamma(m)$$

7.3 Simulations of RP and BPP with a Renyi Source.

The connection between disperser graphs and the simulation of RP (BPP) with imperfect sources is established by:

Theorem 7.3.1 *Suppose there exists an efficiently constructible family $\{G_i\}$ of*

$$\left(\alpha(N_i), M_i^{\mu-1}, t \right)$$

for $t = 1$ ($t = \frac{1}{2}$). Assume $\alpha(N_i) = \Omega\left(\frac{1}{polylog(N_i)}\right)$. Then any RP (BPP) algorithm can be simulated by a $(\mu + \frac{1}{m^c}, \nu)$-Renyi source for any $\nu > 1$ and $0 < c < 1$.

---

[2]Recall that if X is a random variable, then $\|X\|_k = E\left[X^k\right]^{\frac{1}{k}}$, where E denotes the expectation operator.

Proof Let A be an RP (BPP) algorithm which decides a language $\mathcal{L}$, with n random bits, and a constant error probability.

First, by theorems 6.1.4 and 6.1.5, there exists an RP (BPP) algorithm B, which decides $\mathcal{L}$ with n $(2n)$ random bits and error probability bounded by α.

Next, let C be the simulation algorithm for B, induced by the family $\{G_i\}$. By theorem 6.1.3, algorithm C decides $\mathcal{L}$ with m random bits and error probability bounded by $2^{(\mu-1)m}$.

Therefore, given an actual input for C, the cardinality of its nonwitness set N, does not exceed $2^{m\mu}$. Let χ_N be the characteristic vector of N, and $p = (p_0 \dots p_{2^m-1})$ the vector of probabilities induced by a random source on strings of length m. Now if algorithm C gets its random input from the random source, the error probability of the algorithm is just the probability given by the source to the nonwitness set N. By the Holder inequality this can bounded by:

$$\sum_{x \in N} p(x) = \langle p, \chi_N \rangle \le \|p\|_\nu \|\chi_N\|_{\frac{\nu}{\nu-1}} = 2^{\frac{(\nu-1)m}{\nu}(\mu-R_\nu)}$$

So if $R_\nu - \mu \ge \frac{1}{m^c}$, the error probability tends to 0 with m. ∎

Corollary 5.1.2 establishes the existence of dispersers with $\mu = (\log M_i)^{-\eta}$ for any $0 < \eta < 1$, and $\bar{d} = O(polylog(N_i))$. Efficient constructions of such dispersers would imply by theorem 7.3.1, that BPP can be simulated with m bits satisfying $R_\nu > m^{-\eta}$ for any $0 < \eta < 1$ (i.e. $I_\nu > m^\xi$ for any $0 < \xi < 1$). We conjecture that such efficient constructions do exist, and hence that the simulations do exist. However in this work we have only managed to construct dispersers with $\mu > \frac{1}{2}$, and this leads to the following:

Corollary 7.3.2 *Let $\nu > 1$. RP can be simulated by a $(\frac{1}{2} + \epsilon, \nu)$-Renyi source, and BPP can be simulated by a $(\frac{3}{4} + \epsilon, \nu)$-Renyi source.*

Proof Let $\eta < \epsilon$. By theorem 5.4.3 there exists a strongly constructible family G_i of d_i-regular, primitive, $\frac{1}{2} + \eta$ separable digraphs, with:

$$d_i \le (\log |G_i|)^{\frac{1}{\eta}}$$

Let $W_i = W_{G_i}^{c_i \log |G_i|}$. By theorem 2.9.2 one can assume that $\left(\frac{\bar{\lambda}}{\lambda_0}\right)^2 \ge \frac{1}{d_i}$, so that by corollary 5.3.4, W_i is an efficiently constructible

$$\left((\log N_i)^{-\frac{1}{\eta}}, N_i^{\mu-1}, t\right)$$

disperser, with $M_i = N_i^{1+c_i \log d_i}$ and:

$$\mu = \frac{c_i \log d_i}{1 + c_i \log d_i}\left(t\left(\frac{1}{2} + \eta\right) + (1-t)\right) + \frac{c(H(t) + \frac{t}{2})}{1 + c_i \log d_i} + \frac{1}{c_i \log N_i}$$

Now if one sets $c_i \log d_i$ to be a large enough constant (or even $O(polylog(N_i))$), then since $c_i \mapsto 0$ μ can be made asymptotically as close to $\frac{1}{2} + \eta$ as desired. The corollary now follows from theorem 7.3.1 with $t = 1$ in the RP case, and $t = \frac{1}{2}$ in the BPP case. ∎

Corollary 7.3.2 implies immediately simulations of RP and BPP with bit fixing sources:

Corollary 7.3.3   *1. For any $\epsilon > 0$ any RP (BPP) algorithm can be simulated by an adaptive or nonoblivious bit fixing source if the fraction of the fixed bits does not exceed $\frac{1}{2} - \epsilon$ ($\frac{1}{4} - \epsilon$).*

2. For any $\epsilon > 0$ any RP (BPP) algorithm can be simulated by a strong bit fixing source if the fraction of the fixed bits does not exceed $H_2^{-1}\left(\frac{1}{2} - \epsilon\right)$ ($H_2^{-1}\left(\frac{1}{4} - \epsilon\right)$), where H_2^{-1} is the inverse of the binary entropy function, on the interval $[0, \frac{1}{2}]$.

7.4   A Lower Bound on the Renyi Entropy Necessary for the Simulation of RP and BPP.

The attacks of [VV85, CG85] on the imperfect random sources problem use simulations for probabilistic algorithms induced by sampling procedures. It is natural to enquire about the limitations of this approach. A simple counting argument shows that a certain amount of I_ν entropy is required for such simulation algorithms if one does not make any assumption on the structure of the witness set.

Theorem 7.4.1 *A simulation of an RP (BPP) algorithm A induced by an efficiently constructible family of bipartite graphs, which uses m random bits, requires $m^{\Omega(1)}$ I_ν entropy.*

Proof If the original RP (BPP) algorithm A, requires n random bits to decide a string x then the simulating algorithm B can use at most $O(poly(n))$ random bits, to pick a multiset of random inputs for A. The size of the multiset must be $O(poly(n))$. If G is used to sample random inputs for A, then one must have:

$$\begin{aligned}
|V_0(G)| \quad &= N = 2^n \\
|V_1(G)| \quad &= M = 2^m = N^{O(polylog(N))} \\
deg(V_1(G)) \quad &= \bar{d} = O(polylog(N))
\end{aligned}$$

Choose a subset of vertices $S \subset V_1(G)$ such that $|S| < \frac{N}{2\bar{d}}$. Then $\Gamma(S)$ the neighbour set of S satisfies $|\Gamma(S)| < \frac{N}{2}$. Since there are no conditions on the structure of the nonwitness set, only on its size, and this only has to be bounded by $\frac{N}{2}$, we may assume that $\Gamma(S)$ is contained in the nonwitness set of some actual input x of A. Hence S is contained in T, the nonwitness set of x in algorithm B. Hence:

$$|T| = \Omega\left(\frac{N}{polylog(N)}\right)$$

Since one must have $I_\nu > \log|T|$ in order to ensure that the source cannot give the nonwitness set probability 1, the source must have $\Omega(\log N)$ i.e. $m^{\Omega(1)}$ I_ν. ∎

7.5 Matching the Lower Bound for Oblivious Bit Fixing Sources

There is a wide gap between the lower bound of section 7.4 and the upper bounds of section 7.3. We shall show here that this gap can be eliminated in the case of oblivious bit fixing sources.

Theorem 7.5.1 *For any constant $0 < \xi < 1$, any BPP algorithm A, can be simulated by any oblivious bit fixing source whose entropy is bounded from below by m^ξ.*

Theorem 7.5.1 follows from the next two lemmas:

Lemma 7.5.2 *Suppose a BPP algorithm A, can be simulated by oblivious bit fixing sources of rate bounded by a constant α with error probability ϵ. Then for any constant $\beta < \alpha$, A can be simulated by oblivious bit fixing sources of rate bound β and error probability $\left(\frac{\log \beta}{\log \alpha} + 1 \right) \epsilon$.*

Proof Choose a natural number k such that:

$$\alpha^k < \beta$$

Suppose A requires n random bits. The simulation algorithm for rate bound β B will be of the form:

1. B gets $m = n^{2k-1}$ bits from a source of a rate bound β.

2. Produces $2k - 1$ strings of n bits each.

3. Runs A on each string (as random input) and takes a majority vote.

The n bit strings are produced by regarding the m bits as a $2k - 1$ dimensional cube of side length n, and producing one string for each direction by xoring bits in hyperplanes perpendicular to that direction.

For an estimate of the error of B note that a bit produced by xoring a hyperplane is good if the hyperplane contains at least one bit not fixed by the source. Hence if $\alpha_1 \ldots \alpha_{2k-1}$ are the fractions of "good bits" in the strings corresponding to the $2k - 1$ directions of the cube then we have:

$$\prod \alpha_{i=1}^{2k-1} \geq \beta$$

Hence at least k of the α_i are not less than α. The error probability of B is bounded by the probability that A will err on any of these k strings so it is bounded by $k\epsilon$. ∎

Lemma 7.5.3 *For a sufficiently small constant α we have: If BPP algorithm A, can be simulated by n bits of oblivious bit fixing sources of rate bound α with error probability ϵ, then for any $\frac{1}{2} < \delta < 1$ there is a constant k, such that A can be simulated by 2n bits of oblivious bit fixing sources of rate bound $\alpha\delta$, and error probability $k\epsilon$.*

Proof The simulation algorithm B for rate bound $\alpha\delta$ will be of the form:

1. B gets $m = 2n$ bits from a source of a rate bound $\alpha\delta$.

2. Produces $2k - 1$ strings of n bits each (k will be specified later).

3. Runs A on each string (as random input) and takes a majority vote.

The $2k-1$ n bit strings are produced by treating the m bits as vertices of a digraph G which is the l-th power of a d-regular expander (d and l are odd), partitioning the edge set into d^l perfect matching, and from each matching producing a string of n bits by xoring the endpoints of the n edges of the matching.

Suppose Z is the set of vertices of G corresponding to the "good bits". By corollary 2.8.2 we have:

$$|ZZ^c| \geq |Z|(1 - \alpha\delta)(d^l - |\bar{\lambda}|^l)$$

where $|\bar{\lambda}|$ is the eigenvalue of the original expander. Therefore the average number of edges of ZZ^c per matching is bounded below by

$$|Z|(1 - \alpha\delta)(1 - \frac{|\bar{\lambda}|^l}{d^l})$$

It follows that a majority of the matchings contain at least:

$$|Z| \left(2(1 - \alpha\delta)(1 - \frac{|\bar{\lambda}|^l}{d^l}) - 1 \right)$$

edges of ZZ^c.

Since xoring the endvertices of an edge of ZZ^c produces a "good bit", and since $|Z| \geq 2n\alpha\delta$, a majority of the n bits strings contain at least a

$$2\alpha\delta \left(2(1 - \alpha\delta)(1 - \frac{|\bar{\lambda}|^l}{d^l}) - 1 \right)$$

fraction of good bits. Choosing α small enough, and l large enough, the factors $(1 - \alpha\delta)$ and $(1 - \frac{|\bar{\lambda}|^l}{d^l})$ can made as close to 1 as desired, and hence the above bound can be made to exceed α.

Setting $2k - 1 = d^l$ it follows that at least k of the n bit strings produced by B contain at least $n\alpha$ good bits. The error probability of B is bounded by the probability that A errs on any of the k strings and hence by $k\epsilon$. ∎

Proof (of theorem 7.5.1): By corollary 7.3.3 and lemma 7.5.2 we may assume that A can be simulated by a source of rate bound α which is sufficiently small to meet the condition of lemma 7.5.3. Also as the proof of theorem 7.3.1 shows, one can assume that the error is exponentially small. Applying lemma 7.5.3 $c \log n$ times, one gets a probabilistic algorithm which requires $m = n^{1+c}$ random bits from a source of entropy rate bound $\alpha n^{c \log \delta}$, with error probability exponentially small. Hence m^ϵ entropy suffices where:

$$\epsilon = \frac{1 + c \log 2\delta}{1 + c}$$

and by choosing c and δ appropriately ϵ can be made arbitrarily small. ∎

Chapter 8

Conclusion.

In this work we developed a sampling technique based on efficiently constructible disperser graphs. We used this technique to tackle the deterministic amplification and the imperfect random sources problems, for the RP and BPP complxity classes. At the conclusion, we would like to state some of the remaining open questions:

Deterministic amplification.

It was shown how to achieve an exponentially small error with $(3 + \epsilon)n$ random bits (RP), or $(6 + \epsilon)n$ random bits (BPP). The information-theoretic lower bound is $2n$. Bridging this gap is an open (though far from exciting) problem.

Imperfect random sources.

It was shown that $\frac{1}{2}$ $(\frac{3}{4})$ Renyi entropy rate suffices for the simulation of RP (BPP). Recently, this result was improved by Zuckerman [Zuc], who showed that any constant Renyi rate suffices. The sufficiency of an $o(1)$ lower bound is an open question. Our reduction of the simulation problem to the problem of efficiently constructing certain disperser graphs, leads us to conjecture that $n^{\Omega(1)}$ Renyi entropy suffices for the simulation. As we have indicated, this is also a lower bound. Proof or disproof of this conjecture, is the major open question in this field. In fact even simulations of RP (BPP) with specific random sources such as nonoblivious bit fixing sources or SR and PRB sources of $o(1)$ Renyi entropy rate, are open.

LOGSPACE simulations.

The sampling techniques we have given, do not seem to work in logarithmic space. Thus exponential amplification at a linear cost of random bits, and simulations with a constant lower bound on the Renyi entropy rate, are open questions for randomized LOGSPACE computations.

Efficient constructions of disperser graphs.

The problem of giving efficient constructions of disperser graphs is an interesting mathematical problem. It has various applications some of which appeared in this work (for others see for instance [Sip86, UW87]). Among the open questions:

- Give efficient (and hopefully simple) constructions of constant degree $\frac{1}{2}$-separable digraphs.

- Give efficient constructions of the graphs whose existence is established in corollary 5.1.2 (with $\mu = o(1)$ and in particular, with $\mu = (\log M_i)^{-\eta}$ for any

$0 < \eta < 1$).

Bibliography

[ABI86]   N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Jour. of Algorithms*, 7:574, 1986.

[ABO84]   A V Ajtai and M Ben-Or. A theorem on probabilistic constant depth computations. In *STOC*, page 471, 1984.

[AC88]   N. Alon and F. R. K. Chung. Explicit constructions of linear sized tolerant networks. *Discrete Mathematics*, 72:15–19, 1988.

[Adl78]   L Adleman. Two theorems on random polynomial time. In *FOCS*, page 75, 78.

[AKS87]   M. Ajtai, J. Komlós, and E. Szemerédi. Deterministic Simulation in LOGSPACE. In *STOC*, page 132, 1987.

[Alo86a]   N. Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83, 1986.

[Alo86b]   N. Alon. Eigenvalues, geometric expanders, sorting in rounds and ramsey theory. *combinatorica*, 6:207–219, 1986.

[AM85]   N. Alon and V. Milman. λ_1, isoperimetric inequalities for graphs and superconcentrators. *Jour of Combinatorial Theory*, B 38:73, 1985.

[Bac87]   E. Bach. Realistic analysis of some randomized algorithms. In *STOC*, pages 453–461, 1987.

[Blu84]   M. Blum. Independent unbiased coin flips from a correlated source: a finite state markov chain. In *FOCS*, pages 425–433, 1984.

[BNS88]   L. Babai, N. Nisan, and M. Szegedy. Multiparty protocols and logspace hard pseudorandom sequences. In *STOC*, 1988.

[BOL85]   M. Ben-Or and N. Linial. Collective coin flipping robust voting schemes and minimal banzhaf values. In *FOCS*, pages 408–416, 1985.

[Buc86]   M. W. Buck. Expanders and diffusers. *Siam J Alg Disc Meth*, 7(2):282, 1986.

[CG85]   B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. In *FOCS*, pages 429–442, 1985.

[CG86]   B. Chor and O. Goldriech. On the power of two points based sampling. 1986.

[CGH⁺85] B. Chor, O. Goldreich, J. Hastad, J. Friedman, S. Rudich, and R. Smolenski. The bit extraction problem or t-resilient functions. In *FOCS*, pages 396–407, 1985.

[Chu]   F. R. K. Chung. Diameters and eigenvalues. Manuscript.

[CW89]   A. Cohen and A. Wigderson. Dispersers, deterministic amplification, and weak random sources. Submitted to the FOCS, 1989.

[Dia]   P. Diaconis. Group representation in probability and statistics. Preprint.

[FP87]   J. Friedman and N. Pippenger. Expanding graphs contain all small trees. *Combinatorica*, 7(1):71–76, 1987.

[Gan59]   F. R. Gantmacher. *The Theory of Matrices*, volume 2. Chelsea Publishing Company, 1959.

[GG81]   O. Gaber and Z. Galil. Explicit construction of linear sized superconcentrators. *J Comput System Sci*, 22:407, 1981.

[Hal86]   M. Hall. *Combinatorial Theory*. John Wiley and Sons, 2 edition, 1986.

[IZ]   R. Impagliazzo and D. Zuckerman. How to recycle random bits. FOCS 89.

[JM85]   S. Jimbo and A. Maruoka. Expanders obtained from affine transformations. In *STOC*, page 88, 1985.

[KKL88]   J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. In *FOCS*, pages 68–80, 1988.

[KPS85]   R. Karp, N. Pippenger, and M. Sipser. A time randomness tradeoff. In *AMS Conference on Probabilistic Computational Complexity Durham New Hampshire*, 1985.

[KPU88]   D. Krizanc, D. Peleg, and E. Upfal. A time-randomness tradeoff for oblivious routing. In *STOC*, pages 93–102, 1988.

[KRS88]   C. P. Kruskal, L. Rudolph, and M. Snir. A complexity theory of efficient parallel algorithms. Technical Report CS-88-4, Leibniz Center for Research in Computer Science Hebrew University Jerusalem, February 88.

[LLS87]   D. Lichtenstein, N. Linial, and M. Saks. Imperfect random sources and discrete controlled processes. In *STOC*, pages 169–177, 1987.

[LN83]   R. Lidl and H. Niederreiter. *Finite Fields*. Addison-Wesley, 1983.

[Lov90]   L. Lovász. *Communication complexity: a survey, in Path flowers and VLSI*. Springer Verlag, 1990.

[LPS86]   A. Lubotzky, R. Phillips, and P. Sarnak. Explicit expanders and the ramanujan conjecture. In *STOC*, page 240, 1986.

[Lub85]   M. Luby. A simple parallel algorithm for the maximal independent set problem. In *STOC*, page 1, 1985.

[Mar75]   M. A. Margulis. Explicit construction of concentators. *Prob Info Trans*, 9, 1975.

[Min88]   H. Minc. *Nonnegative Matrices*. John Wiley and Sons, 1988.

[MNN89]   R. Motwani, J. Naor, and M. Naor. The probabilistic method yields deterministic parallel algorithms. In *FOCS*, pages 8 – 13, 1989.

[New]   I. Newman. Private vs. common random bits in communication complexity. to appear in IPL.

[Nis90]   N. Nisan. Pseudorandom generators for space bounded computation. In *STOC*, 1990.

[NW88]   N. Nisan and A. Wigderson. Hardness vs. randomness. In *FOCS*, 1988.

[Pip85]   N. Pippenger. On networks of noisy gates. In *FOCS*, page 33, 1985.

[Ren70]   A. Renyi. *Probability Theory*. North-Holland Publishing Company, 1970.

[San]   M. Santha. On using deterministic functions to reduce randomness in probabilistic algorithms. Manuscript.

[Sar]   P. Sarnak. A draft of a book. Preprint.

[Sho88]   V. Shoup. New algorithms for finding irreducible polynomials over finite fields. In *FOCS*, pages 283–290, 1988.

[Sip86]   M. Sipser. Expanders randomness or time vs space. In *Structure in Complexity Theory*, page 325. Springer Verlag, 1986.

[SV84]   M. Santha and U. V. Vazirani. Generating quasi-random sequences from sligthly random sources. In *FOCS*, pages 434–440, 1984.

[SW86]   M. Saks and A. Wigderson. Probabilistic boolean decision trees, and the complexity of evaluating game trees. In *FOCS*, pages 29–38, 1986.

[Syl83]   J. J. Sylvester. On the equation to the secular inequalities in the planetary theory. *Philos. Mag.*, 16(5):267–269., 1883.

[Tan84]   R. M. Tanner. Explicit construction of concentrators fron generalized n-gons. *SIAM J. Algebraic Discrete Methods*, 5:287–293, 1984.

[UW87]   E. Upfal and A. Wigderson. How to share memory in a distributed system. *J. of the ACM.*, pages 116–127, 1987.

[Vaz86]   U. Vazirani. *Randomness Adversaries and Computation.* PhD thesis, University of California Berkeley, 1986.

[VV85]   U. V. Vazirani and V. V. Vazirani. Random polynomial time is equal to semi random polynomial time. In *FOCS*, pages 417–428, 1985.

[Zuc]   D. Zuckerman. General weak random sources. Preprint.