CRYPTOCURRENCY USING BLOCKCHAIN

A PROJECT REPORT

Submitted by

NAME	**REGNO**
PUNITHA VANCHA	**BL.EN.U4AIE19052**
RUTHVIK K	**BL.EN.U4AIE19054**
SAMUDYATHA DEVALLA	**BL.EN.U4AIE19057**

for the course

19AIE203- Data Structures and Algorithms – 2

Guided and Evaluated by

D. RADHA

Asst. Prof(SG),

Dept. of CSE,

AMRITA SCHOOL OF ENGINEERING, BANGALORE

AMRITA VISHWA VIDHYAPEETHAM

BANGALORE-560 035

December-2020

CRYPTOCURRENCY USING BLOCK CHAIN

ABSTRACT

Blockchain is a technology that enables the presence of cryptocurrency. Bitcoin is one of the best-known cryptocurrency, the one for which blockchain technology was actually invented. A cryptocurrency is a medium of exchange of a digital or virtual currency that is secured by cryptography, which makes it nearly impossible to counterfeit or double-spend. Cryptocurrency is implemented using blockchain technology. A blockchain is a type of database. It collects transaction together in groups, also known as blocks, that hold sets of information. Blocks when mined, they are chained onto the previously filled block, forming a chain of data.

Our aim is to create a cryptocurrency for which ledger is implemented using a blockchain.

Blockchain stores all the transactions in the blocks when mined. The user can mine the cryptocurrency, make transaction, view transactions, check if the blockchain is valid and check their wallet. The hashing function used is sha256 which is not complex yet is one of the strongest function.

INTRODUCTION

Blockchain is a type of database. It differs from a typical database in the way it stores information. It structures its data into chunks or blocks that are chained together.

Cryptocurrency is a digital or virtual currency that is secured by cryptography, which makes it hard to double spend. Many cryptocurrencies are decentralized networks based on blockchain technology.

Hashing is a function that converts an input of letters and numbers into an encrypted output which is in alpha numerical value of a fixed length. A hash is created using an algorithm and is essential to blockchain management in cryptocurrency.

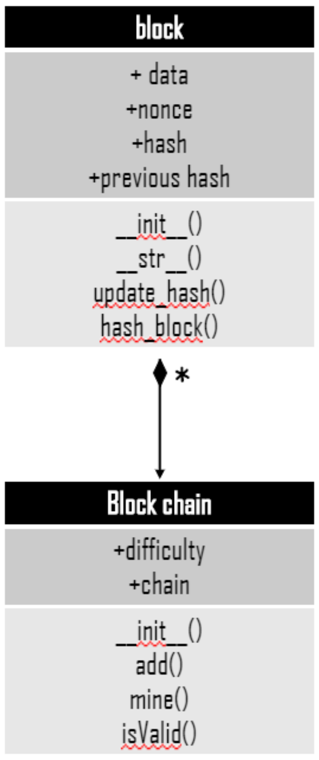
Proof of work algorithm in blockchain, is used to confirm transactions and produce new blocks to the chain. The hash for any data is always trivial, hence to make some proof of work, a difficulty is set. This process includes repeated hashing of block header to find an appropriate hash ID.

Mining is the process of adding transaction records to cryptocurrency public ledger of past transactions. This ledger of past transactions is called the block chain as it is a chain of blocks. The blockchain serves to confirm transactions to the rest of the network as having taken place.

Transactions using cryptocurrency are stored in a ledger made with blockchain technology. Blocks containing transactions are chained together each block contains a hash ID, previous block's hash, data , and an integer variable 'nonce'.The Hash ID is produced by hashing the data, previous block hash, nonce and the hash function used is SHA256. If any data is tampered the hash ID of the block will change.Since it is a distributed ledger, each block has to be verified by all the members and hence tampering the data can be identified easily.The hash for any data is always trivial, hence to make some proof of work, a difficulty is set. Difficulty is given so that our hash ID is lesser than or equal to a given target hash.
Eg: if difficulty is 2, the block chain will accept only those blocks with Hash IDs having 2 zeros in the beginning.
In order to get the target hash, the nonce value is eventually increased until the required hash is obtained. Mining a block means finding an acceptable hashID for a block to add it to the blockchain.
It involves repeated hashing of the block header by increasing the nonce.
In real situations mining takes 15 mins to 1 hr to mine a block.
Generally cryptocurrency such as bitcoin are produced when a miner mines a block or solves some math puzzle.The user is always anonymous.
Every user at the first transaction gets a private key, and a public key based on it.
The transactions are made using the public key which is visible to everybody.
Private key is only known to the user and once lost cannot be found through public key or any means.

System Model – Description of the diagram

Blockchain.py



We import sha256 to hash the blocks in a block chain.
def updateHash(*args)
In this function we can create a list of arguments. We will pass the arguments in hash_self into the args list. We update the hash of the hashing text. So we will hash all the data combined together to encode it in UTF-8 and return them in hexadecimal.

Now in class Block(), we can have have some form of data, in our case it is transaction. Then we have hash, nonce; it is an arbitraray number (where old communications cannot be reused) , hash from previous block . First block will always have no previous blocks.

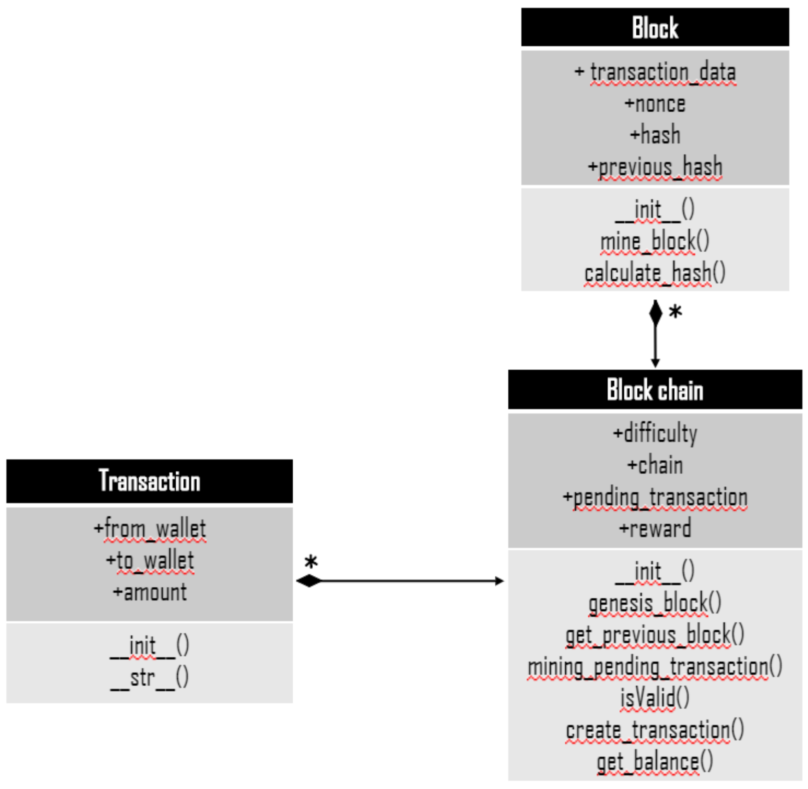
hash_block()
Hash needs to be updated using its data, nonce and its previous hash. We use the Sha256 to do this.
__str__()
We use double underscores of block class to update the way the block is shown in the output and then we return the string of block number.
init()
This function represents a list of blocks which are all chained together. Difficulty is the number of zeros in the front of the hash, more the difficulty, stronger the blockchain.

add()
It appends the blocks in the block chain.

mine()
checks if there is previous hash, if present, then it uses that to mine the next block.
We assign previous has of the block to -1 to get the last block in the chain. There will be an error if there is no previous block. So we loop it until infinity until our nonce produces

isValid()
This function check if the block is present or not and we check if the hash value have the first difficulty number of digits and the previous hash is the same as the next block's hash.

Blockchain_currency.py



Block

Mine_block()
It repeatedly hashes the block header and finds the appropriate hash ID by increasing the nonce

Calculate_hash()
Clacultaes the hash of the arguments using sha256 and returns a 64 bit hash using hexdigest().

BlockChain
Init()
It is like a constructor which has all the variables of the class initialized

Genesis_block()
It is the default first block of the block chain

Get_previous block()
It returns the previous block of the current block and is used for getting the hash of the previous block.

Mining_pending_transaction()
This function is used to mine all the pending transaction in one single block.

isValid()
This is function is used to check if the blockchain is valid or not. If data is tampered it returns false else it returns true.

Create_transaction()
Whenever a transaction is made, this function adds it into pending transaction list which will be added into a block once mined.

Get_balance()
Calculates the balance of the given user and returns it.

Transaction
__init__()
Constructor which initializes the variables.

Str()
Returns the format of the print statement when called

Implementation Details

Hashlib

is hashing function takes variable length of bytes and converts it into a fixed length sequence. This is a one way function. That means, you hash a message, you get a fixed length sequence. But you cannot get the original message from those fixed length sequence.

sha256

SHA, (Secure Hash Algorithms) are set of cryptographic hash functions defined by the language to be used for various applications such as password security etc. It generates an almost-unique **256**-bit signature for a text

Json

JavaScript Object Notation (JSON) is a standardized format commonly used to transfer data as text that can be sent over a network. It's used by lots of APIs and Databases, and it's easy for both humans and machines to read. JSON represents objects as name/value pairs, just like a Python dictionary.

Pprint

The pprint module provides a capability to “pretty-print” arbitrary Python data structures in a form which can be used as input to the interpreter. If the formatted structures include objects which are not fundamental Python types, the representation may not be loadable.

Sample Code

Sample code for blockchain.py

```
#!/usr/bin/python
# -- coding:utf-8 --

from hashlib import sha256


# function to hash the values that uses hash value of the
previous block
def update_hash(*args):
    hashing_text = ""
    h = sha256()
    for arg in args:
        hashing_text += str(arg)

    h.update(hashing_text.encode('utf-8'))
    # returning the hash value
    return h.hexdigest()


# class to create a block
class Block:
    data = None
    hash = None
    nonce = 0
    # 64 bit hash of the previous data
    previous_hash = "0" * 64

    # number is the block number
    def _init_(self, data, number=0):
        self.data = data
        self.number = number

    # hashing the values
    def hash_block(self):
        return update_hash(
            self.previous_hash,
            self.number,
            self.data,
            self.nonce
        )

    # function is used  when the print function is called
    def _str_(self):
        return str("\nBlock#: %s\nHash:\t\t%s\nPrevious Hash:
```

```
%s\nData: %s\nNonce: %s\n"
                   % (self.number,
                      self.hash_block(),
                      self.previous_hash,
                      self.data,
                      self.nonce
                      )
                   )
```

Sample code for blockchain_currency.py

```
# class to chain the blocks
class Blockchain:
    def _init_(self):
        self.chain = [First_Block()]
        self.difficulty = 3
        self.pendingTransaction = []
        self.reward = 10

    # is used to get the last box in the chain that is used to
get the previous hash
    def get_last_block(self):
        return self.chain[len(self.chain) - 1]

    # mining of all pending transactions into 1 block
    def mining_pending_transactions(self, minerRewardAddress):
        # in reality not all of the pending transaction go
into the block the miner gets to pick which one to mine
        new_block = Block(self.pendingTransaction)
        new_block.mine_block(self.difficulty)
        new_block.previous_hash = self.get_last_block().hash

        print("Previous Block's Hash: " +
new_block.previous_hash)
        testChain = []
        for transaction_data in new_block.transaction_data:
            temp = json.dumps(transaction_data._dict_,
indent=5, separators=(',', ': '))
            testChain.append(temp)
        pprint.pprint(testChain)

        self.chain.append(new_block)
        print("Block's Hash: " + new_block.hash)
        print("Block added")
        if minerRewardAddress == 'default':
            self.reward = 0
        else:
            self.reward = 10
        rewardTrans = Transaction("System",
minerRewardAddress, self.reward)
        self.pendingTransaction.append(rewardTrans)
        self.pendingTransaction = []

    # function to check if the blockchain is valid or not by
checking current block's hash and previous block's hash
    def isValid(self):
        for x in range(1, len(self.chain)):
            currentBlock = self.chain[x]
            previous_hash = self.chain[x - 1].hash

            if currentBlock.previous_hash != previous_hash:
                return "The Chain is not valid!"
        return "The Chain is valid and secure"

    # function to add the transaction in pending transaction
list
    def create_transaction(self, transaction):
        self.pendingTransaction.append(transaction)

    # function to get balance of a person by checking all the
transactions
    def get_balance(self, walletAddress):
        balance = 0
        for block in self.chain:
            if block.previous_hash == "":
                # don't check the first block as it has no
data
                continue
            for transaction in block.transaction_data:
                if transaction.from_wallet == walletAddress:
                    balance -= transaction.amount
                if transaction.to_wallet == walletAddress:
                    balance += transaction.amount
        return balance


# class to create a transaction between 2 people for a
specific amount
class Transaction:
    def _init_(self, from_wallet, to_wallet, amount):
        self.from_wallet = from_wallet
        self.to_wallet = to_wallet
        self.amount = amount

    # function for printing the data
    def _str_(self):
        # str(self._class_) + ": " +
        return str(self._dict_)
```

SAMPLE OUTPUT(1 or 2 pages)

Sample output for blockchain.py

```
Press 1 for adding data to block chain
Press 2 to print the block chain
Press 3 for checking if the chain is valid or not
Press 4 for modifying any value in the blockchain
Press 5 to Exit
1
Enter the data to be added in the blockchain:- hello world
Press 1 for adding data to block chain
Press 2 to print the block chain
Press 3 for checking if the chain is valid or not
Press 4 for modifying any value in the blockchain
Press 5 to Exit
1
Enter the data to be added in the blockchain:- how are you
Press 1 for adding data to block chain
Press 2 to print the block chain
Press 3 for checking if the chain is valid or not
Press 4 for modifying any value in the blockchain
Press 5 to Exit
1
Enter the data to be added in the blockchain:- i made money
Press 1 for adding data to block chain
Press 2 to print the block chain
Press 3 for checking if the chain is valid or not
Press 4 for modifying any value in the blockchain
Press 5 to Exit
2

Block#: 1
Hash:           0000c1296c36fde4cad95eaaf1c51bb4a8dc2432250f47064db6391ab374af89
Previous Hash:
0000000000000000000000000000000000000000000000000000000000000000
Data: hello world
Nonce: 11058


Block#: 2
Hash:           00006e30f7dd4694192468d5b6203b42aa91a0fbd796c5eeb33ad45f3f763e65
Previous Hash:  0000c1296c36fde4cad95eaaf1c51bb4a8dc2432250f47064db6391ab374af89
Data: how are you
Nonce: 9252


Block#: 3
Hash:
        00009a6139c487552cfc707d01820e5606a142427d1de069bd37d5756343a6c6
```

```
Previous Hash:
00006e30f7dd4694192468d5b6203b42aa91a0fbd796c5eeb33ad45f3f763e65
Data: i made money
Nonce: 60763

Press 1 for adding data to block chain
Press 2 to print the block chain
Press 3 for checking if the chain is valid or not
Press 4 for modifying any value in the blockchain
Press 5 to Exit
3
The blockchain is valid

Press 1 for adding data to block chain
Press 2 to print the block chain
Press 3 for checking if the chain is valid or not
Press 4 for modifying any value in the blockchain
Press 5 to Exit
4
Enter the block number for which you want to modify2
Enter the modifying value:- i lost money

The blockchain is not valid now

After the change, the block chain is:-

Block#: 1
Hash:           0000c1296c36fde4cad95eaaf1c51bb4a8dc2432250f47064db6391ab374af89
Previous Hash:
0000000000000000000000000000000000000000000000000000000000000000
Data: hello world
Nonce: 11058


Block#: 2
Hash:           822b04cf256f0ae1d009359eabbe918cb7d09b3043caaa6d7c2dca898e54e2db
Previous Hash:  0000c1296c36fde4cad95eaaf1c51bb4a8dc2432250f47064db6391ab374af89
Data: i lost money
Nonce: 9252


Block#: 3
Hash:           0000f1415ddc2cd3380f2de810e13cb33fd06550177ad55ac7a6436ba7f5db56
Previous Hash:
00009a6139c487552cfc707d01820e5606a142427d1de069bd37d5756343a6c6
Data: i made money
Nonce: 92490

The database is set back to previous state when the blockchain was valid

Block#: 1
Hash:           0000c1296c36fde4cad95eaaf1c51bb4a8dc2432250f47064db6391ab374af89
```

```
Previous Hash:
0000000000000000000000000000000000000000000000000000000000000000
Data: hello world
Nonce: 11058


Block#: 2
Hash:           00006e30f7dd4694192468d5b6203b42aa91a0fbd796c5eeb33ad45f3f763e65
Previous Hash:  0000c1296c36fde4cad95eaaf1c51bb4a8dc2432250f47064db6391ab374af89
Data: how are you
Nonce: 9252


Block#: 3
Hash:
        00009a6139c487552cfc707d01820e5606a142427d1de069bd37d5756343a6c6
Previous Hash:
00006e30f7dd4694192468d5b6203b42aa91a0fbd796c5eeb33ad45f3f763e65
Data: i made money
Nonce: 60763

Press 1 for adding data to block chain
Press 2 to print the block chain
Press 3 for checking if the chain is valid or not
Press 4 for modifying any value in the blockchain
Press 5 to Exit
5
Thank you....Visit again later

Process finished with exit code 0


Select an option:-
1.Mine currency
2.Check Balance
3.Send money
4.Check transactions
5.Check if chain is valid or not
6.Exit
1
Who is mining?ruthvik
Previous Block's Hash:
02f2ccfc554897a3b245c2535d10e5dc2aa9deb94d075958bc8fd3ed54a89c93
[]
Block's Hash: 00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
Block added

10 coins are added to ruthvik
```

<u>Sample output for blockchain_currency.py</u>

```
Select an option:-
1.Mine currency
2.Check Balance
3.Send money
4.Check transactions
5.Check if chain is valid or not
6.Exit
2
Whose balance is to be checked?ruthvik
Previous Block's Hash:
00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
[]
Block's Hash: 00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
Block added

ruthvik has 10 Coins on their account

Select an option:-
1.Mine currency
2.Check Balance
3.Send money
4.Check transactions
5.Check if chain is valid or not
6.Exit
3
Who is sending money? ruthvik
Who is receiving money? person1
How much money is to be transferred? 3

Money has been transferred
```

<u>Sample output for blockchain_currency.py</u>

```
Select an option:-
1.Mine currency
2.Check Balance
3.Send money
4.Check transactions
5.Check if chain is valid or not
6.Exit
4
Previous Block's Hash:
00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
['{\n'
 '    "from_wallet": "ruthvik",\n'
 '    "to_wallet": "person1",\n'
 '    "amount": 3.0\n'
 '}']
Block's Hash: 000903471b614f880a1eb81a136c9d06a02b7499f97a9bb54d8919c680bf3d82
Block added

Block 1 :-
```

```
I am the First Block
Block  2 :-
Previous hash:-
02f2ccfc554897a3b245c2535d10e5dc2aa9deb94d075958bc8fd3ed54a89c93
Current hash :-
00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
{'from_wallet': 'System', 'to_wallet': 'ruthvik', 'amount': 10}

Block  3 :-
Previous hash:-
00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
Current hash :-
00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
{'from_wallet': 'System', 'to_wallet': 'default', 'amount': 0}

Block  4 :-
Previous hash:-
00079a20f115f3816dfaa2e7e1c41de151b9848c68d04417b4da1debc5c1c36a
Current hash :-
000903471b614f880a1eb81a136c9d06a02b7499f97a9bb54d8919c680bf3d82
{'from_wallet': 'ruthvik', 'to_wallet': 'person1', 'amount': 3.0}
{'from_wallet': 'System', 'to_wallet': 'default', 'amount': 0}


Select an option:-
1.Mine currency
2.Check Balance
3.Send money
4.Check transactions
5.Check if chain is valid or not
6.Exit
5
The chain is valid

Select an option:-
1.Mine currency
2.Check Balance
3.Send money
4.Check transactions
5.Check if chain is valid or not
6.Exit
6
Thank You....Visit again later

Process finished with exit code 0
```

Conclusion

Blockchain has become a popular technology in a very less time due to cryptocurrencies such as bitcoin. It makes government operations and business more accurate, efficient, secure and cheap and requires fewer middle men. Block chain is not only used for cryptocurrency but also for making online voting systems, store inventories, supply chains ,storing medical records, records of properties ,etc.

Cryptocurrency which is virtual money promises easy transactions between sender and receiver without involving third parties such as banks. Hence it is also cheap. The transaction are completed in lesser time and data cannot be recorder. It has many advantages and yet has its own criticism. Since anonymous transactions are made, it can be used for illegal activities and if the money is stolen, only the money can be traced but not the thief.

REFERENCES

The Growth of Cryptocurrency in India: Its Challenges & Potential Impacts on Legislation

April 2018 DOI: 10.13140/RG.2.2.14220.36486 Project: Virtual Currencies (Cryptocurrency)Authors: Shailak JaniParul Universiy

Cryptocurrencies: market analysis and perspectives

Giancarlo Giudici, Alistair Milne & Dmitri Vinogradov *Journal of Industrial and Business Economics* volume 47, pages1–18(2020)Cite this article

A systematic review of blockchain

Min Xu, Xingtong Chen & Gang Kou *Financial Innovation* **volume 5**, Article number: 27 (2019) Cite this article.

BlockChain Technology Beyond Bitcoin
Sutardja Center for Entrepreneurship & Technology Technical Report Date: October 16, 2015 Authors Michael Crosby, Google Nachiappan, Yahoo Pradhan Pattanayak,Yahoo Sanjeev Verma, Samsung Research America Vignesh Kalyanaraman,Fairchild Semiconductor.

https://www.investopedia.com/terms/c/cryptocurrency.asp
https://www.investopedia.com/terms/b/bitcoin.asp