Adv Topics 2 (Lesieutre)
August 24, 2021

Problem 1. *This week, we'll think about cryptography. Divide yourselves into two groups, and imagine that you want to be able to pass coded messages back and forth without your teacher being able to read them, even if they are intercepted.*

If you could agree on a shared 4-digit secret key (unknown to the teacher), you could use this to encode the message.

a) *Try to have the two groups agree on a secret key by sending some messages back and forth. Remember, I get to read the messages! What is your strategy?*
You could try to use some 2020's pop culture reference that I don't know about, or maybe some State High trivia...

b) *Now imagine that you are trying to agree on a key with a stranger in Amazon's payments processing department, and that anyone on the internet could listen in. Can you think of a new method?*
That's where we're headed next! We'll cover the Diffie–Hellman algorithm for key exchanges.

Problem 2. *Compute the following modular quantities:*

a) $19 \bmod 4$
$19 = 16 + 3 = 4 \cdot 4 + 3$, so it's 3.

b) $(232 + 37) \bmod 11$ *(There are two ways you could do this – do you get the same answer? Why?)*
We could add, and then reduce mod 11:

$$(232 + 37) \bmod 11 = 269 \bmod 11 = 269 - 220 \bmod 11 = 49 \bmod 11 = 5$$

(Here I used a little shortcut that made it easier to do in my head: if you add or subtract a multiple of 11 to something, it doesn't change the value mod 11, and $220 = 20 \times 11$ is an easyish-to-see multiple of 11.)
Or we could reduce mod 11, and then add (and reduce again, if needed):

$$(232 + 37) \bmod 11 = 232 \bmod 11 + 37 \bmod 11 = 1 + 4 = 5.$$

Problem 3. *Compute the following modular exponentials.*

a) $19^2 \bmod 11$
As with addition, it's easier to reduce first. $19 \equiv 8 \pmod{11}$.

$$19^2 \bmod 11 \equiv 8^2 \bmod 11 \equiv 64 \bmod 11 = 9.$$

b) $19^3 \bmod 11$
We can at least recycle what we already know:

$$19^3 \bmod 11 \equiv 8^3 \bmod 11 = 8^2 \cdot 8 \bmod 11 = 9 \cdot 8 \bmod 11 = 72 \bmod 11 = 6.$$

c) $19^4 \bmod 11$
More of the same.

$$19^4 \bmod 11 \equiv 8^4 \bmod 11 = 8^3 \cdot 8 \bmod 11 = 6 \cdot 8 \bmod 11 = 48 \bmod 11 = 4.$$

d) $19^8 \bmod 11$
Let's use a slightly different trick this time.

$$19^8 \bmod 11 \equiv (19^4)^2 \bmod 11 = 4 \cdot 4 \bmod 11 = 5.$$

e) $19^{16} \bmod 11$

$$19^{16} \bmod 11 \equiv (19^8)^2 \bmod 11 = 5^2 \bmod 11 = 3.$$

f) $19^{21} \bmod 11$
Here's our first hint at the main trick.

$$19^{21} \bmod 11 \equiv 19^{16} \cdot 19^4 \cdot 19^1 \bmod 11 = 3 \cdot 4 \cdot 8 = 8$$

g) $19^{73} \bmod 11$
OK, this one isn't so easy. Here's the trick. Keep going with the powers of 2 for a little longer:

$$\begin{aligned} 19^{32} \bmod 11 &\equiv (19^{16})^2 \bmod 11 = 3^2 \bmod 11 = 9 \\ 19^{64} \bmod 11 &\equiv (19^{32})^2 \bmod 11 = 9^2 \bmod 11 = 4. \end{aligned}$$

Now,

$$\begin{aligned} 19^{73} \bmod 11 &= 19^{64} \cdot 19^8 \cdot 19 \bmod 11 = 4 \cdot 5 \cdot 8 \bmod 11 = 20 \cdot 8 \bmod 11 \\ &= 9 \cdot 8 \bmod 11 = 72 \bmod 11 = 6. \end{aligned}$$

That wasn't so bad. Pause a moment to reflect that had we done this more directly, it would've been extremely painful. The number 19^{73} has almost a hundred digits. Even if we avoided calculating that, doing 73 separate multiplications mod11 would've taken an awfully long time. The powers of 2 trick has saved us a lot of trouble.

Problem 4. *Describe a general method for compute* $a^b \bmod n$ *without doing* b *separate multiplications. (For the computer scientists: roughly how many times do you need to multiply?)*

We just did it in an example. The steps are basically:

1. Compute $a^{2^k} \bmod n$ (exponents are powers of two) by repeated squaring. Keep going until 2^k is bigger than b (really, you can stop one step before that).
2. Write b as a sum of powers of 2. (This is the same thing as finding its expression in binary!)
3. Multiply out the corresponding a^{2^k} terms to obtain the final answer.

This takes about $\log_2 b$ steps, which when the numbers get big is a lot less than the b steps that would be required.

Problem 5. *Use Diffie–Hellman key exchange to generate a shared two-digit key with a friend.*

I'll leave this one to you. You can check the notes to remember the process.

Problem 6. *How secure is this, anyway? Think through the Diffie–Hellman algorithm. Which quantities are known to someone eavesdropping on your messages? What else would they need to know or calculate in order to obtain the shared key?*

A listener knows the numbers p, g, A, and B, which are all sent publicly. If they can get x or y from these we're busted. In principle you could solve for x in $A = g^x \pmod p$, just by trying every possible value. But there is no cleverer way than just trying every possible value, which makes it computationally infeasible. So in practice this is secure. This is called the *discrete logarithm problem* and there's no fast algorithm for it. (Unless your nemesis has a quantum computer, in which case, you probably have bigger problems.)

Problem 7. *Let* $\phi(n)$ *denote the Euler* ϕ *function, the number of integers less than* n *and coprime to* n.

a) *Compute* $\phi(10)$.

The numbers less than 10 and coprime to it are $1, 3, 7, 9$. So $\phi(10) = 4$.

b) *Compute* $\phi(13)$.

Every number less than 13 is coprime to it. So this is 12.

c) *What is $\phi(p)$, if p is a prime number?*
As above, you're always going to get $p-1$.

d) *What is $\phi(pq)$, if p and q are both prime numbers?*
This is a little trickier, but this is the one that's used in RSA. Try drawing a $p \times q$ grid in some example and working it out. The answer is $(p-1)(q-1)$.
If you want something to think about – how can you compute $\phi(n)$ if you know the prime factorization of n? (There's a fairly simple formula.)

e) *Compute* $7^{12} \pmod{13}$ *using our method from before. Does your answer agree with the calculation of Fermat's little theorem?*
First, run up the powers of 2. I get $7^1 \equiv 7 \pmod{13}$, $7^2 = 49 \equiv 10 \pmod{13}$, $7^4 \equiv 100 \equiv 9 \pmod{13}$, $7^8 = 81 \equiv 3 \pmod{13}$.
Then

$$7^{12} = 7^8 \cdot 7^4 = 3 \cdot 9 \pmod{13} = 27 = 1.$$

Problem 8. *You plan to receive an encrypted message from me using RSA. First, you need to pick two large primes. If you want to send the message in a single piece (instead of splitting it into smaller pieces), their product needs to have as many digits as the message will have. (To be totally safe, you need each prime to have that many digits, just in case the message is not coprime to n. Let's not.)*

Say you chose the two primes

$$p = 1500450271,$$
$$q = 3628273133.$$

a) *First, you need to compute the product $n = pq$ as well as $\phi(n)$. What is $\phi(n)$?*
We have

$$n = 5444043405671869043.$$

To get $\phi(n)$, use

$$\phi(n) = (p-1)(q-1) = 5444043400543145640.$$

b) *Next you need to generate the encryption and decryption keys. You can either pick d and calculate e, or vice versa. Let's say we pick $d = 65537$, and solve for e. What is e? (We'll want to ask a computer to do this.)*
I get $e = 1425949448612594993$.

c) *With those numbers in hand, you publish n and e, but keep d secret. I now write my message in the form of a number M and compute $E = M^e \pmod{n}$. Again this is easy, because I can use fast modular exponentiation. Here's the message I send you:*

$$E = 5219361229139918678$$

d) *Now decrypt the message given in the previous part. First find M from E using the RSA algorithm. Note that the decryption key is $d = 65537 = 2^{16} + 2^0$, so you can find this using the repeated squaring algorithm mod n without very much work, as long as you can square mod n.*

You simply compute

$$M = E^d \pmod{n} = 1409030523151811.$$

e) *To turn it into text, write M as a base-10 number. Each two digits in M correspond to an English letter via the code $A = 01$, $B = 02$, $C = 03$, $\ldots$, $Z = 26$. What is the message?*

The letters are

$$14 \cdot 09 \cdot 03 \cdot 05 \cdot 23 \cdot 15 \cdot 18 \cdot 11.$$

That's `NICEWORK`.