Thaler Book Exercises

CHAPTER 3 and 4

1 Introduction

This document presents a collection of exercises from Chapters 3 and 4 of Thaler's book. The exercises focus on various topics in polynomial functions, interactive proofs, and their applications in theoretical computer science. The reader is encouraged to explore these problems to deepen their understanding of the material. This is a temporary version, and further revisions may follow.

2 Exercises

2.1 Chapter 3 Exercises

Exercise 3.1. Let A, B, C be $n \times n$ matrices over a field $\mathbb{F}$. In Section 2.2, we presented a randomized algorithm for checking that $C = A \cdot B$. The algorithm picks a random field element r, lets $x = (r, r^2, \ldots, r^n)$, and outputs EQUAL if $Cx = A \cdot (Bx)$, and NOT-EQUAL otherwise. Suppose instead that each entry of the vector x is chosen independently and uniformly at random from $\mathbb{F}$. Show that:

- If $C_{ij} = (AB)_{ij}$ for all $i = 1, \ldots, n$, $j = 1, \ldots, n$, then the algorithm outputs EQUAL for every possible choice of x.
- If there is even one $(i, j) \in [n] \times [n]$ such that $C_{ij} \neq (AB)_{ij}$, then the algorithm outputs NOT-EQUAL with probability at least $1 - \frac{1}{|\mathbb{F}|}$.

Exercise 3.2. In Section 2.1, we described a communication protocol of logarithmic cost for determining whether Alice's and Bob's input vectors are equal. Specifically, Alice and Bob interpreted their inputs as degree-n univariate polynomials p_a and p_b, chose a random $r \in \mathbb{F}$ with $|\mathbb{F}| \gg n$, and compared $p_a(r)$ to $p_b(r)$. Give a different communication protocol in which Alice and Bob interpret their inputs as multilinear rather than univariate polynomials over $\mathbb{F}$. How large should $\mathbb{F}$ be to ensure that the probability Bob outputs the wrong answer is at most $\frac{1}{n}$? What is the communication cost in bits of this protocol?

Exercise 3.3. Let $p = 11$. Consider the function $f : \{0,1\}^2 \to \mathbb{F}_p$ given by $f(0,0) = 3, f(0,1) = 4, f(1,0) = 1$, and $f(1,1) = 2$. Write out an explicit expression for the multilinear extension $\tilde{f}$ of f. What is $\tilde{f}(2,4)$?

Now consider the function $f : \{0,1\}^3 \to \mathbb{F}_p$ given by $f(0,0,0) = 1, f(0,1,0) = 2, f(1,0,0) = 3, f(1,1,0) = 4, f(0,0,1) = 5, f(0,1,1) = 6, f(1,0,1) = 7$, and $f(1,1,1) = 8$. What is $\tilde{f}(2,4,6)$? How many field multiplications did you perform during the calculation? Can you work through a calculation of $\tilde{f}(2,4,6)$ that uses "just" 20 multiplication operations? Hint: see Lemma 3.8.

Exercise 3.4. Fix some prime p of your choosing. Write a Python program that takes as input an array of length 2^ℓ specifying all evaluations of a function $f : \{0,1\}^\ell \to \mathbb{F}_p$ and a vector $\mathbf{r} \in \mathbb{F}_p^\ell$, and outputs $\tilde{f}(\mathbf{r})$.

2.2 Chapter 4 Exercises

Exercise 4.1. Recall that Section 4.3 gave a doubly-efficient interactive proof for counting triangles. Given as input the adjacency matrix A of a graph on n vertices, the IP views A as a function over domain $\{0,1\}^{\log_2 n} \times \{0,1\}^{\log_2 n}$, lets $\widetilde{A}$ denote its multilinear extension, and applies the sum-check protocol to the $(3 \log n)$-variate polynomial

$$g(X, Y, Z) = \widetilde{A}(X, Y) \cdot \widetilde{A}(Y, Z) \cdot \widetilde{A}(X, Z)$$

A 4-cycle in a graph is a quadruple of vertices (a, b, c, d) such that $(a, b), (b, c), (c, d)$, and (a, d) are all edges in the graph. Give a doubly-efficient interactive proof that, given as input the adjacency matrix A of a simple graph, counts the number of 4-cycles in the graph.

Exercise 4.2. Here is yet another interactive proof for counting triangles given as input the adjacency matrix A of a graph on n vertices: For a sufficiently large prime p, define $f : \{0,1\}^{\log_2 n} \times \{0,1\}^{\log_2 n} \times \{0,1\}^{\log_2 n} \to \mathbb{F}_p$ via $f(i,j,k) = A_{i,j} \cdot A_{j,k} \cdot A_{k,i}$, where here we associate vectors in $\{0,1\}^{\log_2 n}$ with numbers in $\{1, \ldots, n\}$ in the natural way, and interpret entries of A as elements of $\mathbb{F}_p$ in the natural way. Apply the sum-check protocol to the multilinear extension $\widetilde{f}$. Explain that the protocol is complete and has soundness error at most $\frac{3 \log_2 n}{p}$.

What are the fastest runtimes you can give for the prover and verifier in this protocol? Do you think the verifier would be interested in using this protocol?

Exercise 4.3. This question has 5 parts.

(a) Section 4.2 gave a technique to take any Boolean formula $\phi : \{0,1\}^n \to \{0,1\}$ of size S and turn ϕ into a polynomial g over field $\mathbb{F}$ that extends ϕ. Apply this technique to the Boolean formula in Figure 4.16. You may specify the resulting extension polynomial g by drawing the arithmetic circuit computing g or by writing out some other representation of g.

(b) Section 4.2 gives an interactive proof for counting the number of satisfying assignments to ϕ by applying the sum-check protocol to g. For the polynomial g you derived in Part (a) that extends the formula in Figure 4.16, provide the messages sent by the honest prover if the random field element chosen by the verifier in round 1 is $r_1 = 3$ and the random field element chosen by the verifier in round 2 is $r_2 = 4$. You may work over the field $\mathbb{F}_{11}$ of integers modulo 11.

(c) Imagine you are a cheating prover in the protocol of Part (b) above and somehow you know at the start of the protocol that in round 1 the random field element r_1 chosen by the verifier will be 3. Give a sequence of messages that you can send that will convince the verifier that the number of satisfying assignments of ϕ is 6 (the verifier should be convinced regardless of the random field elements r_2 and r_3 that will be chosen by the verifier in rounds 2 and 3).

(d) You may notice that the extension polynomial g derived in Part (a) is not multilinear. This problem explains that there is a good reason for this. Show that the ability to evaluate the multilinear extension $\tilde{\phi}$ of a formula ϕ at a randomly chosen point in $\mathbb{F}^n$ allows one to determine whether or not ϕ is satisfiable.

(e) Propose a scheme where the verifier would want to "check" that the prover's answer is accurate after receiving a few messages from the prover. The scheme should be a (potentially) logarithmic time solution to checking the answer against the value of the formula ϕ.