Luigi Russo

FFPP 2021

Foundations and Frontiers of Probabilistic Proofs · Worksheets

FFPP 2021

Worksheets and Solutions by L. Russo

Contents

Preface

This document contains some of my solutions to homework proposed during the 2021 Summer School on *Foundations and Frontiers of Probabilistic Proofs* co-organized by MSRI, ETH Zurich, UC Berkeley.

I am always happy to receive feedback. Also, no document is error-free: if you do find any, I would greatly appreciate it if you let me know; you can open an issue on GitHub for this.

Luigi Russo
Rome (IT)

This document was last updated on August 7, 2021 at 16:30.

Introduction

Proofs are at the foundations of mathematics. Viewed through the lens of theoretical computer science, verifying the correctness of a mathematical proof is a fundamental computational task. Indeed, the P versus NP problem, which deals precisely with the complexity of proof verification, is one of the most important open problems in all of mathematics.

The complexity-theoretic study of proof verification has led to exciting reenvisionings of mathematical proofs. For example, probabilistically checkable proofs (PCPs) admit local-to-global structure that allows verifying a proof by reading only a minuscule portion of it. As another example, interactive proofs allow for verification via a conversation between a prover and a verifier, instead of the traditional static sequence of logical statements. The study of such proof systems has drawn upon deep mathematical tools to derive numerous applications to the theory of computation and beyond.

In recent years, such probabilistic proofs received much attention due to a new motivation, delegation of computation, which is the emphasis of this summer school. This paradigm admits ultra-fast protocols that allow one party to check the correctness of the computation performed by another, untrusted, party. These protocols have even been realized within recently-deployed technology, for example, as part of cryptographic constructions known as succinct non-interactive arguments of knowledge (SNARKs).

This summer school has provided an introduction to the field of probabilistic proofs and the beautiful mathematics behind it. Two complementary courses have been offered:

A. a foundations course, covering the "classics" of probabilistic proofs. The material includes seminal results that have found a diverse set of applications in theoretical computer science.

B. a frontiers course, covering contemporary results in probabilistic proofs. The focus of this course is on proof protocols for delegating computations.

For the first course on Interactive Proofs, the lectures have been taken by prof. Alessandro Chiesa and are available at https://www.youtube.com/playlist?list=PLGkwtcB-DfpyjJfxPUdwWpg_ygk2OIp9-, while the worksheets have been discussed with prof. Nick Spooner.

The topics covered have been: Sumcheck Protocol, Interactive Proofs (IP) for PSPACE, Doubly-Efficient IPs, Zero-Knowledge IPs, Interactive Oracle Proofs (IOP), Linear-Size IOPs for Circuits and Machines, and the limitations of both IPs and IOPs

For the second course on Probabilistic Checkable Proofs, instead, the lectures have been taken by prof. Tom Gur and are available at \dot , while the worksheets have been discussed with Marcel Dall'Agnol.

The topics covered have been: ...

A Interactive Proofs

A0.1 (Restriction to a line). A *line* in $\mathbb{F}_n$ is a function $g : \mathbb{F} \to \mathbb{F}_n$ of the form $g(z) = (a_1 z + b_1, ..., a_n z + b_n)$ for some choice of coefficients $a_1, b_1, ..., a_n, b_n \in \mathbb{F}$. The *restriction* of an n-variate polynomial $f(x_1, ..., x_n)$ over $\mathbb{F}$ to the line g is defined as the univariate polynomial $h(z) := f(g(z))$. Prove that for every line g, the degree of h is at most the total degree of f.

$h(z) = f(g(z)) = f(a_1 z + b_1, ..., a_n z + b_n) = f(z'_1, ..., z'_n)$, where $z'_i = a_i z + b_i$. Since $\deg(z)$ is at least $\deg(z')$, the total degree of h is at most the total degree of f.

Next we prove the Schwartz–Zippel Lemma: for every non-zero n-variate polynomial f of total degree at most d over a field $\mathbb{F}$ and every finite set S in $\mathbb{F}$, $\Pr_{a_1,...,a_n \leftarrow_\$ S}[f(a_1, ..., a_n) = 0] \leq \frac{d}{|S|}$. This fundamental lemma is used numerous times in this course.

A0.2 (Zeroes of univariate polynomials). Let $\mathbb{F}$ be a field and f a non-zero univariate polynomial over $\mathbb{F}$ of degree at most d. Prove that f has at most d roots in $\mathbb{F}$ (you may use without proof the fact that $\mathbb{F}[X]$ is a Euclidean domain). (In particular, for every finite set $S \subseteq \mathbb{F}, \Pr_{a \leftarrow_\$ S}[f(a) = 0] \leq \frac{d}{|S|}$.) Give an example of a finite field F and polynomial $f \in \mathbb{F}[X]$ that has strictly fewer than $\deg(f)$ roots in $\mathbb{F}$.

Assume f is a non-zero polynomial of degree at most d that has $n \geq d+1$ roots. Then it must be the case we can rewrite f as $\alpha_0(x - \alpha_1)(x - \alpha_2)...(x - \alpha_n) = \alpha_0 \prod_{i=1}^{n}(x - \alpha_i)$, where $\alpha_i \in \mathbb{F}$ and $\alpha_0 \neq 0$. But this implies that f has degree $n > d$ because of the term $\alpha_0 x^n$. A polynomial of degree d has at most d distinct roots. It implies that for every finite set $S \subseteq \mathbb{F}$, $\Pr_{a \leftarrow_\$ S}[f(a) = 0] \leq \frac{deg(f)}{|S|} \leq \frac{d}{|S|}$.)

A0.3 (Zeroes of multivariate polynomials). Let $\mathbb{F}$ be a field and f a non-zero n-variate polynomial over $\mathbb{F}$ of degree at most d. Prove that, for every finite set S in $\mathbb{F}$, f has at most $d|S|^{n-1}$ roots in S^n. (Hint: rely on the prior problem, and use induction.) Conclude from this the Schwartz–Zippel Lemma.

Proof strategy: prove by induction.

A1.1 (Importance of randomness). Prove that if a language $\mathcal{L}$ has an interactive proof with a deterministic verifier, then $\mathcal{L} \in$ NP.

First, note that when the verifier is deterministic we can only consider proofs with perfect soundness, i.e. $\forall x \notin \mathcal{L}$, the verifier rejects the proof. An NP proof for $x \in \mathcal{L}$ is just a transcript $(\alpha_1, \alpha_2, ..., \alpha_n)$ of the interaction Prover-Verifier on input x. Indeed, the verifier of NP can simply check that the transcript is valid and makes the verifier of IP to accept, namely: $V(x, \alpha_1) = \alpha_2$, and $V(x, \alpha_1, \alpha_2, \alpha_3) = \alpha_3$, ..., and finally $V(\alpha_1, ..., \alpha_n) = 1$.

A1.2 (Sequential repetition). Suppose that L has an interactive proof (P, V) with perfect completeness and soundness error $1/2$. Let (P_t, V_t) be the t-wise sequential repetition of (P, V): the new prover P_t and the new verifier V_t respectively simulate the old prover P and old verifier V for t times one after the other, each time with fresh randomness; V_t accepts if and only if V accepts in all t repetitions. Prove that (P_t, V_t) is an interactive proof for L with perfect completeness and soundness error 2^{-t}.

Each execution is independent. It means that whenever $x \in \mathcal{L}$, the honest prover can convince with probability 1 the verifier at each round. If $x \notin \mathcal{L}$, the malicious prover can convince the verifier to accept with probability at most $1/2$ in each round, but since these runs are independent, the overall probability to convince the verifier is at most the probability to successfully cheat for t consecutive times, i.e. 2^{-t}.

A1.3 (Invertible matrices). Let $\mathbb{F}$ be a finite field. Show that the language

$$\text{INV}_{\mathbb{F}} := \{M \in \mathbb{F}^{n \times n} : \exists A \in F^{n \times n} \text{ s.t. } M A = I\}$$

has an interactive proof with perfect completeness, soundness error $1/2$, and $O(n)$ total communication, where the verifier runs in time $O(n^2)$. (Assume that sampling field elements and performing basic field operations have unit cost.)

The verifier picks a random vector $x \in \mathbb{F}^n$ and computes in $O(n^2)$ steps the product $y = Mx$. Then it sends y to the prover. The prover computes $x' = M^{-1}y$ and sends it to the verifier that checks whether $x = x'$. Completeness is straigthforward since for every $M \in \mathcal{L}$, it always holds that the honestly computed x' is equal to x. As for the soundness, when the matrix M is not invertible, it means there are at least two vectors x_1, x_2 such that $Mx_i = y$.

An alternative approach is the following. The verifier picks a random vector $y \in \mathbb{F}^n$ and sends it to the prover. The prover computes $x = M^{-1}y$ and sends this vector to the verifier that accepts if $Mx = y$. We have seen that this latter protocol is public-coin, but is not zero-knowledge, while the former is private-coin and zero-knowledge.

A2.1 (Sumcheck with tensor weights). We consider an extension of the sumcheck problem where the summand is multiplied by weights that have a product structure. Specifically, given $n \cdot |H|$ field elements $\{\delta_{i,\alpha} \in \mathbb{F}\}_{i\in[n],\alpha\in H}$, we consider statements of the following form:

$$\sum_{\alpha_1,...,\alpha_n\in H} \delta_{1,\alpha_1} \cdots \delta_{n,\alpha_n} \cdot p(\alpha_1, ..., \alpha_n) = \gamma.$$

Show that the sumcheck protocol can be extended to support the above statement, with the same completeness and soundness guarantees.

Slighty modify the sumcheck protocol to "inject" the term δ_{i,α_i}.

A2.2 (Efficient multilinear extension). The multilinear extension of a boolean function $f : \{0,1\}^n \to \{0,1\}$ over a field $\mathbb{F}$ is the unique multilinear polynomial $\mathrm{MLE}_{\mathbb{F}}(f) \in \mathbb{F}[X_1, ..., X_n]$ that agrees with f on $\{0,1\}^n$:

$$\mathrm{MLE}_{\mathbb{F}}(f)(X_1, ..., X_n) := \sum_{b_1,...,b_n\in\{0,1\}} f(b_1, ..., b_n) \prod_{i\in[n]\ b_i=1} X_i \prod_{i\in[n]\ b_i=0} (1 - X_i).$$

Prove that evaluating a multilinear extension at a single point is in linear time. Namely, give an algorithm that given a boolean function f (represented as a string of $2n$ bits), finite field $\mathbb{F}$, and evaluation point $(\alpha_1, ..., \alpha_n) \in \mathbb{F}^n$, computes the evaluation of $\mathrm{MLE}_{\mathbb{F}}(f)$ at $(\alpha_1, ..., \alpha_n)$ in $O(2^n)$ field operations.
Hint: The multilinear extension can be evaluated by summing term by term in $O(n \cdot 2^n)$ field operations while maintaining a state of $O(1)$ field elements in memory. How can you use more memory to speed up the computation?

Precompute the exponentially large table of the products (use dynamic programming). Then, compute the summation.

A2.3 (Efficient sumcheck). We analyze the running time of the honest prover in the sumcheck protocol, when proving statements of the form $\sum_{\alpha_1,...,\alpha_n \in H} p(\alpha_1, ..., \alpha_n) = \gamma$.

- Prove that the honest prover can be realized in $O(d \cdot |H|^n \cdot |p|$ operations, where d is the individual degree of p and $|p|$ is the number of operations to evaluate p at any point in $\mathbb{F}^n$.
- Consider the special case where $H = \{0, 1\}$ and p is multilinear. Prove that, if p is specified via its evaluations on $\{0, 1\}^n$, the honest prover can be realized in $O(2^n)$ field operations.

Add proof.

We work out Shamir's original proof that IP = PSPACE, which relies on fully quantified boolean formulas with a special structure. We say that a fully quantified boolean formula is simple if every occurrence of every variable is separated from its quantification point by at most one universal quantifier ($\forall$) and arbitrarily many other symbols. We denote by TSQBF language obtained by considering only fully quantified boolean formulas that are simple.

A3.1 (Warmup on TSQBF). Which of the following fully quantified boolean formulas are simple? What is their value?

Add exercise and its solution.

A3.2 ....

Add exercise and its solution.

A3.3 (IP for TSQBF). Outline an interactive proof for TSQBF. Hint: show that simple formulas have "nice" arithmetizations.

Add solution.

A4.1 (Layered circuits). Prove that any arithmetic circuit of depth d and size S can be transformed into a layered arithmetic circuit of depth d and size $O(S^2)$.

Add solution.

A4.2 (GKR for any set of gates). The GKR protocol that we saw in class applies to layered arithmetic circuits, which by default involve two gates: addition gates and multiplication gates. Now suppose that we instead consider layered circuits where gates are selected from a gate set of bivariate polynomials $\{g_k(X, Y)\}_k$. (The special case of addition and multiplication thus corresponds to the gate set $\{g_1(X, Y) = X + Y, g_2(X, Y) = X \cdot Y\}$.) How would you modify the GKR protocol to support the evaluation of such circuits?

Add solution.

A4.3 (GKR for formulas).

Add exercise and its solution.

B Probabilistic Checkable Proofs

B0.1 (Basics of linear codes).

Add exercise and its solution.

B0.2 (Identity testing). Fix an arbitrary string $s \in \Sigma^n$ and $\epsilon = \epsilon(n) \in \{0, 1\}$. Show that the query complexity of detecting whether an unknown string $x \in \Sigma^n$ is equal to s or differs from s in at least an ϵ fraction of locations is $\Theta(1/\epsilon)$. That is:

- Construct an algorithm that makes $O(1/\epsilon)$ queries to x, and always accepts if $x = s$ and rejects with probability at least $2/3$ if x is ϵ-far from s.

Consider the following algorithm: first we sample a coordinate $i \in [n]$ uniformly at random, then we query x_i and we reject if $x_i \neq s_i$. If $x = s$ this algorithm never rejects. If x is ϵ-far from s, instead, it rejects with probability at least ϵ, by definition. If we repeat t times the previous procedure, the probability to accept when x is ϵ-far from s is at most $(1-\epsilon)^t \leq e^{-t\epsilon}$. For $t := \frac{2}{\epsilon}$, this probability is at most $e^{-2} \leq 1/3$.

- Argue that no algorithm making $o(1/\epsilon)$ queries satisfies both conditions.

Add the solution of the second case.

B0.3 (Hadamard code). The code $\text{Had} : \mathbb{F}^k \to \mathbb{F}^{|\mathbb{F}|^k}$ is defined as $\text{Had}(x) := (\langle x, y\rangle)_{y \in \mathbb{F}^k}$ (i.e., the encoding of x is the linear function $\text{Had(x)} : \mathbb{F}^k \to \mathbb{F}$ where $\text{Had}(x)(y) = \langle x, y\rangle$). Show that Had has relative distance $1 - 1/|\mathbb{F}|$. (Despite its exponential block length, this code has important features that will be useful in this course: local testability and local decodability.)

Let $x_1, x_2 \in \mathbb{F}^k$. $\langle x_1, y\rangle = \langle x_2, y\rangle$ that can be rewritten as $\langle x_1 - x_2, y\rangle = 0$ is a linear equation on k variables that is satisfied exactly in the subspace $y^\perp$. The fraction of elements of $\mathbb{F}^k$ that satisy it is therefore: $\frac{|y^\perp|}{|\mathbb{F}|^k} = \frac{|\mathbb{F}|^{k-1}}{|\mathbb{F}|^k} = \frac{1}{|\mathbb{F}|}$.

B1.0 (From many to 2 queries). Prove that $\mathcal{L}$ has a PCP with perfect completeness, soundness error $1 - \frac{1-\epsilon}{q}$, alphabet Σ^q, proof length $l + 2^r$, and query complexity 2. (In other words, one can always reduce query complexity to 2, incurring a loss in soundness error and alphabet size.)

Add solution.

B1.2 (Lower bound on soundness error). Suppose that there exists $x \notin \mathcal{L}$ such that for every choice of verifier randomness $\rho \in \{0,1\}^r$ there exists a proof $\pi \in \Sigma^l$ such that $V^\pi(x;\rho) = 1$. Prove that $\epsilon \geq 2^{-q \log |\Sigma|}$.

Add solution.

B1.3 (More on lower bounds). The Exponential Time Hypothesis (ETH) states that 3SAT cannot be decided by any deterministic algorithm running in time $2^{o(n)}$. Prove that, assuming ETH, if $\mathcal{L} = 3\text{SAT}$ and $r + q \log |\Sigma| = o(n)$, then $\epsilon \geq 2^{-q \log |\Sigma|}$. (Hint: prove that ETH implies the assumption to the prior problem.)

Add solution.

B2.1 (Affine function testing). A function $f : \mathbb{F}^n \to \mathbb{F}$ is affine if there exists a vector $a \in \mathbb{F}^n$ and constant $\beta \in \mathbb{F}$ such that $f(x) = \sum_{i \in [n]} a_i x_i + \beta$. Design and analyze a 4-query test for the set of affine functions. Hint: reduce the problem to linearity testing, and rely on the BLR test for linear functions.

Add solution.