EE 374: Blockchain Foundations **Stanford, Winter 2023**

Theory Exercise 1

Jan 31, 2023

Problem 1

Let $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ be a collision-resistant hash function and define $G(x) = H(H(x))$. Use a reduction to show that G is a collision-resistant hash function.

Problem 2

Given a collision resistant hash function $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$, create a hash function $G : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ with the following properties:

1. G is collision resistant.
2. The first two bits of G are predictable.

Write out explicitly what this G function is, then define and prove the above two properties formally. For the first property, use a reduction.

Problem 3

Construct a correct but insecure signature scheme. Write the full implementation of your signature scheme in pseudocode.

Problem 4

Consider the complete transaction graph ***accepted and stored in the database of an honest node*** illustrated in Figure 1. The circles represent transactions and the numbers within them are the txids. The outgoing edges are outputs, and the incoming edges are inputs. For each transaction, edges appear ordered from top to bottom. For this graph:

1. Identify the coinbase transactions.
2. Identify the double spending transactions. For each double spending transaction, identify the transaction it is conflicting with.
3. Identify the UTXO set.
4. Identify the transactions for which the Weak Conservation Law does not hold.
5. Identify the outputs of transactions that are partially, but not fully, spent.

To identify transactions, use their txids. To identify an output, use outpoint notation. Note that transaction 6 has four outputs.

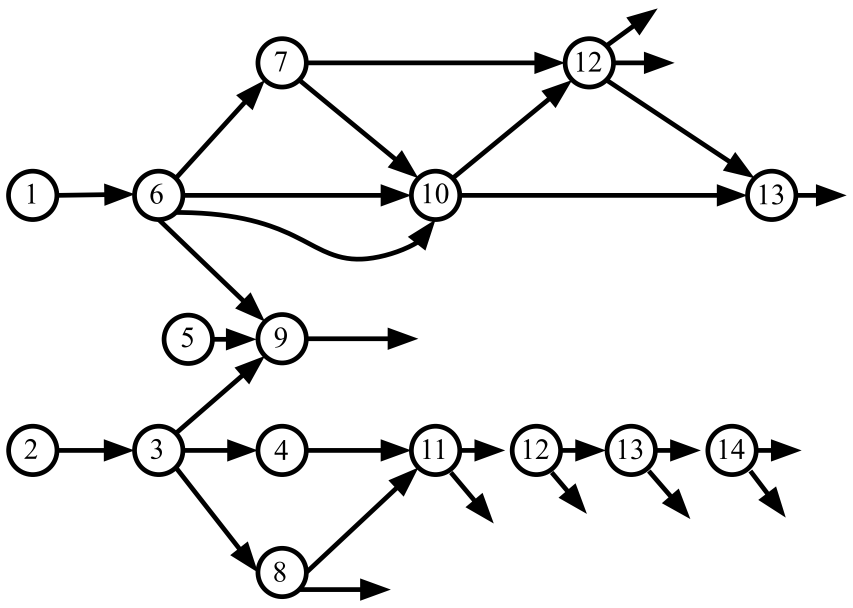

Figure 1: The transaction graph of Problem 4.

Problem 5

The honest proof-of-work algorithm begins by sampling a uniformly random κ-bit nonce, then iterates for a polynomial number of repetitions while incrementing the nonce. Prove that, if a constant number of honest parties execute this algorithm a polynomial number of times, the probability that they query the hash using the same nonce is negligible.

Reference

Algorithm 1 The collision-finding game for a hash function H.

```
1: function COLLISION-GAME$_{H,\mathcal{A}}(\kappa)$
2:     $x_1, x_2 \leftarrow \mathcal{A}(1^\kappa)$
3:     return $H_\kappa(x_1) = H_\kappa(x_2) \wedge x_1 \neq x_2$
4: end function
```

Algorithm 2 The existential forgery game for a signature scheme (Gen, Sig, Ver).

```
function existential-forgery-game$_{Gen,Sig,Ver,\mathcal{A}}(\kappa)$
    $(pk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
    $M \leftarrow \emptyset$
    function $\mathcal{O}(\mathrm{m})$
        $M \leftarrow M \cup \{m\}$
        return $\mathsf{Sig}(sk, m)$
    end function
    $m, \sigma \leftarrow \mathcal{A}^{\mathcal{O}}(pk)$
    return $\mathsf{Ver}(pk, \sigma, m) \wedge m \notin M$
end function
```

Algorithm 3 The proof-of-work algorithm.

```
function $\textsc{PoW}_{H,T}$
    $\mathsf{ctr} \overset{\$}{\leftarrow} \{0, 1\}^\kappa$
    while true do
        $B \leftarrow \mathsf{ctr}$
        if $H(B) \le T$ then
            return B
        end if
        $\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$
    end while
end function
```