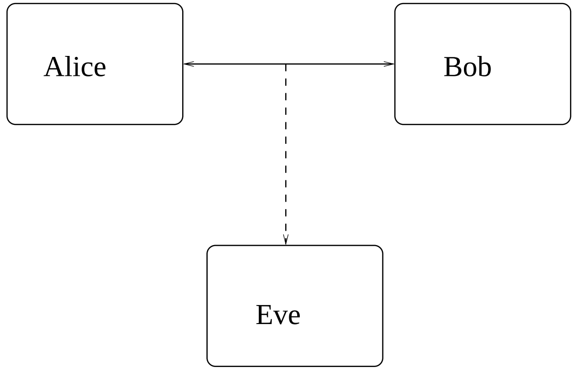
Complexity Theory
Lecture 10

Anuj Dawar

University of Cambridge Computer Laboratory
Easter Term 2010

http://www.cl.cam.ac.uk/teaching/0910/Complexity/

Cryptography



Alice wishes to communicate with Bob without Eve eavesdropping.

Private Key

In a private key system, there are two secret keys

e – the encryption key
d – the decryption key

and two functions D and E such that:

for any x,

$$D(E(x,e),d) = x$$

For instance, taking $d = e$ and both D and E as *exclusive or*, we have the *one time pad*:

$$(x \oplus e) \oplus e = x$$

One Time Pad

The one time pad is provably secure, in that the only way Eve can decode a message is by knowing the key.

If the original message x and the encrypted message y are known, then so is the key:

$$e = x \oplus y$$

Public Key

In public key cryptography, the encryption key e is public, and the decryption key d is private.

We still have,

for any x,

$$D(E(x, e), d) = x$$

If E is polynomial time computable (and it must be if communication is not to be painfully slow), then the function that takes $y = E(x, e)$ to x (without knowing d), must be in FNP.

Thus, public key cryptography is not *provably secure* in the way that the one time pad is. It relies on the existence of functions in FNP − FP.

One Way Functions

A function f is called a *one way function* if it satisfies the following conditions:

1. f is one-to-one.
2. for each x, $|x|^{1/k} \leq |f(x)| \leq |x|^k$ for some k.
3. $f \in \mathsf{FP}$.
4. $f^{-1} \notin \mathsf{FP}$.

We cannot hope to prove the existence of one-way functions without at the same time proving P ≠ NP.

It is strongly believed that the RSA function:

$$f(x, e, p, q) = (x^e \bmod pq, pq, e)$$

is a one-way function.

UP

Though one cannot hope to prove that the RSA function is one-way without separating P and NP, we might hope to make it as secure as a proof of NP-completeness.

Definition
A nondeterministic machine is *unambiguous* if, for any input x, there is at most one accepting computation of the machine.

UP is the class of languages accepted by unambiguous machines in polynomial time.

UP

Equivalently, UP is the class of languages of the form

$$\{x \mid \exists y R(x, y)\}$$

Where R is polynomial time computable, polynomially balanced, *and* for each x, there is *at most one* y such that $R(x, y)$.

UP One-way Functions

We have

$$\mathsf{P} \subseteq \mathsf{UP} \subseteq \mathsf{NP}$$

It seems unlikely that there are any NP-complete problems in UP.

One-way functions exist *if, and only if,* $\mathsf{P} \neq \mathsf{UP}$.

Space Complexity

We've already seen the definition $\mathsf{SPACE}(f(n))$: the languages accepted by a machine which uses $O(f(n))$ tape cells on inputs of length n. *Counting only work space*

$\mathsf{NSPACE}(f(n))$ is the class of languages accepted by a *nondeterministic* Turing machine using at most $f(n)$ work space.

As we are only counting work space, it makes sense to consider bounding functions f that are less than linear.

Classes

$\mathsf{L} = \mathsf{SPACE}(\log n)$

$\mathsf{NL} = \mathsf{NSPACE}(\log n)$

$\mathsf{PSPACE} = \bigcup_{k=1}^{\infty} \mathsf{SPACE}(n^k)$

The class of languages decidable in polynomial space.

$\mathsf{NPSPACE} = \bigcup_{k=1}^{\infty} \mathsf{NSPACE}(n^k)$

Also, define

co-NL – the languages whose complements are in NL.

co-NPSPACE – the languages whose complements are in NPSPACE.

Inclusions

We have the following inclusions:

$$\mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PSPACE} \subseteq \mathsf{NPSPACE} \subseteq \mathsf{EXP}$$

where $\mathsf{EXP} = \bigcup_{k=1}^{\infty} \mathsf{TIME}(2^{n^k})$

Moreover,

$$\mathsf{L} \subseteq \mathsf{NL} \cap \text{co-}\mathsf{NL}$$

$$\mathsf{P} \subseteq \mathsf{NP} \cap \text{co-}\mathsf{NP}$$

$$\mathsf{PSPACE} \subseteq \mathsf{NPSPACE} \cap \text{co-}\mathsf{NPSPACE}$$

Establishing Inclusions

To establish the known inclusions between the main complexity classes, we prove the following.

- $\mathsf{SPACE}(f(n)) \subseteq \mathsf{NSPACE}(f(n))$;
- $\mathsf{TIME}(f(n)) \subseteq \mathsf{NTIME}(f(n))$;
- $\mathsf{NTIME}(f(n)) \subseteq \mathsf{SPACE}(f(n))$;
- $\mathsf{NSPACE}(f(n)) \subseteq \mathsf{TIME}(k^{\log n + f(n)})$;

The first two are straightforward from definitions.

The third is an easy simulation.

The last requires some more work.