Deree
School of Graduate and Professional Education

Big Data Architectures
Winter 2024

Term Project

Due: Monday March 25, 2024, 11:59.

In this project you will use Spark, Kafka, and MongoDB to develop a network of applications for generating transactions, mining them into blocks of a blockcain, which are then transmitted through Kafka and eventually stored into a MongoDB for subsequent processing and answering queries about them.

Background

A blockchain is a data structure, which is implemented as a sequence of blocks chained together with each block containing one or more immutable transactions. Any modification to any of the constructed blocks including the transactions they contain is prohibitively expensive. A blockchain is a means of achieving consensus between parties who do not share any trust relationships among themselves. It is the infrastructure behind cryptocurrencies like Bitcoin, Ethereum, and several more.

The blocks of a blockchain are linked together by a field, which is a hash value, the result of applying a cryptographic hash function to a block. The linking of the blocks is achieved by including in each block the hash value of the previous block of the chain.

For the purposes of the project each block of a blockchain contains the following data

1. A sequence number: Sequence numbers start from 0. The very first block of the blockchain (the genesis block) has sequence number 0.
2. A list of one or more transactions: They are the actual payload data to be included in the blockchain.
3. A nonce: value that is determined through a process called mining, as explained below.
4. Its hash value.
5. The time it took for mining the block.

For a block to be added to the blockchain its construction must be proved, i.e., that it has not created out of thin air. The construction proof takes the form of a computationally intense process, called mining. There are various mining approaches, in this project we will use the one called Proof of Work (PoW). PoW mining is demanding in terms of computing power, and, consequently, of electricity. It has been deemed as a negative aspect of blockchains and a point of harsh criticism for their use. Mining entails the solution to a difficult mathematical problem as explained below. The process is called mining to parallel the labor intense process of mining for precious metals in the real world.

Cryptographic hash functions

Cryptographic hash functions are one-way functions used heavily in cryptography, with certain properties that are beyond the scope of this project. They are called one-way

because given an input string it is easy to calculate its hash value, while the opposite is computationally infeasible.

The cryptographic hash of a given string (might be a block of data encoded as a string) is a value that is expressed as a sequence of bits. There are many cryptographic hash functions, in this project we will make use of SHA-256. Here is how the SHA-256 hash value of a string "ACG ICT 6107" is calculated in Python:

```
from hashlib import sha256

s256 = sha256()
s256.update('ACG ICT 6107'.encode())
digest = s256.hexdigest()

print(digest)
```

The output, which is also called the "digest" of the original string, is

3dc34f34eab1b83f09d9efa669f5cb8c673976d8ddf43c095d8869a6dafc0378

If the string "ACG ICT 6107" is slightly perturbed to "ACG ICT 6107!" then its digest is completely different

490ef69072739a515c50a295c9fb65da09204439727d76bef777f87b518fac30

Hashlib's sha256 allows accumulation of strings to be digested later, i.e.,

```
from hashlib import sha256

s256 = sha256()
s256.update('ACG ICT 6107'.encode())
s256.update('Term project'.encode())
digest = s256.hexdigest()

print(digest)
```

and the digest of the two juxtaposed strings is

cbb675c8629893f60f7b6ff47309f93eaaf639fd70900a485b228c655801e14a

The digest produced by the SHA-256 hash function is a 256-bit number which is represented as 64 hexadecimal digits (4 bits each) in the above examples.

Mining

For the purposes of this project PoW mining will be used. Then the problem of mining is to find an integer value *n* (called nonce), such that the digest of the quintuple

1. the sequence number of the block,

2. the transactions the block contains,
3. the value of the nonce,
4. the value of the digest of the previous block

has a certain number of leading zeros. The only thing that can vary in the previous list (contents of the block) is the value of the nonce. The number of leading zeros determines the difficulty level of block mining. The more the number of leading zeros required the more difficult the mining problem becomes. Fr the purposes of the project we assume the level of difficulty to be 3, i.e., 3 leading zeros of the digest.

To find a nonce that solves the PoW mining problem one needs to exhaustively try all possible integer values, from 0 to $2^{32} - 1$ (assuming 32-bit integers) until he hits one that satisfies the requirement of leading zeros (for the given difficulty level). This is because the cryptographic hash functions break any relations between their input and the digest. This process can be heavily parallelized.

In the Bitcoin network, miners across the world (possibly putting efforts together by forming mining farms) compete to mine the next block. The first to find the solution to the mining problem is the winner and gets a reward according to the Bitcoin rules. For the purposes of this project different Spark workers will compete to find a nonce that solves the mining problem. The first of these workers to solve the problem will be the winning one and the one whose nonce and time-to-mine will be kept with the block.

In general, it is not guaranteed that a nonce exists that solves the PoW mining problem. The Bitcoin blockchain takes this into account and uses additional information in such cases but for the purposes of this project the difficulty levels that will be used will result to a nonce that satisfies the difficulty requirement (number of leading zeros of the block digest).

Once a nonce is found the block is constructed as a quintuple that contains

1. The block serial number.
2. The list of transactions that are contained in the block.
3. The value of the nonce that resulted to the successful mining of the block.
4. The block's digest.
5. The time it took to mine the block.

The very first block of the chain, the genesis block, is hand crafted. You may consider that the only transaction it contains is the string 'Genesis block', its sequence number is 0 and the hash of the previous block is the string '0' as no previous block exists. Once the genesis block is constructed and its digest computed, additional blocks can be constructed and added to the blockchain containing transactions from a stream of transactions.

Term project requirements

[10%] Write a server in file tr-server.py, which runs forever and generates a transaction (an arbitrary string for the purposes of the project) every second. Make sure that the generated transactions are unique, i.e., no string is repeated. The transactions are emitted at port 9999.

Deree
School of Graduate and Professional Education

Big Data Architectures
Winter 2024

[50%] Write a Spark Streaming application in file mine.py to accept transactions coming from the server, collect them in 2-minutes intervals, and mine a block that includes them. The level of parallelism for mining should depend on the number of cores the processor of your machine. When mining blocks, observe the performance of the cores of your machine and take a copy of the performance monitor as the mining process progresses.

When a new block is mined your application must print the whole blockchain from the genesis block to the newly mined one.

[10%] Each time a block is mined all information about the block is written to a Kafka topic "Blocks" as a JSON object, which must contain the following:

1. The block serial number.
2. The number of transactions that are contained in the block.
3. The block nonce.
4. The block's digest.
5. The time it took to mine the block.

Topic "Blocks" contains two partitions: 0 and 1. Mined blocks with even serial numbers are written to partition 0 and blocks with odd serial numbers are written to partition 1.

[10%] Write a Python application in file app0.py that reads partition 0 of topic "Blocks" and adds its contents to a MongoDB collection "blocks". Similarly for a second application app1.py that reads partition 1 of "Blocks" and writes its contents to the same MongoDB collection.

[20%] Write a Python application in file mongoq.py that answers the following:

1. For a given block serial number, what is its nonce value, digest, and number of transactions contained in the block.
2. Which of the blocks mined so far has the smallest mining time.
3. Which is the average and cumulative mining time of all blocks mined so far.
4. Which block(s) has the largest number of transactions in them.

Submission

Each team should submit

1. All Python code.
2. A Powerpoint presentation containing
    a. A short description of the problem.
    b. The architecture of the solution.
    c. The design decisions made.
    d. Screenshots of the running application.
    e. Lessons learnt.

    The presentation should not be more than 15 slides excluding the first slide that should contain the name and logo of the team, and the names of its members, and the last slide that should thank the audience and prompt for questions.

3. A file Team.txt containing a clear statement stating which parts of the application where implemented by which team member and a self-assessment of the percentage of contribution of each team member to the project (e.g., Efthimiou 45%, Papadopoulou 55%). Statements of the form “both members contributed to all parts” will not be accepted.
4. A README.txt file with instructions on how to run the network of applications.

To submit your work add

- all source code files,
- the Powerpoint presentation,
- the file Team.txt
- the file README.txt

into a zip compressed file with name <Lastname>_<Lastname>.zip (e.g., Efthimiou_Papadopoulou.zip) and submit it to Blackboard. Names in the filename must appear in alphabetical order.

Penalties

Violation of any naming conventions will result into 30 points reduced from your grade.

Submission of files in excess to those asked in tis handout will result into 20 points reduced from your grade.

References

Check https://www.blockchain.com/explorer for the Bitcoin blockchain.