Weekly Report - Survey On Mining Algorithms

MSc Project
Runchao Han

November 21, 2017

1 Progress of the Last Week

1. Searched state-of-the-art mining algorithms
2. Understanded most of those mining algorithms
3. Set up the programming environment of OpenCL/CUDA
4. Downloaded their source code(most are C/C++) and ran them successfully(including their CUDA/OpenCL versions if exists). This includes:
   (a) ethminer(CUDA/OpenCL)
   (b) cuckoo
   (c) scrypt
   (d) cryptonight
5. Made the initial decision of choosing mining algorithms

2 The Choice of Target Mining Algorithms

Currently I suggest to target at Ethash and Cuckoo, and maybe Scrypt. The reason of chooseing Ethash and Cuckoo are:

1. Parallelisable
2. Widely adopted
3. Current implementations are robust, especially the Ethash

Scrypt is not parallelisable at present, but ASICs for scrypt have emerged. Currently scrypt is not adopted to be the exact PoW algorithm, but combined with other functions to form stronger algorithms, like Momentum.

After the search of mining algorithms, most existing mining algorithms are memory-bound, except Bitcoin. This is because ASICs make the cryptocurrency decentralised. The consensus among cryptocurrency communities is that GPU mining is the de facto standard. Though lower hashrate is produced by purely GPU mining, the difficulty of mining a block is lower according

to the hashrate of the whole network, which keeps the speed of finding a block is statistically constant(e.g. A block is mined in the Bitcoin network per 10 minutes). This will keep the mining profitable.

State-of-the-art mining algorithms are all parallelisable to reach the memory bandwidth limit. Currently no mining machines with huge memory bandwidth are produced(maybe it's difficult).

3 Introduction

Currently Proof-of-Work (PoW) algorithm is the most popular consensus algorithm for public blockchains because of its scalability, where peers get coins by "mining" on the blockchain network. "Mining" in PoW based cryptocurrencies means to find hash values which satisfy the requirement to get the right to append a block on the blockchain. This report focuses on the first task of the MSc Project, which is to make a survey on state-of-the-art mining algorithms.

4 Background of PoW

This section introduces the theory of PoW from its cryptography basis to this probability based consensus, including Hash Functions, HashCash and PoW. In addition, its challenges and corresponding solutions are discussed.

4.1 Hash Functions

A hash function is any function that maps arbitrary size data to fixed size data. The values returned by a hash function are called hash values. An example of the the usage is a data structure called hash table, widely used in computer software for rapid data lookup. It can be expressed as:

$$H(x) = h$$

`H()` is the hash function which processes an arbitary string `x` then get a fixed length string `h`. Widely used hash functions are listed below:

1. Secure Hash Algorithm (SHA) family, including SHA-0 to SHA-3 with different parameters.
2. MD5 Message-Digest Algorithm

Since hash functions are essentially many-to-one functions for footprinting strings, some requirements should be met:

1. **Compression:** H can be applied to a block of data of any size, but produces a fixed-length output
2. **One-way property (pre image resistant):** H(x) is easy to compute for any given x. For any given h, it is hard to compute x such that H(x)=h
3. **Weak collision resistance (2nd preimage resistant):** Given x, it is hard to find y != x such that H(y)=H(x)
4. **Strong collision resistance (collision resistance):** It is hard to find two different messages, x != y, such that H(y)=H(x)

This online tool can do popular hash function calculations.

4.2 HashCash

HashCash was proposed to resist Denial of Service (DoS) Attack and spam emails. It requires a client to do a specific calculation which is hard to compute but easy to verify before invoking essential operations of the server. For example, a spam email sending machine should do this calculation every time before it sends an email, which will make the spam sending uneconomic, but not for normal individuals.

One-way functions like hash functions are suitable for implementing HashCash. For example, the server requires the client to find a value whose hash value starts with at least three zeros. Both clients and servers do not know answers at first. The client continuously generates random strings and computes hash values of them to find a valid string. Based on the random nature of hash functions, the probability of finding a valid value is fixed so that the required work of the client is also fixed.

4.3 PoW Consensus - Double SHA256 in Bitcoin as an Example

Supported by the HashCash mechanism, the consensus is obtained on a peer-to-peer (P2P), unstable and untrusted network by PoW. According to the example above, HashCash can change the difficulty simply by changing required number of 0 at the beginning of hash values. On the Bitcoin network, every node calculates HashCash values, where the difficulty is dynamic according to the total calculating power on the network. That is, if the total computing power is bigger, the difficulty will increase, which always keeps the average time to find a valid value about 10 minutes. This process is called "Mining" because it is similar to mine gold in a goldmine.

The time of 10 minutes is an average value based on probabilitic theories which is a practical way to avoid the double spending problem. 10 minutes is enough for nodes in the whole network to receive the latest block when no other valid hash values are found.

The hash function of Bitcoin's PoW is Double SHA256:

$$H(x) = SHA256(SHA256(x))$$

4.4 Challenges of PoW - Centralisation of Computing Power

At the beginning of Bitcoin, only CPU was used for mining, then followed GPU mining because SHA256 is easy to parallelise. To mine more coins with less electric power, FPGAs and ASICs were designed, which caused the centralisation of computing power.

4.5 Solutions

To avoid the centralisation, memory-bound and non-parallelisable algorithms are proposed, to mitigate the performance gap between CPU/GPU and FPGA/ASIC, or even make FPGA/ASIC unavailable. Requirements for this more democrat approach are:

1. Memory-hard or memory-bound
2. Non-parallelisable
3. Hard to compute but easy to verify
4. Dynamic difficulty

5 State-of-the-art Democrat PoW Algorithms

5.1 Scrypt

Scrypt[4] is a sequential memory-hard hash function adopted by Litecoin's PoW consensus. The original paper gave thorough definitions to the memory-hard algorithms and sequential memory-hard algorithms. It also introduces the $ROMix_H$ function class and the $SMix$ function class which are all sequential memory-hard based on the Random Oracle model[1]. Finally it proposed the *Scrypt* algorithm which achieved the best performance.

5.2 Ethash Based on DAG Generation

Ethash is a memory-bound mining algorithm resistant to ASICs[2]. The general route that the algorithm takes is as follows:

1. There exists a seed which can be computed for each block in the blockchain. In other words, a seed can be computed by a specific algorithm for each block.
2. From the seed, a 16 MB pseudorandom cache is computed. "Light clients"(network nodes who needn't store the whole blockchain) can verify a block only by the cache.
3. From the cache, we can generate a 1 GB dataset, with the property that each item in the dataset depends on only a small number of items from the cache. Full clients and miners store the dataset. The dataset grows linearly with time.
4. Mining involves grabbing random slices of the dataset and hashing them together.
5. Verification can be done with small memory by using the cache to regenerate the specific pieces of the dataset that you need.

The large dataset is updated once every 30000 blocks, so the vast majority of a miner's effort will be reading the dataset, not making changes to it. The cache and the dataset can be pre-computed.

The difficulty of the mining is dynamic which maintains that a new block is generated every 15 seconds.

5.3 Cuckoo Based on Graph Theory

Cuckoo is a memory bandwidth bound mining algorithm based on graph theory[5], adopted by aeternity.

Cuckoo Cycle aims to minimize performance-per-dollar differences on different hardware architectures and make mining on commodity hardware more cost effective. This is to be achieved by making main memory latency a bottleneck, since DRAM latencies have remained relatively stable while cpu-speed and memory bandwidth vary highly across hardware architecture and process technology.

The basic thinking is to construct a random bipartite graph with randomly chosen edges. The bipartile graph is implemented by two hashtables. A user randomly insert edges to it, until finding a cycle with the fixed length L. This process is easy to verify but hard to solve with frequent memory access.

5.4 Momentum Based on Hash Birthday Collision

Momentum is a mining algorithm based on hash function's birthday collision, adopted by MemoryCoin. Birthday collision means two known strings have the same hash value.

Assuming a cryptographically secure hashing function $Hash(x)$ and a sequential memory-hard hash function $BirthdayHash(x)$ like `scrypt` are defined. The algorithm can be described as follows:

1. Given a block of data D, calculate $H = Hash(D)$
2. Find nonce A and B such that $BirthdayHash(A + H) = BirthdayHash(B + H)$
3. If $Hash(H + A + B) < TargetDifficulty$, then a result has been found, otherwise keep searching.

5.5 MemoHash

Memohash[3] is a memory-hard hash function, but not a PoW function. It gave concrete definitions of a memory-hard function, as well as proposed two sequential memory hard functions `SeqMemoHash` and `RandMemoHash`. Little information about this algorithm exists, so I haven't understood it very well yet.

6 Planned Accomplishments

1. New item: give internal deadline for deliverable.
2. Person and Scheduled Task name:
   - hours spent by person
   - description of what was done (task doesn't need to be complete)
3. Scheduled Task name:
   - hours spent by person
   - description of what was done (task doesn't need to be complete)

7 Miscellaneous

- A website is found to check the hash rates of different cryptocurrencies with information about CPU/GPU/ASICs.

8 Next Week's Plan

- Make a detailed illustration of chosen mining algorithms
- Read paper about optimising and parallelising algorithms
- Read source code of chosen algorithms
- Learn about GPU programming(OpenCL/CUDA)

References

[1] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of the ACM (JACM)*, 51(4):557–594, 2004.

[2] Ethereum. Ethash.

[3] Sergio Demian Lerner. Strict memory hard hashing functions.

[4] Colin Percival and Simon Josefsson. The scrypt password-based key derivation function. Technical report, 2016.

[5] John Tromp. Cuckoo cycle: a memory-hard proof-of-work system. *IACR Cryptology ePrint Archive*, 2014:59, 2014.