Problem 1 – RSA Cryptography

1. [3] Suppose that we have a set of messages \( \langle m_1, \ldots, m_k \rangle \), encrypted with RSA under public key \( \langle e, n \rangle \). The corresponding ciphertexts are \( \langle c_1, \ldots, c_k \rangle \). Suppose further that someone tells us that one of the plaintext messages \( m_j \) shares a common factor with \( n \). Does this help an attacker to recover the plaintexts \( \langle m_1, \ldots, m_k \rangle \), given the ciphertexts \( \langle c_1, \ldots, c_k \rangle \) and the public key \( \langle e, n \rangle \)?

2. [2] Is it wise to send RSA-encrypted messages \( m \in \mathbb{Z}_n \setminus \mathbb{Z}_n^* \)? Hint: That’s corollary to the preceding problem.

3. [8] (Protocol Failure of RSA.) Suppose that the message \( m \) is sent to two users \( U_1 \) and \( U_2 \) with public keys \( \langle e_1, n \rangle \) and \( \langle e_2, n \rangle \), respectively. Call the two ciphertexts \( c_1 \) and \( c_2 \), both encrypting the same plaintext \( m \). Show that

   - Each user can obtain the other users private key.
   - In case that \( e_1 \) and \( e_2 \) are relatively prime, any attacker can recover \( m \) out of \( c_1 \) and \( c_2 \) and the two public keys.

Problem 2 – RSA Random Self Reducibility

Given a probabilistic polynomial time algorithm \( A(n, e, RSA(x, n, e)) \) that inverts the RSA-function on a polynomial subset \( T \subseteq \mathbb{Z}_n^* \), i.e., \( \Pr [A(n, e, RSA(x, n, e) = x)] > \epsilon \) for \( RSA(x, n, e) \in T \).

[15] Show that there exists a probabilistic polynomial time algorithm \( A'(n, e, RSA(x, n, e)) \) that inverts the RSA-function on all inputs with probability greater than \( 1 - \delta \) for any \( \delta > 0 \).
Problem 3 – P $\neq$ NP is not enough

Proving $P \neq NP$ would be a major break-through. However, even after such a proof a lot of questions would remain open which are of central interest. In particular, we will look at the complexity of FACTORING which is the computational problem of finding the prime factors for any given positive integer.

- Define the decision problem $FBIT$ with $(x, p) \in FBIT$ iff $x$ are a positive integer such that the $p$th bit of the largest prime-factor of $x$ is set to 1.

- Define $PRIMES$ as the language of all prime numbers – it is known that $PRIMES \in NP \cap coNP$.

1. [10] Prove the following statement: $P = NP \cap coNP$ implies $\text{FACTORING} \in \text{FP}$ ($\text{FP}$ is the class of functions which are computable within polynomial time).
   
   (a) Prove $\text{FBIT} \in \text{NP}$ and $\text{FBIT} \in \text{coNP}$ separately by using the assumption $\text{PRIMES} \in \text{P}$. (Hint: The two machines to prove this will be very similar)
   
   (b) Conclude that $\text{PRIMES} \in \text{P}$ implies $\text{FBIT} \in \text{NP} \cap \text{coNP}$.
   
   (c) Prove that $P = \text{NP} \cap \text{coNP}$ implies $\text{FACTORING} \in \text{FP}$.

2. [3] Based on this: What is the relationship of the statements $P \neq NP$, $P \neq NP \cap coNP$ and FACTORING $\not\in$ FP?

Problem 4 – Upward Translations

[5] Prove the following statement:

$$P = NP \Rightarrow EXP = NEXP$$

Problem 5 – $\text{DSPACE}(n) \neq \text{NP}$

Although we are currently unable to prove that either $\text{PSPACE} \neq \text{NP}$ or $\text{PSPACE} = \text{NP}$, we can show the following statement:

$$\text{DSPACE}(n) \neq \text{NP}$$

Note, that the inequality is the only relationship between $\text{DSPACE}(n)$ and $\text{NP}$ that we are able to prove.

[10] Prove that the two classes are different. Hint: In the lecture we proved that $\text{NP}$ is closed under log-space reductions. Show that $\text{DSPACE}(n)$ is not closed under log-space reductions and conclude that the two classes are in fact different.

[5] Can you apply the same proof-technique to other classes? For example can you prove that

---

1In fact, a more recent result shows $\text{PRIMES} \in \text{P}$. Beforehand, it was known that $\text{PRIMES} \in \text{ZPP}$, i.e., the class of randomized polynomial time algorithms, which are expected to produce a definite result within a constant number of trials.
The definitions are: $\text{EXP} = \bigcup_{c=1}^{\infty} \text{DTIME}(2^{cn})$ and $\text{E} = \bigcup_{c=1}^{\infty} \text{DTIME}(2^{cn})$.

[5] Can you generalize? Argue why $\text{P}$ is defined as the set of all decision problems which are solvable within polynomial time. Also, explain why $\text{EXP}$ is preferred over $\text{E}$.

A note on $\text{DSPACE}(n) \neq \text{NP}$: By the same argument, you can separate $\text{NSPACE}(n)$ from $\text{NP}$ and $\text{P}$ and other classical classes. The classes of nondeterministic linear-space Turing machines coincides with those languages that are recognizable by context sensitive grammars. Thus, the class of context sensitive languages is different from $\text{P}$, $\text{NP}$, $\text{PSPACE}$ . . .

Problem 6 – Existential Second Order Logic

1. Express the following properties in existential second order logic (ESO):
   
   • [6] SubgraphIsomorphism: given two graphs $G$ and $G'$, does $G$ contain a subgraph isomorphic to $G'$? The graphs are encoded as 2-ary relations.
   
   • [6] Sat. Encode a Sat instance by two 2-ary relations $P(x, y)$ and $N(x, y)$, determining whether the variable $x$ occurs in clause $y$ positively or negatively.
   
   • [6] NotAllEqualSat, which is a restriction of 3Sat requiring a special truth assignment such that in each clause of the formula, the truth values of all literals must not be identical.

2. [6] Show that NotAllEqualSat is NP–complete.


Problem 7 – BITGUESS

We define a new complexity class BITGUESS as the set of all problems that are solvable by the following special class of Turing machines, which works in two phases:

• First, the machine guesses polynomially many bits (with respect to the original input $x$) nondeterministically and writes these bits on a single tape.

• Afterwards, the machine uses the original input $x$ and the nondeterministically guessed bits to decide within LOGSPACE whether to accept or reject the input.

An input $x$ is accepted in this model, if the second phase accepts $x$ for at least one possible guess in the first phase.

Problem 8 – 0/1 Laws

1. [3] Express the 4-CLIQUE problem in first order logic.

2. [5] Based on this, express the $k$ – CLIQUE problem in first order logic.

3. [3] Remember from the lecture that first order logic has a 0/1 law. In consequence, the $k$ – CLIQUE problem is true for a large random graph with probability of 0 or 1 (as the graph size goes to infinity). Which one is it?