

# Blockchain Technologies

## Alternative Consensus



# PROOF OF WORK REVIEW

- In Bitcoin, miners use their computing resources to solve a cryptographic puzzle.
- The puzzle is:  $\text{Hash}(\text{Prv}, \text{Tx}, \text{nonce}) < \text{threshold}$
- It is difficult to solve, but not too hard!
- The probability of a minor to become a winner is proportional to its computing power.
- It tries to prevent centralization with the assumption of distributed processing powers around the world.

# THE FIRST PAPER: MOTIVATED BY SPAM EMAILS!

## Pricing via Processing or Combatting Junk Mail

Cynthia Dwork and Moni Naor

IBM Almaden Research Center  
650 Harry Road  
San Jose, CA 95120

@ 1993




**Abstract.** We present a computational technique for combatting junk mail in particular and controlling access to a shared resource in general. The main idea is to require a user to compute a moderately hard, but not intractable, function in order to gain access to the resource, thus preventing frivolous use. To this end we suggest several *pricing functions*, based on, respectively, extracting square roots modulo a prime, the Fiat-Shamir signature scheme, and the Ong-Schnorr-Shamir (cracked) signature scheme.

To send a message  $m$  at time  $t$  to destination  $d$ , the sender computes  $y = f_s(h(m \cdot t \cdot d))$  and sends  $y, m, t, d$  to  $d$ . The recipient's mail program verifies that  $y = f_s(h(m \cdot t \cdot d))$ . If the verification fails, or if  $t$  is significantly different from the current time, then the message is discarded and (optionally) the sender is notified that transmission failed. If the verification succeeds and the message is timely, then the message is routed to the reader.

# PROOF OF WORK IS POWERFUL!

- It works very well in providing security!
- It is very good at preventing DoS attacks.
- It is the main technique in guaranteeing Bitcoin's security by steering the incentives of the nodes.

# PROOF OF WORK HAS SOME DRAWBACKS

- Electricity consumed by PoW.
- Hardware for PoW is not useful in any other way.
- Centralization treats.  mining pools

# ALTERNATIVE CONSENSUS

- Alternative consensus: other methods of verification other than proof of work (PoW).
- Possible goals:

ASIC Resistance

Pool Resistance

Prevention of Waste of Energy

...

- One should be careful about the security requirements.



# Overview of Some Alternatives to Proof-of-Work

# WHAT IS A GOOD PUZZLE?

- Cheap to verify
- Adjustable difficulty (adaptation to network size)
- No shortcuts available to the solution
- Small players should have incentive to participate.



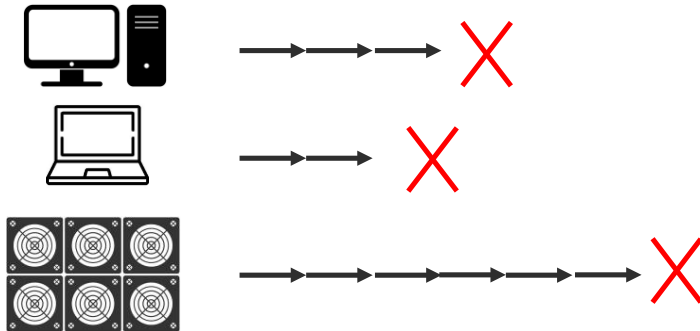
# ASIC RESISTANCE PROPERTY

- Why?: let ordinary people mine on their laptop or mobile phone.
- Making the eco-system as distributed as possible.
- Industrial miners treat the security.
- For a good puzzle, ASIC performance should not be much better than general purpose hardware.
- Be careful about future ASIC improvements.



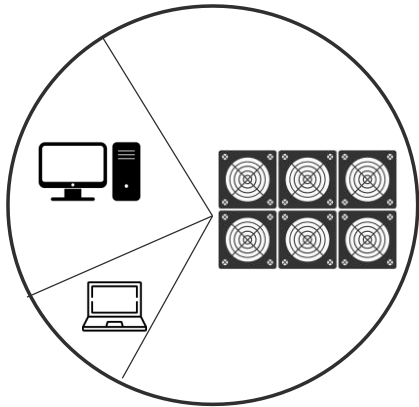
# A BAD PUZZLE EXAMPLE: SEQUENTIAL PROOF OF WORK

- This will result in the powerful miner to always win.



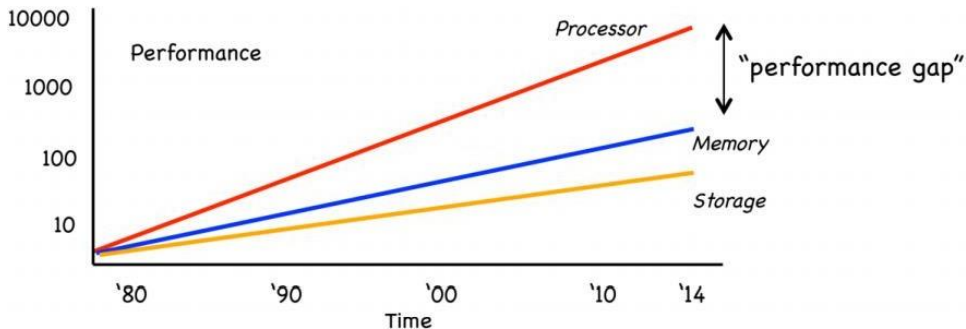
# A GOOD EXAMPLE: WEIGHTED SAMPLING

- The dart target portion is proportional to the processing power.



# PROOF OF MEMORY

- Memory-hard puzzles are more hardware-improvement resistant than computation-hard puzzles.



# A MEMORY-HARD PUZZLE EXAMPLE: SCRIPT

- It is a memory-hard hash function
- Constant Time-Memory Tradeoff
- Used in Litecoin, Dogecoin, Gridcoin, Auroracoin

## STRONGER KEY DERIVATION VIA SEQUENTIAL MEMORY-HARD FUNCTIONS

@ 2009

COLIN PERCIVAL

**ABSTRACT.** We introduce the concepts of memory-hard algorithms and sequential memory-hard functions, and argue that in order for key derivation functions to be maximally secure against attacks using custom hardware, they should be constructed from sequential memory-hard functions. We present a family of key derivation functions which, under the random oracle model of cryptographic hash functions, are provably sequential memory-hard, and a variation which appears to be marginally stronger at the expense of lacking provable strength. Finally, we provide some estimates of the cost of performing brute force attacks on a variety of password strengths and key derivation functions.



# SCRIPT: STEP 1

- Input:  $x$        $H(.)$ : SHA2

$$V_1 = H(x)$$

$$V_2 = H(V_1) = H(H(x))$$

$$V_3 = H(V_2) = H^3(x)$$

$\vdots$

$$V_N = H(V_{N-1}) = H^N(x)$$

$V_1$	$V_2$	$V_3$			
		$\dots$			
					$V_N$

$$N = 36$$

## SCRIPT: STEP 2

- Input:  $x$

$$A = H^{N+1}(x)$$

- For  $N$  iterations

$$i = A \bmod N < \boxed{36}$$

$$A = H(A \text{ XOR } V_i)$$

- Output  $A$

 index

good idea: store last ones instead of

$V_1$	$V_2$	$V_3$			
		...		$V_i$	
					$V_N$

$$N = 36$$

# WHY IS CRYPT MEMORY-HARD?

- Input:  $x$

$$A = H^{N+1}(x)$$

- For  $N$  iterations

$$i = A \bmod N$$

$$A = H(A \text{ XOR } V_i)$$

- Output  $A$

$V_1$		$V_3$		$V_5$	
		...			
		$V_{i-1}$	$V_i$		

$$N = 36$$



# A RELATED PAPER...

## Scrypt Is Maximally Memory-Hard

Joël Alwen<sup>1</sup>, Binyi Chen<sup>2</sup>, Krzysztof Pietrzak<sup>1</sup>, Leonid Reyzin<sup>3(✉)</sup>,  
and Stefano Tessaro<sup>2</sup>

<sup>1</sup> IST Austria, Klosterneuburg, Austria  
{jalwen,pietrzak}@ist.ac.at

<sup>2</sup> UC Santa Barbara, Santa Barbara, USA  
{binyichen,tessaro}@cs.ucsb.edu

<sup>3</sup> Boston University, Boston, USA  
reyzin@cs.bu.edu

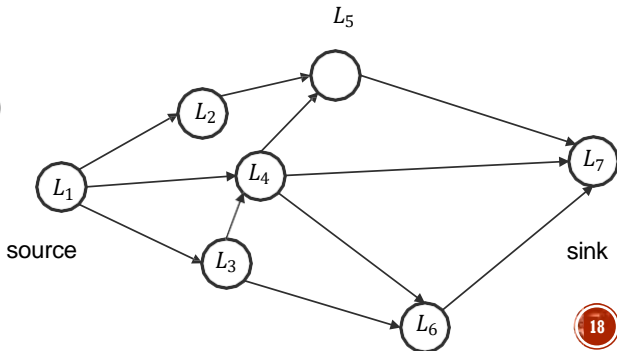
**Abstract.** Memory-hard functions (MHFs) are hash algorithms whose evaluation cost is dominated by memory cost. As memory, unlike computation, costs about the same across different platforms, MHFs cannot be evaluated at significantly lower cost on dedicated hardware like ASICs. MHFs have found widespread applications including password hashing, key derivation, and proofs-of-work.

# DAG-BASED MEMORY HARD FUNCTIONS

- Consider a DAG  $G = (V, E)$  with a source and a sink.
- Assign the label of the source based on the input and name it  $L_1$ .
- Label of the other nodes is characterized by an ordinary hash function  $H(.)$  as follows:

$$L_v = H(\{L_{v'} : v' \in \text{pred}(v)\})$$

- Output will be  $L_{\text{sink}}$



# PROOF OF USEFUL WORK

- Question: Is there a way to benefit society from the results of processing in mining?
- Candidates:

Protein folding  
Optimization problems

...

- The main question: how to decompose the solution to many challenges in a decentralized way?
- No admin should be involved.

# PROOF OF USEFUL WORK EXAMPLE: PRIMECOIN

- Miners look for large prime numbers with a special format called Cunningham primes ( $p_{i+1} \sim 2p_i$ ).
- Actually, they find long chains of prime numbers.
- The findings will be available on the ledger for use in other applications.
- An important question: do we have many large Cunningham chains of arbitrary lengths?



TODO 0\_29 34:00

# CUNNINGHAM PRIMES

- Cunningham chains of the first kind

$$p_{i+1} = 2p_i - 1$$

1531, 3061, 6121, 12241, 24481

- Cunningham chains of the second kind

$$p_{i+1} = 2p_i + 1$$

2, 5, 11, 23, 47

- bi-twin chains

211049, 211051, 422099, 422101, 844199, 844201



## ANOTHER INTERESTING EXAMPLES

- Gridcoin

Incentivizing sharing compute power for solving scientific problems based on BOINC platform.



- FoldingCoin

Like gridcoining but for the Folding@home network project.



# GRIDCOIN



<https://coincheckup.com/coins/gridcoin/about>

# PROOF-OF-DEEP-LEARNING

## Energy-recycling Blockchain with Proof-of-Deep-Learning

Changhao Chenli\*, Boyang Li\*, Yiyu Shi, Taeho Jung

*Department of Computer Science and Engineering*

*University of Notre Dame*

Notre Dame, Indiana, USA

{cchenli, Boyang.Li.258, yshi4, tjung}@nd.edu

- Main idea: distributing DNN training tasks between miners.
- First the training data is released between the miners by the model requester.
- Miners commit to their trained models by sending hashes of their models.
- Then the test data is released, and miners validate their models and submit the accuracy.
- Miners validate the models, and reward is given to accurate ones.



# PROOF-OF-STAKE

- In PoS, the chance of proposing a block is proportional to the coins of each miner (also called validator here).
- The idea has appeared first on an online forum (2011)

**Bitcoin Forum**simple machines forum

March 02, 2021, 06:08:43 AM

Welcome, **Guest**. Please login or register.




**News:** Latest Bitcoin Core release: [0.21.0](#) [Torrent]

[HOME](#) [HELP](#) [SEARCH](#) [LOGIN](#) [REGISTER](#) [MORE](#)

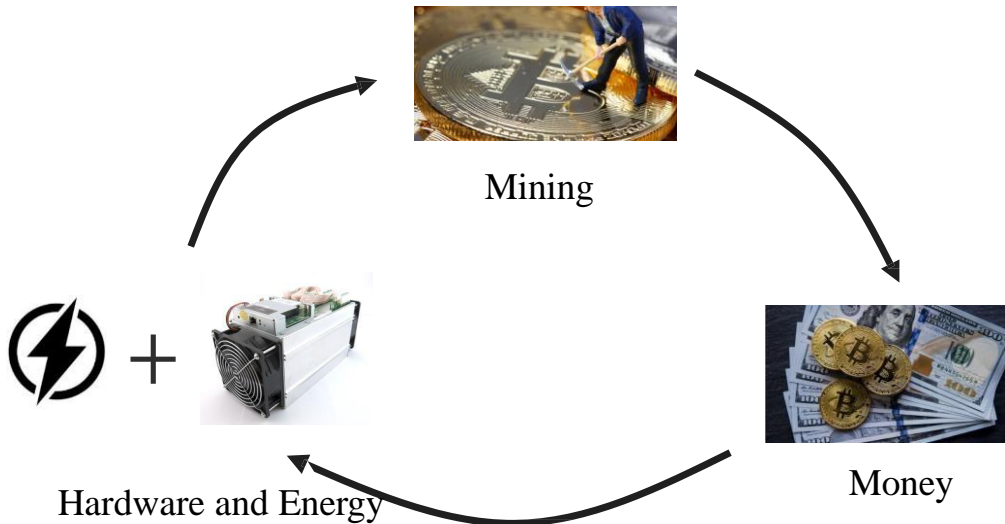
Bitcoin Forum > Bitcoin > Development & Technical Discussion > **Proof of stake instead of proof of work**

« previous topic next topic »  
print

Pages: [1] 2 » All

Author	Topic: Proof of stake instead of proof of work. (Read 32130 times)	
<b>QuantumMechanic</b> Member 	 <b>Proof of stake instead of proof of work</b> July 11, 2011, 04:12:45 AM <i>Merited by</i> <a href="#">ETFbitcoin (3)</a> , <a href="#">Vod (2)</a> , <a href="#">webtricks (2)</a> , <a href="#">d5000 (1)</a> , <a href="#">drays (1)</a>	#1
Activity: 110 Merit: 19 	<p>I've got an idea, and I'm wondering if it's been discussed/ripped apart here yet:</p> <p>I'm wondering if as bitcoins become more widely distributed, whether a transition from a proof of work based system to a proof of stake one might happen. What I mean by proof of stake is that instead of your "vote" on the accepted transaction history being weighted by the share of computing resources you bring to the network, it's weighted by the number of bitcoins you can prove you own, using your private keys.</p> <p>For those that don't want to be actively verifying transactions, and so that not all private keys need to be facing the network, votes could be delegated to other addresses via some kind of nonstandard Bitcoin transaction. In this way, voting power would accumulate with trusted delegates instead of miners. New bitcoins and transaction fees could be randomly and periodically distributed to delegates, weighted by the number of votes they've accumulated, thereby incentivising diversity of the delegates and direct voters.</p>	

# THE UNNECESSARY STEP



# EXAMPLE: PEERCOIN

Peercoin which was developed by Sunny King is a pioneer in PoS systems.

- The main rule for proposing a new block

staking: More coins you have, y



difficulty

$$\text{hash}(\text{prev\_block\_data}, \text{time in seconds}, \text{UTXO}) \leq d \cdot \text{coins}(\text{UTXO}) \cdot \text{timeweight}(\text{UTXO})$$

- There is no nonce.
- One attempt per second.
- One attempt per UTXO.
- Coinage determines the timeweight of each utxo.
- $d$  represents the difficulty parameter.
- $\text{prev\_block\_data}$  is the hash of the previous block.

#coins in that single utxo

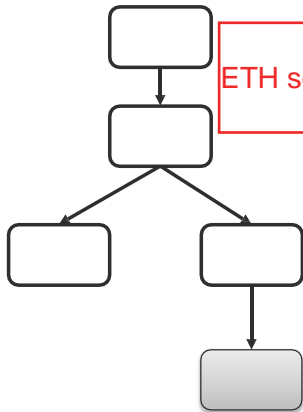
A single UTXO = a single unspent transaction

time since the transaction occurred

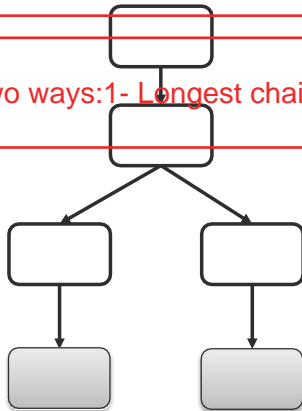
\*\*\* Encourages using the older

# NOTHING AT STAKE PROBLEM

Because you can easily spend some time mi



ETH solves the problems in two ways: 1- Longest chain appro



In PoW the attacker should distributed its hash power between the forks.

In PoS the attacker can extend on both the forks.

# STAKE COMPOUNDING PROBLEM

- In PoS, new coins are directly given to the owners of previous coins.
- This was also true for PoW, but in an indirect manner.
- This results in the rich get richer, compromising the security of the system.
- Consider an adversary with 30% of stake at the beginning, that can wait until getting 51% of the stake and applying an attack on the network.



# PROOF OF BURN

This must be included in the formula too.

- In PoB one should destroy coins (in a provable manner) to participate in the block mining lottery.
- Sending a small amount to an unspendable address (Eater address).
- E.g. a randomly generated public address without any private key available.



# PROOF OF



Proof of Importance

Proof of Devotion

Proof of Bid

Proof of Authority



Proof of Activity

Proof of Elapsed Time