

## CAS CS 538. Problem Set 3

**Problem 1.** Let  $p > 2$  be a prime. Recall that  $\mathbb{Z}_p$  is a field of integers modulo  $p$ , and  $\mathbb{Z}_p^*$  is the multiplicative group of that field. Recall Fermat's little theorem: for all  $a \in \mathbb{Z}_p^*$ ,  $a^{p-1} \equiv 1 \pmod{p}$ . You may also use the following fact (without proof—see any number theory textbook or me if you'd like to see the proof): there exists a generator  $g$  such that  $g, g^2, g^3, \dots, g^{p-1}$  (all modulo  $p$ ) are all distinct and, therefore, cover all of  $\mathbb{Z}_p^*$ . In other words, there exists  $g \in \mathbb{Z}_p^*$  such that the map  $\mathbb{Z}_p^* \rightarrow \mathbb{Z}_p^*$  given by  $x \mapsto g^x \pmod{p}$  is a bijection.

Let  $q > 2$  also be a prime, and  $n = pq$ .  $Z_n^*$  is a multiplicative group (using multiplication modulo  $n$ ). Its elements are all those integers  $x, 0 < x < n$  that are relatively prime with  $n$ . Euler totient function  $\phi(n)$  is defined to be the size of  $Z_n^*$ . Euler's theorem generalizes Fermat's little theorem: for all  $a \in Z_n^*$ ,  $a^{\phi(n)} \equiv 1 \pmod{n}$ .

(a) Derive (and prove) the values of  $\phi(p)$  and  $\phi(n)$  for the  $p, n$  above.

(b) Show that exponents in  $Z_p^*$  (resp. in  $Z_n^*$ ) work modulo  $p-1$  (resp. modulo  $\phi(n)$ ): in other words, if  $x \equiv y \pmod{p-1}$  then, for any  $a$ ,  $a^x \equiv a^y \pmod{p}$  (hint: write  $x$  as  $(p-1)k_x + r$  and  $y$  as  $(p-1)k_y + r$ ); respectively, if  $x \equiv y \pmod{\phi(n)}$  then, for any  $a \in Z_n^*$ ,  $a^x \equiv a^y \pmod{n}$ .

(c) Show that if  $g$  is a generator of  $Z_p^*$ , then  $g^x \equiv 1$  if and only if  $(p-1) \mid x$  (Hint: let  $r$  be the remainder of  $x$  modulo  $p-1$ . Consider  $g^r$ .)

(d) Let  $a = g^x \pmod{p}$ , where  $g$  is again a generator. Show that if  $x$  is even, then  $a$  has a square root modulo  $p$ . Now show the converse: if  $a$  has a square root modulo  $p$ , then  $x$  is even. (Hint: suppose  $x$  is odd. Represent  $r$  as  $g^y$ . Then  $g^{2y} \equiv g^x \pmod{p}$ . Now use the previous part to show that  $x$  is even.) Thus, we can tell from the exponent whether or not the value is a square.

(e) We'd also like to be able to tell if a value is a square modulo  $p$  without having to know its discrete logarithm. Show that if  $a$  is a square, then  $a^{(p-1)/2} \equiv 1 \pmod{p}$ . Now show that if  $a$  is a non-square, then  $a^{(p-1)/2} \not\equiv 1 \pmod{p}$ .<sup>1</sup> (Hint: first write  $a$  as  $g^x$ ; then use the previous part.)

(f) Show that if  $p \equiv 3 \pmod{4}$ , and  $a$  has a square root, then the value  $a^{(p+1)/4}$  is a square root of  $a$ . (We thus have a simple algorithm to compute square roots for half the primes. The algorithm to compute square roots in other case, if  $p \equiv 1 \pmod{4}$ , is a little bit more complex, but still very efficient.)

**Problem 2.** Given  $n = pq$  (but not  $p, q$ ) and  $\phi(n)$  factor  $n$ .

(Hint: given  $a, b$  one can find  $x, y$ , such that  $xy = a, x + y = b$  by solving a quadratic equation; namely, a quadratic equation with variable  $z$  and roots  $x, y$  can be written as  $(z-x)(z-y) = 0$ )

**Problem 3.** Prove that if the discrete logarithm assumption holds, then it is hard to compute  $x$  given  $g^{xy}$  and  $g^y$ . Formally, prove that for any poly-time algorithm  $A$ , there exists a negligible

---

<sup>1</sup>In fact, it can be shown that it is  $-1$  in this case, and the value of  $a^{(p-1)/2}$  is called the Legendre symbol of  $a$  and is often written as  $\left(\frac{a}{p}\right)$ .

function  $\eta$  such that, if you generate random  $k$ -bit  $p$  and its generator  $g$  and select a random  $x, y \in \mathbb{Z}_p^*$ ,  $\Pr[A(p, g, g^{xy} \bmod p, g^y \bmod p) = x] \leq \eta(k)$ .

**Problem 4.**

Recall that CRT works not only for a product of two primes, but also for any product of two (or more) relatively prime integers. Therefore, for any odd  $n$  that is not prime or prime power, a square  $s \in QR_n \subset \mathbb{Z}_n^*$  has at least four square roots (this is just like in class: because  $n$  can be represented as  $n_1 n_2$  where  $n_1$  and  $n_2$  are relatively prime and odd; then  $s$  has at least two roots  $(r_1, -r_1)$  modulo  $n_1$  and at least two roots  $(r_2, -r_2)$  modulo  $n_2$ ). On the other hand, if  $n$  is an odd prime, then  $s$  has two square roots. You can use these facts to build a probabilistic primality test. As always, in problems below you can use the previous part in the next part, even if you haven't proven it.

(a) Show a polynomial-time algorithm to check if  $n$  is a perfect power (i.e., if there exist integers  $a, b > 1$  such that  $a^b = n$ ). The algorithm's running time should be polynomial in the length of the binary representation of  $n$  (not in  $n$  itself). (Hint: this is very straightforward, if you are thinking of something complicated, you are probably on the wrong track. No knowledge of number theory needed here.)

(b)

Assume you have a polynomial-time algorithm  $A$  that, on input a prime  $p$  and a square  $s \in \mathbb{Z}_p^*$ , is guaranteed to find some square root of  $s$ . (You already devised such an algorithm for  $p \equiv 3 \pmod{4}$  on HW2; a similar, though slightly more complex algorithm, exists for  $p \equiv 1 \pmod{4}$ .) We don't know anything about how the algorithm will behave if  $p$  is not prime—it will output some value, which may or may not be a square root of  $s$ .

Given  $A$ , devise a (randomized) polynomial-time algorithm  $T$  with the following properties. Given an integer  $n > 0$ ,

- If  $n$  is prime,  $T(n)$  always outputs “Prime”
- If  $n$  is composite,  $T(n)$  outputs “Composite” with probability at least  $1/2$  (the probability should be at least  $1/2$  not just for a *random*  $n$ , but for *every*  $n$ ).

You should, of course, explain why your algorithm works as specified above.

(c)  $T$  is a probabilistic primality test, but makes errors with probability up to  $1/2$ . How do you reduce the probability of error to  $1/2^{100}$ ?