**Carnegie Mellon University** Instructor: Venkatesan Guruswami
15-252: More GITCS Term: Spring 2021

PROBLEM SET 6
Due date: Friday, March 19 (by midnight EST)

INSTRUCTIONS

- You are allowed to collaborate with one other student taking the class, or do it solo.
- Collaboration is defined as discussion of the lecture material and solution approaches to the problems. Please note that *you are not allowed to share any written material and you must write up solutions on your own.* You must clearly acknowledge your collaborator in the write-up of your solutions.
- Solutions must be submitted on gradescope. Typesetting in LaTeX is recommended but not required. If submitting handwritten work, please make sure it is a legible and polished final draft.
- You should not search for solutions on the web. More generally, you should try and solve the problems without consulting any reference material other than what we cover in class and any provided notes.
- Please start working on the problem set early. Though it is short, the problem(s) might take some time to solve.

---

The FFT algorithm gives a way to evaluate a polynomial $f(x)$ of degree $(n-1)$, where $n$ is a power of 2, at all $n$'th roots of unity, using $O(n \log n)$ (complex) operations. But what if we want to evaluate the polynomial $f(x)$ at $n$ *arbitary* input points $u_0, u_1, \ldots, u_{n-1}$ (assume to be integers)? The naive approach of computing each $f(u_i)$ separately will take $O(n^2)$ operations. This exercise shows how one can also do this task in near-linear time.

To do so, we shall use the result (which we will assume without proof) that given two input polynomials $A(x), B(x)$ of degree $< n$ in the coefficient representation, one divide $A(x)$ by $B(x)$ and compute the remainder polynomial (in coefficient representation) using $O(n \log n)$ operations. For example, the remainder of $4x^3 + x^2 - 2x + 3$ when divided by $x^2 + x + 2$ is

$$(4x^3 + x^2 - 2x + 3) \mod (x^2 + x + 2) = -7x + 9 \ .$$

That is, as with polynomial multiplication, polynomial division with remainder, can also be computed in $O(n \log n)$ complexity.

Using the above, show how one can evaluate an input degree $(n-1)$ polynomial $f(X)$, given in coefficient representation, at any set of $n$ input points $\{u_0, u_1, \ldots, u_{n-1}\}$ using $O(n \ (\log n)^2)$ operations.

<u>Hint</u>: If you define $f_0(x) = f(x) \mod ((x-u_0)(x-u_1)\cdots(x-u_{n/2-1}))$ and $f_1(x) = f(x) \mod ((x-u_{n/2})(x-u_{n/2+1})\cdots(x-u_{n-1}))$ , then $f(u_i) = f_0(u_i)$ if $i < n/2$, and $f_1(u_i)$ if $i \geq n/2$. Use this to give a divide-and-conquer algorithm, and carefully account for the time to compute the product polynomials such as $(x-u_0)(x-u_1)\cdots(x-u_{n/2-1})$ that arise in the recursion.