Blockchain Foundations
Midterm
Date TBD

1. The exam has 4 questions with a total of 100 points. You have 80 minutes to take the exam. Questions have different numbers of points so please allocate your time to each question accordingly.
2. Please write the answer in the designated area underneath each question. If you need more room for your answer, please indicate as such, and continue your response on the blank page at the end of all questions. No additional pages will be allowed.
3. Scratch paper will be provided and collected at the end of the exam, but will not be graded.
4. All answers should be justified, unless otherwise stated.
5. The exam is closed book but you are allowed one double-sided sheet of notes. A list of variables and reference information is provided at the end of the exam. No other materials are allowed.

Good luck!

Name:
SUID:

(22 points) Problem 1

For the following questions, choose the one most fitting answer among the four choices. No justifications are required for this problem. 2 points for a correct answer, 0 points for an incorrect answer. 0.5 point for leaving the answer blank. Knowing you don't know something has value.

1. In a system with constant n, t, Δ, q, which reconfiguration increases the density of honestly successful queries in the unit of time?

   (a) Increase k

   (b) Decrease k

   (c) Increase T

   (d) Decrease p

   (c)
   Increasing T increases p which results in more successful queries by honest parties.

2. If an adversary alters the public key of a coinbase transaction of a proof-of-work protocol in transit during gossiping, this will invalidate:

   (a) The signature on the coinbase transaction

   (b) The signature on the block

   (c) The suitability of the block nonce

   (d) The weak conservation law

   (c)
   Changing the coinbase will affect the block hash. That makes the block nonce fail PoW.

3. In the UTXO model, an input of a transaction tx contains an outpoint that always points to:

   (a) A coinbase transaction tx′

   (b) A tuple (blockid, index) of the previous block on the chain

   (c) The output of a different transaction tx′

   (d) The output of a different transaction tx′, or an output of that same transaction tx

   (c)
   All transaction inputs are sourced from the outputs of other transactions in the

UTXO model. The same transaction cannot point to itself, as it needs to know its own hash.

4. At the epoch boundary, if the difference between the timestamp recorded on the tip and the timestamp recorded m blocks ago is larger than the expected epoch time duration:

   (a) The parameter p is indirectly increased by adjusting the target.

   (b) The parameter Δ is decreased so that more blocks have sufficient time to be transmitted across the diameter of the gossiping network.

   (c) The parameter k is decreased to maintain the same liveness level, since successful queries are becoming rarer.

   (d) A request is placed towards the honest participants to increase their q value so that mining power is appropriately increased to keep f constant.

(a)
The idea of adjusting T depending on how fast blocks are being mined is known as "variable difficulty." The values Δ and q cannot be changed, as they are provided to us from the model. Decreasing k will make the system insecure.

5. A correct signature scheme must necessarily:

   (a) Ensure the adversary cannot guess the secret key with non-negligible probability.

   (b) Ensure the adversary cannot forge a signature on a new message that she did not request from the oracle except with negligible probability.

   (c) Ensure the honest party can verify honestly produced signatures on messages that start and end with the 0 bit.

   (d) Ensure that the key generation function is collision-resistant.

(c)
The correctness of a signature scheme deals with the verifiability of *honest* signatures. Correctness is not security.

6. When a rational miner holds a mempool that is smaller than the block size limit:

   (a) He includes all the transactions of his mempool into the new block.

   (b) He waits for the mempool to fill up before he begins mining a block, to save on mining resources.

   (c) He includes only the transactions with the top fee/byte score into the new block.

   (d) He places half of the transactions in the current new block, and saves the rest for the next block to minimize the variance of his profits.

(a)
Since the whole mempool fits into the block, including everything is beneficial to maximize fees gained.

7. If the recent chunks of an honestly adopted chain have small chain quality:

   (a) Ledger safety has been violated.

   (b) Common Prefix has been violated for sufficiently large values of k.

   (c) Chain Growth must have recently dropped to a velocity of $\tau = 0$.

   (d) Ledger liveness may be deteriorating.

(d)
The lack of recent honest blocks can indicate that honest transactions aren't getting confirmed. Hence, ledger liveness may deteriorate.

8. How can a non-temporary probabilistic polynomial majority adversary spend an honest party's money?

   (a) By issuing coinbase transactions that violate the system's macroeconomic policy.

   (b) By issuing double spending transactions that spend the honest party's money, while mining using the Nakamoto Race to ensure a ledger safety violation.

   (c) By creating a transaction that spends the honest party's money and ensuring it is confirmed in a timely manner.

   (d) She cannot.

(d)
Spending an honest party's money with non-neglibible probability requires access to their secret key.

9. The fan-out situation, in which numerous forks are created even though there is an honest majority, occurs in a proof-of-work system when:

   (a) The network delay Δ is extremely large.

   (b) The system is misconfigured with a too high target T.

   (c) The system is misconfigured with a too high query success probability p.

   (d) All of the above.

(d)
All are conditions that increase the chance of a fan-out situation.

10. If the k Common Prefix chain virtue is violated in a proof-of-work longest chain protocol, we can deduce that:

(a) Ledger safety has been violated.

(b) Ledger liveness has been violated.

(c) Chain quality of recent chunks has been reduced to $\mu = 0$.

(d) None of the above.

(d)
A Common Prefix violation *may* lead to a ledger safety violation, but this is not necessary, as the blocks can be empty.

11. A majority selfish miner, over the long run, can achieve a chain quality of:

(a) $\mu = 0$

(b) $\mu = \frac{1}{3}$

(c) $\mu = \frac{1}{2}$

(d) $\mu = 1$

(a)
Simply by mining on their own chain without regards to others, a majority selfish miner can achieve a chain quality of $\mu = 0$ in the long run.

(23 points) Problem 2

Let $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ be a collision-resistant hash function. Define $G : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ to be the hash function obtained by invoking $H(x)$, flipping the last bit of $H(x)$, and then returning the result. Namely, $G(x) = H(x)[:-1] \,\|\, (H(x)[-1] \oplus 1)$. Examine whether the function G is collision-resistant.

Hint: If you choose to prove collision resistance, use a computational reduction. We suggest that you write out the exact pseudocode of the reduction and calculate the relationship of the probabilities of success between the adversaries. If you choose to disprove collision resistance, construct a pathological case in which H is collision-resistant, but G is not collision-resistant, and prove the latter assuming the former.

We prove that G is also collision resistant. For the sake of contradiction, assume that G weren't collision resistant. Then, there exists a PPT adversary $\mathcal{A}$ which can find $x_1 \neq x_2$ so that $G(x_1) = G(x_2)$ with non-negligible probability in κ. Construct a PPT adversary $\mathcal{A}'$ as follows:

Algorithm 1 Adversary $\mathcal{A}'$ that breaks H.

```
function A'(1^κ)
    x_1, x_2 ← A(1^κ)
    return (x_1, x_2)
end function
```

If $\mathcal{A}$ is successful in finding $x_1 \neq x_2$ for which $G(x_1) = G(x_2)$, then $H(x_1) = H(x_2)$ since $H(x)$ can be fully determined given $G(x)$ simply by reversing the last bit. The same goes for when $\mathcal{A}$ is unsuccessful - if $G(x_1) \neq G(x_2)$, then $H(x_1) \neq H(x_2)$. As a result, $\mathcal{A}'$ has the same non-negligible probability of success in breaking H as $\mathcal{A}$ does in breaking G. This contradicts the collision resistance of H, as desired. ■

(25 points) Problem 3

We are working in a UTXO longest chain system with a block reward of 50 units and a confirmation rule of $k = 6$. Consider the transaction graph illustrated in Figure 1. Transaction ids are displayed in the circles, values of outputs are written above the outputs, and owners of outputs are written below the outputs. Any outputs with no owners indicated belong to the adversary. Transactions with no inputs are coinbase transactions. For the coinbase transactions, you can assume that no "height" is needed to be included in them for validity, and that there is no maturation restriction, so they can be spent within the same block. In the questions below, please denote your outpoints in $(\mathsf{txid}, \mathsf{index})$ notation.

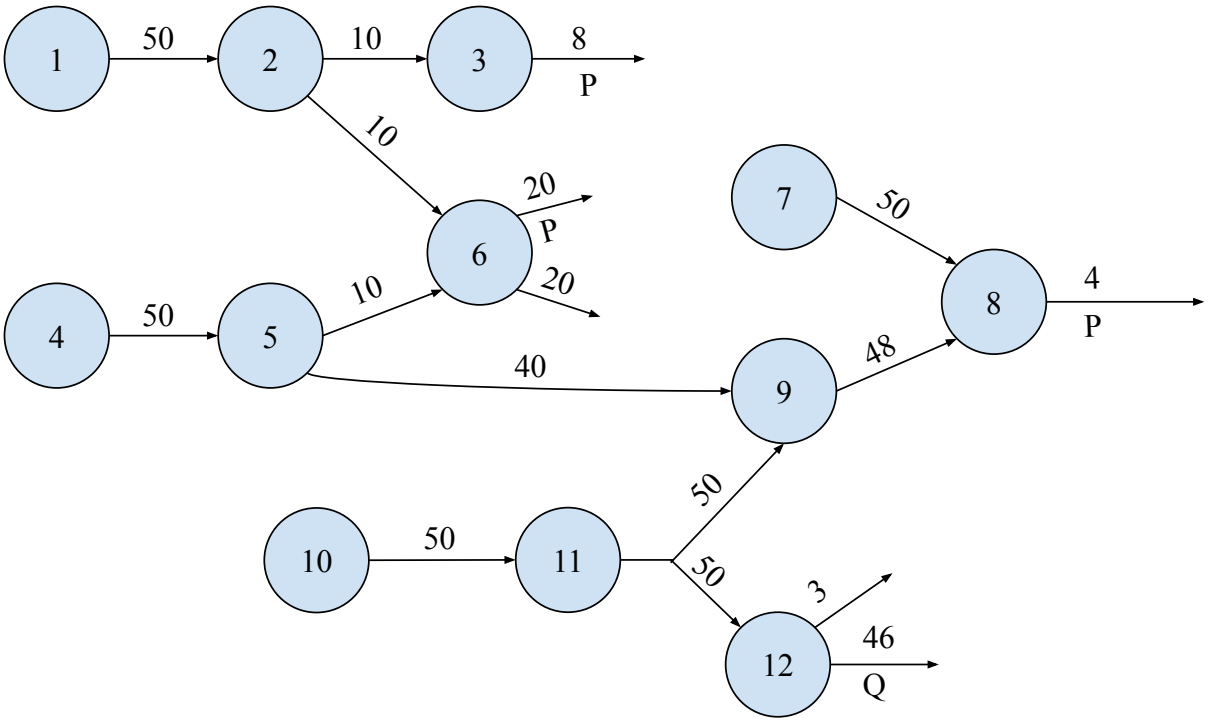


Figure 1: The transaction graph.

Remark. The second input to transaction 9 and the first input to transaction 12 are the same outpoint $(11, 0)$.

1. (7 points) Honest party P has adopted a chain $C_P = (\mathcal{G}, B_1, B_2, B_3)$. Genesis has no transactions, block B_1 only contains transaction 1, block B_2 only contains transaction 4, and block B_3 only contains transaction 10. Party P receives the transactions in the order 2, 3, 5, 6, 7, 8, 9, 11, 12 from the network. The party does not receive any new blocks for now. According to the state derived from P's new mempool, what is the UTXO set and how much money do P and Q have respectively?

Party P's mempool will consist of transactions $2, 3, 5, 11, 12$. Transaction 6 is missing because it violates the conservation law. Transaction 7 is missing because it is a coinbase transaction not attached to a block. Transaction 8 is missing because the outpoint from 7 wasn't valid. Transaction 9 is missing because the outpoint from 11 doesn't exist yet.

The UTXO set is

$$\{(2,1),(3,0),(5,0),(5,1),(12,0),(12,1)\}$$

P will have 8 units from outpoint $(3,0)$, and Q will have 46 units from outpoint $(12,1)$.

2. (7 points) A rational party R (who may play dishonestly and is trying to maximize his profits) has also adopted the same chain $C_R = C_P$ and attempts to mine a block on top of $C_R[-1]$, having seen all the 12 transactions in the graph. The block size limit is such that up to 5 transactions (1 coinbase + 4 other transactions) can fit per block. Which transactions will R choose to include in the block he creates? According to the state derived from R's newly mined block B_4, how much money does R have?

To maximize profits, R will pick the transactions $7, 5, 11, 9, 8$ in that order to maximize transaction fees. Even though 7 is a coinbase transaction paying the adversary, it is still possible for R to include it (without changing the output). This is beneficial, since a later transaction 8, spending 7, pays very highs fees. The transaction fees sum to $0+0+0+42+94 = 136$ units. Hence, R will end up with 136 units in total. This is the maximum profit achievable in this situation.

3. (5 points) After block B_4 is mined, another 10 *empty* blocks get mined and broadcast sequentially by the adversary on top of B_4 and P adopts the chain C'_P containing all of them. What is the k-confirmed ledger according to P?

The k-confirmed ledger consists of the transactions in blocks that have at least k blocks proceeding them in the longest chain. In this case, the k-confirmed blocks are $\mathcal{G}, B_1, B_2, B_3, B_4$ and 4 empty blocks. Hence, in order, $L^P = (1, 4, 10, 7, 5, 11, 9, 8)$.

4. (6 points) Assuming no further blocks or transactions have arrived in the meantime, what is P's new mempool now?

P's mempool will only contain transactions 2 and 3. Formerly, P's mempool had transactions $2, 3, 5, 11, 12$. Transactions $5, 7, 8, 9, 11$ were included in B_4, so they will no longer appear in P's mempool. This leaves transactions 2, 3 and 12. However, transaction 12 would be a double spend of outpoint $(11, 0)$ since B_4

included transaction 9. This leaves us with transactions 2 and 3, in that order.

(30 points) Problem 4

Consider the blocktree illustrated in Figure 2. The picture shows all the blocks that have been successfully mined in a static difficulty proof-of-work longest chain protocol during the time interval $[0, 31]$. The chains C_1, C_2, C_3, C_4, C_5 are marked at their tips on the diagram. Block $C_1[0]$ is the genesis block. Blue blocks have been mined by an honest party, whereas red blocks have been mined adversarially. Within each box, the time at which the block was mined is displayed. Consider a scenaro where the network delay is $\Delta = 1.5$, there are $n = 5$ parties and $q = 1000$ is the mining power. We are using a $\kappa = 256$ bit function to mine. The adversary is constantly mining with her maximum mining power. The confirmation and Common Prefix parameter is $k = 2$.

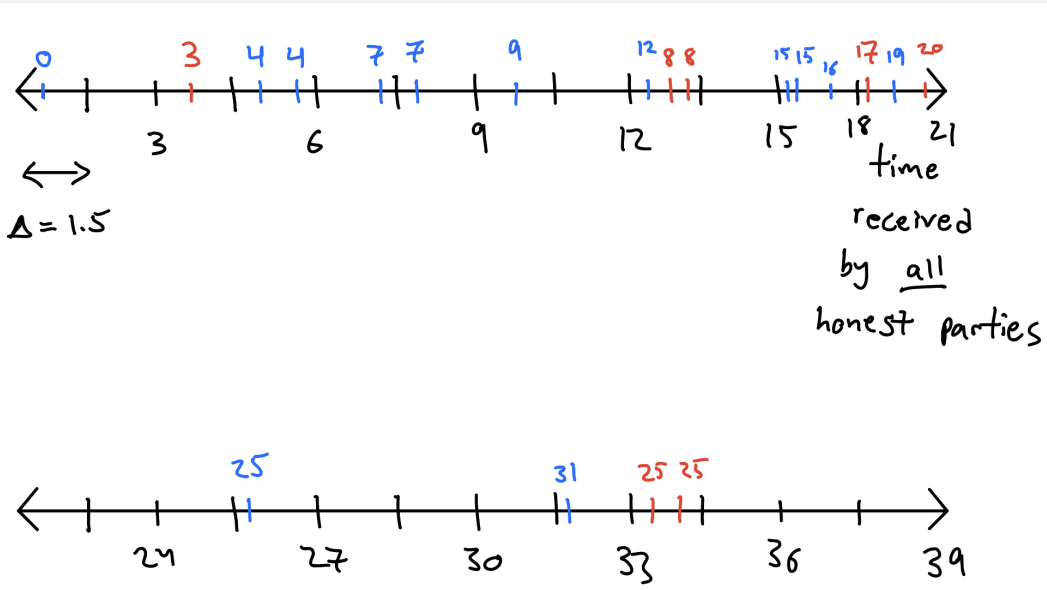
1. (4 points) How many successful queries have occurred? Of those, which ones are convergence opportunities?

   There are 20 blocks in the blocktree. Hence, 20 successful queries occurred. Of them, only blocks at times $0, 9, 12, 19, 25$ are convergence opportunities, because they are the only honestly mined blocks which are Δ-separated from all other honestly mined blocks. 31 could also be accepted as a convergence opportunity if we assume no block was mined within Δ delay afterwards.

2. (4 points) Calculate the chain quality of C_1 and C_5.

   Chain C_1 has 3 honest blocks and 1 adversarial block, resulting in a chain quality of $\frac{3}{4}$. Chain C_5 has 6 honest blocks and 6 adversarial blocks, resulting in a chain quality of $\frac{6}{12} = \frac{1}{2}$.

3. (7 points) Draw an example timeline during which Common Prefix with $k = 2$ was violated. In this timeline, highlight when each party received each block. Write out the concrete predicate describing Common Prefix that was violated. At which times was it violated, and between the chains of which parties? Why had the particular honest parties adopted these tips at the particular times, and which other chains had they seen at those times?



There are many possible valid timelines. Any timeline that respects the network delay of $\Delta = 1.5$, and the fact that honest parties build on top of the longest chain will work.

Consider the two honest parties P_1, P_2 that mined blocks at time 15. Common Prefix for $k = 2$ will be violated between the two honest parties that mined blocks at time 15. They will disagree on whether C_2 or $C_5[:7]$ is the longest chain. As a result, they will also disagree on whether their second to last block in their longest chain was queried at time 8 or 9. In particular,

$$C_2[:-2] \not\preceq C_5[:7],$$

violating Common Prefix for $k = 2$.

4. (7 points) For the above Common Prefix violation, describe an example with the ledgers and transactions adopted by the honest parties in which safety was violated. Write out the concrete predicate describing safety that was violated. During which times was safety violated and among which honest parties? What was the ledger of each honest party at those times?

Consider the two honest parties P_1, P_2 that mined blocks at time 15. Let party P_1's ledger consists of the transactions in $C_2[:5]$, and party P_2's ledger consists of the transactions in $C_5[:5]$. Assume that the blocks mined at time 8 and time 9 are non-empty and contain completely different transactions. Then, neither

$$L_{15}^{P_1} \preceq L_{15}^{P_2}$$

nor

$$L_{15}^{P_2} \preceq L_{15}^{P_1},$$

resulting in a ledger safety violation at time 15.

5. (8 points) Based on the observed blocktree, give your best estimation for the values of t and T. You should calculate the exact values for t and T by plugging in the numbers, but you don't need to do the final numerical calculation if it requires a calculator.

7 out of the 20 blocks are adversarial. Hence, we expect roughly $\frac{2}{5}$ of the parties to be adversarial. Since $n = 5$, we expect $t = 2$.

Now, we compute T. There are 20 blocks mined in 31 units of time. Hence, roughly $\frac{20}{31}$ blocks are mined per unit time. We expect there to be $p(nq)$ blocks to be mined per unit time as well. As a result, $pnq = \frac{20}{31}$ which means

$$\frac{T}{2^{256}} = p = \frac{20}{31nq} = \frac{20}{31(5)(1000)} = \frac{1}{7750}.$$

Solving this equation gives $T = 2^{256-\log_2(7750)} \approx 2^{243}$.

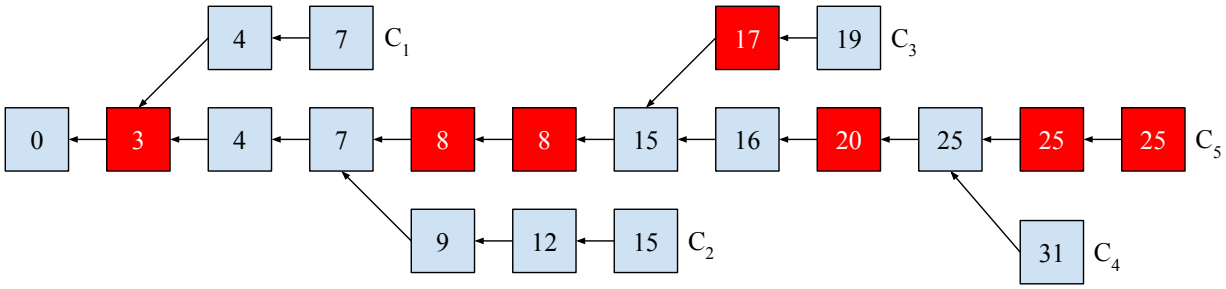


Figure 2: The blocktree graph.

Extra page for answers

Reference

Some helpful definitions are provided below.

Definition 1 (Collision Resistance)**.** *A hash function* $H : \{0,1\}^* \longrightarrow \{0,1\}^\kappa$ *is* collision resistant *if for all PPT adversaries* $\mathcal{A}$*,*

$$\Pr[\textit{collision-game}_{H,\mathcal{A}(\kappa)} = 1] = \mathrm{negl}(\kappa)\,.$$

The game is defined in Algorithm 2.

Algorithm 2 The collision-finding game for a hash function H.

```
1: function COLLISION-GAME_{H,A}(κ)
2:     x_1, x_2 ← A(1^κ)
3:     return H_κ(x_1) = H_κ(x_2) ∧ x_1 ≠ x_2
4: end function
```

Definition 2 (Correct Signature)**.** *A signature scheme* (*Gen*, *Sig*, *Ver*) *is* correct *if, for all* $m \in \{0,1\}^*$*, whenever* $(sk, pk) \leftarrow \textit{Gen}(1^\kappa)$*, we have that* $\textit{Ver}(pk, m, \textit{Sig}(sk, m)) = 1$*.*

Definition 3 (Secure Signature)**.** *A signature scheme* (*Gen*, *Sig*, *Ver*) *is* secure *if for all PPT adversaries* $\mathcal{A}$*,*

$$\Pr[\textit{existential-forgery-game}_{(\textit{Gen},\textit{Sig},\textit{Ver}),\mathcal{A}}(\kappa) = 1] = \mathrm{negl}(\kappa)\,.$$

The game is defined in Algorithm 3.

Algorithm 3 The existential forgery game for a signature scheme (Gen, Sig, Ver).

```
1: function existential-forgery-game_{(Gen,Sig,Ver),A}(κ)
2:     (pk, sk) ← Gen(1^κ)
3:     M ← ∅
4:     function O(m)
5:         M ← M ∪ {m}
6:         return Sig(sk, m)
7:     end function
8:     m, σ ← A^O(pk)
9:     return Ver(pk, σ, m) ∧ m ∉ M
10: end function
```

Definition 4 (Weak Conservation Law)**.** *A transaction* tx *satisfies the Weak Conservation Law if*

$$\sum_{i \in \mathsf{tx}.\textit{ins}} i.v \geq \sum_{o \in \mathsf{tx}.\textit{outs}} o.v\,.$$

Definition 5 (Velocity)**.** *The* velocity τ *of a chain of an honest party* P *between times* $r_1 < r_2$ *is the ratio* $\frac{|\mathcal{C}^P_{r_2}| - |\mathcal{C}^P_{r_1}|}{r_2 - r_1}$.

Definition 6 (Common Prefix)**.** *The Common Prefix virtue, parametrized by* $k \in \mathbb{N}$*, is satisfied if for all honest parties* P_1, P_2 *and for all times* $r_1 \leq r_2$*, it holds that* $\mathcal{C}^{P_1}_{r_1}[:-k] \preceq \mathcal{C}^{P_2}_{r_2}$.

Definition 7 (Chain Quality)**.** *The Chain Quality of a chain chunk* $\mathcal{C}$ *is defined as the ratio of the honestly produced blocks divided by the total blocks within that chain chunk.*

Definition 8 (Ledger Liveness (informal))**.** *A ledger system is* live *if, whenever an honest party attempts to inject a transaction to the ledger, then this transaction will make it to all honest ledgers "soon".*

Definition 9 (Ledger Safety)**.** *A ledger system is* safe *if, for all honest parties* P_1, P_2 *and for all times* r_1, r_2*, it holds that* $L^{P_1}_{r_1} \preceq L^{P_2}_{r_2}$ *or* $L^{P_2}_{r_2} \preceq L^{P_1}_{r_1}$.

Our variables.

- κ: The security parameter
- $\mathcal{A}$: The uniform PPT adversary
- Π: The honest protocol
- H: The hash function
- $\mathcal{G}$: The genesis block, an *honestly* mined reference block with 0 height
- Δ: The maximum network delay
- T: The mining target
- p: The probability of a successful query
- n: The total number of parties (includes both honest and adversarial)
- t: The number of adversarial parties
- q: The hashing power of a single party per unit of time
- k: The Common Prefix parameter, in blocks
- μ: The Chain Quality parameter, as a proportion
- τ: The velocity, in blocks per unit of time
- m: Epoch duration, in blocks

Terminology.

- The proof-of-work equation: $H(B) \leq T$.

- A *query* is a fresh call to the hash function H.
- A *successful query* is a fresh (honest or adversarial) query to the random oracle H that satisfies the proof-of-work equation.
- A *convergence opportunity* is an *honest* successful query which is spaced at least Δ apart from all other *honest* successful queries. The genesis block was computed during a convergence opportunity.
- A *negligible function* is eventually smaller than all inverse polynomials.
- A block tree has the *Common Prefix* virtue with parameter k if, for any two chains C_1, C_2 currently adopted by honest parties, $C_1[:-k]$ is a prefix of C_2.
- The *k-confirmation rule* reports the transactions in $C[:-k]$ as confirmed.
- The xor operator $b_1 \oplus b_2$ between two bits b_1 and b_2 returns 1 iff $b_1 \neq b_2$.

Algorithms.

Algorithm 4 The mining algorithm.

```
1: function MINE(s, x̄)
2:     ctr ←$ {0,1}^κ
3:     while true do
4:         B ← s ‖ x̄ ‖ ctr
5:         if H(B) ≤ T then
6:             return B
7:         end if
8:         ctr ← ctr + 1
9:     end while
10: end function
```

Chain addressing notation.

- $|\mathcal{C}|$: Chain length
- $\mathcal{C}[i]$: i^{th} block in the chain (0-based). The block height is i.
- $\mathcal{C}[-i]$: i^{th} block from the end.
- $\mathcal{C}[0]$: Genesis (by convention honest).
- $\mathcal{C}[-1]$: The tip.
- $\mathcal{C}[i{:}j]$: Chain chunk from block i (inclusive) to j (exclusive).
- $\mathcal{C}[:j]$: Chain chunk from the beginning and up to block j (exclusive).
- $\mathcal{C}[i:]$: Chain chunk from block i (inclusive) onwards.
- $\mathcal{C}[:-k]$: The stable chain.