HOMEWORK 4 MATHS 141 WINTER 2018

1. Introduction

This problem set finishes our study of modular arithmetic and begins our study of asymptotic analysis.

2. Problems

Problem 1. Solve the following system of equations:

$$\begin{aligned} x &\equiv 2 \mod 5 \\ x &\equiv 3 \mod 11 \\ x &\equiv 4 \mod 13 \\ x &\equiv 7 \mod 23 \end{aligned}$$

Hint: It may be easier to work with the larger numbers first.

Problem 2. Suppose we can solve a discrete logarithm problem by:

$$a^t \equiv 1 \mod n$$

Now suppose t is even, $t = 2k$.

(a) Show that n is composite and the numbers

$$\gcd(a^k - 1, n) \text{ and } \gcd(a^k + 1, n)$$

are factors of n. That is:

$$\gcd(a^k \pm 1, n) | n.$$

The problem we encounter here is that we may have

$$\gcd(a^k \pm 1, n) = 1$$

We know this happens in the case where n is prime and we have $t = n - 1$ by Fermat's little theorem.

(b) Can you think of a criterion that prevents this?

(c) Factor 35 by showing $2^{24} \equiv 1 \mod 35$ and computing the relevant gcd. Spoiler alert: You should get 5 and 7.

Problem 3. Let $\alpha > 0$ be a positive real number. Use the triangle inequality to show

$$\sum_{j=1}^{n} j^{\alpha} \in O(n^{\alpha+1})$$

Problem 4. Let $k > 0$ be a positive integer. Prove
(a)

$$(\log(n))^k \in O(n)$$

(Bonus) For a big bonus let $\alpha > 0$ be a positive real number, show

$$(\log(n))^k \in O(n^\alpha)$$

Problem 5. Use Stirling's approximation to show

$$\log(n!) \in \Theta(n \log(n))$$

We gave a weak form of this problem in homework 2. You may to do a touch of outside reading in order to get a usable form of Stirling's approximation. Please show as much work as possible to and justify all constants bounding $\log(n!)$.

Problem 6. Define the three relations O, Ω, Θ on functions with domain and codomain $\mathbb{R}$ by

$$\begin{aligned} fOg &\iff f \in O(g) \\ f\Omega g &\iff f \in \Omega(g) \\ f\Theta g &\iff f \in \Theta(g) \end{aligned}$$

(a) Show O and Ω are reflexive and transitive.

(b) Show that Θ is an equivalence relation on functions.

3. Notes

In this problem set we finish up our modular arithmetic and begin our studyof asymptotic analysis. In problem one we have a classic style Chinese Remainder Theorem problem on Integers. In modern mathematics we write the set of equivalence classes of the relations

$$nRm \iff N|(n-m)$$

as $\mathbb{Z}/N$. We call this set "zee mod N" or "zed mod N" if you're an English speaker from literally anywhere other than the United States. What the Chinese Remainder Theorem Really tells us is that when $N = p_1 \cdots p_k$ is broken up into its prime factors we can write an isomorphism

$$\mathbb{Z}/N \simeq \mathbb{Z}/p_1 \times \mathbb{Z}/p_2 \times \cdots \mathbb{Z}/p_k$$

An isomorphism is a bijective function that also maintains the equivalence class structures. Generally speaking if we have a set A which allows two types of operations (addition and multiplication or boolean and/or, if-then/and, etc) where one operation distributes over another we call this type of set a ring. When there is a way to factor using the operation that distributes (i.e. prime factorization or DeMorgan's Laws, etc) then we can write

$$A/R \simeq A/R_1 \times \cdots \times A/R_k$$

where the equivalence relation R is factored into $R = R_1 R_2 \cdots R_k$. An enormous amount of modern geometry is done in this way where we describe many spaces as subsets of $n \times n$ matrices with special properties. Since matrices carry an addition and multiplication we can add equivalence relations to them. When we consider the equivalence classes of such a set (called a Lie group pronounces lee since Sophus Lie was from Scandinavia) the resultant space is called an orbifold. That's your fun math word for the week.

Problem two tells us the classical part of Shor's algorithm. Shor's algorithm, of course, is the famous quantum algorithm which is meant to destroy all modern encryption. Such fears are terribly overblown, but if we were to have a fully functional quantum computer it could solve the discrete logarithm problem very fast, that's the essence of Shor's algorithm. Once the discrete logarithm prolem is solved, we revert to the classical part, i.e. the Euclidean algorithm to acquire the actual factors of a number. This part, we know is extremely fast. Fast enough, in fact, that we can do it by hand for relatively small numbers. A supercomputer would have no trouble crushing Euclid's algorithm given the solution to the discrete logarithm problem.

Problems three through five are getting our algebraic chops in tune to compute asymptotic growths of some functions. I try to obey the heuristic

$$1 < \log(n) < n^{\alpha} < \beta^n < n! < n^n$$

Obviously this is not exhaustive, but all functions on the left are big O of all functions on the right, and functions on the right are big omega of functions on the left. Additionally, nothing across these transitive inequalities can be big theta.

And, since we couldn't let it go, you can prove many of these inequalities by induction.

Problem six brings us back to the heart of the course. Everything is tied together here. Big theta is an equivalence relation on function of real numbers.