CS 70 Discrete Mathematics and Probability Theory
Spring 2015 Vazirani HW 5

Reminder: Deadline for HW4 self-grade is Thursday, February 19 at noon.

Reading: Note 6.

Due Monday February 23

1. **Multiplicative Inverses**

   The goal of this question is to give a more transparent proof (without appealing to the gcd algorithm) that if $\gcd(a,n) = 1$ then $a^{-1} \pmod{n}$ exists. Fill in the details of the following outline of the proof: Note that you are trying to show that $ka \equiv 1 \pmod{n}$ for some k. First observe that it is sufficient to show that $ja \pmod{n}$ must be distinct for $j = 0, 1, \ldots, n-1$. Assume for contradiction that this in not true, and show that it contradicts the condition that $\gcd(a,n) = 1$.

   Answer:

   It is sufficient to prove that $ja \pmod{n}$ is distinct for $j = 0, \ldots, n-1$ because then the mapping $j \to ja \pmod{n}$ would be a one-to-one mapping from the set $\{0, \ldots, n-1\}$ to the set $\{0, \ldots, n-1\}$. But any one-to-one mapping from an n element set to an n element set must be a bijection. Therefore there is a number that is mapped to 1. In other words, there exists a k such that $ka = 1 \pmod{n}$, which is exactly what we wanted.

   Now let us prove that $ja \pmod{n}$ is distinct for $j = 0, \ldots, n-1$. For the sake of contradiction assume this is not the case. Then $ja \equiv j'a \pmod{n}$ for some $j \not\equiv j' \pmod{n}$. This means that $n \mid (j-j')a$, i.e. $\frac{(j-j')a}{n}$ is an integer. But note that a and n do not have any common factors. So if we write out the prime decompositions of $(j-j')$, a, and n, none of the primes that appear in a would cancel out the primes that appear in n. This means that even without a, the fraction should result in an integer, i.e. $\frac{j-j'}{n}$ is an integer. But this means that $n \mid j-j'$ or in other words $j \equiv j' \pmod{n}$, which is a contradiction.

2. **Combining moduli**

   Suppose we wish to work modulo $n = 40$. Note that $40 = 5 \times 8$, with $\gcd(5,8) = 1$. We will show that in many ways working modulo 40 is the same as working modulo 5 and modulo 8, in the sense that instead of writing down $c \pmod{40}$, we can just write down $c \pmod{5}$ and $c \pmod{8}$.

   (a) What is $8 \pmod{5}$ and $8 \pmod{8}$. Find a number $a \pmod{40}$ such that $a \equiv 1 \pmod{5}$ and $a \equiv 0 \pmod{8}$.

   (b) Now find a number $b \pmod{40}$ such that $b \equiv 0 \pmod{5}$ and $b \equiv 1 \pmod{8}$.

   (c) Now suppose you wish to find a number $c \pmod{40}$ such that $c \equiv 2 \pmod{5}$ and $c \equiv 5 \pmod{8}$. Find c by expressing it in terms of a and b.

   (d) Repeat to find a number $d \pmod{40}$ such that $d \equiv 3 \pmod{5}$ and $d \equiv 4 \pmod{8}$.

   (e) Compute $c \times d \pmod{40}$. Is it true that $c \times d \equiv 2 \times 3 \pmod{5}$, and $c \times d \equiv 5 \times 4 \pmod{8}$

Answer:

(a) $8 \equiv 3 \pmod 5$ and $8 \equiv 0 \pmod 8$. We can find such a number by considering multiples of 8, i.e. $0, 8, 16, 24, 32$, and find that if $a = 16$, $16 \equiv 1 \pmod 5$. Therefore 16 satisfies both conditions.

(b) We can find such a number by considering multiples of 5, i.e. $0, 5, 10, 15, 20, 25, 30, 35$, and find that if $b = 25$, $25 \equiv 1 \pmod 8$, so it satisfies both conditions.

(c) We claim $c \equiv 2a + 5b \equiv 37 \pmod{40}$. To see that $c \equiv 2 \pmod 5$, we note that $b \equiv 0 \pmod 5$ and $a \equiv 1 \pmod 5$. So $c \equiv 2a \equiv 2 \pmod 5$. Similarly $c \equiv 5b \equiv 5 \pmod 8$.

(d) We can repeat the same procedure as above, and find that $d = 3a + 4b \equiv 28 \pmod{40}$.

(e) $c \times d = 37 \times 28 \equiv 36 \pmod{40}$. Note that if $a \equiv b \pmod n$ and $c \equiv d \pmod n$ then $a \times c \equiv b \times d \pmod n$. Therefore we can multiply $c \equiv 2 \pmod 5$ and $d \equiv 3 \pmod 5$ to get $c \times d \equiv 2 \times 3 \pmod 5$. Similarly we can multiply these equations $\pmod 8$ and get $c \times d = 5 \times 4 \pmod 8$.

3. **CRT**

We generalize the previous question in two successive steps. In part (a) we work with arbitrary n_1 and n_2 instead of 5 and 8, and in part (b) we generalize to arbitrary k instead of $k = 2$.

(a) Suppose $\gcd(n_1, n_2) = 1$. Prove that given $r_1 \in \{0, 1, \ldots, n_1 - 1\}$ and $r_2 \in \{0, 1, \ldots, n_2 - 1\}$, there is a unique $c \pmod{n_1 n_2}$, such that $c \equiv r_1 \pmod{n_1}$ and $c \equiv r_2 \pmod{n_2}$.

(b) Given $n_1, \ldots, n_k$ with $\gcd(n_i, n_j) = 1$, and $r_i : \ 0 \le r_i \le n_i - 1$, prove that there is a unique $c \pmod{n_1 \cdot \ldots \cdot n_k}$, such that $c \equiv r_i \pmod{n_i}$ for $i = 1, \ldots, k$.

Congratulations, you just proved the Chinese Remainder theorem!

Answer:

(a) We follow the previous question, and let $m_1 \equiv n_1^{-1} \pmod{n_2}$. Then $n_1 m_1 \equiv 1 \pmod{n_2}$ and $n_1 m_1 \equiv 0 \pmod{n_1}$.
Similarly let $m_2 \equiv n_2^{-1} \pmod{n_1}$. Then $n_2 m_2 \equiv 1 \pmod{n_1}$ and $n_2 m_2 \equiv 0 \pmod{n_2}$.
Now $c \equiv r_1 n_2 m_2 + r_2 n_1 m_1 \pmod{n_1 n_2}$ has the desired property. Because

$$c \equiv r_1 n_2 m_2 \equiv r_1 \pmod{n_1}$$

and

$$c \equiv r_2 n_1 m_1 \equiv r_2 \pmod{n_2}.$$

We then need to prove that this c is a unique solution. Suppose there is another solution $c' \not\equiv c \pmod{n_1 n_2}$. Let $d = c' - c$. Then $d \not\equiv 0 \pmod{n_1 n_2}$ and satisfies the equations $d \equiv 0 \pmod{n_1}$ and $d \equiv 0 \pmod{n_2}$. This means that n_1 and n_2 both divide d. Since n_1 and n_2 share no common factors, $n_1 n_2$ should also divide d, which means that $d \equiv 0 \pmod{n_1 n_2}$, a contradiction. Therefore there is only one solution.

(b) We prove this by induction on k.

Base case When $k = 1$, There is only 1 solution $\pmod{n_1}$ to $c \equiv r_1 \pmod{n_1}$, which is $c = r_1$

Induction hypothesis Suppose there exists a unique $c \pmod{n_1 \cdot \ldots \cdot n_k}$, such that $c \equiv r_i \pmod{n_i}$ for $i = 1, \cdots, k$.

Induction step Let $N_1 = n_1 \cdot \ldots \cdot n_k$ and $N_2 = n_{k+1}$. Now using the induction hypothesis there is a unique $R_1 \pmod{N}_1$ such that $c \equiv r_1 \pmod{n_1}$, $\ldots$, $c \equiv r_k \pmod{n_k}$ if and only if $c \equiv R_1 \pmod{N_1}$.

Note that we want solutions that satisfy both $c \equiv r_1 \pmod{n_1}, \ldots, c \equiv r_k \pmod{n_k}$ and $c \equiv r_{k+1} \pmod{n_{k+1}}$. We just showed that these are exactly the set of solutions that satisfy $c \equiv R_1 \pmod{N_1}$ and $c \equiv r_{k+1} \pmod{N_2}$. Now we appeal to part (a). It tells us that there is a unique $R \pmod{N_1 N_2}$ such that $c \equiv r_1 \pmod{n_1}, \ldots, c \equiv r_{k+1} \pmod{n_{k+1}}$ if and only if $c \equiv R \pmod{N_1 N_2}$. Note that $N_1 N_2 = n_1 \ldots n_{k+1}$. This is exactly what we wanted to prove.

4. **Consecutive composites**

Given any positive integer k, describe a way to write down k consecutive composite numbers. *Hint: The Chinese remainder theorem might be helpful!*

Answer:

There are many answers to this. Here are two possible ones, one using the Chinese Remainder Theorem. They both use the same idea, which is to construct a number that has $2, 3, \cdots, k+1$ as divisors.

Answer 1:

Consider all primes p_i that are smaller than or equal to $k+1$. Since p_i are all primes, we can find a unique $c \pmod{\prod p_i}$ such that $c \equiv 0 \pmod{p_i}$. Then $c+2, c+3, c+4, \cdots, c+(k+1)$ is such a sequence since c and any number less than or equal to $k+1$ share at least one common prime divisor. Note that we need c to be positive, since although $c+k$ shares a factor with k, we want that factor to not be the whole $c+k$. This would be impossible if $c > 0$. CRT says such a c is unique $\pmod{\prod p_i}$, but arbitrary shifts of length $\prod p_i$ are allowed, so we can always have $c > 0$.

Answer 2:

$(k+1)!+2, (k+1)!+3, (k+1)!+4, \cdots, (k+1)!+(k+1)$ is also a sequence of k consecutive composite integers.

5. **Binary GCD**

(a) Prove that the following statements are true for all $m, n \in \mathbb{N}$.

$$\begin{aligned} \text{If } m \text{ is even and } n \text{ is even}, &\quad \gcd(m,n) = 2\gcd(m/2, n/2). \\ \text{If } m \text{ is even and } n \text{ is odd}, &\quad \gcd(m,n) = \gcd(m/2, n). \\ \text{If } m, n \text{ are both odd and } m \geq n, &\quad \gcd(m,n) = \gcd((m-n)/2, n). \end{aligned}$$

(b) Fill in the missing part of the following template to get an alternative algorithm for computing the gcd.

gcd(m,n):
1. If $m = 0$, return n. If $n = 0$, return m.
2. If m is even and n is even, return $2 \cdot \gcd(m/2, n/2)$.
3. If m is even and n is odd, return $\gcd(m/2, n)$.
4. If m is odd and n is even, return $\gcd(m, n/2)$.
5. ??????????.

Prove that the resulting algorithm correctly computes the gcd.

Answer:

(a) *Approach.* If $a|b$ and $b|a$, then $a = b$.

Proof. m,n are both even: if d divides both $m/2$ and $n/2$, then $2d$ divides both m and n, so $2 \cdot \gcd(m/2,n/2) | \gcd(m,n)$. Moreover, $d' = \gcd(m,n)$ is even, and since d' divides both m and n, $d'/2$ divides both $m/2$ and $n/2$, whence $d'/2 | \gcd(m/2,n/2)$, i.e., $\gcd(m,n) | 2 \cdot \gcd(m/2,n/2)$.

m is even and n is odd: This can be verified as before: if d divides both m and n, then d is odd, and so d divides $m/2$, whence $\gcd(m,n) | \gcd(m/2,n)$; the other direction is trivial. The case where m is odd and n is even is symmetrical.

m,n are both odd: We know from class that when $m \leq n$, $\gcd(m,n) = \gcd(n-m,m)$. (This is the basis of the Euclidean algorithm.) Since $n-m$ is even and m is odd, applying the observation of the paragraph above yields $\gcd(m,n) = \gcd(n-m,m) = \gcd((n-m)/2,m)$. The case where $m > n$ is symmetrical.

□

(b) The full algorithm is as follows:

$\gcd(m,n)$:
1. If $m = 0$, return n. If $n = 0$, return m.
2. If m is even and n is even, return $2 \cdot \gcd(m/2,n/2)$.
3. If m is even and n is odd, return $\gcd(m/2,n)$.
4. If m is odd and n is even, return $\gcd(m,n/2)$.
5. If $m \geq n$ then return $\gcd((m-n)/2,n)$ else return $\gcd((n-m)/2,m)$.

We first prove that the algorithm terminates. We can see that $m+n$ is always greater than 0, and each recursive call either decreases the value of $m+n$ keeping it above 0 or terminates. Thus the algorithm will always terminate.

Since the algorithm terminates in a finite amount of recursive calls, we can now prove that it returns the correct result after terminating. We do this by inducting on n, the number of recursive calls the algorithm takes to terminate.

Base case When the algorithm terminates in 1 recursive call, m or n must be 0. Since $\gcd(0,k) = \gcd(k,0) = k$, the algorithm will return the correct answer.

Inductive hypothesis If the algorithm returns the correct answer on all inputs using n recursive calls, it will also return the correct answer on all inputs needing $n+1$ recursive calls.

Inductive step We prove by cases, considering if m and n are even. Each case is proved in the previous part to be equal to another recursive call of gcd, which will give the correct answer by the inductive hypothesis. Therefore if the function takes $n+1$ recursive calls, it correctly computes the gcd of two numbers.

In practice, an important reason that the binary gcd algorithm might be better than the Euclidean algorithm is that the constant factors in the binary gcd algorithm seem to be smaller, and the operations used in the binary gcd algorithm are better suited for computer implementation. Dividing by 2, multiplying by 2, and testing whether an integer is even are extremely fast operations when the inputs are represented in binary, and this makes the binary gcd algorithm fairly attractive in software. It is also potentially attractive in hardware, since we don't need to implement a modular reduction circuit.

6. **Midterm 1**

   Solve the question from midterm 1 for which you got the least points.

   Answer: See Midterm 1 solutions.

7. **Midterm 1**

   Solve the question from midterm 1 for which you got the second least points.

   Answer: See Midterm 1 solutions.