Analysis of Variance of Graph-Clique Mining for Scalable Proof of Work

Anada, Hiroaki; Matsushima, Tomohiro; Su, Chunhua; Meng, Weizhi; Kawamoto, Junpei; Bag, Samiran; Sakurai, Kouichi

Published in:
Information Security and Cryptology. Inscrypt 2018

Link to article, DOI:
10.1007/978-3-030-14234-6_6

Publication date:
2019

Document Version
Peer reviewed version

Analysis of Variance of Graph-Clique Mining for Scalable Proof of Work

Hiroaki Anada[1], Tomohiro Matsushima[2], Chunhua Su[3], Weizhi Meng[4], Junpei Kawamoto[2], Samiran Bag[5], and Kouichi Sakurai[2]

[1] Department of Information Security, University of Nagasaki, Japan. anada@sun.ac.jp
[2] Department of Informatics, GraduateSchool and Faculty of Information Science and Electrical Engineering, Kyushu University, Japan. sakurai@inf.kyushu-u.ac.jp
[3] Division of Computer Science, University of Aizu, Japan chsu@u-aizu.ac.jp
[4] Department of Applied Mathematics and Computer Science. Technical University of Denmark, Denmark weme@dtu.dk
[5] School of Computing Science, Newcastle University, UK samiran.bag@ncl.ac.uk

Abstract. Recently, Bitcoin is becoming one of the most popular decentralized cryptographic currency technologies, and Bitcoin mining is a process of adding transaction records to Bitcoin's public ledger of past transactions or blockchain. To obtain a bitcoin, the mining process involves compiling recent transactions into blocks and trying to solve a computationally difficult puzzle, e.g., proof of work puzzle. A proof of work allows miners the ability to quantify how much work a given proof contains. Basically, the required time for mining is decided in advance, but problems will occur if the value is large for dispersion. In this paper, we first accept that the required time between consecutive blocks follows the exponential distribution. That is, the variance is stable as long as the expected time is fixed. Then, we focus on the graph clique mining technique proposed by the literature, like Tromp (BITCOIN 2015) and Bag-Ruj-Sakurai (Inscrypt 2015), which is based on a computational difficulty problem of searching cliques of undirected graphs, where a clique is a subset of vertices. In particular, when the clique size is two, graph clique mining can be used to gain Bitcoins. The previous work also claimed that if the clique size is parameterized and increased, even if the expected time is fixed, the variance would not be stable. However, no qualitative or quantitative results were given to support their claim. Motivated by this issue, in this work, we propose a simple search algorithm for graph cliques mining, and perform a small scale evaluation on Bitcoin and Graph cliques's solo mining to investigate the variance issue.

Keywords: Blockchain, Proof of work, Graph-Clique Mining, Bitcoin, Mining competition.

1 Introduction

Before the year of 2009, currency transactions were conducted through trusted third parties such as banks and credit card companies, but Bitcoin [11], one cryp-

tographic currency released in 2009, allows a decentralized digital currency without a central bank or single administrator. Bitcoin system guarantees the legitimacy of a transaction without requiring a trusted agency. Transactions are verified by network nodes through cryptography and recorded in a public distributed ledger called a blockchain. It is regarded as an open ledger that epitomizes a general consensus among the online participants with respect to historicity of all validly executed transactions over the Bitcoin network. A newly constructed block gets appended to the already existing block chain after an approximately constant time interval (e.g., 10 minutes) [1,14].

A proof of work (PoW) in the context of blockchain is a piece of data that is difficult to generate due to the cost and time-consumption, but is easy for others to verify. To generate a proof of work can be a random process with a low probability, which means that many efforts should be made before a valid PoW is obtained. In particular, Bitcoin uses the Hashcash proof of work system. In order for a block to be accepted by network participants, miners must complete a proof of work that covers all of the data in the block. The difficulty of this work is adjusted so as to limit the rate at which new blocks can be generated by the network to one every 10 minutes. Due to the very low probability of successful generation, this makes it unpredictable which worker computer in the network will be able to generate the next block. In other words, under the incentive of getting a bitcoin reward, bitcoin miners have to repeat mining competition for each block. PoW's computational nature allows miners to quantify how much work a given proof contains.

In this paper, we consider the statistical time dispersion of mining competition in such PoW system. Regarding Bitcoin mining, the expected time required for mining is decided as 10 minutes in advance. However, an extremely lucky miner may finish the mining competition in a short time, i.e., much shorter than the expected time, or an extremely unlucky miner may take a longer time while cannot find any. The difficulty is expected to rise with the popularity of Bitcoin, but the following three problems would occur [9,12].

1. It is known that the utility of money is concave. Thus, the time variance in the supply of money would result in the difficulty of finance management (or plans) and the decrease of a person's utility.
2. Bitcoin blocks are not published at fixed time intervals, but are randomly found in a Poisson process. As payment is not made regularly, it is technically difficult to validate whether all systems are working properly.
3. The Bitcoin model differs from the mint model in a sense that it uses a finality confirmation structure via mining competition. That is, a high time dispersion may cause much stress among all mining participants.

Regard the convenience and security of a virtual currency network, it is desirable that the time variance required for mining is small enough according to the above three issues. However, in Bitcoin's PoW system, by given a hash value, we have to find the input of a hash function where the size of the problem space is constant

irrespective of the number of trials. The time distribution required for mining can be regarded as an exponential distribution, so that the time dispersion depends only on the expected time required for mining. In the context of Bitcoin, this can be considered as one of the important tasks to make the time dispersion scalable, by properly setting it to a (desirable) small value that is as small as possible [9,12].

In the literature, the proof-of-work algorithms proposed by Bag et al. [2] and Tromp [15] are based on a computationally difficult problem of searching cliques in an undirected graph, where a clique is a subset of fully connected vertices. It is worth noting that the problem of searching for a clique of the specified number of vertices (size) is NP complete [6]. In the previous study [2], they utilized the problem of finding the largest clique in a big graph as a replacement for the existing Bitcoin PoW scheme. They handled a graph having $O(2^{30})$ vertices and $O(2^{48})$ edges, which is constructed deterministically using the set of transactions executed within a certain time slot. They then proposed an enhanced algorithm to solve this PoW puzzle by doing $O(2^{80})$ hash calculations. Their scheme forces both computing power and memory of a miner. Taking the advantage of the graph clique search problem, the time variance required for mining is scalable with the size of cliques.

1.1 Our contributions

Motivated by this challenge, in this work, we propose a simple search algorithm for graph cliques mining, and perform a small scale evaluation on both Bitcoin and Graph cliques's solo mining to investigate the variance issue. Our contributions can be summarized as follows.

- Firstly, we conduct a theoretical evaluation of solo mining. Our interest is that the graph clique mining can become a Bitcoin mining scheme when the clique size is two. Our theoretical evaluation validates this observation.
- Secondly, we propose a easy-to-use search algorithm for mining graph cliques. Although our algorithm is not performed the fastest as compared with the existing search algorithms, it is much easier to implement.
- Further, we perform an evaluation to test the performance of our algorithm in the context of Bitcoin, i.e., exploring graph cliques via solo mining and investigating the variance issue.

1.2 The organization of this paper

In Section 2, we introduce the notation and primitives used in this paper, including hash function, Bitcoin mining technique [11], and various mining approaches, i.e., solo mining and pooled mining. In Section 3, we conduct a theoretical analysis on solo mining time in Bitcoin, based on the existing research [8,12]. In Section 4, we analyze the existing studies on graph clique mining like [2], discuss the mining time variance compared to Bitcoin, and perform an evaluation on Bitcoin solo mining and Graph cliques's solo mining.

Finally, we conclude this work with future directions in Section 5.

2 Preliminaries

In this section, we introduce the notations used in this paper, and summarize key requirements for cryptographic hash functions. Then, we make a brief introduction on Bitcoin mining [11], two mining ways of solo mining and pooled mining [12], as well as mining competition.

Bitcoin is a decentralized cash system that does not depend on a centralized server. The corresponding public key can be used to publicly verify the authenticity of the transaction. The process of Bitcoin mining involves compiling recent transactions into blocks and trying to solve a computationally difficult puzzle [11]. Bitcoin network maintains a publicly auditable ledger called Bitcoin block chain that is aimed at preventing double spending of Bitcoins. A Bitcoin block is constructed by users called miners and it requires one to execute a nontrivial amount of computation.

For an undirected graph with a finite number of vertices, a clique is a complete subgraph of a graph. Clique problem involves finding two types of cliques: maximal clique and maximum clique. The former is one that cannot be extended to form a clique of bigger size, while the latter is a clique that has the size equal to that of the largest clique in the same graph. Clique problem is defined as the problem of finding the largest clique in a graph or listing all maximal cliques in the graph. When the number of vertices is k, we say it is a clique of size k or a k-clique. When a finite number of vertices is given, the problem of searching for one clique of size k can be known as 'k-clique search problem'.

Solo mining refers to the process of calculating hashes individually, in order to find a valid block whose reward will be paid entirely to the person in ownership of the hashing computer. Pooled mining refers to a joint effort between several miners to work on finding blocks together, and split the rewards among the participants in proportion to their contribution.

2.1 Cryptographic hash function

The cryptographic hash function is a function H that takes an input of an arbitrary-length message and outputs a fixed length bit string, which is called 'hash value'. It has the following three major features:

1. *Pre-image resistance.* When the value h is given, finding the input m such that $h = H(m)$ is computationally difficult.
2. *Second pre-image resistance.* Given an input m_1, it should be difficult to find a different input m_2 such that $H(m_1) = H(m_2)$. Hash functions are vulnerable to second-preimage attacks without this property.
3. *Collision resistance.* It should be difficult to find two different messages m_1 and m_2 such that $H(m_1) = H(m_2)$. Such a pair is called a cryptographic hash collision. To defend against birthday attacks, strong collision resistance is desirable, which requires a hash value at least twice as long as that required for pre-image resistance.

At the analysis in Section 2.3, we assume that the hash function H is a random oracle.

2.2 Background on Bitcoin Mining

Bitcoin mining used in this paper refers to how to search for a hash value in relation to Bitcoin transactions described in the original paper [11].

In particular, Bitcoin's network has a timestamp server, which is responsible for hashing the data (e.g., transaction information) to be time-stamped using the SHA-256 algorithm and broadcasting the hash value throughout the network. Bitcoin mining is intentionally designed to be resource-intensive and difficult, so that the number of blocks found each day by miners remains steady. Individual blocks must contain a proof of work to be considered valid. The mining process requires miners to perform competitive computation in finding a solution for a puzzle, based on the broadcasted hash value. The primary purpose of mining is to allow Bitcoin nodes to reach a secure, tamper-resistant consensus. Bitcoin mining is difficult because the SHA-256 hash of a block's header must be lower than or equal to the target in order for the block to be accepted.

To obtain a Bitcoin, miners have to search for a *nonce* (described later) that satisfies a condition, and if they find the correct solution, then they have to broadcast that *nonce* and the solution to the whole network. Only by doing this, a miner can become the winner of the computational competition. In other words, miners perform some computation on the data, and then send the timing data to the time stamp server. This is required by the server to decide who found the solution *nonce* first. It should be noted that the time consumption of propagating *nonce* to the entire computer network is much shorter than the time consumption for completing a mining process. Due to this, the required time for propagation can be neglected.

In the mining process, we have two types of data: the data in a blockchain include all received transactions up to now; and the data of transaction information from the last time-stamp to the next received time-stamp. The use of timestamp is to prove that the existence of transaction at the time when the transaction is timestamped. To obtain a reward, a miner has to concatenate *nonce* to these two values, perform a hash calculation, and search for a *nonce* that can make the hash value less than or equal to a predetermined threshold. Assume that the hash value of an agreed blockchain is B, and the data of all the transaction histories are T. Let D denote the value determined from the adjustment of the mining difficulty. Then the goal of mining process is to search for a *nonce* (a string) that can satisfy the following conditional expression (the concatenation of the string a and b is written as $a \parallel b$).

$$H(B \parallel T \parallel nonce) < D. \tag{1}$$

2.3 Mining Ways and Competition

There are two major mining ways [12]: solo mining and pooled mining. Solo mining is a solo process where a miner completely does his task of mining operations without joining a pool. These blocks are mined and generated in a way to the task completed by the miner's credit. In contrast, pooled mining refers to a scenario that most miners do the mining in pools, which is the pooling of resources by miners, who share their processing power over a network, to split the reward equally, according to the amount of work they contributed to the probability of finding a valid block.

On the other hand, the mining process consists in repeatedly computing hashes of variants of a data structure called a block header, until one is found whose numerical value is low enough. When this happens, it allows releasing a valid block, for which the miner is rewarded with bitcoins in an amount (known as mining competition). To be a winner, miners have to solve the above computational problem (1), which allows them to chain together blocks of transactions.

A graph is a set of vertices V and set of edges $E(\subset V \times V)$, which can be determined by (V, E). In this paper, we denote the number of vertices $|V|$ as N. Thus, a subset C of V is a clique of size k if $|C| = k$ and for any $(v_1, v_2) \in C \times C$ s.t. $v_1 \neq v_2$, $(v_1, v_2) \in E$ holds.

In our paper, we use the random graphs proposed in [4]. If we set a constant number $0 \leq p \leq 1$, then the probability $\Pr[(v_1, v_2) \in E]$ is p, which determines the probability of $(v_1, v_2) \in V \times V$ being an edge. Note that the coin tossings are independent of each other in repeated trials (the probability of becoming"head" is p). A random graph determined by (N, p) is written as $\mathcal{G}_{N,p}$. For all cliques of $\mathcal{G}_{N,p}$, let $Z(\mathcal{G}_{N,p})$ be the maximum value of the clique size. According to the previous study [7], the asymptotic behavior of $Z(\mathcal{G}_{N,p})$ can be represented as follows.

$$Z(\mathcal{G}_{N,p}) = \frac{2\log_e N}{\log_e(1/p)} + O(\log_e n).$$

Furthermore, according to the work [10], given a value of k, the probability $\Pr[Z(\mathcal{G}_{N,p}) \geq k]$ can be evaluated by combinatorics. For example, we have:

$$\Pr[Z(10^{10}, 0.25) = 30] > 0.9997.$$

As mentioned earlier, it is worth noting that the problem of searching for a clique of the specified number of vertices (size) is NP complete [6]. Based on these facts, Bag et al. [2] advised to use the maximum clique search problem for mining against the random graph $\mathcal{G}_{N,p}$, which can be determined from the transaction history decisively. However, in fact, we estimate the value k of $Z(\mathcal{G}_{N,p})$ in advance for the random graph $\mathcal{G}_{N,p}$, and prefix k-clique search problem. In particular, we adopt the following value for k in this work.

$$k := \frac{2\log_e N}{\log_e(1/p)}.$$

Then, it is important to know how to determine the number of vertices and edges of the graph. In this work, we denote the number of vertices as $N := |V|$, where $N = 2^n$ (power of 2) for benefiting bit shift. For the purpose of replacing Bitcoin's proof of work $n = 30$, $N = 2^{30}$ is appropriate according to the work [2]. Another issue is how to define the sides, we assume that a set of transaction histories that a miner wishes to capture is $\{T_s; s = 0, \ldots, N_t - 1\}$, and denote the order number of the transaction history as $N_t = 2^\nu$ (the power of 2).

For the purpose of positioning it as generalization of Bitcoin mining using graph clique, we should slightly change the way of setting sides. First of all, for N vertices, adding an integer value v_l to each vertex as follows.

$$\begin{aligned} v_l &\stackrel{\text{def}}{=} (T_{l/2^{n-\nu}} \cdot 2^{n-\nu}) \parallel (l \% 2^{n-\nu}), \\ & l = 0, \ldots, N-1. \end{aligned}$$

As an example, if $n = 4$ and $\nu = 2$, the number of vertices is $N = 16$ and the number of transaction histories is $N_t = 4$. For $\{v_l; l = 0, \ldots, N-1\}$, we determine the edges in the adjacency matrix $A = (A_{i,j})_{0 \le i,j < N}$ as follows.

$$\begin{aligned} A_{i,j} &\stackrel{\text{def}}{=} \begin{cases} 0 \text{ if } i = j, & \\ 1 \text{ if } H(v_i \parallel v_j) = (0^m \parallel x), & 0 \le i \le j < N, \\ 0 \text{ otherwise}, & \end{cases} \\ A_{i,j} &\stackrel{\text{def}}{=} A_{j,i}, & 0 \le j < i < N. \end{aligned}$$

Here, $0^m \parallel x$ represents the concatenation of m 'zero' strings and arbitrary string x, and m is a parameter. According to the work [2], we set $m = 12$ for $n = 30$. Viewing the hash function H to be a random oracle, the parameter p is estimated as follows.

$$p = 2^{-m}.$$

In the mining in $\mathcal{G}_{N,p}$, the miner has to find a solution for the clique search problem. The solution denotes a submatrix of the adjacency matrix A, at which all the components are 1. The miner sets the already agreed hash value as B, the transactions as $T := T_0 \parallel \cdots \parallel T_{N_t-1}$, and the solution as a string *clique*. Then we have the following value as the next agreed hash value that should be included in the blockchain.

$$H(B \parallel T \parallel \textit{clique}). \tag{2}$$

As a result, the above graph clique mining process can be regarded as Bitcoin mining, if we assume that $n = 256$ (means the bit length of SHA-256's end region) and $k = 2$.

3 Our Analysis of Bitcoin Mining Time

In this section, we review the existing studies [3,8,12] regarding the probability distribution of time interval, which the winner of the mining competition follows during Bitcoin mining.

3.1 Bitcoin Solo Mining Time: Exponential Distribution

In Bitcoin solo mining, a miner's evaluation of each *nonce* (Expression (1)) does not use the evaluation results before the evaluation (i.e. memoryless trials [12]). Hence we stand on the following assumption.

– A miner samples *nonce* uniformly at random every time.

According to the general theory of probability distribution, memoryless continuous probability distribution is limited to the exponential distribution [5]. In the following discussion, we show this derivation briefly (for details, see [16], etc.). When $\Delta\ x$ is sufficiently small, the probability of occurrence of an event between time x and $x + \Delta x$, which is denoted by $P(x \leq t \leq x + \Delta x)$, can be obtained by the definition of the probability density function $f(x)$:

$$P(x \leq t \leq x + \Delta x) = f(x)\Delta x. \tag{3}$$

On the other hand, the same probability is described in another way by the above assumption as

$$\text{(the probability that the event does not occur until a time } x) \tag{4}$$
$$\times\text{(the probability that the event occurs between } x \text{ and } x + \Delta x) \tag{5}$$

That is,

$$P(x \leq t \leq x + \Delta x) = (1 - \int_0^x f(t)dt) \times \lambda\Delta x, \tag{6}$$

where λ denotes the average number of occurrence in the Bernoulli trials per a unit time, which is a constant. Note here that, due to the above assumption, the second factor $\lambda\Delta x$ does not depend on x.

Therefore, we obtain the following integral equation:

$$f(x) = \lambda - \lambda \int_0^x f(t)dt$$

We then differentiate these two sides and solve the differential equation. The solution is the following function.

$$f(x) = \lambda e^{-\lambda x}. \tag{7}$$

This is an exponential function. That is, the probability density function is limited to the probability density function of the exponential distribution (7).

In terms of the above ground-truth, from the point at which the winner appears in the i-th slot of Bitcoin mining competition, the time interval x_i until the next winner appears in the $i+1$-th slot of Bitcoin mining competition should follow the exponential distribution [12]. That is, if set the time interval $x_i, i = 1, 2, \ldots$, to be handled by the random variable X, then X follows the exponential distribution. Also, $(X_i)_{i=1,2,\ldots}$ becomes a Poisson process when we treat x_i as probabilistic variable X_i [12,8]. Some studies based on actual data like [3] indicate that the time for solo mining in Bitcoin follows the exponential distribution.

3.2 Bitcoin mining time variance

In the exponential distribution (7) of X, the expected value is $E_f(X) = 1/\lambda$ and the variance is $V_f(X) = 1/\lambda^2$. Therefore, if we set the difficulty level D (Expression (1)), then the expected value is set as $1/\lambda$. By performing adjustment which is the case for Bitcoin, the expected value $1/\lambda = 10$[min], and therefore the variance must be $1/\lambda^2 = 100$[min^2] (standard deviation is therefore $1/\lambda = 10$[min]) .

Hence, for Bitcoin mining, as long as the expected time is fixed, there might be a problem that the variance cannot be reduced, as we mentioned in Introduction.

3.3 Bitcoin Mining Time: Relationship with Geometric Distribution

The exponential distribution is a type of continuous probability distribution, and the geometric distribution is type of discrete probability distribution, while these two can become equivalent via conversion (see an example in [13]). In general, the probability mass function $f(i)$ of a geometric distribution can be determined by one parameter $p,\ 0 \leq p \leq 1$, as below.

$$f(i) = (1-p)^{i-1}p. \tag{8}$$

We denote the random variable that follows the geometric distribution as Y, and we denote the expected value as $E_f(Y) =: \mu$. Then, the variance is represented as $V_f(Y) = \sigma^2 = \mu^2 - \mu$.

4 Our Experimental Analysis of Graph Clique Mining Time

In this section, we try to analyze the probability distribution for the graph clique mining, where the time interval at which the winner of the mining competition should follow.

4.1 Graph Clique/ Solo Mining Time

In the case of graph clique solo mining, Expression (6) established by Bitcoin solo mining does not hold anymore. This is because the probability multiplication factor (5) would not be $\lambda \Delta x$ in Expression (6) anymore.

4.2 Time Variance of Graph Creek Mining

In the previous study [2], they set the number of vertices to be constant (under the parameter settings), and the graph's edges is subjected to the clique search and the transaction history. If an efficient algorithm is available for searching cliques, then the problem space can become smaller. This is because the vertex out of any clique becomes known in the process of searching the clique. Therefore, in the graph clique search problem, the time variance required for mining is expected to be smaller than that required for Bitcoin.

4.3 Experimental Evaluation

In this section, we begin by introducing experimental results on validating the theoretical analysis of the Bitcoin solo mining and then discuss experimental results regarding the graph clique solo mining.

Table 1 describes our experimental environment with a 64-bit Linux machine. Python version 3.5 was used as the programming language for algorithm implementation. Due to the availability, we used 12 cores of CPU, but we did not particularly leverage the parallel processing capability in the evaluation. The size of available memory was 62.9 GB.

Table 1. Experimental environment and settings.

Programming Language	Python 3.5
CPU	Intel Core i7-3960X CPU3.30GHz×12
RAM	62.9GB
OS	64bit, Linux

4.4 Experiments on Bitcoin Solo Mining and Results

Algorithm for Bitcoin Solo Mining. The algorithm used to evaluate the expression (1) iteratively, that is, our iterative generation of *nonce*, was to use the random function provided by Python 3.5 with the time as a seed.

For Bitcoin solo mining, the hash value B is fixed, and the data T is set to the value of the random function (seeding time). In addition, we set the value D specified for mining difficulty adjustment to 2^{228} (SHA-256 hash value, leading 28 bits is 0). The number of trials to find *nonce* should satisfy the expression (1) (the number of trials is to find a solution for different T) was set to 800 times (or trials).

Table 2. Experimental Results of Bitcoin Solo Mining

Experiment number	Expected value μ[sec]	Standard deviation σ[sec]
Number 0	538.6	535.3

In particular, Table 2 shows that the theoretical standard deviation can be estimated from the average value as $\sqrt{\mu^2 - \mu} = \sqrt{(538.6)^2 - 538.6} = 538.1$. It is approximately equal to $\sigma = 535.3$ (root of unbiased variance), indicating that it follows the geometric distribution, as well as the exponential distribution (as these two can be adjusted to be equal). In addition, Figure 1 shows the approximate shape of the exponential distribution; that is, the time consumption required by Bitcoin solo mining.

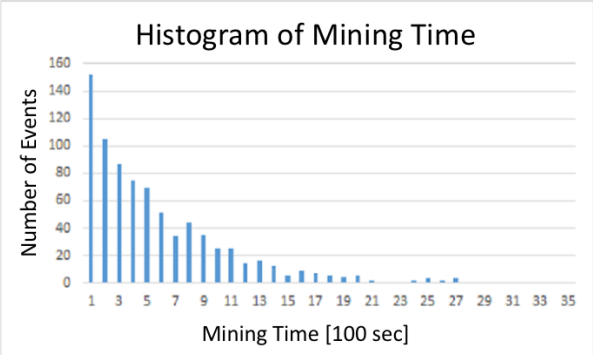

Fig. 1: Time Consumption of Bitcoin Solo Mining

4.5 Experiments on Graph Clique Solo Mining

Algorithm of graph clique solo mining. We used a naive algorithm to explore the clique search problem as below.

1. Compute the values attached to the vertex $v_l, l = 0, \ldots, N-1$,
2. Initialize all the components $A_{i,j}$ in adjacency matrix $A = (A_{i,j})_{0 \le i,j < N}$ with -1.

3. Search for cliques of size k; the hash value $H(v_i \parallel v_j)$ is evaluated only for $A_{i,j} = -1$.

Experimental Result. Table 3 indicates how we set parameters for the graph clique mining. The transaction history data $T_s, s = 0, \ldots, N_t - 1$ were generated by the random function (using seeding time).

In this work, we set the number of trials to find a solution (for different T_s) to 200 times. It is worth noting that the parameter value m^* (for Number 2 in Table 3) is not an integer value but a value slightly smaller than 7, calculated as $\frac{(0^6 \| x)_{10}}{2^{256}} < 3/2^7$. The component of the adjacency matrix is assumed to be 1 (the vertices i and j are connected by an edge). This is because if $m^* = 7$, then no solution was found in the experiment. Table 4 details our experimental results.

Table 3. Experimental parameters of graph clique solo mining

Experiment Number	n	N	ν	N_t	m	k
Number 1	14	16384	8	256	8	4
Number 2	14	16384	8	256	m^*	5

Table 4. Experimental Results on Graph Clique Solo Mining

Experiment number	Expected value μ[sec]	Standard deviation σ[sec]	ratio R
Number 1	327.56	313.24	96%
Number 2	255.94	187.49	73%

4.6 Discussion on Experimental Results

For Bitcoin solo mining, the time variance required for mining is determined by the expected value. It is comparable with the time variance of graph clique mining. That is, for an expected value μ obtained from the graph clique mining experimentally, if the expected value of Bitcoin mining were the same, then the variance for Bitcoin mining can be estimated based on the probability mass function (8) as $\sigma^2_{\mathrm{BC}} := \mu^2 - \mu$. On the other hand, the value of the unbiased variance can be obtained by the experiment for graph clique mining as σ^2. The following value R (also in Table 4) indicates the ratio.

$$R \stackrel{\mathrm{def}}{=} \frac{\sigma}{\sqrt{\mu^2 - \mu}}. \tag{9}$$

In the experiment with Number 1, the ratio R_1 is computed as below.

$$\begin{aligned} R_1 &= \sigma_1 / \sqrt{\mu_1^2 - \mu_1} \\ &= 313.24 / \sqrt{(327.56)^2 - (327.56)} \\ &= 0.95722 \approx 0.96. \end{aligned}$$

For the same expected value μ_1, it is found that the difference between the standard deviation σ_1 of the graph clique mining and the standard deviation of Bitcoin mining ($\sqrt{\mu_1^2 - \mu_1}$) is $1 - 0.96 = 0.04$ (i.e. 4%).

In the experiment with Number 2, the ratio value of R_2 is computed as below.

$$\begin{aligned} R_2 &= \sigma_2 / \sqrt{\mu_2^2 - \mu_2} \\ &= 187.49 / \sqrt{(255.94)^2 - (255.94)} \\ &= 0.73399 \approx 0.73 \end{aligned}$$

Similarly, for the same expected value μ_2, the difference between the standard deviation σ_2 of the graph clique mining and the standard deviation of Bitcoin mining ($\sqrt{\mu_2^2 - \mu_2}$) is $0.73 = 0.27$ (i.e. 27%).

5 Conclusion and Future Work

In this work, we firstly conducted a theoretical analysis on Bitcoin solo mining and graph clique mining, and then proposed a simple search algorithm for graph cliques mining. We accepted that the required time between consecutive blocks follows the exponential distribution. In the evaluation, we perform a small scale evaluation on Bitcoin and graph clique solo minings to validate the correctness of our theoretical evaluation. We investigated the variance issue. It is found experimentally that the the standard deviation of unbiased variance of the graph clique mining is reduced compared with the standard deviation (dispersion) of Bitcoin mining.

In future, we plan to conduct a more complete theoretical evaluation on graph clique solo mining. We also plan to do experiments to study graph clique solo mining. In addition, we plan to compare the Bitcoin pooled mining and the graph clique-based pooled mining under various conditions.

Acknowledgement

In the first stage of this research, Hiroaki Anada, Junpei Kawamoto and Kouichi Sakurai were supported by JSPS Kiban(B) JP15H02711. Hiroaki Anada, Chunhua Su and Kouichi Sakurai are supported by JSPS Kiban(B) JP18H03240. Chunhua Su is also supported by JSPS Kiban(C) JP18K11298. Samiran Bag is supported by the ERC starting grant, no. 306994. The authors would like to thank all anonymous reviewers for their insightful comments and suggestions.

References

1. Antonopoulos, A.M.: Mastering Bitcoin: Unlocking Digital Crypto-Currencies. O'Reilly Media, Inc., 1st edn. (2014)
2. Bag, S., Ruj, S., Sakurai, K.: On the application of clique problem for proof-of-work in cryptocurrencies. In: Information Security and Cryptology - 11th International Conference, Inscrypt 2015, Beijing, China, November 1-3, 2015, Revised Selected Papers. pp. 260–279 (2015), http://dx.doi.org/10.1007/978-3-319-38898-4_16
3. bitcoinwiki: Confirmation, https://en.bitcoin.it/wiki/Confirmation, accessed 15 Dec, 2016
4. Erdős, P., Renyi, A.: On the evolution of random graphs. In: Publication of the Mathematical Institute of the Hungarian Academy of Sciences. pp. 17–61 (1960)
5. Feller, W.: An Introduction to Probability Theory and Its Applications, vol. 1. Wiley (January 1968)
6. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA (1979)
7. Grimmett, G.R., McDiarmid, C.J.H.: On colouring random graphs. Math. Proc. Cambridge Philos. Soc. **77**, 313–324 (1976)
8. Kraft, D.: Difficulty control for blockchain-based consensus systems. Peer-to-Peer Networking and Applications **9**(2), 397–413 (2016). https://doi.org/10.1007/s12083-015-0347-x, http://dx.doi.org/10.1007/s12083-015-0347-x
9. Matsushima, T., Anada, H., Kawamoto, J., Bag, S., Sakurai, K.: Evaluation of bitcoin-mining for search problem of graph cliques. In: Hinokuni Symposium on Information 2016, Miyazaki, Japan, March 2–3, 2016. pp. 4B–2 (2016)
10. Matula, D.: On the complete subgraph of random graph. Combinatory Mathematics and Its Applications pp. 356–369 (1970)
11. Satoshi Nakamoto: Bitcoin: A peer-to-peer electronic cash system (2008), http://bitcoin.org/bitcoin.pdf
12. Rosenfeld, M.: Analysis of bitcoin pooled mining reward systems. CoRR **abs/1112.4980** (2011), http://arxiv.org/abs/1112.4980
13. Saito, R.: Deriving exponential distribution from geometric distribution, http://chianti.ucsd.edu/~rsaito/ENTRY1/WEB_RS3/PDF/JPN/Texts/half_life1_1.pdf, accessed 15 Dec, 2016
14. Swan, M.: Blockchain: Blueprint for a New Economy. O'Reilly Media, Inc., 1st edn. (2015)
15. Tromp, J.: Cuckoo cycle: A memory bound graph-theoretic proof-of-work. In: Financial Cryptography and Data Security - FC 2015 International Workshops, BITCOIN, WAHC, and Wearable, San Juan, Puerto Rico, January 30, 2015, Revised Selected Papers. pp. 49–62 (2015)
16. WIKIPEDIA: Memorylessness, https://en.wikipedia.org/wiki/Memorylessness, accessed 15 Dec, 2016