Theory of Computation

Solutions to Homework 5

Problem 1. Let p, q be two distinct primes. Recall that the RSA function, shown on pages 551–558 in the slides, is $x^e \bmod pq$ for an odd e relatively prime to $\phi(pq)$. Show that the RSA function is not secure when q is restricted to be $p+2$. That is, given the binary representations of pq, e and $x^e \bmod pq$ as inputs, show how to compute $x \bmod pq$ in time polynomial in the input length, provided the following conditions hold:

1. $q = p + 2$.

2. p and q are distinct primes.

3. e is odd and relatively prime to $\phi(pq)$.

Proof. Knowing $q = p + 2$ and the value of pq, one is able to recover p, q and therefore $\phi(pq) = (p-1)(q-1)$ in polynomial time by a binary search for the value $x \in \{1, \ldots, pq\}$ satisfying $x(x+2) = pq$.

We separate the discussion into four cases according to value of $\gcd(x, pq) \in \{1, p, q, pq\}$, or equivalently, $\gcd(x^e, pq)$ as p, q are distinct primes.

1. $x^e \bmod pq$ is relatively prime to pq.

2. $x^e \bmod pq$ is a multiple of p but not a multiple of q.

3. $x^e \bmod pq$ is a multiple of q but not a multiple of p.

4. $x^e \bmod pq$ is zero.

In case one, one applies the Euclidean algorithm to find an integer d with $ed \equiv 1 \bmod \phi(pq)$. Now $x \bmod pq$ equals $x^{ed} \bmod pq$, which can be computed from $x^e \bmod pq$ by the method of recursive doubling.

In case two, $x^e \bmod q$ can be easily computed given $x^e \bmod pq$. Then one finds by the Euclidean algorithm an integer d with $ed \equiv 1 \bmod (q-1)$ and computes $x^{ed} \bmod q$ by recursive doubling. This finds $x \bmod q$ since it equals $x^{ed} \bmod q$. Now knowing $x \bmod q$ and $x \equiv 0 \bmod p$ and observing that p is relatively prime to q, one can find $x \bmod pq$ by applying the Euclidean algorithm as in the Chinese remainder theorem.

Case three is symmetric to case two.

Finally, in case four, pq divides x and $x \bmod pq$ is zero. □

Problem 2. Show that if SAT has no polynomial circuits, then coNP $\neq$ BPP. (Hint: Adleman's theorem states that all languages in BPP have polynomial circuits.)

Proof. Assume that SAT has no polynomial circuits. As all languages in BPP have polynomial circuits by Adleman's theorem, NP $\neq$ BPP. Hence

$$\text{coNP} \neq \text{coBPP} = \text{BPP}.$$

$\square$