Homework 25

Joe Baker, Brett Schreiber, Brian Knotten

March 22, 2018

44

a

Given a one-way function f, it is possible to create a new one-way function g which runs in $O(n^2)$ time as follows:
On input x of size n:

Split the input x into $log(n)$ chunks: $x_1, x_2 ... x_{log(n)}$.

Return $f(x_1)||f(x_2)...||f(x_{log(n)}$ where $||$ is the concatenation of the bitstrings.

First, g runs in $O(n^2)$ time, because f performing $log(n)$ computations. So g is the complexity of f multiplied by $log(n)$. Assuming f runs in at least linear time, the $log(n)$ multiplier is negligible.

b

f is one way $\Rightarrow f_U$ is one-way. This can be proved by contrapositive, that f_U is not one-way $\Rightarrow f$ is not one way. Assume f_U is not one-way. Then there exists an algorithm A_U which given y can produce the x such that $f_U(x) = y$ in polynomial time. Then you can construct an algorithm A which given y can produce the x such that $f(x) = y$ in polynomial time.

$A =$ on input x:

1. Generate $r =$ some number of random bits.
2. Construct the string $x' := x||r$.
3. Run A on x' to get y.
4. If $f(y) = x$, return y, else, go back to step 1.

A will halt in polynomial time because

45