CS60094 Computational Number Theory, Spring 2017–2018

Mid-Semester Test

23–February–2018 | CSE-107, 09:00am–11:00am | Maximum marks: 40

Roll no: ____________ **Name:** ______________________________

[*Write your answers in the question paper itself. Be brief and precise. Answer* *__all__* *questions.*
If you use any algorithm/result/formula covered in the class, just mention it, do not elaborate.]

1. Let f_n denote the n-th Fibonacci number. Prove that the smallest positive integers on which the Euclidean GCD algorithm requires n steps are f_{n+2} and f_{n+1}. In other words, if $\gcd(a,b)$ with $a > b > 0$ takes n steps, then $a \geqslant f_{n+2}$, and $b \geqslant f_{n+1}$. **(4)**

Solution We can prove this by induction on n. Base case is when $n = 1$, so a is a multiple of b. The minimum for such a and b is when $a = 2 = f_3$, and $b = 1 = f_2$.

Assuming that the hypothesis holds for $n = m$, we show it holds for $n = m+1$. Let $a = qb + r_1$. Now, if $\gcd(a,b)$ takes $n+1$ steps, $\gcd(b, r_1)$ takes n steps. Thus according to induction hypothesis, $b \geqslant f_{n+2}$ and $r_1 \geqslant f_{n+1}$. Since $q \geqslant 1$, we have $a \geqslant b + r_1 \geqslant f_{n+2} + f_{n+1} = f_{n+3}$. This completes the inductive step.

2. Solve the following parts with appropriate justifications. Here, a, b, c are arbitrary integers.

(a) Prove that if $a|b$ and $b|c$, then $a|c$. **(1)**

Solution There exist integers λ and μ such that $b = \lambda a$, $c = \mu b$. But then $c = \mu\lambda a$.

(b) Prove that if $a|(bc)$ and $\gcd(a,b) = 1$, then $a|c$. **(2)**

Solution There exist integers λ and μ such that $1 = \lambda a + \mu b$. Hence, $c = \lambda ac + \mu bc$. Since a divides the RHS, it must divide c.

(c) Using the rules of Jacobi-symbol computation, show that the congruence $x^2 \equiv 286 \pmod{563}$ is not solvable. **(3)**

Solution We have $\left(\frac{286}{563}\right) = \left(\frac{2}{563}\right)\left(\frac{143}{563}\right) = -\left(\frac{143}{563}\right) = \left(\frac{563}{143}\right) = \left(\frac{-9}{143}\right) = -\left(\frac{3^2}{143}\right) = -1.$

3. Use Hensel's lifting to prove the following: If a is a quadratic residue of an odd prime p, then it is also a quadratic residue of p^k for any positive integer k. **(3)**

Solution Since a is a quadratic residue of p, the congruence $f(x) = x^2 - a \equiv 0 \pmod{p}$ has a solution say x_0, $0 < x_0 < p$.

Now, $f'(x) = 2x$, and $f'(x_0) = 2x_0 \not\equiv 0 \pmod{p}$, as p is an odd prime. Hence, we can (uniquely) lift x_0 to p^2, and then to p^3, and so on to p^k for any integer $k \geqslant 2$.

4. The Discrete Fourier Transform (DFT) requires the use of complex numbers, which can result in a loss of precision due to round-off errors. For some problems, the answer is known to contain only integers, and it is desirable to utilize a variant of the DFT, based on modular arithmetic in order to guarantee that the answer is computed exactly. Let n be the number of points, of which the DFT is taken. In this exercise, we develop a strategy where the modulus p is of length $O(\lg n)$. Answer the following parts in this context.

(a) Suppose that we search for the smallest $k \in \mathbb{N}$ such that $p = kn + 1$ is prime. Give a simple heuristic argument why we expect k to be $O(\lg n)$. How does the expected length of p compare to the length of n? **(3)**

Solution From the prime number theorem, between 1 and N there are about $N/\ln N$ prime numbers. Hence, the probability that a random number from 1 and N is prime is $\frac{(N/\ln N)}{N} = \frac{1}{\ln N}$. If $N = n \ln n$, then this probability is about $\frac{1}{\ln n}$.

Hence, if we vary k from 1 to $O(\lg n)$, then in the desired form of one more than a multiple of n, we would expect one number to be prime.

The bit length of p is $\approx \lg k + \lg n$. We expect $k = O(\lg n)$, so the expected bit length of p is $\lg n + O(\lg \lg n)$.

(b) Let g be a generator of Z_p^*, and let $w \equiv g^k \pmod{p}$. State the DFT operation modulo p using w. **(2)**

Solution Since $w^n \equiv g^{kn} \equiv g^{p-1} \equiv 1 \pmod{p}$, we can simply replace the complex n-th root of unity by this value to obtain the formulation.

(c) Let p and w be supplied as inputs to the DFT algorithm. Show that the DFT takes time $O(n \lg n)$, under the assumption that operations on words of $O(\lg n)$ bits take unit time. **(2)**

Solution Consider the numbers $1, w, w^2, \ldots, w^{n/2}, w^{n/2+1}, \ldots, w^{n-1}$ modulo p. We have $w^{n/2} \equiv -1 \pmod{p}$, $w^{n/2+1} \equiv -w \pmod{p}$, and so on. Thus, we can simply apply the recursion to evaluate the input polynomial at these n points. Hence, the recurrence is $T(n) = 2T(n/2) + O(n)$. This implies $T(n) = O(n \lg n)$.

5. (a) Prove that the polynomial $f(x) = x^3 + 2x + 2$ is irreducible over $\mathbb{F}_3$. **(2)**

Solution Since $\deg(f) = 3$, the polynomial must have a root in $\mathbb{F}_3$ if it is reducible. But $f(0) \equiv f(1) \equiv f(2) \equiv 2 \pmod 3$.

(b) Define $\mathbb{F}_{27} = \mathbb{F}_{3^3} = \mathbb{F}_3(\theta)$, where $f(\theta) = \theta^3 + 2\theta + 2 = 0$. Take the element $\gamma = \theta + 2 \in \mathbb{F}_{27}$. Determine whether γ is a primitive element of $\mathbb{F}_{27}$. **(4)**

Solution Since $|\mathbb{F}_{27}^*| = 26 = 2 \times 13$. Also $\gamma \neq 1$. So it suffices to compute γ^2 and γ^{13} to determine whether γ is primitive. We have $\theta^3 + 2\theta + 2 = 0$, that is, $\theta^3 = \theta + 1$. We then have:

$$\begin{array}{rcl}
\gamma & = & \theta + 2, \\
\gamma^2 & = & \theta^2 + \theta + 1, \\
\gamma^3 & = & \theta^3 + 2 \;=\; \theta, \\
\gamma^9 & = & \theta^3 \;=\; \theta + 1, \\
\gamma^{12} & = & \gamma^9 \times \gamma^3 \;=\; (\theta+1)\theta \;=\; \theta^2 + \theta, \\
\gamma^{13} & = & \gamma^{12} \times \gamma \;=\; (\theta^2+\theta)(\theta+2) \;=\; \theta^3 + 2\theta \;=\; 1.
\end{array}$$

Since $\gamma^{13} = 1$, γ is <u>not</u> a primitive element of $\mathbb{F}_{27}$.

(c) Determine whether the element $\delta = \theta^2 \in \mathbb{F}_{27}$ is a normal element of $\mathbb{F}_{27}$. **(4)**

Solution We have

$$\begin{array}{rcl}
\delta & = & \theta^2, \\
\delta^3 & = & \theta^6 = (\theta+1)^2 = \theta^2 + 2\theta + 1, \\
\delta^9 & = & \theta^6 + 2\theta^3 + 1 = (\theta^2 + 2\theta + 1) + 2(\theta + 1) + 1 = \theta^2 + \theta + 1,
\end{array}$$

that is,

$$\begin{pmatrix} \delta \\ \delta^3 \\ \delta^9 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ \theta \\ \theta^2 \end{pmatrix}.$$

The transformation matrix has non-zero determinant $1 - 2 \equiv 2 \pmod 3$, so δ <u>is</u> a normal element of $\mathbb{F}_{27}$.

6. Take an extension field $\mathbb{F}_q = \mathbb{F}_{p^n}$ for a prime p and for $n \geqslant 2$. Suppose that p is small, so the basic arithmetic operations in $\mathbb{F}_p$ can be assumed to run in $O(1)$ time. Let $\theta_0, \theta_1, \theta_2, \ldots, \theta_{n-1}$ constitute an arbitrary $\mathbb{F}_p$-basis of $\mathbb{F}_q$. For all $i, j \in \{0, 1, 2, \ldots, n-1\}$, write

$$\theta_i \theta_j = \sum_{k=0}^{n-1} t_{i,j,k} \theta_k$$

with $t_{i,j,k} \in \mathbb{F}_p$. Suppose that the n^3 elements $t_{i,j,k}$ are precomputed and stored. Finally, let

$$1 = \sum_{k=0}^{n-1} c_k \theta_k,$$

and suppose that the n elements $c_k \in \mathbb{F}_p$ are also precomputed and stored.

Let $\alpha = a_0\theta_0 + a_1\theta_1 + a_2\theta_2 + \cdots + a_{n-1}\theta_{n-1}$, $a_i \in \mathbb{F}_p$, be an element of $\mathbb{F}_q^*$ expressed in the given basis. We want to compute the inverse of α again in the given basis, that is, the element $\beta = b_0\theta_0 + b_1\theta_1 + b_2\theta_2 + \cdots + b_{n-1}\theta_{n-1} \in \mathbb{F}_q^*$ with $\alpha\beta = 1$. Fermat's little theorem implies $\beta = \alpha^{q-2}$. Since each multiplication in $\mathbb{F}_q$ can be done by table lookup in $O(n^3)$ time, and the exponentiation can be done by a square-and-multiply algorithm having $\log_2(q-2) \approx n \log_2 p$ iterations, the overall running time is $O(n^4)$. Propose an $O(n^3)$-time algorithm to compute $\beta = \alpha^{-1}$. **(10)**

Solution We use linear algebra to solve this problem. We need to determine the unknown quantities $b_j \in \mathbb{F}_p$.

1. Since $\alpha\beta = 1$, we have

$$\begin{aligned}\sum_{k=0}^{n-1} c_k\theta_k &= \left(\sum_{i=0}^{n-1} a_i\theta_i\right)\left(\sum_{j=0}^{n-1} b_j\theta_j\right) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_i b_j \theta_i\theta_j = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\left(a_i b_j \sum_{k=0}^{n-1} t_{i,j,k}\theta_k\right) \\ &= \sum_{k=0}^{n-1}\left(\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_i b_j t_{i,j,k}\right)\theta_k = \sum_{k=0}^{n-1}\left[\sum_{j=0}^{n-1}\left(\sum_{i=0}^{n-1} a_i t_{i,j,k}\right) b_j\right]\theta_k\end{aligned}$$

   For all $j, k \in \{0, 1, 2, \ldots, n-1\}$, we compute $s_{j,k} = \sum_{i=0}^{n-1} a_i t_{i,j,k}$.

2. We have the following set of n linear equations in the variables $b_0, b_1, b_2, \ldots, b_{n-1}$:

$$\sum_{j=0}^{n=1} s_{j,k} b_j = c_k$$

   for $k = 0, 1, 2, \ldots, n-1$. We solve the system modulo p to obtain $b_0, b_1, b_2, \ldots, b_{n-1}$.

Step 1 requires the computation of n^2 elements $s_{j,k}$ of $\mathbb{F}_p$, each involving an n-fold sum over $\mathbb{F}_p$, so this step takes a total of $O(n^3)$ time. Finally, the $n \times n$ linear system of equations over $\mathbb{F}_p$ can be solved in Step 2 by Gaussian elimination using $O(n^3)$ operations in $\mathbb{F}_p$.