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: $x_1, x_2...x_{log(n)}$.
for each x_i:
Compute $f(x_i)$, keeping track of the number of steps f takes. After n^2 steps, just output 0.
Return $f(x_1)||f(x_2)...||f(x_{log(n)}$ where $||$ is the concatenation of the bitstrings.
Some of these substrings will be 0.

First, g runs in $O(n^2)$ time, because f performing $log(n)$ computations. So g is the complexity of f multiplied by $log(n)$. Since we stop f after n^2 steps, the total runtime is $n^2 * log(n) = O(n^2)$.

Second, g is a one way function, since $g = f_U$, and f_U is one-way as proved below.

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 y:

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

A will halt in polynomial time because

45