# Energy Efficient Hardware Design for Crypto Mining Algorithms

Dingyuan Cao, Vishnu Srinivasan

April 30, 2022

## 1 Introduction

In the past few years, there has been a large trend emerging in the growth of crypto-currencies and the mining techniques that go along with them. Large coins such as Bitcoin, have to be mined and this is generally a tedious and time consuming process. Unlike mining with a traditional pick-axe, Bitcoin mining refers to performing and solving complex math puzzles, and if solved, they produce new Bitcoins on the Bitcoin network. For some context, bitcoin mining is performed by high-powered computers that solve complex computational math problems; these problems are so complex that they cannot be solved by hand and are complicated enough to tax even incredibly powerful computers. Given the fact that these currencies are gaining mass popularity and are slowly being adapted on scale, it is logical for these mining methods to be more sustainable and less energy taxing as it will become a hurdle on the long run if it continues this way.

Conventional methods for mining for bitcoin have astronomically high energy consumption rates. For instance, according to recent surveys, the amount of electricity consumed by Bitcoin related mining was equal to about 0.55 percent of the world's annual electricity production, or the equivalent of a small country like Sweden's energy needs. While this accounts for both the usage and mining of bitcoin, it is apparently clear that mining takes and consumes much more energy than the usage of Bitcoin and there have been clear needs emerging for finding better approaches in the same.

For some context, the images and info graphics below show how much energy Bitcoin uses.

## 2 Background

### 2.1 Cryptographic Hash Functions

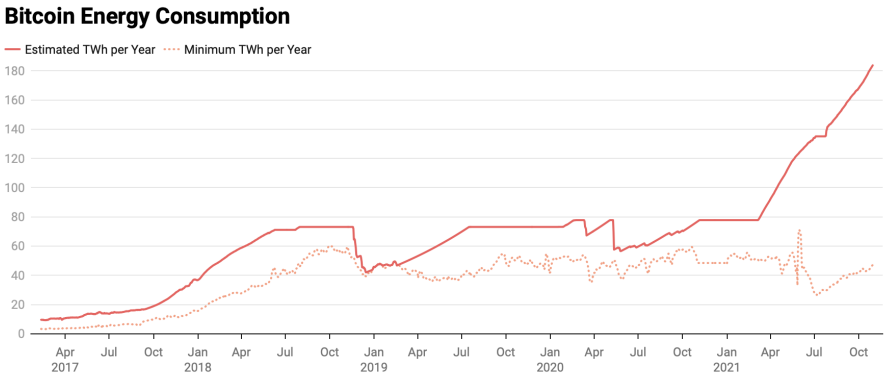
Hash Functions are widely used in computer systems. It takes an arbitrary of input data, and produce an output based on a set of transformation. Given the input data and the hash function, the output of the function is always the same. However, for a given output, there could potentially be multiple inputs corresponding to this same output. The property gives hash functions the ability to decrease the range of the input data, extracting (or "digest") the information inside input data.

## Annualized Total Bitcoin Footprints

| Carbon Footprint | Electrical Energy | Electronic Waste |
|---|---|---|
| 87.28 Mt CO2 | 183.74 TWh | 26.26 kt |
| Comparable to the carbon footprint of **Bangladesh**. | Comparable to the power consumption of **Thailand**. | Comparable to the small IT equipment waste of **the Netherlands**. |

## Single Bitcoin Transaction Footprints

| Carbon Footprint | Electrical Energy | Electronic Waste |
|---|---|---|
| 869.94 kgCO2 | 1831.45 kWh | 261.70 grams |
| Equivalent to the carbon footprint of **1,928,083** VISA transactions or **144,990** hours of watching Youtube. | Equivalent to the power consumption of an average U.S. household over **62.77** days. | Equivalent to the weight of **1.60** iPhones 12 or **0.53** iPads. (Find more info on e-waste here.) |



In a cryptographic setting, the property of hash functions are strengthened. A good cryptographic hash functions should have following properties:

- Non-reversibility. It is easy to compute the hash value of a given input, but it should be impossible reconstruct input data from the hash output.
- Diffusion, or avalanche effect. A slight change in the input data, for example a bit flip, will result in huge change in the output. In other words, the output should appear to be random generated, and the behavior of the hash function is unpredictable.
- Determinism. For same input and hash function, the output should always be the same.
- Collision resistance. Given an input $x_0$ and its output $y_0$, it should be hard to find a different input $x_1 \neq x_0$ such that $H(x_1) = y_0$.
- Non-predictable. The output of the function should not be easily predictable based on the input.

With these good properties, cryptographic hash functions can be used as message digest and digital signature. More and more hash functions have been developed, such as MD5, SHA-256 etc.

## 2.2 Merkle Tree

Merkle Tree is a tree of hash values. In a Merkle Tree, each leaf node is a hash of an input data block. To compute the hash value of a given node, all of its children nodes are evaluated (hashed), then it compute the hash of the concatenation of all its direct children, and store as its own value. After all the computations, the hash for the root node is computed, denoted as the hash of the Merkle Tree, also root hash.

When someone wants to check the integrity of the data they received from an untrusted source, the root hash can be retrieved from a trusted source. Then, a merkle tree can be constructed and computed based on the untrusted data. By comparing the root hash of the constructed merkle tree and the one obtained from trusted authority, the integrity of the data can be verified. Another good property about merkle tree is that partial integrity can be checked by computing hash values of internal nodes. when a hash value of one of the internal nodes is verified, all of its children's integrity is verified.

## 2.3 Bitcoin Mining Algorithm

The algorithim used for Bitcoin mining is SHA-256. This is a member of the SHA-2 set of functions created by the NSA and SHA stands for Secure Hashing Algorithm. The Sha-256 algorithm is based on the Merkle-Damgard construction method, according to which the initial index is divided into blocks immediately after the change is made, and those, in turn, into 16 words. Additionally, this algorithm has some strong technical parameters such as a block size indicator of 64, maximum message length of 33, standad word size of 4, speed of 140 MiB/s, and 64 iterations per cycle. This is the method used in the Bitcoin mining proof of work algorithm.

### 2.4 Process Variation

Process variation refers to transistor parameter variations caused by manufacturing process. As the advancement of technology node, it becomes more and more difficult to manage the process precision, and small shift in parameters can have a huge impact on the performance of transistors. Two major parameters suffering process variations are the threshold voltage ($V_{th}$) and the effective gate length ($L_{eff}$). These parameters are key to transistor performance, and variation on these parameters will generally make the processor run slower , because the clock speed is limited by the slowest path in the pipeline. On the other hand, $V_{th}$ also affect power consumption, for example, low-$V_{th}$ transistors typically consume more energy than high-$V_{th}$ transistors, though they are faster than the latter ones.

### 2.5 Dynamic Voltage And Frequency Scaling

There has always been a relationship between the frequency of a processor and the voltage supplied here. When the voltage is high, the frequency increases and when the voltage gets low, the frequency reduces. This translates into a debate of performance versus power consumption, where the higher the power consumed, the performance increases and vice versa. However, in most real world systems and applications, there are constraints that need to be met in order to provide reasonable and tangible use cases. Due to this, there has been work done in the area and this has led to the emergence of methods to address this and the most prominent one has been Dynamic Voltage and Frequency Scaling. Dynamic Voltage and Frequency Scaling (DVFS) is a method to save energy consumption of electronic devices and to protect them against overheating by automatic sensing and adaptation of their energy consumption.

## 3 Motivation

With the inherent heterogeneity caused by process variation, different cores in a CMP chip will exhibit different performance and power properties. This creates design opportunities for scheduling and power management algorithms. On the other hand, crypto-mining algorithms are really power hungry, which means it would be good to reduce their power consumption. Combining these two observations, we will test crypto-mining programs in a process variation setting, to see the power and performance of such workload.

## 4 Design

We are going to test cryptocurrency mining algorithm on a process variation aware system[9].

### 4.1 High-Level System Design

Two major high-level design issues should be considered here. First is whether all cores must run at a same frequency, or each of the cores can be assigned with a different

frequency. The second is whether the system supports Dynamic Voltage and Frequency Scaling (DVFS), or it has to be locked at a given frequency. Combining these two design factors, we get 4 different configurations of system:

- Uniform Frequency + No DVFS: In this configuration, all the system parameters are set at the very beginning, and the system could not dynamically adjust frequencies in execution. This means that all cores must run on the frequency of the slowest core, and the only difference between cores is the power consumption. The scheduler can try to minimize the power consumption under the given frequency.
- Non-Uniform Frequency + No DVFS: In this configuration, all cores run on their own fastest frequency possible, and each core has different power consumption and frequency. So a scheduler can try to maximize the performance of such system or try to minimize energy consumption.
- Uniform Frequency + DVFS: In this configuration, all cores run on a same frequency and can be regulated by DVFS. Given a power budget, the scheduler can try to optimize performance under this budget.
- Non-Uniform Frequency + DVFS: In this configuration, all cores can run on a different frequency and DVFS can be applied independently on each of them. This gives the scheduler the most freedom to schedule cores and save energy. Scheduler should maximize performance under power budget in this setting (power budget can be tuned for either energy-saving or performance).

## 4.2 LinOpt Algorithm

LinOpt Algorithm tries to solve scheduling problem as a linear optimization problem.

The linear optimization problem is formulated as follow: for $N$ independent variables $x_1, \ldots, x_N$, maximize objective function:

$$g = a_1 x_1 + a_2 x_2 + \cdots + a_N x_N$$

$x_1, \ldots, x_N$ are subject to a set of primary constraints $x_1 \geq 0, \ldots, x_N \geq 0$ and any number of additional constraints:

$$b_1 x_1 + b_2 x_2 + \cdots + b_N x_N \leq B$$
$$\vdots$$

In this problem, we want to find the best voltage value for each cores such that under given power budget we can maximize the performance of the system. The throughput of the system is the average of throughput of each core:

$$TP = \frac{tp_1 + tp_2 + \cdots + tp_N}{N}$$

By definition, $tp_i = f_i \times ipc_i$, where $f_i$ corresponds to the frequency of core $i$ and $ipc_i$ corresponds to instruction-per-cycle of core $i$. Although frequency can affect ipc, the variation of ipc is much larger between different threads than that is affected by frequency, so in this case we assume ipc to be stable through the execution. Frequency is largely a linear function of voltage. with these assumption, we can rewrite the equation above:

$$TP = \frac{a_1}{N}v_1 + \frac{a_2}{N}v_2 + \cdots + \frac{a_N}{N}v_N$$

where $v_i$ corresponds to the voltage of core $i$. The voltages of the cores are constrained by the physical restrictions of cores, and we have:

$$V_{low} \leq v_1, v_2, \ldots, v_N \leq V_{high}$$

Because the limitation of the linear problem, we cannot analytically generate the power $p$ based on the voltage. Instead, we use least square root to fit our linear estimation to the experiment data. for every $p_i$ and $v_i$, we have $p_i = f(v_i) \approx b_i v_i + c_i$. the voltage of each core is constrained by the maximum power for each core, also the total power budget for the chip. Thus we have:

$$b_1v_1 + b_2v_2 + \cdots + b_nv_n + c < P_{target},\ c = \sum c_i,\ \forall i \in 1..N$$
$$b_iv_i + c_i < P_{coremax},\ \forall i \in 1..N$$

With all these constraints, we can use any linear optimization algorithm to solve this problem.

# 5 Methodology

We use Gem5 system and processor simulator[3] to model a large CMP chip with 20 out-of-order cores. We use the analytical power model integrated with Gem5. The configuration of our experiment is summarized below:

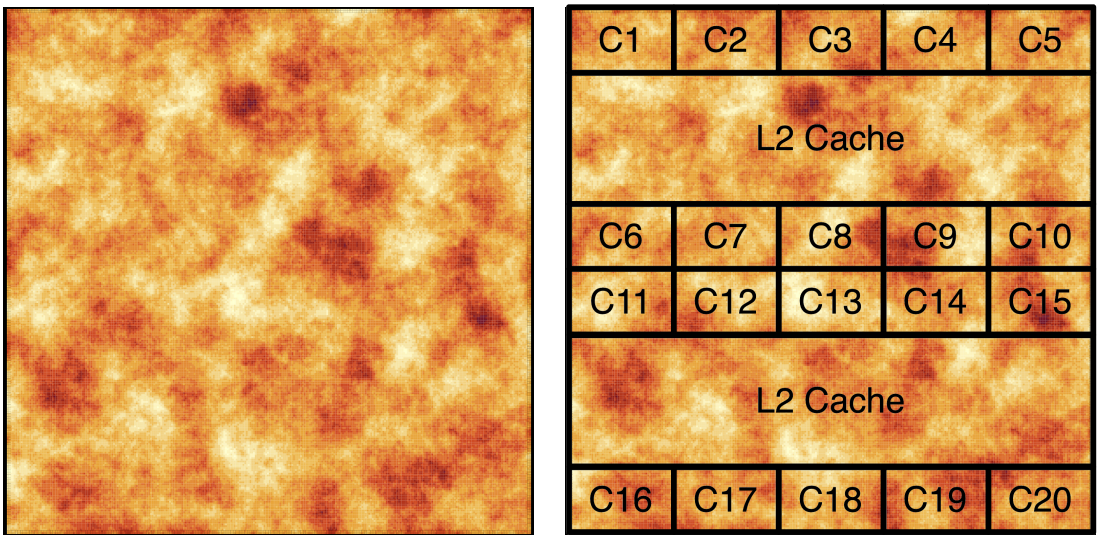
| Config. | Parameters |
|---|---|
| Overall | CMP with 20 out-of-order RISCV-like processor, 4GHz (nominal) |
| Branch Predictor | Tournament branch predictor, 8k globle history buffer, 2k local histroy buffer |
| Pipeline | issue/commit width: 8/8 |
| Cache | 32kB L1D/L1I, shared 10MB L2, shared 32MB L3, line size: 64B |
| Memory | DDR3 1600MHz, 8 channel, 8 bank per channel |

## 5.1 Process Variation Model

We use the VARIUS model[8] to simulate process variation within die. First, we use VARIUS model to generate a spatial map for $V_{th}$ and $L_{eff}$ variation (Figure 1(a)).

Then, we superpose the chip floor plan on top of the variation map (Figure 1(b)). With this variation map, combining critical path model described in [8], we can derive frequency and power characteristics of each core. This serves as the profiling process in real world applications.

| Config. | Parameter |
|---|---|
| $V_{DD}$ | 0.6-1.0V, nominal voltage is 1.0V |
| Die size | 340mm$^2$ |
| Number of dies | 20 |
| $V_{th}$ | $\mu : 250mV, \sigma/\mu : 0.12, \phi : 0.5$ |




(a) Variation Map.

(b) Variation Map with Floorplan.

Figure 1: Variation Model

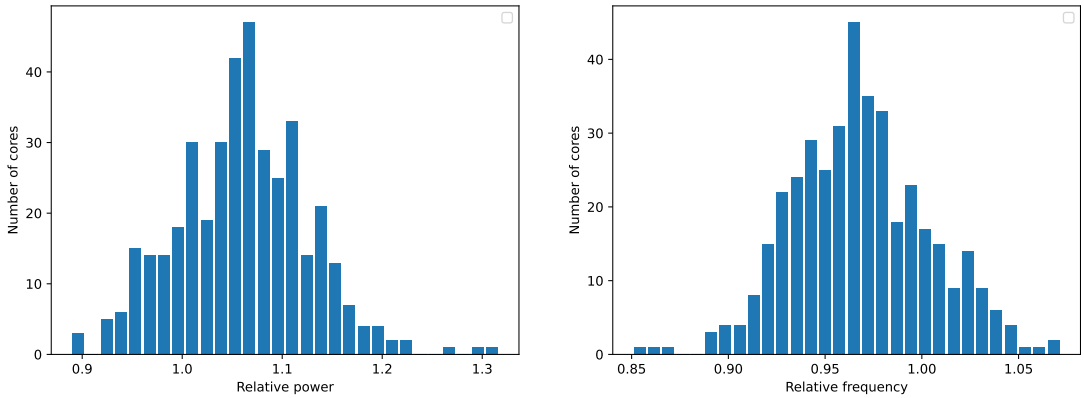
### 5.2 Workloads

We select applications from SPECint (*bzip*2, *crafty*, *gap*, *gzip*, *mcf*, *parser*, *twolf*, and *vortex*), to evaluate our system under a range of settings. We usea open source crypto-mining hashing poof of work solver[1] to simulate the mining process. In each experiment run, 2 to 20 applications are chosen, and the crypto-mining application is guaranteed to be chosen in each run. Each experiment is repeated 10 times, and the result is the average outcome of the 10 trails.

## 6 Evaluation

### 6.1 Power and Performance Variation

First, we evaluate the process variation effect on the power and frequency of the chips. We evaluate 400 cores with process variation, and estimate the frequency and power based on $V_{th}$ and $L_{eff}$ variation map. The result is shown in figure 2.



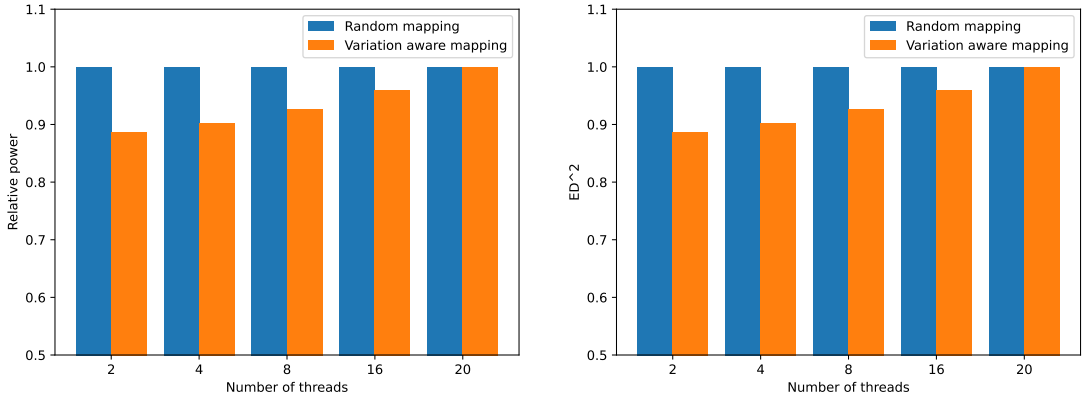

(a) Power Variation. (b) Frequency Variation.

Figure 2: Variation Effect on Power and Frequency

From these figures we can see that the power and frequency property vary greatly across cores. For example, the most power hungry core can consume 48% more energy than the most power efficient one. This shows great opportunity of variation aware scheduling algorithm.

## 6.2 Uniform Frequency with No DVFS

Under this configuration, the scheduler has limited control over hardware. The results are shown in figure 3. All the numbers are normalized to random mapping.



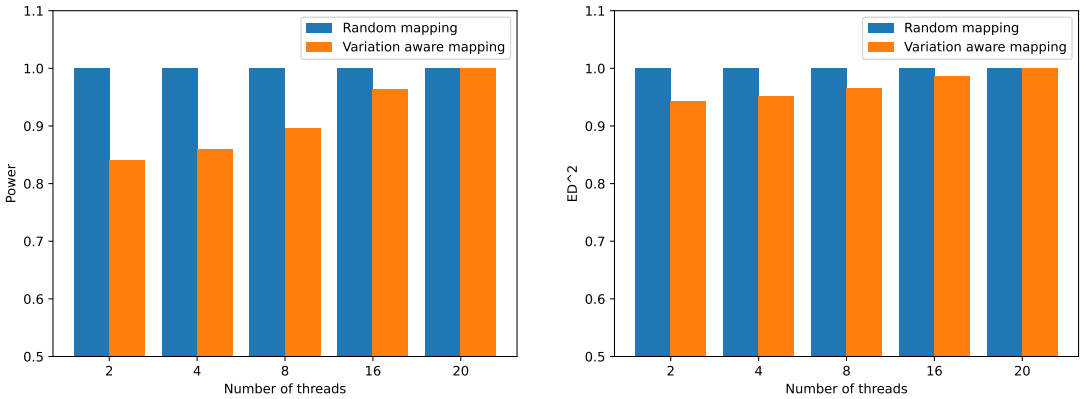

(a) Power (b) $ED^2$

Figure 3: Power and $ED^2$ in uniform frequency setting.

From the figure, we can see that as the number of threads grows, the performance gain is decreasing. This is because that when the number of threads is small, the scheduler have more opportunity to schedule the threads to the most energy efficient cores, while under high load there's little space for scheduling. $ED^2$ shows similar improvement as power under this setting.

## 6.3 Nonuniform Frequency with no DVFS

Compared to the previous setting, cores in this setting can run under different frequencies. This means that faster cores will no longer be constrained by the slower cores. We use the scheduler to try to minimize power of the system, and the results are shown in figure 4.




(a) Power (b) $ED^2$

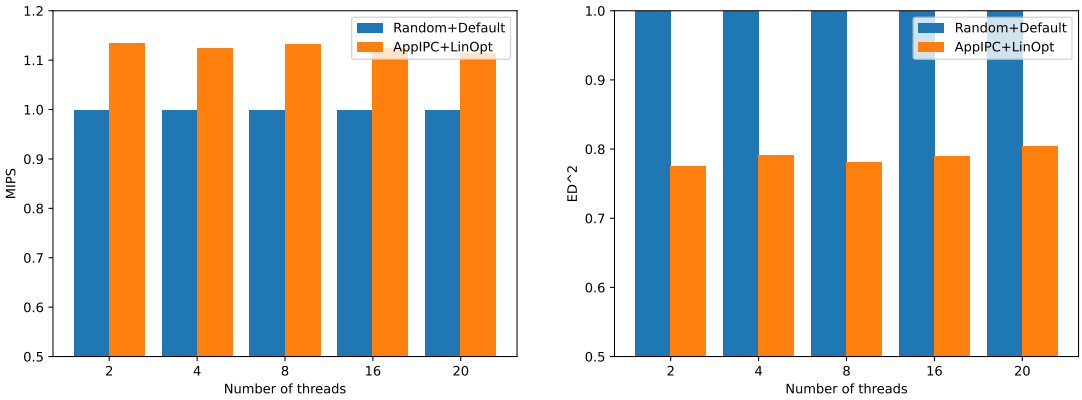
Figure 4: Power and $ED^2$ in nonuniform frequency setting.

Under this setting, the power saving is similar to the results in 6.2. However, the $ED^2$ is a little bit worse than that in the uniform frequency setting. This is because under this configuration, we maybe end up mapping a thread to an energy efficient but way slower core, thus affect the overall $ED^2$.

## 6.4 Nonuniform Frequency and DVFS

In this part, we test our system in a DVFS setting. The power budget of the system is set to 50W, and we want to maximize performance and throughput under this budget. First, as a naive approach, we can map workloads with higher IPC to higher performance cores. This is because for those workloads with lower IPCs, the performance is more likely to be constrained by memory performance, which is out of the scope of this project. Then, we apply the linear optimization technique mentioned in 4.2, to further tune the performance. The results are shown in figure 5.

We can see from the figures, after adapting linear optimization and application IPC aware mapping, the overall performance (MIPS) increased 11-13%, and $ED^2$ decreased by more than 20%, compared to the baseline system, which maps workload randomly and adapt default DVFS setting (not variation aware). Limited by time, we did not implement the simulated annealing technique to compare with linear optimization algorithm, but the linear optimization algorithm itself shows great advantage against baseline system.




(a) The MIPS of the system.

(b) $ED^2$

Figure 5: MIPS and $ED^2$ in nonuniform frequency and DVFS setting.

# 7 Conclusion

In this project, we implemented a process variation aware system, and applied scheduling and mapping algorithms to minimize power or maximize performance under given power budget. We adopted linear optimization algorithm described in [9], and tested several workloads on the system, including a crypto-mining benchmarks. The results showed that there are great potential in scheduling for process variation system, and linear optimization can greatly improve the energy efficiency of the system.

# 8 Related Works

This project is mainly based on [9]. We re-implemented the system with similar configurations in Gem5, and tested crypto-mining algorithms in the system. The variation model is adapted from [8]. Additionally, there has been a lot of work done in the domain in order to compare the different workloads and energy consumption of crypto mining algorithms. According to [4], talks about the different power consumption metrics and utilization numbers in regards to bitcoin and compares it to large scale metrics in the world. This paper also then compares the different metrics for the top twenty coins and looks into them individually, basing their metrics on the mining devices and the hash hit rate. This work provides some important and useful context in regards to the baselines that generally exist cross coins in the field right now, and how some projects utilize this to emerge as a low energy alternate. Similarly, in [6], the energy and power consumption of different coin mining are measured and evaluated in regards to different units (energy per coin etc). Additionally, it also looks into more energy efficient methods to manage power and energy in other crypto currencies, namely Monero and does a deep dive into this and looks into more efficient mining approaches here. From a transaction verification point of view, there has been work done in [11] where the work proposed the development of a Proof OF Work nonce calculation methods where these methods enable the acceleration of the transaction verification process especially in solo mining,

Moving away from generalized analysis and methods, the works from [10] looks

into a novel method to reduce the energy costs associated with mining. Unlike the traditional Proof of Work Method, this work proposes a new method known as Proof Of Contribution which is a modified version of Proof Of Work in order to improve mining efficiency. This method is equally or similarly secure as Proof Of Work, and based on experiments conducted this has a better overall result in regards to energy efficiency and power management metrics. Similar to this, there have been other methods that have been built upon the existing Proof Of Work consensus, such as the work in [5] which talks about methods to reduce energy consumption by large amounts in mining processes by utilizing a system of exclusive computation "rights" for runner ups in certain situations, and this is able to reduce the overall mining energy consumption by about 50 % at its best performing case.

There has also been work in non traditional set ups in order to reduce the energy consumption of crypto mining algorithms. For instance, [2] talks about the utilization of a group of quantum servers on which to build a quantum enabled blockchain architecture. This novel method utilizes a different hardware and system set up to extract energy savings and is able to achieve a reasonable set of results for the same. This builds upon the premise of using smarter and more advanced communication networks to help with the mining process, and this is also noticeable in the methods employed by [7], where the work talks about using Distributed Generation Resources to monitor the power management of mining protocols and programs in order to reduce the overall energy consumption.

# References


[1] Equihash proof-of-work solvers. https://github.com/tromp/equihash. Accessed: 2021-12-12.

[2] Adam J Bennet and Shakib Daryanoosh. Energy-efficient mining on a quantum-enabled blockchain using light. *Ledger*, 4, Jul 2019.

[3] Nathan Binkert, Bradford Beckmann, Gabriel Black, Steven K. Reinhardt, Ali Saidi, Arkaprava Basu, Joel Hestness, Derek R. Hower, Tushar Krishna, Somayeh Sardashti, Rathijit Sen, Korey Sewell, Muhammad Shoaib, Nilay Vaish, Mark D. Hill, and David A. Wood. The gem5 simulator. *SIGARCH Comput. Archit. News*, 39(2):1–7, aug 2011.

[4] Ulrich Gallersdörfer, Lena Klaaßen, and Christian Stoll. Energy consumption of cryptocurrencies beyond bitcoin. *Joule*, 4(9):1843–1846, 2020.

[5] Noureddine Lasla, Lina Alsahan, Mohamed Abdallah, and Mohamed Younis. Green-pow: An energy-efficient blockchain proof-of-work consensus algorithm. 2020.

[6] Jingming Li, Nianping Li, Jinqing Peng, Haijiao Cui, and Zhibin Wu. Energy consumption of cryptocurrency mining: A study of electricity consumption in mining cryptocurrencies. *Energy*, 168:160–168, 2019.

[7] Mohamed A. Mohamed, Seyedali Mirjalili, Udaya Dampage, Saleh H. Salmen, Sami Al Obaid, and Andres Annuk. A cost-efficient-based cooperative allocation of mining devices and renewable resources enhancing blockchain architecture. *Sustainability*, 13(18), 2021.

[8] Radu Teodorescu, Brian Greskamp, Jun Nakano, Smruti Sarangi, Abhishek Tiwari, and Josep Torrellas. Varius: A model of parameter variation and resulting timing errors for microarchitects. 01 2007.

[9] Radu Teodorescu and Josep Torrellas. Variation-aware application scheduling and power management for chip multiprocessors. In *Proceedings of the 35th Annual International Symposium on Computer Architecture*, ISCA '08, page 363–374, USA, 2008. IEEE Computer Society.

[10] T. Xue, Y. Yuan, Z. Ahmed, K. Moniz, G. Cao, and C. Wang. Proof of contribution: A modification of proof of work to increase mining efficiency. 1:636–644, 2018. Cited By :21.

[11] Muhammad Muneeb Omair Shafiq Zeeshan Raza, Irfan ul Haq. Energy efficient multiprocessing solo mining algorithms for public blockchain systems. *Scientific Programming*, 2021.