**https://lifeinplaintextblog.wordpress.com/use-matlab-to-implement-a-simple-blockchain/**

**Implement a simple blockchain with MATLAB**

This article is an attempt from the perspective of a programmer to understand a small part of the blockchain. Blockchain is a distributed way of storing data. Below we use MATLAB to simulate this way of storing data, omitting the distributed feature.

Realize the block with MATLAB

A block is actually a data structure that can be used to store any type of data. In most practical applications of blockchain, the stored data is usually transaction data. In MATLAB, we can use a class to represent the block (Block)

classdef Block < handle

properties

index % block number

data % transaction data

previousHash % hash of the previous block

selfHash % own hash

nonce % A random number will be explained later

end

methods

function obj = Block(index, data, previousHash) % constructor

if nargin == 2 % genesis block

obj.index = index ;

obj.data = data ;

elseif nargin == 3

obj.index = index ;

obj.data = data ;

obj.previousHash = previousHash;

end

end % function end

function str = getCombined(obj)

% This function converts all data on the block except nonce and selfHash into characters.

% Is then used to calculate selfHash

str = strcat(num2str(obj.index),obj.previousHash,strjoin(obj.data));

end % function end

end % method end

end % class def end

In the Block class definition, attributes include

index (block number),

data (transaction data),

selfHash (block's own Hash value),

previousHash (hash value of the previous block), and a

random value (nonce )

The Block constructor can accept 2 or 3 parameters. When two parameters are provided, the constructed Block object is called genesis block (Genesis block), which is the first block on the entire blockchain; when it accepts three When there are two parameters, the third parameter is the hash value from the previous block, which will be detailed later.

What is hash algorithm and mining

For simplicity, the Hash algorithm can be understood as a mapping algorithm, which maps a string of characters into another fixed-length string is a functional relationship, using MATLAB language

function output = HashOperation(input)

...

end

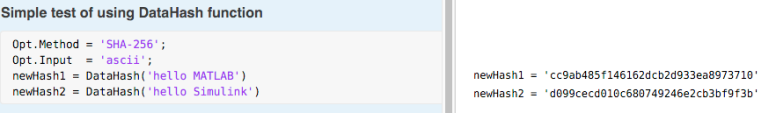
Given the input, the Hash algorithm can quickly calculate the fixed-length output; but the reverse is extremely difficult, that is, the output is known, and it is difficult to reversely find what the input is.

Given the input, the Hash algorithm can quickly calculate the fixed-length output; but the reverse is extremely difficult, that is, the output is known, and it is difficult to reversely find what the input is.

This article uses a user-written DataHash function in Mathworks File Exchange as the Hash algorithm

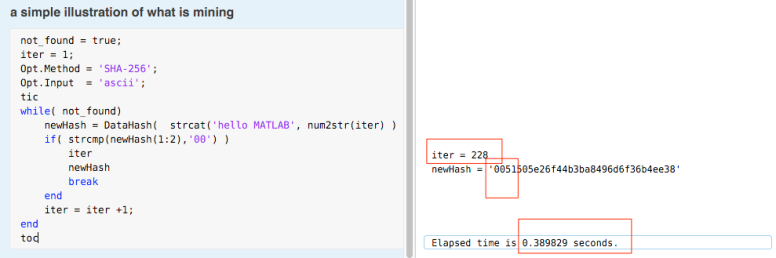
<https://www.mathworks.com/matlabcentral/fileexchange/31272-datahas>

Test this DataHash function in the live editor, and different inputs are mapped into fixed-length newHash1 and newHash2 strings



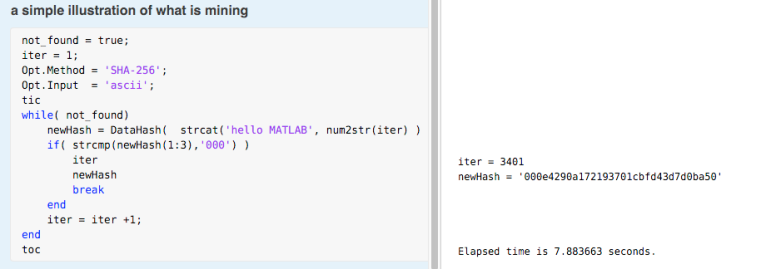
With this DataHash function, you can roughly understand what mining is. We observe the one-to-one relationship between newHash1 and'hello MATLAB' above, and newHash1 is a combination of extremely irregular characters

Now it is required to solve such a problem: It is stipulated that the first two digits of newHash start with 00, and what can be the input string. Note that we only specify the first two digits of newHash, not all the digits. There may be many inputs that meet this requirement, but still because even if the output is known, it is difficult to find the input backwards, so we can only use brute force exhaustion. For example, use'hello MATLAB' and an integer as strcat, and then calculate DataHash. If it does not meet the requirements, try the next integer until the first two digits of newHash are 00. The example in Live Script is as follows:



We didn’t get the newHash that met the requirements until we exhausted to the 228th number, which took 0.39 seconds in total

If it is more demanding, it is required that the first three numbers of newHash must be zero, and it must be exhausted to 3401 to get the newHash that meets the conditions. It takes 7.88 seconds



This exhaustive method to find the hash that meets the conditions is the essence of mining. Since there is no connection between each cycle, these operations can be parallelized. This is the essence of the mining machine. The more stringent the requirements for the initial letter of newHash, the mining The difficulty, that is, the greater the time-consuming.

Use MATLAB to implement blockchain: connect block objects into chains

Now the blocks can be connected into a blockchain, we use a new class called Block Chain to simulate it. First, it is stipulated that the first block in the chain can be directly generated, which is the genesis block. This is the constructor of the above block. The number of input parameters equal to 2 is the returned object.

classdef BlockChain < handle

properties

totalCount% is used to record the number of blocks

blockArray% The object array is used to hold the blockchain

end

methods

function obj = BlockChain()

obj.blockArray =[ Block(0,'Genesis Block')];

% The first genesis block is generated when the blockchain object is constructed

obj.totalCount = 1;

obj.calculateGensisBlockHash();

% Calculate the hash of the genesis block

end

function bc = getLatest(obj)

bc = obj.blockArray(end);

end

function calculateGensisBlockHash(obj)

gb = obj.blockArray(1);

Opt.Method = 'SHA-256';

Opt.Input = 'ascii';

str = strcat(num2str( gb.index),gb.data )

% The calculation method is to use all the contents as strcat

gb.selfHash = DataHash (str, Opt); % hash

end

function addBlock(obj,newBlock)

% When the miner successfully 'digs out' a block that meets the requirements

if obj.validateNewBlock(newBlock)% call this function

obj.blockArray(end+1) = newBlock;

% Add this block to the blockchain

end

end

function tf = validateNewBlock(obj,newBlock)

% Verify whether the newly added block meets the requirements

newHash = DataHash( strcat( newBlock.getCombined(), num2str(newBlock.nonce) ));

if(strcmp(newHash(1:2),'00') && strcmp(newBlock.selfHash,newHash))

tf= true;

else

tf = false;

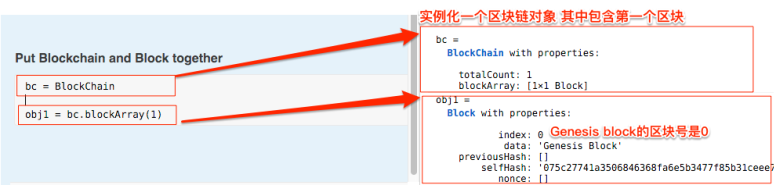
end

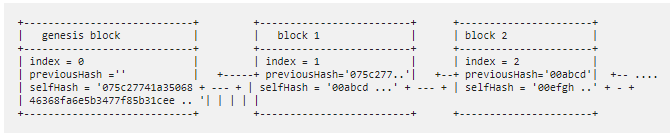
end

end

end

Demonstrate and instantiate the Block Chain class in the live editor. In its constructor, generate the first genesis block and calculate its hash value.





Simulate mining with MATLAB

We gathered the mining behavior and related data into a class called Miner. The key to this class is

(1) Miner needs to ask about the data of the latest block of the blockchain before mining to form the block to be added.

(2) Only blocks that meet the requirements can be added to the blockchain. Given index, previousHash, and data, the hash of a block is fixed, which does not necessarily meet the requirements. All we have to add a value obtained by brute force solution, this value and The other data in the block is strcat, and the hash obtained can meet the conditions. This is the process of mining.

classdef Miner < handle

properties

blockchain% miner has a reference to the blockchain

end

methods

function obj = Miner(blockchain)

obj.blockchain = blockchain;

end

function mine(obj,newData)

% Get the last block on the current blockchain

latestBlock = obj.blockchain.getLatest();

%(1) is used to construct a newest block

% The required information is index, and the selfHash of the previous block is regarded as its previousHash

newBlock = Block(latestBlock.index+1,...

newData,...

latestBlock.selfHash);

% Violent solution method to find a selfHash that meets the requirements

not\_found = true;

iter = 1;

Opt.Method = 'SHA-256';

Opt.Input = 'ascii';

while( not\_found)

newHash = DataHash( strcat( newBlock.getCombined(), num2str(iter) ));

if( strcmp(newHash(1:2),'00') )

newBlock.nonce = iter% (2) Violent solution

newBlock.selfHash = newHash% If the assignment selfHash is found

obj.blockchain.addBlock(newBlock);% Add yourself to the blockchain

break

end

path = path +1;

end

end

end

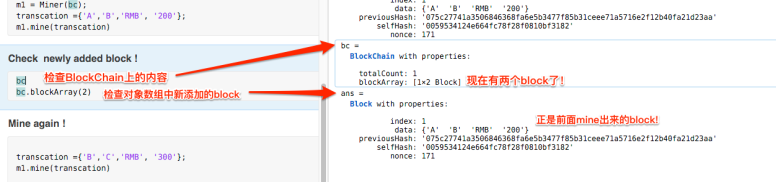
end

Let’s demonstrate with Live Editor

Initialize a blockchain and mine the first block except the genesis block



Check the blockchain



Dig the second block



