CS225: Pseudorandomness Prof. Salil Vadhan

Lecture 1: Introduction

February 1, 2007

Based on scribe notes by Saurabh Sanghvi.

1 Course Overview

Over the past few decades, *randomization* has become one of the most pervasive paradigms in computer science. Its widespread uses include:

Algorithm Design: For a number of important algorithmic problems, the most efficient algorithms known are randomized. For example:

- PRIMALITY. This was shown to have a randomized polynomial-time algorithm in 1977. It wasn't until 2002 that a deterministic polynomial-time algorithm was discovered. (We will see this algorithm, but not its proof.)
- APPROXIMATE COUNTING. Many approximate counting problems (e.g. counting perfect matchings in a bipartite graph) have randomized polynomial-time algorithms, but the fastest known deterministic algorithms take exponential time.
- UNDIRECTED S-T CONNECTIVITY. This was shown to have a randomized logspace algorithm in 1979. It wasn't until 2005 that a deterministic logspace algorithm was discovered (using tools from the theory of pseudorandomness). We will cover this result in class.
- PERFECT MATCHING. This was shown to have a randomized polylogarithmic-time parallel algorithm in the late 1970's. Deterministically, we only know polynomial-time algorithms.

Cryptography: Randomization is central to cryptography. Cryptography is concerned with protecting secrets, and how can something be secret if it is deterministically fixed? For example, we assume that cryptographic keys are chosen at random (e.g. uniformly from the set of n-bit strings). In addition to the keys, it is known that often the cryptographic algorithms themselves (e.g. for encryption) must be randomized to achieve satisfactory notions of security (e.g. that no partial information about the message is leaked).

Combinatorial Constructions: Randomness is often used to prove the *existence* of combinatorial objects with a desired properties. Specifically, if one can show that a randomly chosen object has the probability with nonzero probability, then it follows that such an object must, in fact, exist. A famous example due to Erdös is the existence of *Ramsey graphs*: A randomly chosen n-vertex graph has no clique or independent set of size $2 \log n$. We will see several other applications of the probabilistic method in the course, such as with two important objects mentioned below: expander graphs and error-correcting codes.

Though these *applications* of randomness are interesting and rich topics of study in their own right, they are not the focus of the course. Rather, we ask the following:

Main question: Can we reduce or even eliminate the use of randomness in these settings?

We have several motivations for doing this.

- Complexity Theory: We are interested in understanding and comparing the power of various kinds computational resources. Since randomness is such a widely used resource, we want to know how it relates to other resources such as time, space, and parallelism. In particular, we ask: *Can every randomized algorithm be derandomized with only small loss in efficiency?*
- Using Physical Random Sources: It is unclear whether the real world has physical sources of perfect randomness. We may use sources that seem to have some unpredictability, like the low order bits of a system clock or thermal radiation noise, but these sources will generally have biases and (more problematically) correlations. This reasoning leads us to ask questions like: What can we do with a source of biased and correlated bits?
- Explicit Constructions: Probabilistic constructions of combinatorial objects often do not provide us with efficient algorithms for using those objects. Indeed, the randomly chosen object often has a description that is exponential in the relevant parameters. Thus, we look for *explicit* constructions — ones that are deterministic and efficient. Finding such explicit constructions is also considered to demonstrate that we have achieved an improved understanding of the object at hand. The best known explicit construction of n-vertex Ramsey graphs gives no clique or independent set of size $2^{\log^{o(1)} n}$ (STOC 2006) and is based on the theory of pseudorandomness (namely randomness extractors).
- Unexpected Applications: In addition, the theory of pseudorandomness has turned out to have many applications to problems that seem to have no connection to derandomization. These include data structures, distributed computation (e.g. leader election), circuit lower bounds in complexity theory, reducing interaction in interactive protocols, saving memory in streaming algorithms, and more. We will see some of these applications in the lectures and the homework assignments.

The paradigm we will use to study the main question is that of *pseudorandomness*: efficiently generating objects that "look random" using little or no randomness.

Specifically, we will study four "pseudorandom" objects:

Pseudorandom generators (PRGs): A PRG is an algorithm that takes as input a short, perfectly random *seed* and then returns a (much longer) sequence of bits that "look random." That the bits output cannot be perfectly random is clear — the output is determined by the seed and there are far fewer seeds than possible bit sequences (since the sequence is longer than the seed). Thus, "looks random" must be formalized. We will formalize this notion as saying that a sequence of bits looks random if no efficient algorithm can distinguish the bits from those of a truly random sequence. The modern theory of pseudorandom generators originated in cryptography (where they have numerous applications); in this course, we will see their role in derandomizing algorithms.

Note that asserting that a function is a PRG is a statement about something that efficient algorithms can't do (in this case, distinguish two sequences). But proving that efficient algorithms cannot compute things is typically out of reach for theoretical computer science; indeed this is why the **P** vs. **NP** question is so hard. Thus, in this course, we will settle for conditional statements. An ideal theorem would be something like: "If $\mathbf{P} \neq \mathbf{NP}$, then pseudorandom generators exist." (The assumptions we make won't exactly be $\mathbf{P} \neq \mathbf{NP}$, but things of the same flavor.)

Randomness Extractors: A randomness extractor takes as input a source of biased and correlated bits, and then produces a sequence of almost-uniform bits as output. Ideally, extractors would be deterministic, but as we will see this proves to be impossible for general sources of biased and correlated bits. Nevertheless, we will get close—producing extractors that are only "mildly" probabilistic.

Expander Graphs: Expanders are graphs with two seemingly contradictory properties: they have very low degree (e.g. constant in the number of vertices), but they also are "well-connected" in some well defined sense. For example, one might say that the graph cannot be bisected without cutting a large (say, constant) fraction of the edges.

It is not obvious that such graphs exist, but in fact it can be shown that a random graph of degree 3 is 'good' expander with high probability. (This is another application of the probabilistic method.) Expander graphs have numerous applications in theoretical computer science. They were originally studied for their use in designing fault-tolerant networks (e.g. for telephone lines), which are networks that maintain good connectivity even when links or nodes fail. But they also have less obvious applications, such as an $O(\log n)$-time algorithm for sorting in parallel.

Most of these applications of expander graphs really need *explicit constructions*, and these proved much harder to find. We will see some explicit constructions in this course, but they do not always match the bounds given by the probabilistic method (in terms of the degree/expansion tradeoff).

Error-Correcting Codes: Error-correcting codes (ECCs) are tools for communicating over noisy channels. Specifically, they specify a way to encode messages into longer, redundant codewords so that even if the codeword gets somewhat corrupted along the way, it is still possible for the receiver to decode the original message. In his landmark paper that introduced the field of coding theory, Shannon also proved the existence of good error-correcting codes via the probabilistic method. That is, a random mapping of n-bit messages to $O(n)$-bit codewords is a "good" error-correcting code with high probability. Unfortunately, these probabilistic codes are not feasible to actually use — a random mapping requires an exponentially long description, and we know of no way to decode such a mapping efficiently. Again, explicit constructions are needed.

In this course, we will focus on the problem of *list decoding*. Specifically, we will consider scenarios where the number of corruptions is so large that unique decoding is impossible; at best one can produce a short list that is guaranteed to contain the correct message.

Each of the above objects has been the center of a large and beautiful body of research, but until recently these corpora were largely distinct. An exciting recent development has been the realization that all four of these objects are almost *the same* when interpreted appropriately. Their intimate connections will be a major focus of the course, tying together the variety of constructions

and applications that we will see.

The surprise and beauty of these connections has to do with seemingly different nature of each of these objects. PRGs, by asserting what efficient algorithms cannot do, are objects of complexity theory. Extractors, with their focus on extracting the entropy in a correlated and biased sequence, are information-theoretic objects. Expander graphs are of course combinatorial objects (as defined above), though they can also be interpreted algebraically, as we will see. Error-correcting codes involve a mix of combinatorics, information theory, and algebra. Because of the connections, we obtain new perspectives on each of the objects, and make substantial advances on our understanding of each by translating intuitions and techniques from the study of the others.