



Technion IIT
Department of Computer Science
Spring 2009–10

Course 236649:

**RESEARCH LABORATORY IN
THEORETICAL COMPUTER SCIENCE**

Eli Ben-Sasson

Table of Contents

Lecture 2: A constructive proof of the Lovász Local Lemma

2.1	Statement of the constructive version of the lemma	2-1
2.2	Proof of the constructive version of the lemma	2-2

Lecture 3: Local search on 2-CNFs and random 3-CNFs

3.1	Random walk applied to satisfiable 2-CNFs	3-1
3.2	Random walk applied to random 3-CNFs with small clause density	3-2

Lecture 6: The Parallel Repetition Theorem

6.1	Statement of the theorem and main lemma	6-1
-----	---	-----

Lecture 7: The Parallel Repetition Theorem Part II

7.1	Conditioning on a frequent event does not significantly change a product distribution's marginal	7-1
7.2	Sampling from similar distributions in a coordinated manner	7-2

Lecture 8: The Parallel Repetition Theorem Part III

8.1	Defining the pair of strategies	8-1
8.2	Similar distributions	8-2

Lecture 9: The Parallel Repetition Theorem Part IV

Lecture 11: Deterministic polynomial time primality test

11.1	The Agarwal-Biswas randomized primality test	11-1
------	--	------

Lecture 12: Deterministic polynomial time primality test — Part II

12.1	Overview of the proof of Theorem 12.1	12-1
12.2	Part 1 — The field F and polynomial $P(Y)$	12-2
12.3	Part 3 — Its all about the roots of $P(Y)$	12-3

Lecture 13: Deterministic polynomial time primality test — Part III

13.1	A small set of roots for $P(Y)$	13-1
13.2	A large set of roots for $P(Y)$	13-1
13.2.1	S is large	13-2
13.2.2	Elements of S are roots of $P(Y)$	13-2

References

2.1 Statement of the constructive version of the lemma

The first topic of our research laboratory this year is the novel constructive proof of the Lovász Local Lemma (LLL) which appeared originally in [Erdos and Lovász \[1975\]](#).

Theorem 2.1 (Lovász Local Lemma). *Let \mathcal{A} be a finite set of events in a probability space. For $A \in \mathcal{A}$ let $\Gamma(A)$ be a subset of \mathcal{A} satisfying that A is independent from the collection of events $\mathcal{A} \setminus (\{A\} \cup \Gamma(A))$. If there exists an assignment of reals $x : \mathcal{A} \rightarrow (0, 1)$ such that for all $A \in \mathcal{A}$*

$$\Pr[A] \leq x(A) \prod_{B \in \Gamma(A)} (1 - x(B)), \quad (1)$$

then the probability of avoiding all events in \mathcal{A} is at least $\prod_{A \in \mathcal{A}} (1 - x(A))$, which is positive.

Last week we presented the “classical” proof of the lemma, which is nonconstructive (cf. [\[Alon and Spencer, 1992, Chapter 5\]](#)). This week we start the novel constructive proof of this lemma due to [Moser \[2009\]](#); [Moser and Tardos \[2010\]](#). By “constructive” we mean that if there is an effective way to describe the probability space, then we can find an element of this space that avoids all events in \mathcal{A} . To effectively describe the probability space we assume there exists a finite set of mutually independent random variables $\mathcal{P} = \{P_1, \dots, P_n\}$ (we do not assume these variables are identically distributed) such that each event A is determined by a subset of \mathcal{P} , denoted $\text{vbl}(A)$. In this case, the *dependency graph* $G = G_{\mathcal{A}}$ of \mathcal{A} is the graph on vertex set \mathcal{A} with an edge between $A \neq B$ iff $\text{vbl}(A) \cap \text{vbl}(B) \neq \emptyset$. In what follows let $\Gamma(A)$ denote the set of neighbors of A in $G_{\mathcal{A}}$ and let $\Gamma^+(A) = \{A\} \cup \Gamma(A)$. We assume that we can sample each variable P_i according to its distribution. Consider the following “local-search” algorithm that essentially starts with a random assignment and as long as some event $A \in \mathcal{A}$ is violated, tries to “locally” correct the situation by resampling $\text{vbl}(A)$.

Definition 2.2 (Local search algorithm). Given \mathcal{P}, \mathcal{A} as described above, the local search algorithm proceeds as follows:

Initialize For all $P_i \in \mathcal{P}$ let $\alpha(P_i)$ be a random evaluation of P_i . Let $\alpha = (\alpha(P_1), \dots, \alpha(P_n))$.

Correct While $\exists A \in \mathcal{A}$ that is violated under α , pick an arbitrary violated event A and resample all variables $P \in \text{vbl}(A)$. Call the modified assignment α .

Output α .

The constructive version of LLL says that the local-search algorithm terminates quickly on expectation. (This algorithm can be derandomized under further assumptions, cf. [Moser and Tardos, 2010, Theorem 1.4].)

Theorem 2.3 (Lovász Local Lemma — constructive version). *Let $\mathcal{P}, \mathcal{A}, \Gamma$ be as defined above. If there exists an assignment of reals $x : \mathcal{A} \rightarrow (0, 1)$ such that for all $A \in \mathcal{A}$*

$$\Pr[A] \leq x(A) \prod_{B \in \Gamma_{\mathcal{A}}(A)} (1 - x(B)), \quad (2)$$

then the local search algorithm described above finds an assignment that does not violate any event of \mathcal{A} after an expected number of at most $\sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}$ correction steps.

2.2 Proof of the constructive version of the lemma

Our proof closely follows Moser and Tardos [2010]. The *log* of an execution of the local-search algorithm is a mapping $C : \mathbb{N} \rightarrow \mathcal{A}$ which records the sequence of events $A \in \mathcal{A}$ that are corrected during the execution. (If the algorithm terminates, C is partial and defined only up to the number of correction steps.)

The proof goes by defining a sequence of trees that “explain” the sequence of events recorded in the log. Then, we bound the size of the log by bounding the probability that any particular explanation appears.

Definition 2.4 (Proper witness tree). A *witness tree* $\tau = (T, \sigma)$ is a finite rooted tree T that is labeled by $\sigma : V(T) \rightarrow \mathcal{A}$ and satisfies the following condition: The children of $u \in V(T)$ receive labels from $\Gamma^+(\sigma(u))$. If distinct children of a vertex always receive distinct labels we say the witness tree is *proper*.

Next, we associate a witness tree with each entry of the log C , the tree associated with the t^{th} entry of the log “explains” how we arrived at the need to correct A at time t .

Definition 2.5 (Explanation of execution log). Given a log $C : \mathbb{N} \rightarrow \mathcal{A}$ and $t \in \mathbb{N}$ such that $C(t) = A$, let $\tau_C^{(t)}(t)$ be the witness tree that has a single vertex (the root), labeled by A . For $i = t - 1, t - 2, \dots, 1$ let $\tau_C^{(i)}(t)$ be defined as follows:

1. If there exists a vertex in $\tau_C^{(i+1)}(t)$ that is labeled by an event B such that $\text{vbl}(B) \cap \text{vbl}(C(i)) \neq \emptyset$, then choose a vertex v among all such vertices with maximal distance from the root (breaking ties arbitrarily), attach a new child to v and label it by $C(i)$. The resulting tree is $\tau_C^{(i)}(t)$.
2. Otherwise — when no vertex in $\tau_C^{(i+1)}(t)$ is labeled by an event B with $\text{vbl}(B) \cap \text{vbl}(C(i)) \neq \emptyset$ — set $\tau_C^{(i)}(t) = \tau_C^{(i+1)}(t)$.

Finally, the *explanation* of the t^{th} entry of the execution log C is $\tau_C(t) = \tau_C^{(1)}(t)$.

Problem 2.1. Prove: If τ is the explanation of the t^{th} entry of C then τ is a *proper* witness tree.

Problem 2.2. Prove: The probability that $\tau = (T, \sigma)$ appears as the explanation of some entry of the log is at most $\prod_{v \in V(T)} \Pr[\sigma(v)]$. Hints:

- Assume that the source of randomness for our algorithm is given by n infinite sequences, one sequence per variable P_i . Each time we sample P_i , we read the next entry from the relevant sequence, denoted S_i . Let S_{ij} denote the j^{th} element of S_i .
- Let v_1, \dots, v_m be an ordering of $V(T)$ such that for all $i < j$ the event that labels v_i occurred in the execution log before the event that labels v_j .
- Argue, by induction on the order of vertices, that the event labeling each v_i depends on a subset of the S_{ij} 's and that all of these subsets are disjoint.

The following process will be used later on to bound the probability that a specific witness tree τ appears in the log. A similar process was first analyzed in [Watson and Galton \[1875\]](#) in an attempt to understand why certain aristocratic family names become extinct. Today, Galton–Watson processes are used, among other things, in the study of population genetics and epidemiology.

Definition 2.6 (Galton–Watson branching process). Fix $A \in \mathcal{A}$. Consider the following process for generating a proper witness tree. Start with a single vertex labeled by A . In each subsequent round, for each vertex v generated in the previous round, and for each $B \in \Gamma^+(\sigma(v))$, add to v a child labeled B with (independent) probability $x(B)$.

Problem 2.3. In what follows, let $x'(B) = x(B) \prod_{B' \in \Gamma(B)} (1 - x(B'))$. For τ a fixed proper witness tree with root labeled by A , show that the probability p_τ that the Galton–Watson process yields τ is precisely

$$p_\tau = \frac{1 - x(A)}{x(A)} \prod_{v \in V(\tau)} x'(\sigma(v)).$$

Problem 2.4. Complete the proof of [Theorem 2.3](#) by showing that the expected number of times that A appears in the execution log C is at most $\frac{x(A)}{1-x(A)}$. Hints:

- Let \mathcal{T}_A denote the (infinite) set of proper witness trees with root A (recall each such tree is finite). Let N_A be the random variable that counts the number of times that A appears in the execution log.
- Notice that each appearance of A in the log is explained by a *distinct* proper witness tree. Conclude

$$\mathbf{E}[N_A] = \sum_{\tau \in \mathcal{T}_A} \Pr[\tau \text{ appears in the execution log}].$$

- Bound the right hand side of the above expression using [Problem 2.2](#), [Problem 2.3](#) and our assumption that $\Pr[A] \leq x'(A)$ for all $A \in \mathcal{A}$.

In this session we will study two additional results associated with a variant of the local search algorithm of [Definition 2.2](#). In what follows we consider formulas in conjunctive normal form (CNF). Thus, the random variables will be denoted by x_1, \dots, x_n (in the previous session they were denoted by P_1, \dots, P_n), each variable is distributed uniformly over $\{0, 1\}$, and the events A_1, \dots, A_m are clauses. The variant we consider, called henceforth the *random walk algorithm*, is the following.

Definition 3.1 (Random walk algorithm). Given a CNF $\mathcal{A} = \{A_1, \dots, A_m\}$ over variables $\{x_1, \dots, x_n\}$ the random walk algorithm proceeds as follows:

Initialize Let $\alpha \in \{0, 1\}^n$ be a random assignment.

Correct While some clause is violated under α , pick an arbitrary violated clause A and a uniformly random variable x_i appearing in A and flip the assignment to x_i . Call the modified assignment α .

Output α .

The first result we shall study shows that the random walk algorithm finds an assignment to a satisfiable 2-CNF in expected quadratic time, and the second result shows that it finds in linear time, with high probability, a satisfying assignment to a random 3-CNF with a small number of clauses.

3.1 Random walk applied to satisfiable 2-CNFs

The results of this section are due to [Papadimitriou \[1991\]](#). Recall that a k -CNF is a CNF formula such that each clause contains at most k literals.

Problem 3.1. Prove: If \mathcal{A} is a satisfiable 2-CNF with n variables then the random walk algorithm applied to \mathcal{A} finds a satisfying assignment after an expected number of at most $O(n^2)$ correction steps. Hints:

- Fix an arbitrary assignment $\beta \in \{0, 1\}^n$ that satisfies \mathcal{A} .
- Let X_0, X_1, \dots be the sequence of random variables measuring the Hamming distance between β and the assignment of the local search algorithm, where X_i measures this distance after i correction steps. What can be said of $\mathbf{E}[X_i - X_{i-1} | X_{i-1}]$?

3.2 Random walk applied to random 3-CNFs with small clause density

The results of this section appeared in [Alekhovich and Ben-Sasson \[2006\]](#). We start with the definition of a random k -CNF.

Definition 3.2 (Random k -CNF). For $\Delta \geq 0$ and integers k, n and , let $\text{CNF}[k, n, \Delta]$ denote the following distribution over k -CNF formulas. A formula sampled from this distribution, denoted $\mathcal{A} \sim \text{CNF}[k, n, \Delta]$, is obtained by sampling independently $\lceil \Delta \cdot n \rceil$ clauses uniformly from all $2^k \cdot \binom{n}{k}$ clauses of size k .

There are many interesting open questions regarding random 3-CNF formulas, such as how exactly does the *clause density* Δ relate to the probability of a formula being satisfiable and to the ease/hardness of finding a satisfying assignment. A related question is to explain the empirically noticed success of simple heuristics in finding a satisfying assignment. Here's an example regarding the random walk algorithm.

Theorem 3.3 (Random walk requires linear time of random CNFs with small clause density). *For $\Delta < 1.63$ and $\mathcal{A} \sim \text{CNF}[k, n, \Delta]$, the random walk algorithm finds an assignment satisfying \mathcal{A} after $O(n)$ correction steps, with high probability. The probability is taken with respect to the random CNF and the coin tosses used by the algorithm.*

Empirically, it seems like the random walk algorithm requires linear time even for higher values of Δ but we have no rigorous theoretical explanation for this behavior.

The proof of [Theorem 3.3](#) has two parts. First, we argue in the next problem that every CNF that has a weighted assignment that “greatly” satisfies \mathcal{A} can be solved in short time by the random walk algorithm. Then we argue that a random \mathcal{A} has, with high probability, a “greatly” satisfying assignment.

Let \mathcal{A} be a satisfiable k -CNF over n variables. View each clause A as a vector in $\{-1, 0, 1\}^n$ by setting the i^{th} coordinate of A to be 1 if x_i appears positively in A , -1 if x_i appears negatively in A , and 0 if x_i does not appear in A . Recall the ℓ_1 norm of a vector $v \in \mathbb{R}^n$ is $\|v\|_1 = \sum_i |v_i|$ and $\|v\|_\infty = \max_i |v_i|$. Notice \mathcal{A} is satisfiable iff there exists $\beta \in \{-1, 1\}^n$ such that $\langle A, \beta \rangle > -\|A\|_1$ for all $A \in \mathcal{A}$, where $\langle \alpha, \beta \rangle = \sum_i \alpha_i \cdot \beta_i$.

Problem 3.2. Prove: If there exists a *weighted assignment* $\beta \in \mathbb{R}^n$ such that $\langle A, \beta \rangle \geq 1$ for all $A \in \mathcal{A}$, then the random walk algorithm finds a satisfying assignment to \mathcal{A} in time $O(\|\beta\|_1 \cdot \|\beta\|_\infty)$ with probability at least $1 - \exp(-\Omega(\|\beta\|_1 / \|\beta\|_\infty))$. Hints:

- A *submartingale* is a sequence of random variables X_0, X_1, \dots, X_m satisfying for all $0 \leq i < m$ $\mathbf{E}[X_{i+1} | X_i] \leq X_i$.
- Azuma's inequality (cf. [Alon and Spencer \[1992\]](#)): For X_0, X_1, \dots, X_m a submartingale satisfying $|X_{i+1} - X_i| \leq 1$, $0 \leq i < m$, and $\lambda > 0$ we have

$$\Pr[X_m > X_0 + \lambda\sqrt{m}] \leq \exp(-\lambda^2/2).$$

To get to the second part of the proof we need a couple of definitions.

Definition 3.4 (Pure literal decomposition). A variable is said to be *pure* in \mathcal{A} if it appears in \mathcal{A} only positively, or only negatively. Let S_0 be the set of pure literals in \mathcal{A} . (Notice that every clause containing a literal in S_0 can be removed from \mathcal{A} without affecting the satisfiability of \mathcal{A} .) Let $\mathcal{A}^{(1)}$ denote the CNF resulting from removing from \mathcal{A} all clauses that contain a pure literal from S_0 . For $i \geq 1$ define S_i to be the set of pure literals in $\mathcal{A}^{(i)}$ and let $\mathcal{A}^{(i+1)}$ be the CNF obtained from removing from $\mathcal{A}^{(i)}$ all clauses that contain a literal from S_i .

In what follows let $|\mathcal{A}|$ denote the number of clauses in \mathcal{A} . For $\mathcal{A}' \subseteq \mathcal{A}$ let $\text{Vars}(\mathcal{A}')$ denote the set of variables that appear in some clause of \mathcal{A} (either positively or negatively).

Definition 3.5 (Expanding CNF). A CNF \mathcal{A} is said to be (r, c) -expanding if for all subformulas $\mathcal{A}' \subseteq \mathcal{A}$, $|\mathcal{A}'| \leq r$ we have $|\text{Vars}(\mathcal{A}')| \geq c \cdot |\mathcal{A}'|$.

We need the following claim from [Alekhovich and Ben-Sasson \[2006\]](#) (the proof can be found there).

Lemma 3.6. *For $\Delta < 1.63$ there exists a constant d such that for a random CNF $\mathcal{A} \sim \text{CNF}[k, n, \Delta]$, with high probability the subformula $\mathcal{A}^{(d)}$ is $(|\mathcal{A}^{(d)}|, 7/4)$ -expanding.*

Problem 3.3. Prove [Theorem 3.3](#) using [Problem 3.2](#) and [Lemma 3.6](#). Hints:

- Construct an assignment $\beta_0 \in \mathbb{R}^n$ that satisfies the requirements of [Problem 3.2](#) with respect to the CNF $\mathcal{A} \setminus \mathcal{A}^{(d)}$. Assume this CNF can be partitioned into d formulas $\mathcal{A}'_1, \dots, \mathcal{A}'_d$ such that each clause of \mathcal{A}'_i contains a literal from S_i .
- Construct an assignment $\beta_1 \in \mathbb{R}^n$ that satisfies the requirements of [Problem 3.2](#) with respect to $\mathcal{A}^{(d)}$. Use the expansion assumption.
- Combine β_0, β_1 into one assignment that satisfies [Problem 3.2](#) with respect to \mathcal{A} .

We end this topic with the following open question.

Problem 3.4. Can the constructive proof of the Lovász Local Lemma proved in the previous session be used to analyze the expected running time of either the local search algorithm, or the random walk algorithm, on a satisfiable 2-CNF? Can it be used to bound the expected running time of these algorithms on $\mathcal{A} \sim \text{CNF}[k, n, \Delta]$ for larger values of Δ than shown above? For more information on this latter question, see [\[Luby et al., 1998, Section 3\]](#).

The next few sessions will be devoted to a proof of the Parallel Repetition Theorem of Raz [1998]. In our previous session we started discussing this theorem and recalled the necessary information-theoretic definitions using [Cover and Thomas, 1991, Sections 2.1–2.6] as our reference. We shall study the simplified proof of this theorem due to Holenstein [2009] and closely follow the cleaner exposition of Holenstein’s proof given in Rao [2008]. Let us start by defining and stating formally the theorem itself.

6.1 Statement of the theorem and main lemma

The following notion of a game arose from the study of probabilistically checkable proofs (PCPs) as a method for obtaining tight inapproximability results on NP-complete problems. The connection of games to PCP and hardness of approximation will be discussed in the recitation and is described, e.g., in [Guruswami and O’Donnell, 2005, Lecture 11].

Definition 6.1 (Two-prover one-round game). A *two-prover one-round game* G , called for simplicity a *game*, is defined by (i) a distribution (X, Y) on a set of questions $\mathcal{X} \times \mathcal{Y}$, (ii) a set of answers $\mathcal{A} \times \mathcal{B}$, and (iii) a decision predicate $V : \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{B} \rightarrow \{0, 1\}$. (We associate the decision 1 with “accept”/“win” and 0 with “reject”/“lose”.) A *strategy* is a pair of randomized functions $a : \mathcal{X} \times \mathcal{R} \rightarrow \mathcal{A}, b : \mathcal{Y} \times \mathcal{R} \rightarrow \mathcal{B}$. The *value* of the game is

$$v(G) = \max_{X, Y, R} \mathbf{E} [V(X, Y, a(X, R), b(X, R))]$$

where the maximum is taken over any pair of strategies a, b . The *answer length* of G is $\log |\mathcal{A}| + \log |\mathcal{B}|$.

Problem 6.1. Prove: For every game G there exists a pair of *deterministic* strategies $a : \mathcal{X} \rightarrow \mathcal{A}, b : \mathcal{Y} \rightarrow \mathcal{B}$ such that $\mathbf{E}_{X, Y} [V(X, Y, a(X), b(X))] = v(G)$. (The reason we choose to allow randomized strategies is that later on it will be easier for us to construct randomized strategies for proving the parallel repetition theorem.)

The parallel repetition theorem is concerned with the value of the game G repeated n times *in parallel*. By this we mean that n question-pairs $(x_1, y_1), \dots, (x_n, y_n)$ are sampled independently according to the distribution (X, Y) and all n questions are sent in one batch to each “player”, i.e., one player receives (x_1, \dots, x_n) and the other receives (y_1, \dots, y_n) . The players each respond with n answers, denoted (a_1, \dots, a_n) and (b_1, \dots, b_n) respectively and

now the game is a “win” if and only if all n -rounds of it are “wins”, i.e., $V(x_i, y_i, a_i, b_i) = 1$ for $i = 1 \dots n$.

Definition 6.2 (n -wise repeated game). Given a game $G = \langle (X, Y), \mathcal{X}, \mathcal{Y}, \mathcal{A}, \mathcal{B}, V \rangle$ as defined above, the n -wise repeated game G^n is the game defined by (i) the product distribution $(X, Y)^n$ on the set of questions $\mathcal{X}^n \times \mathcal{Y}^n$, (ii) the set of answers $\mathcal{A}^n \times \mathcal{B}^n$ and (iii) the decision predicate $V^n : \mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{A}^n \times \mathcal{B}^n$ given by

$$V^n(x_1, \dots, x_n, y_1, \dots, y_n, a_1, \dots, a_n, b_1, \dots, b_n) = \prod_{i=1}^n V(x_i, y_i, a_i, b_i).$$

At first sight it seems obvious that $v(G^n) \leq (v(G))^n$. After all, what better strategy than repeating n times the best strategy for G ? Alas, as shown in the previous session, our intuition regarding the parallel repetition is false, and for every n there exist games satisfying $v(G^n) \geq v(G)/6$ Feige and Lovász [1992]. (We showed an example of a game due to Feige [1998] for which $v(G^2) = v(G)$.) In spite of this, repeated games do in fact result in an exponential decrease in game-value as shown in the following celebrated Parallel Repetition Theorem of Raz [1998]. The version we quote below, due to Holenstein [2009], has better parameters* and is the one we shall prove.

Theorem 6.3 (Parallel repetition). *There is a universal constant $\alpha > 0$ such that for every game G with value $1 - \epsilon$ and answer length c , the value of the n -wise repeated game G^n is at most $(1 - \epsilon/2)^{\alpha \epsilon^2 n/c}$.*

The proof of this theorem uses the Main Lemma stated next. Fix any pair for strategies for the repeated game. For $S \subset [n]$ let W_S be the event that the game is won in the coordinates in S , i.e., $V(x_i, y_i, a_i, b_i) = 1$ for all $i \in S$.

Lemma 6.4 (Main lemma). *Let G be a game with value at most $1 - \epsilon$ and answer size c . let $S \subset [n]$, $|S| = k$ and $\gamma > 0$ be such that*

$$\Pr[W_S] \geq 2^{-\gamma^2(n-k)+kc}. \tag{3}$$

Then for i chosen uniformly from outside S we have

$$\mathbf{E}_{i \notin S} \left[\Pr[W_{\{i\}} | W_S] \right] \leq 1 - \epsilon + 25\gamma. \tag{4}$$

Problem 6.2. Prove the Parallel Repetition Theorem using the Main lemma.

We shall start the proof of a simple case of the theorem in next session. As a warm-up,

*The bound on the value of the repeated game in Raz’s paper was $(1 - \epsilon/2)^{\alpha \epsilon^{32} n/c}$. Figuring out the best exponent of ϵ in various games is an interesting question with connections to the “unique games conjecture”, but we shall not pursue these connections in this course.

Problem 6.3. Prove Propositions 11, 12, 13, 15 and facts 21, 24 of Rao [2008] (we use the notation given there):

Proposition 11 Let D, F be two random variables over a set S such that $|D - F| \leq \epsilon$. Let g be any function on S . Then $|g(D) - g(F)| \leq \epsilon$.

Proposition 12 Let A, B, C be random variables over S with $A \stackrel{\epsilon_1}{\approx} B \stackrel{\epsilon_2}{\approx} C$. Then $A \stackrel{\epsilon_1 + \epsilon_2}{\approx} C$.

Proposition 13 Let A, A', B, B' be random variables such that $|A - A'| = 0$. Then for every a , $|(B|A = a) - (B'|A' = a)| \leq |AB - A'B'| / \Pr[A = a]$.

Proposition 15 Let X, Y be independent random variables and E be an event that depends only on Y , i.e., given the value of Y we know whether E holds. Then $XY|E$ are independent random variables, i.e., for every x, y we have $\Pr[X = x \text{ and } Y = y|E] = \Pr[X = x|E] \cdot \Pr[Y = y|E]$.

Fact 21 Let A, B be random variables in some probability space. Let A' be another random variable such that $|A - A'| \leq \epsilon$. Then $|\{AB\} - \{A'\}\{B|A'\}| \leq \epsilon$.

Fact 24 If V is a random variable, E is any event with $\Pr[E] \geq 2^{-d}$, and $\tilde{V} = V|E$, then $D(\tilde{V}||V) \leq d$.

From here on we focus on proving the main lemma, [Lemma 6.4](#), which we have shown in [Problem 6.2](#) to imply the parallel repetition [Theorem 6.3](#). The proof goes by way of contradiction. Assume that [Equation 4](#) does not hold. We shall construct a pair of strategies for the two players that wins the game with probability greater than its value, thereby reaching a contradiction. The plan is for the two players, upon receiving x, y respectively, to each complete her input into a tuple x^n, y^n such that (i) the pair (x, y) is the i^{th} coordinate of x^n, y^n and (ii) W_S holds for x^n, y^n . They can then use the assumption that [Equation 4](#) does not hold to win on x, y with probability greater than $1 - \epsilon$.

There are a few obstacles we need to overcome to implement our proof and we shall deal with them one at a time.

7.1 Conditioning on a frequent event does not significantly change a product distribution's marginal

If the players are to assume that their inputs can be viewed as the i^{th} coordinate of the distribution $(X^n, Y^n)|W_S$ then it better be the case that the i^{th} marginal of $(X^n, Y^n)|W_S$, i.e., the i^{th} coordinate of a sample from this conditional distribution, be close to the original distribution (X, Y) . The following lemma says that for most $i \notin S$ this is true.

Lemma 7.1 (Frequent events do not change the expected marginal distribution). *Let U_1, \dots, U_n be independent random variables and E be an event in the same probability space that holds with probability at least 2^{-d} . Then*

$$\mathbf{E}_{i \in [n]} [\| \{U_i\} - \{U_i | E\} \|] \leq \sqrt{\frac{d}{n}}.$$

Problem 7.1. Prove [Lemma 7.1](#). Hints: Square the left hand side, use [Fact 24](#) and the following facts that are given without proof (we continue to use the numbering from [Rao \[2008\]](#)):

Fact 23 $D(V||U) \geq |V - U|^2$.

Fact 25 If U_1, \dots, U_n are independent random variables and V_1, \dots, V_n are other random variables then

$$\sum_{i=1}^n D(V_i||U_i) \leq D(V_1, \dots, V_n||U_1, \dots, U_n).$$

Lemma 7.1 implies that for a “typical” $i \notin S$ the marginal distribution $X_i, Y_i | W_S$ is close to the distribution X, Y . Fix i to be a typical coordinate. So when the first player sees x and the second player sees y , they can each think of their inputs as having been drawn jointly from the distribution $X^n, Y^n | W_S$. Next we devise a strategy for the two players in an unrealistically simple case.

Problem 7.2. Assume that the distribution $X^n | W_S$ is independent of the distribution $Y^n | W_S$, i.e.,

$$\Pr[X^n = x^n, Y^n = y^n | W_S] = \Pr[X^n = x^n | W_S] \cdot \Pr[Y^n = y^n | W_S].$$

Assume furthermore that **Equation 4** does not hold, i.e., that for all $i \notin S$ we have

$$\Pr[W_{\{i\}} | W_S] > 1 - \epsilon + 25\gamma.$$

Prove that the value of the game G is strictly greater than $1 - \epsilon$.

The first assumption above is unrealistic for two reasons. First, even for $S = \emptyset$ it could be that X and Y are dependent and this implies that X^n and Y^n are dependent. Indeed, this happens in typical uses of the Parallel Repetition Theorem in the context of soundness-amplification of PCP systems, cf. [Guruswami and O’Donnell, 2005, Lecture 11]. Second, even if X and Y are independent, the event W_S depends on both X^n and Y^n . So conditioning on this event may cause X^n and Y^n to become dependent.

7.2 Sampling from similar distributions in a coordinated manner

The next problem we need to overcome, which appears in other settings as well, is that of sampling in a coordinated manner. Suppose we have several players that cannot communicate but have shared randomness (i.e., they all view the same random string). Each player receives a description of a distribution over a set \mathcal{A} as her input and these distributions are pairwise close. Can each player sample from her distribution and yet have all players end up with the same element of \mathcal{A} ?

Lemma 7.2 (Correlated noncommunicating sampling). *Let A_1, \dots, A_ℓ be distributions on a finite set \mathcal{A} such that $|A_1 - A_i| \leq \epsilon_i$ for $i \in \{2, \dots, \ell\}$. Then ℓ noncommunicating players can use shared randomness to sample B_1, \dots, B_ℓ such that for every $i \in \{2, \dots, \ell\}$:*

- B_i is distributed identically to A_i
- $\Pr[B_1 \neq B_i] \leq 2\epsilon_i$

Consequently, the probability that $B_i \neq B_j$ for some $i, j \in [\ell]$ is at most $2 \sum_{i=2}^{\ell} \epsilon_i$.

Problem 7.3. Prove [Lemma 7.2](#). Hints: Draw (on a mental sheet of paper) a histogram where, for each $a \in \mathcal{A}$ the height of the bar corresponding to a is $\Pr[A_i = a]$. Draw (on the same piece of paper) a similar histogram for the distribution A_1 . Use shared randomness to sample a point on the piece of paper.

Let us see, as an example, how [Lemma 7.2](#) can be used to prove yet another unrealistic simple case of [Lemma 6.4](#).

Problem 7.4. Assume [Equation 4](#) does not hold, i.e., that for all $i \notin S$ we have

$$\Pr[W_{\{i\}} | W_S] > 1 - \epsilon + 25\gamma.$$

Furthermore, assume there is a random variable R ranging over a set \mathcal{R} (this variable may depend on both X_i and Y_i) and two functions $f : \mathcal{X} \times \mathcal{R} \rightarrow \mathcal{X}^n$ and $g : \mathcal{Y} \times \mathcal{R} \rightarrow \mathcal{Y}^n$ such that for every $x \in \mathcal{X}, y \in \mathcal{Y}$ we have

- The distribution $(f(R, x), g(R, y))$ is ϵ_0 -close to the distribution $(X^n, Y^n | W_S$ and $X_i = x$ and $Y_i = y)$.
- The two distributions *(i)* $(R | X_i = x)$ and *(ii)* $(R | Y_i = y)$, are each ϵ_1 -close to the distribution *(iii)* $(R | X_i = x$ and $Y_i = y)$.
- The marginal distribution $\{X_i Y_i | W_S\}$ is γ -close to the distribution $\{XY\}$.

Devise a protocol that wins the game G with probability at least $1 - \epsilon + 20\gamma - O(\epsilon_0 + \epsilon_1)$.

In the next session we shall combine the two unrealistic cases, that of [Problem 7.2](#) and that of [Problem 7.4](#) to obtain a strategy for winning G with probability greater than $1 - \epsilon$ using only the assumption that [Equation 4](#) does not hold.

After constructing strategies for two unrealistically simple cases of [Lemma 6.4](#) we finally start to prove this lemma without making any additional assumptions. As outlined in the previous session, we shall assume [Equation 4](#) does not hold and construct a pair of strategies that will cause the players to win a single round of G with probability that exceeds $1 - \epsilon$.

8.1 Defining the pair of strategies

The plan is for both players to use shared randomness in order to eventually generate tuples $X^n|W_S$ and $Y^n|W_S$. The way they generate these tuples will seem a bit convoluted and we describe the process next. From here on we assume without loss of generality that we have won the last k games, i.e., that $S = \{n - k + 1, \dots, n\}$ and for notational simplicity let $\bar{S} = \{1, \dots, n - k\} = [n] \setminus S$.

A pair of strategies for the n -wise repeated game induces a distribution over tuples X^n, Y^n, A^n, B^n , obtained by first sampling X^n, Y^n and then sampling A^n, B^n in response (notice the strategies may be randomized). We are going to look at this distribution conditioned on certain events and it is useful to “forget” the fact that A^n is a response to X^n (and B^n a response to Y^n) and view the situation as an arbitrary distribution over tuples X^n, Y^n, A^n, B^n . Let $Q \in (\mathcal{X} \times \mathcal{Y})^S$ denote random queries to the last k rounds (that we shall soon assume have been won) and let $A \in \mathcal{A}^S, B \in \mathcal{B}^S$ denote their respective answers.

Using shared randomness, both players are going to generate one query (either X_i or Y_i) for each $i \in \bar{S}$. This is done by tossing a fair coin $V_i \in \{0, 1\}$ and then generating X_i if $V_i = 0$ and Y_i if $V_i = 1$. For future use in our proof, we shall call the complementary query (the one not generated using shared randomness) by U_i . To sum, for $i \in \bar{S}$ let (T_i, U_i) denote a pair of random queries for the i^{th} coordinate as follows:

$$(T_i, U_i) = \begin{cases} (X_i, Y_i) & V_i = 0 \\ (Y_i, X_i) & V_i = 1 \end{cases} \quad (5)$$

Let $R = (Q, (V_1, T_1), \dots, (V_{n-k}, T_{n-k}))$ and for $j \in \bar{S}$ let R^{-j} be R after removing (V_j, T_j) from it. Notice that R defines for every $i \in S$ both queries and for $i \in \bar{S}$ the information in R gives one (random) query, either X_i (when $T_i = 0$) or Y_i (when $T_i = 1$).

Definition 8.1 (Strategies for winning a single round). Assume the existence of a pair of strategies for G^n . The 1-round strategy of the players on respective inputs x, y and a shared random string is:

First player Use the shared random string to sample R^{-i} and A conditioned on W_S and $X_i = x$. Let r^{-i}, a respectively denote the values of these two random variables. Next, sample X^n conditioned on $X_i = x, R^{-i} = r^{-i}, A = a$ and W_S . (This sampling does not use shared randomness.) Use the strategy for G^n to compute an answer to X^n and reply with the i^{th} coordinate of this answer.

Second player Use the shared random string to sample R^{-i} and A conditioned on W_S and $Y_i = y$. Let r'^{-i}, a' respectively denote the values of these random variables. Sample Y^n conditioned on $Y_i = y, R^{-i} = r'^{-i}, A = a'$ and W_S . (This sampling does not use shared randomness.) Use the strategy for G^n to compute an answer to Y^n and reply with the i^{th} coordinate of this answer.

Problem 8.1. Prove: The distribution X^n conditioned on $((X_i, R^{-i}, A) = (x, r^{-i}, a)$ and W_S) is independent of Y_i . Similarly, the distribution on Y^n conditioned on $((Y_i, R'^{-i}, A') = (y, r'^{-i}, a')$ and W_S) is independent of X_i . Hints: Use Proposition 15 given as part of **Problem 6.3**.

Problem 8.2. Suppose the queries given to the two players are distributed exactly according to X_i, Y_i conditioned on W_S . Then the distribution on X^n (Y^n , respectively) described in **Definition 8.1** is equal to the distribution on X^n (Y^n , respectively) conditioned on W_S .

8.2 Similar distributions

The main claim needed to complete our proof is the following. Informally, it says that when using shared randomness, each player can individually generate AR^{-i} and end up with the same instance of this random variable, with high probability.

Lemma 8.2 (Main technical). *For $i \in \bar{S}$ consider the following three distributions that use the same shared randomness to obtain AR^{-i} :*

- $\{X_i\} \{AR^{-i} \mid X_i \text{ and } W_S\}$
- $\{Y_i\} \{AR^{-i} \mid Y_i \text{ and } W_S\}$
- $\{X_i Y_i\} \{AR^{-i} \mid X_i Y_i \text{ and } W_S\}$

Then, the expectation over $i \in \bar{S}$ of the probability that not all three instances of AR^{-i} are identical, is bounded by 24γ .

Problem 8.3. Complete the proof of **Lemma 6.4**. Hints:

- Use **Lemma 7.1** to argue that in expectation over $i \in \bar{S}$, the distribution $\{X_i Y_i \mid W_S\}$ is γ -close to the distribution XY .
- Use Fact 21 given in **Problem 6.3** and the previous bullet to show that in expectation over $i \in \bar{S}$, the distribution $\{X_i Y_i\} \{AR^{-i} \mid X_i Y_i W_S\}$ is γ -close to $\{X_i Y_i AR^{-i} \mid W_S\}$.

- Invoke [Lemma 8.2](#) to argue that the random variable AR^{-i} generated by both players using shared randomness is likely to be identical.
- Complete the proof using [Problem 8.2](#).

In this lecture, our final on the parallel repetition theorem, we prove the main technical [Lemma 8.2](#) thereby completing the proof of the Parallel Repetition [Theorem 6.3](#). For this we recall our assumption stated as [Equation 3](#) in [Lemma 6.4](#) which says that

$$\Pr[W_S] \geq 2^{-\gamma^2(n-|S|)+|S|c}.$$

Our starting point is that if we sample RA conditional on W_S , then for typical i we can generate U_i — the query missing from R in the i^{th} coordinate — from R . In other words, having sampled R , even conditioning on the answers A to the coordinates in S , the marginal distribution for typical $i \notin S$ is close to the distribution XY .

To show this we need the following corollary of [Lemma 7.1](#). Its proof will be presented in class (it can also be found as Corollary 27 in [Rao \[2008\]](#)).

Lemma 9.1 (Frequent events do not change the expected conditional marginal distribution).

Let R, U_1, \dots, U_n, A be random variables and E an event with $\Pr[E] \geq 2^{-d}$ satisfying

- For every r , the variables U_1, \dots, U_n are independent conditioned on $R = r$.
- The random variable A is supported on at most 2^h values.

Then

$$\mathbf{E}_{i \in [n]} [|\{RA \mid E\} \{U_i \mid R\} - \{RAU_i \mid E\}|] \leq \sqrt{\frac{d+h}{n}}. \quad (6)$$

Problem 9.1. Use [Lemma 9.1](#) to argue that

$$\mathbf{E}_{i \in \bar{S}} [|\{RA \mid W_S\} \{U_i \mid R\} - \{RAU_i \mid W_S\}|] \leq \gamma. \quad (7)$$

Problem 9.2. Use Proposition 13 stated in [Problem 6.3](#) to argue that

$$\mathbf{E}_{i \in \bar{S}} [|\{R^{-i}AX_iY_i \mid W_S\} - \{R^{-i}AY_i \mid W_S\} \{X_i \mid Y_i\}|] \leq 2\gamma. \quad (8)$$

Problem 9.3. Argue, using the problems stated in this session, that

$$\mathbf{E}_{i \in \bar{S}} [|\{X_iY_i\} \{R^{-i}A \mid X_iY_iW_S\} - \{X_iY_i\} \{R^{-i}A \mid Y_iW_S\}|] \leq 6\gamma. \quad (9)$$

And, by symmetry,

$$\mathbf{E}_{i \in \bar{S}} [|\{X_i Y_i\} \{R^{-i} A \mid X_i Y_i W_S\} - \{X_i Y_i\} \{R^{-i} A \mid X_i W_S\}|] \leq 6\gamma. \quad (10)$$

You may use, without proof, the following statement (we essentially proved it in [Problem 8.3](#)).

$$\mathbf{E}_{i \in \bar{S}} [|\{X_i Y_i \mid W_S\} - \{X_i Y_i\}|] \leq \gamma.$$

Finally,

Problem 9.4. Complete the proof of [Lemma 8.2](#) using the previous problem and [Lemma 7.2](#).

For the remainder of the course we will study the deterministic polynomial time primality testing algorithm of [Agrawal, Kayal, and Saxena \[2004\]](#). Intuitively, the deterministic algorithm *derandomizes* a primality testing algorithm suggested by [Agrawal and Biswas \[2003\]](#). We start with the randomized algorithm, but first say a few words on *randomness* and *derandomization*.

Derandomization is a process that converts a randomized algorithm into a deterministic one. A central current quest of Theoretical Computer Science is to understand the power of randomness in computation and in particular show that randomness can be eliminated from various algorithmic tasks without significantly increasing other computational resources such as running time or memory. In this context, the deterministic primality testing algorithm presented in [[Agrawal et al., 2004](#)] is a great success in the history of derandomization. First, primes are fundamental objects and understanding their properties is a great mathematical challenge. Second, primes play a crucial role in many practical areas of computation, e.g., public key cryptography. Finally, one of the first randomized algorithms to be discovered was the primality test of [Rabin \[1980\]](#). This algorithm added randomness to a deterministic algorithm of [Miller \[1976\]](#). Miller's algorithm is deterministic and runs in polynomial time but its correctness relies on an unproven (though widely believed) assumption called the *Extended Riemann Hypothesis*. Primality testing was cited as one of the most important problems for which randomness offers significant savings in computational resources. The deterministic algorithm of [Agrawal et al. \[2004\]](#) shows this is no longer the case.

11.1 The Agrawal-Biswas randomized primality test

Problem 11.1. Prove: $n \in \mathbb{N}^+$ is prime iff n divides $\binom{n}{i}$, denoted $n | \binom{n}{i}$, for all $i \in \{1, \dots, n-1\}$. Conclude n is prime iff for all $a \in \mathbb{Z}_n$,

$$(x+a)^n \equiv x^n + a \pmod{n}. \text{ Equivalently, } \mathcal{P}_{n,a}(x) \triangleq (x+a)^n - x^n - a \equiv 0 \pmod{n}. \quad (11)$$

Problem 11.2. Suppose n is a composite that is not a power of a prime. Prove: For any prime divisor p of n we have $\mathcal{P}_{n,1}(x) \not\equiv 0 \pmod{p}$. Hint: Let b be the largest power such that $p^b | n$. Consider the coefficient of x^{p^b} in $(x+1)^n$ and argue it is not divisible by p .

Testing [Equation 11](#) requires time $\Omega(n)$ which is exponential in the size of our input, since integer n is typically described using $\approx \log n$ bits. To reduce the running time we use randomness. Specifically, we reduce $\mathcal{P}_{n,1}(x)$ modulo a *random* low-degree polynomial and

check the resulting equivalence. This leads to the following algorithm. Recall a polynomial is called *monic* if the coefficient of its highest degree term is 1. We use the notation $f(x) \equiv g(x) \pmod{n, Q(x)}$ to denote $f(x) = g(x)$ in the ring $\mathbb{Z}_n[x]/Q(x)$.

Agarwal-Biswas-Primality(n)

1. If $n = a^b$ for integers a and $b > 1$, output “COMPOSITE”.
2. Else, randomly choose a monic polynomial $Q(x) \in \mathbb{Z}_n[x]$ with $\deg Q(x) \leq \lceil \log n \rceil$.
3. Output “PRIME” iff $\mathcal{P}_{n,1}(x) \equiv 0 \pmod{n, Q(x)}$, else output “COMPOSITE”.

Problem 11.3. Analyze the asymptotic running time and prove it is polynomial in $\log n$. To compute $p^n(x) \pmod{q(x)}$ use repeated squaring. I.e., start with $p^{2^0}(x) = p(x) \pmod{q(x)}$ and then compute $p^{2^{i+1}}(x) \pmod{q(x)}$ assuming $p^{2^i}(x) \pmod{q(x)}$ is known. Use the fact that division of degree k polynomials with coefficients in \mathbb{Z}_n requires polynomial time in k and $\log n$.

Problem 11.4. Prove the algorithm has only one-sided error: If n is prime the algorithm always outputs “PRIME”.

To prove the other direction we rely on the following fact. Recall a polynomial $P(x) \in F[x]$ over a field F is *irreducible* over F if it cannot be written as a product $P(x) = R(x) \cdot T(x)$ with $\deg R < \deg P$. Irreducible polynomials are to polynomials what primes are to integers — every monic polynomial over F can be written in a unique way as a product of monic irreducible polynomials over F .

Proposition 11.1. *The number of monic irreducible polynomials of degree d over the prime field \mathbb{Z}_p is at least $\frac{p^d}{d} - p^{d/2}$.*

Problem 11.5. Prove: If n is a composite that is not a prime power and is not divisible by a prime smaller than 17, then the algorithm outputs “COMPOSITE” with probability at least $1/\text{poly}(\log n)$. Hints:

- Let p be a prime divisor of n . Apply **Problem 11.2** and argue it suffices to prove $\mathcal{P}_{n,1}(x) \not\equiv 0 \pmod{p, Q'(x)}$ where $Q'(x) = Q(x) \pmod{p}$.
- Use **Proposition 11.1** and the assumption $p > 16$ to bound from below the probability p_1 that $Q'(x)$ is irreducible.
- Bound from above the probability p_2 that $Q'(x)$ is monic irreducible and divides $\mathcal{P}_{n,1}(x)$.
- Argue $p_1 - p_2 \geq 1/\text{poly}(\log n)$.

Remark. The analysis in [Agrawal and Biswas \[2003\]](#) proves a stronger statement. Namely, it shows that on input a composite n the algorithm outputs “COMPOSITE” with probability $\geq 2/3$. For details see the proof of Theorem 3.2 in [Agrawal and Biswas \[2003\]](#).

The derandomization of [Agrawal, Kayal, and Saxena \[2004\]](#) replaces the large sample space of polynomials $Q(x)$ with a much smaller space \mathcal{Q} , $|\mathcal{Q}| \leq \text{poly}(\log n)$. The crucial point in the proof is that if $\mathcal{P}_{n,1}(x) \equiv 0 \pmod n, Q(x)$ for all $Q(x) \in \mathcal{Q}$, then n is a prime power and hence would be detected by step 1 of Agarwal-Biswas-Primality.

Today we follow [Agrawal, Kayal, and Saxena \[2004\]](#) and derandomize the primality testing algorithm of [Agrawal and Biswas \[2003\]](#) presented last week. We will complete the derandomization in our next and last session. Our exposition follows closely the description given in a [Agrawal's lecture notes](#):

<http://www.cse.iitk.ac.in/~manindra/CS681/> .

The deterministic primality testing algorithm follows from the main Theorem of [Agrawal et al. \[2004\]](#), stated below. In what follows, the *order* of integer n modulo integer r , denoted $o_r(n)$, is the minimal integer k such that $n^k \equiv 1 \pmod{r}$.

Theorem 12.1. *Suppose $n, r \in \mathbb{N}^+$ satisfy $o_r(n) > 4 \log^2 n$ and n has no divisors that are less than or equal to r . If*

$$(x + a)^n \equiv x^n + a \pmod{n, x^r - 1}, \quad \forall a \in \{0, 1, \dots, \lceil 4\sqrt{r} \log n \rceil\}, \quad (12)$$

Then n is a prime power, i.e., $n = p^b$ for prime p and integer b .

We omit the proof of the following technical claim which is proved as Lemma 4.3 in [Agrawal et al. \[2004\]](#).

Proposition 12.2. *There exists integer $r \leq 16 \log^5 n$ such that $o_r(n) > 4 \log^2 n$.*

Problem 12.1. Use [Theorem 12.1](#), [Proposition 12.2](#) and [Problem 11.3](#) to construct a deterministic polynomial time primality testing algorithm. What is the exponent of the polynomial bounding the running time?

12.1 Overview of the proof of [Theorem 12.1](#)

From here on we assume (i) n is composite, (ii) $r \leq 16 \log^5 n$, (iii) $o_r(n) > 4 \log^2 n$ and (iv) $p > r$ is a prime divisor of n . Intuitively, the proof is composed of the following steps.

1. Construct a finite field F of characteristic p and a low-degree polynomial $P(Y)$.
2. [Equation 12](#) implies $P(Y)$ has many roots in F .
3. The existence of many roots for $P(Y)$ implies n is a prime power. QED.

Details follow. We start with the first part, continue with the third and conclude with the second and main part of the proof.

12.2 Part 1 — The field F and polynomial $P(Y)$

Recall p is a prime that divides n . Let F_p be the prime field with p elements. Let $F_p[x]$ be the ring of polynomials with coefficients in F_p . The following proposition shows the existence of a “good” finite field F . Intuitively, a field is “good” if the set of elements $\{x^{n^i} \mid i \geq 0\}$ is sufficiently large. A proof of the following proposition can be found in [Lidl and Niederreiter, 1997, Chapter 2].

Proposition 12.3. *The polynomial $x^r - 1$ has a factor $h(x) \in F_p[x]$ with the following properties:*

- $h(x)$ is irreducible over $F_p[x]$ of degree d and $1 < d < r$.
- The ring $F_p[x]/h(x)$ is a finite field, denoted henceforth as F . Elements of this field are polynomials of degree $< d$ in variable x with coefficients in F_p .
- $x \in F$ is an r^{th} root of unity, i.e., r is the smallest positive integer such that $x^r =_F 1$, where $=_F$ denotes equality in F . This implies $x^1, x^2, \dots, x^r =_F 1$ are distinct elements of F .
- For any $j \geq 0$ and $f(x) \in F$ we have $(f(x))^{p^j} =_F f(x^{p^j})$.

Remark. The last part of the previous proposition follows by induction on j from **Problem 11.1** and Fermat’s Little Theorem: $a^p \equiv a \pmod{p}, \forall a \in \mathbb{Z}_p$.

We stress that the primality testing algorithm does not need to compute p , $h(x)$ or F . We merely use their existence and properties in our analysis. Next, let us argue that F is indeed “good”.

Problem 12.2. Prove: If $\text{o}_r(n) = k$, then $x^n, x^{n^2}, \dots, x^{n^k}$ are distinct elements of F .

Problem 12.3. Let

$$T = \{x^{n^i p^j} \mid i, j \geq 0\} \subseteq F; \quad t = |T|.$$

Prove:

- $t \geq 4 \log^2 n$. Hint: Use the previous problem.
- There exist $0 \leq i, i', j, j' \leq \sqrt{t}$ such that $(i, j) \neq (i', j')$ and $x^{n^i p^j} = x^{n^{i'} p^{j'}}$ over F . Hint: Pigeonhole principle.

Problem 12.4. Let i, i', j, j' be as defined in the previous problem. Let $P(Y) \in F[Y]$ be defined by

$$P(Y) = Y^{n^i p^j} - Y^{n^{i'} p^{j'}}.$$

Prove: $\deg P(Y) \leq n^{2\sqrt{t}}$.

12.3 Part 3 — Its all about the roots of $P(Y)$

Theorem 12.1 follows by showing $P(Y)$ has many roots.

Problem 12.5. Prove: If $P(Y)$ has more than $n^{2\sqrt{t}}$ distinct roots in F , then n is a prime power.

Today we complete the proof of [Theorem 12.1](#). Given [Problem 12.5](#), all we need to do is show that $P(Y)$ defined in [Problem 12.4](#) has more than $n^{2\sqrt{t}}$ roots in F . We start by getting a few roots from [Equation 12](#).

13.1 A small set of roots for $P(Y)$

Problem 13.1. Prove: For all $i > 0$,

$$((x+a)^n \equiv x^n + a \pmod{n, x^r - 1}) \implies \left((x^i + a)^n \equiv x^{ni} + a \pmod{n, x^r - 1} \right).$$

Hint: substitute $y = x^i$ and use the fact that $(x^r - 1)$ divides $x^{ri} - 1$ for all $i > 1$.

Problem 13.2. Using the previous problem, conclude

$$((x+a)^n \equiv x^n + a \pmod{n, x^r - 1}) \implies \left((x+a)^{n^i} \equiv x^{n^i} + a \pmod{n, x^r - 1} \right).$$

Problem 13.3. Prove:

$$(f(x) \equiv g(x) \pmod{n, x^r - 1}) \implies \hat{f}(x) =_F \hat{g}(x),$$

where $\hat{f}(x) \in F$ is $f(x) \pmod{p, h(x)}$ and $\hat{g}(x) \in F$ is analogously defined.

Problem 13.4. Let

$$S_0 = \{x + a \mid a \in \{0, 1, \dots, \lceil 4\sqrt{r} \log n \rceil\}\}.$$

Prove: The elements of S_0 are roots of $P(Y)$. Hint: Use [Problem 13.2](#), [Problem 13.3](#) and the last part of [Proposition 12.3](#).

13.2 A large set of roots for $P(Y)$

Recall the definition of t from [Problem 12.3](#). Consider the set

$$S = \left\{ \prod_{i=1}^{\sqrt{t} \log n} (x + a_i) \mid a_i \in \{0, 1, \dots, \lceil 4\sqrt{r} \log n \rceil\} \text{ and } a_i \neq a_{i'}, \forall i \neq i' \right\}.$$

We are going to prove that $S \subseteq F$ is large and all its elements are roots of $P(Y)$. This will complete the proof of [Theorem 12.1](#).

13.2.1 S is large

Problem 13.5. Consider the following set $S' \subset F[Y]$, where $F[Y]$ is the ring of polynomials in variable Y with coefficients from the field F :

$$S' = \left\{ \prod_{i=1}^{\sqrt{t} \log n} (Y + a_i) \mid a_i \in \{0, 1, \dots, [4\sqrt{r} \log n]\} \text{ and } a_i \neq a_{i'}, \forall i \neq i' \right\}$$

We stress that Y is a formal variable, not an element of F . Prove: $|S'| > n^{2\sqrt{t}}$. Hint: Use the inequality $\binom{n}{k} > (n/k)^k$ for $n > k > 1$

Problem 13.6. Let $q_1(Y), q_2(Y)$ be distinct elements of S' . Prove $q_1(x) \neq_F q_2(x)$. Hints:

- Bound the degree of $q(Y) = q_1(Y) - q_2(Y)$. We will show the number of distinct roots of $q(Y)$ in F is greater than its degree, hence $q(x) =_F 0$.
- Consider the pair of maps $\psi, \phi : F \rightarrow F$ given by

$$\psi(f(x)) = (f(x))^n \text{ and } \phi(f(x)) = (f(x))^p.$$

Prove $\phi(f(x)) = f(\phi(x))$ for all $f(x) \in F$ and $\psi(f(x)) = f(\psi(x))$ for all $f(x) \in S$.

- Let $\psi^1(f(x)) = \psi(f(x))$ and for $i > 1$ let $\psi^i(f(x)) = \psi(\psi^{i-1}(f(x)))$. Let ϕ^j be similarly defined. Prove: $\psi^i \phi^j(f(x)) = f(\psi^i \phi^j(x))$ for all $f(x) \in S$. Hint: Use [Problem 13.2](#).
- Using the bound on $\deg q(y)$ and the definition of T and t from [Problem 12.3](#), prove $q(y)$ is identically zero.

Problem 13.7. Conclude $|S| > n^{2\sqrt{t}}$.

13.2.2 Elements of S are roots of $P(Y)$

Problem 13.8. Prove: All elements of S are roots of $P(Y)$. Hint: Use the reasoning applied in [Problem 13.6](#).

Problem 13.9. Give a formal proof of [Theorem 12.1](#).

References

1. Manindra Agrawal and Somenath Biswas. Primality and identity testing via chinese remaindering. *J. ACM*, 50(4):429–443, 2003. URL <http://doi.acm.org/10.1145/792538.792540>.
2. Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004.
3. Mikhail Alekhnovich and Eli Ben-Sasson. Linear upper bounds for random walk on small density random 3-CNFs. *SIAM J. Comput.*, 36(5):1248–1263, 2006. ISSN 0097-5397.
4. Noga Alon and Joel Spencer. *The Probabilistic Method*. John Wiley, 1992. ISBN 0-471-53588-5.
5. Thomas M. Cover and Joy Thomas. *Elements of Information Theory*. Wiley, 1991.
6. P. Erdos and L. Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. *Infinite and finite sets*, 2:609–627, 1975.
7. Uriel Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45(4):634–652, July 1998.
8. Uriel Feige and László Lovász. Two-prover one-round proof systems: their power and their problems (extended abstract). In *STOC '92: Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 733–744, New York, NY, USA, 1992. ACM. ISBN 0-89791-511-9.
9. Venkatesan Guruswami and Ryan O’Donnell. The pcp theorem and hardness of approximation. Lecture notes, 2005.
10. Thomas Holenstein. Parallel repetition: Simplification and the no-signaling case. *Theory of Computing*, 5(1):141–172, 2009.
11. Rudolf Lidl and Harald Niederreiter. *Finite fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, second edition, 1997. ISBN 0-521-39231-4.
12. Michael Luby, Michael Mitzenmacher, and Mohammad Amin Shokrollahi. Analysis of random processes via and-or tree evaluation. In *SODA*, pages 364–373, 1998.
13. Gary L. Miller. Riemann’s hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13(3):300–317, December 1976.
14. Robin A. Moser. A constructive proof of the lovász local lemma. In Michael Mitzenmacher, editor, *STOC*, pages 343–350. ACM, 2009. ISBN 978-1-60558-506-2.
15. Robin A. Moser and Gábor Tardos. A constructive proof of the general lovász local lemma. *J. ACM*, 57(2), 2010.
16. Christos H. Papadimitriou. On selecting a satisfying truth assignment (extended abstract). In *SFCS '91: Proceedings of the 32nd annual symposium on Foundations of computer science*, pages 163–169, Washington, DC, USA, 1991. IEEE Computer Society. ISBN 0-8186-2445-0.
17. Michael Rabin. Probabilistic algorithm for testing primality. *J. Number Theory*, 12:128–138, 1980.
18. Anup Rao. Parallel repetition in projection games and a concentration bound. In Richard E. Ladner and Cynthia Dwork, editors, *STOC*, pages 1–10. ACM, 2008. ISBN 978-1-60558-047-0.

19. Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, June 1998.
20. H. W. Watson and Francis Galton. On the probability of the extinction of families. *Journal of the Anthropological Institute of Great Britain*, 4:138–144, 1875. URL <http://galton.org/essays/1870-1879/galton-1874-jaigi-family-extinction.pdf>.