



A Review of Hashing Algorithms in Cryptocurrency

Erkan ÜNSAL^{*}, Humar KAHRAMANLI ÖRNEK² and Şakir TAŞDEMİR³

¹Computer Engineering/Institute of Sciences, Selçuk University, Turkey

²Computer Engineering/Institute of Sciences, Selçuk University, Turkey

³Computer Engineering/Institute of Sciences, Selçuk University, Turkey

[*erkanunsal@engineer.com](mailto:erkanunsal@engineer.com) Email of the corresponding author

Abstract – In this study, it is aimed to make a detailed examination of the hashing algorithms utilized in cryptocurrencies. In this direction, basic information about hashing which forms the basis of hashing algorithms used in cryptocurrencies and plays an important role in many fields such as cryptology and blockchain, its purpose, structure, working style and usage areas are given. Additionally, to present hashing more clearly, the example of modulo operation, which is one of the easy to understand hashing functions, is visualized with the table and the resulting collision situation is schematized. By mentioning the hash table used with hashing to store and retrieve data items or records, the first step of the study is completed. Since the hashing algorithms used in cryptocurrencies are cryptographic, in the continuation of our study, after examining the features that should have by understanding the purpose and structure of cryptographic hashing algorithms more clearly, it is determined how an ideal cryptographic hashing algorithm should be. Subsequently, as the most important part of the study, cryptographic hashing algorithms SHA256, Ethash, Scrypt, Equihash, RandomX, X11, Lyra2Z and Lyra2REv2, which are designed to be utilized only by a certain cryptocurrency or play a fundamental role in the formation of several cryptocurrencies, are discussed one by one. The study was concluded by examining the creators of these algorithms, for what purposes they were created, their features, structures, working methods and areas of use. The specified aim was achieved by sharing the results obtained at the end of the study.

Keywords – Hash, Hashing, Algorithm, Cryptographic, Cryptocurrency

I. INTRODUCTION

In this study, it is aimed to examine in detail the hashing, which has an important role in many fields such as cryptology, blockchain and cryptocurrency, and the hash table used with hashing to store and retrieve data items or records, the cryptographic hashing algorithms and their properties that can fulfill the special requirements for cryptographic applications, as well as the ones used in cryptocurrencies from these algorithms.

In chapter 2, hashing and hash table, in chapter 3, cryptographic hashing algorithms, their features and cryptographic hashing algorithms used in

cryptocurrencies are discussed in detail. The results are provided in the last section.

II. HASHING

Any function that is used to control the integrity of the data and that matches fixed size values as a result of a special algorithmic processing of all its bits from the first sector to the last sector of the random size data to which it belongs is called hashing [1]. The root of the word hash is the same as the Arabic word hashish. And inspired by the deformation it has caused on humans, it has taken this name because of the deformation caused on the information entering the hashing function [2].

The hash function generates a fixed-length hash value regardless of the data it hashes. It works unidirectionally, that is, it is not possible to reach the source data from the hash value. Even small data changes produce very different hash information, and this feature is called the avalanche effect [3].

The values returned by a hashing function are called hash values, hash codes, or simply hashes. Values are often used to index a fixed-size table called a hash table. Using a hashing function to index a hash table is called hashing or scatter storage addressing. The hashing function and associated hash tables are used in data storage and retrieval applications to access data with a small and nearly constant access time [4]. It is generally used in the database for operations such as quickly finding a data searched in a table or speeding up data comparison processes, detecting the same or similar records in a large file, finding similar sequences in a DNA sequence [5].

The modulo operation is an easy to understand hashing function. Accordingly, mod 10 results of numbers 1, 2, 5, 6, 7, 10, 23, 34, 48, 59, 61, 72, 85, 96, 107, 210, 323, 434, 548, 659, 761, 872, 999 are listed and grouped in Table 1.

Table 1. Hashing

Bunch (Bouquet)	Numbers		
0	10	210	
1	1	61	761
2	2	72	872
3	23	323	
4	34	434	
5	5	85	
6	6	96	
7	7	107	
8	48	548	
9	59	659	999

As shown in Table 1, the numbers are all hashed into a single digit number. Certainly, there are multiple numbers hashed up to the same number. This situation is called collision.

Considering the example, the collision situation is schematized in Figure 1. In case of collision, data belonging to the same key start to branch as a linked list from the key they collide. To exemplify, when 2 different data come to key 2, these data will build a linked list from key 2 onwards.

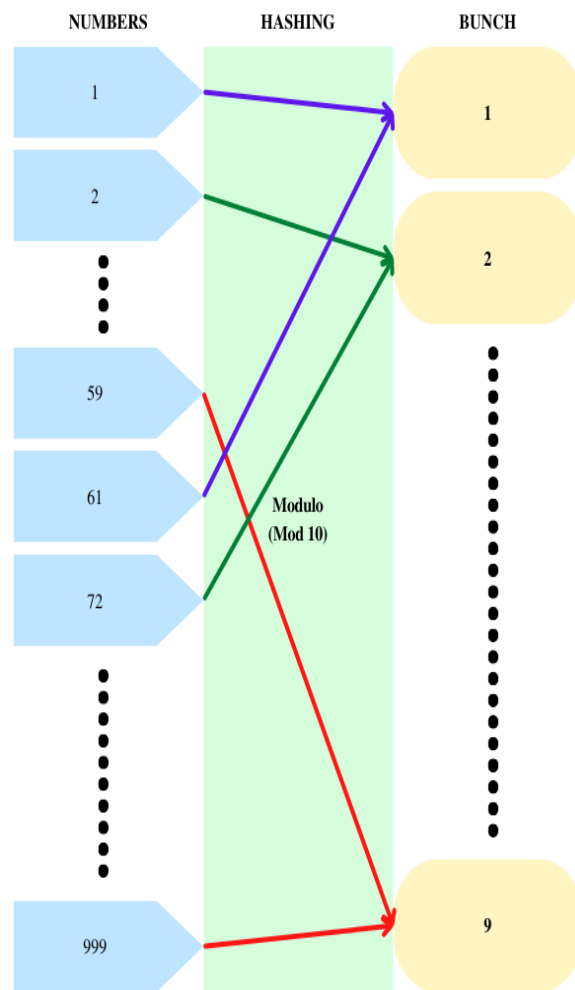


Fig. 1 Collision

A good hashing function should be really fast to compute, and it should minimize output value collision.

A. Hash Table

The hashing function is used with hash tables to store and retrieve data items or data records. It also returns the key associated with each data or record into a hash code used to index the hash table. When adding an item to the table, the hash code may index an empty field, in which case the item is added to the table there. If the hash code is indexing a full field, some kind of collision resolution is required: the new item can be skipped, replaced by the old item, or added to the table in another location by a specified procedure. This procedure depends on the structure of the hash table.

Hash tables are also used to implement associative arrays and dynamic sets [6].

III. CRYPTOGRAPHIC HASHING ALGORITHMS

Cryptographic hashing algorithms are a set of operations that convert any length of input data into a fixed-length output. It is a hashing algorithm that can fulfill specific requirements for a cryptographic application [7].

A. Features of Cryptographic Hashing Algorithms

The output of cryptographic hashing algorithms must have certain properties. These properties are called pre-image resistance, second pre-image resistance, and collision resistance, respectively [8].

The pre-image resistance specifies that the hashing algorithm should be a one-way algorithm. That is, it should not be possible for an attacker to determine the original data from a given hash value. The second pre-image resistance should be difficult for any message given to the attacker to find another message that is different from the given message and has the same hash. Collision resistance means that each message has 12 unique hash values and that it is difficult for an attacker to find two messages with the same hash value. It is implied with the phrase "hard" or "hard to find" that the computer needs a long time to perform this operation and requires a large amount of memory [9]. To exemplify, calculating a message via its hash value requires many years and a very large amount of memory for a computer with today's technology standards. Therefore, the calculation made is considered to be inapplicable. As the computing power of computers has grown over the decades, some hashing algorithms that were previously assumed to be secure (having all the features of pre-image resistance, second pre-image resistance, collision resistance) are now considered cracked [10]. As the computational power increased and crypto analysis of hashing algorithms was performed, some hashing algorithms standards were also revised as they were weak. It is desired to have hashing algorithms that produce output safely and quickly [11].

In practical cryptography applications, difficulty is defined as the degree to which enemies attacking the system cannot break the system as long as the security of the system is important. According to the definition, the need for security may vary depending on the nature of the application [12]. It can be assumed that the effort of the enemies is proportional to the value that will be gained by cracking the application. The effort required to crack the system increases very quickly with the

length of the hash. Thus, by increasing the hash length by a few tens of bits, it is possible to increase the effort required to crack the system by the thousands of times. This situation will make cracking the system inefficient according to the value to be obtained.

In theoretical meaning, the difficulty is mathematically defined. A security system that cannot be cracked in asymptotic polynomial time can be considered hard to break. Although such definitions are important in terms of demonstrable security, they may be far from practical. An exponential time algorithm that is considered slow in theory may run fast enough in practice to crack the system. Similarly, a polynomial time algorithm that is considered fast in theory may not be used because it is too slow in practice [13].

An ideal cryptographic hashing algorithm should provide these four properties: (1) It should be easy to calculate the hash for any message. (2) It must be difficult to compose the message to correspond to a hash. (3) It should be difficult to change the message in the way that the hash does not change. (4) It should be hard to find two different messages with the same hash.

B. Hashing Algorithms Utilized by Cryptocurrencies

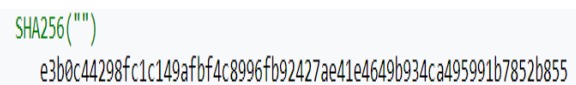
Some cryptographic hashing algorithms are designed for use only by a particular cryptocurrency or have played a fundamental role in the creation of several cryptocurrencies.

1) SHA256

It is one of the algorithms created based on SHA2. SHA2(Secure Hashing Algorithm 2) is a set of cryptographic hashing algorithms designed by the United States National Security Agency (NSA) and first published in 2001.

The secret to the number 256 in this algorithm is that the algorithm resizes your input to 256 bits with 64 characters no matter what size it is. It is collision-proof [14].

An example of SHA256 is shown in Figure 2.



```
SHA256("")
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Fig. 2 SHA256

The SHA256 cryptographic hashing algorithm is used in the infrastructure of Bitcoin, Bitcoin Cash,

Counterparty, LBRY, Mazacoin, Namecoin, Peercoin and Bitcoin cryptocurrencies.

SHA256 is one of the most reliable hashing algorithms used today. It is used in many cryptocurrency infrastructures that use the blockchain structure. The security and accuracy of the blocks in these blockchains is ensured by using hashing algorithms that pay attention to all previous blocks and convert data of different lengths into bit strings of a certain length. Blocks that keep a list of transactions created by nodes in the network also contain the hash value of the previous block [15]. For each new block created, a ledger is kept at the nodes on the distributed network. Blockchain technology and distributed ledger technologies have characteristics related to continuity, scalability, energy consumption, security, privacy, and protection of personal and sensitive data [16].

The integrity of Bitcoin transactions depends on the collision resistance and the pre-image resistance of the SHA256 hashing algorithm. Bitcoin mining is entirely based on double SHA256 processing of a specific type of input.

In the Bitcoin network, the block information containing the current transactions between users, the block number, the "nonce" value, and the hash code of the previous block are combined and converted into a hash code in accordance with the SHA256 standard.

The hash code of 2,684 transfer transactions in block 629,334 created in the Bitcoin network is displayed in Figure 3.

```
"000000000000000000000000bfc7df45983354cc12847745459  
a0e4c834f77ddb4c9b"
```

Fig. 3 Hash Code

As it is seen, more than 2,600 process information is converted to a hash output of only 64 characters. Thus, instead of examining all the data for any change in the block, it is possible to control only with this hash data.

Blocks in the Bitcoin network contain the hash codes of the previous block. Therefore, when the data in the block is manipulated, the retrospective change will be understood immediately, since the hash code will also differ.

2) *Ethash*

The creators of this algorithm are Vitalik Buterin and Thaddeus Dryja. Both developed this system between 2013 and 2014. It is the algorithm that renders many cryptocurrencies possible such as Ethereum, Ethereum Classic, WBTC, Metaverse ETP, Expanse, Musicoin, Ellatism, Elementrem, WhaleCoin, DaxxCoin, Bowhead and Ethereum Fog. This algorithm takes advantage of very advanced computing techniques that increase the level of security in the network [17]. Although its full name is Ethash-Dagger-Hashimoto, it should be noted that Dagger-Hashimoto is the mining algorithm that served as the basis for the creation of the current Ethash.

Dragger is an algorithm developed by the same Ethereum creator, Buterin. At the beginning of its development, the algorithm's build occupied about 1GB(gigabyte) in storage, but over the years its storage has been expanded to 4GB or 5GB. The structure of this algorithm allows the Hashimoto mining process, which is another algorithm that forms the basis of the Ethash algorithm. Hashimoto is the second basic algorithm of Ethash, developed by Thaddeus Dryja in order to perform hash mining on the system it is applied to. The algorithm itself increases RAM(Random-Access Memory) consumption and thus limits ASICs(Application-Specific Integrated Circuit). From the combination of the two algorithms comes another unique algorithm that offers the ability to design a new mining system that is too complex for ASIC miners to implement effectively. In fact, its structure is so advanced that many developers are trying to implement it as an alternative option to Scrypt. The operation of Ethash is completely different from the basic operation of Dragger-Hashimoto, the algorithm on which it is based. However, there are some key features that remain in place.

To arrive at current performance, Ethash reviewed 23 versions of its algorithm. However, there are some unchanging aspects such as the use of SHA3-256(Keccak-256) and SHA3-512(Keccak-512) algorithms.

When such a high number of updates are seen, it can be thought that the developers are not satisfied or the project is incomplete, but each update corresponds to the solution of the problems that the algorithm can present. Updates were made to improve, secure and fix the algorithm, but above all to increase the difficulty of implementation in

ASIC. In this way, an increasingly GPU(Graphics Processing Unit)-friendly algorithm was created.

3) *Scrypt*

It is a password-based key derivation function created by Colin Percival for the Tarsnap online backup service [18]. This algorithm is specifically designed to require large amounts of memory, making it expensive to carry out large-scale custom hardware attacks. It is inspired by SHA256 and is simpler and faster than SHA256. It needs memory. The need for memory is a compelling factor for performing large-scale custom hardware attacks. A simplified version of the Scrypt algorithm, implemented by an anonymous programmer with the ArtForz username, was used as a proof-of-work scheme, first at Tenebrix and then by a number of cryptocurrencies, including Fairbrix and Litecoin. It is also the hashing algorithm used in the infrastructure of many cryptocurrencies such as Auroracoin, Bitconnect, Coinye and Dogecoin.

Scrypt's large memory requirements are provided by a large vector of bit strings generated as part of the algorithm. After the vector is created, its elements are accessed in pseudo-random order and combined to produce the derived key. The entire vector of a simple application needs to be kept in RAM, so it can be accessed as needed.

Since the elements of the vector are generated algorithmically, each element can be generated as required at once, therefore, each element can be generated as required on the fly, in a way which will significantly reduce memory requirements. However, generating each element is intended to be computationally expensive, and the elements are expected to be accessed many times throughout the execution of the algorithm. In this way, the speed must be renounced in a significant to a considerable extent in order to avoid large memory requirements. Such a time-memory swap is usually available in computer algorithms: speed can be increased at the cost of using more memory, or memory requirements can be reduced at the cost of performing more operations and lengthening the required process. The idea behind the Scrypt is to intentionally make this swap costly in both directions. Thus, an attacker can use an application that does not require a lot of resources but runs very slowly, or they can use an application that runs faster to be in parallel but has a huge memory requirement and is therefore more expensive.

4) *Equihash*

It is a memory-hard proof-of-work algorithm introduced by the University of Luxembourg's Interdisciplinary Center for Security, Reliability and Trust (SnT) at the 2016 Symposium on Network and Distributed Systems Security. The main goal of its developers was to get rid of ASICs in the network, as they threaten centralization due to high computing power density. Along with the Ethash algorithm, Equihash was the pioneer of the GPU mining community. The algorithm is based on the generalization of the birthday problem that finds colliding hash values. It has serious time-space swaps but admits vulnerability to unpredictable parallel optimizations [19].

In an attempt to worsen the cost-performance swaps of designing custom ASIC applications, it is designed in such a way that parallel applications are congested by memory bandwidth. The ASIC resistor in Equihash is based on the assumption that commercially sold hardware already has fairly high memory bandwidth, so improvements by custom hardware may not be worth the development cost.

It is the hashing algorithm used in the infrastructure of many cryptocurrencies such as Bitcoin Gold, ZCash, Bitcoin Private, Komodo, ZenCash, ZClassic, BitcoinZ, Hush, Zero, Bitgem, Zcash.

5) *RandomX*

RandomX is a proof-of-work algorithm optimized for general purpose GPUs and CPUs(Central Processing Unit). Its main feature is random code execution along with several techniques. This combination will be fully appreciated by developers because it will minimize the efficiency advantage of dedicated hardware.

RandomX uses a virtual machine that executes programs on a special instruction set. These programs can be instantly assembled to the CPU's native machine code. As a result, the outputs of the executed programs are combined into a 256-bit result using the Blake2b cryptographic hashing algorithm.

RandomX can operate in two main modes that differ in memory requirements. Fast mode(2181 MB(Megabyte)) and light mode(268 MB) require shared memory. Both modes can be used interchangeably as they provide the same results in the end. Quick mode is suitable for mining.

It is an algorithm utilized in the infrastructure of many cryptocurrencies such as Monero, Quantum Resistant Ledger, Dynasity Coin, Dero, LOKI, WOWNERO and ITALOCOIN.

6) X11

The X11 algorithm was developed in 2014 by Evan Duffield, the main developer of the Darkcoin cryptocurrency (later Dash). Initially, he was tasked with creating an algorithm that would protect cryptocurrencies from the private mining devices of ASICs, which are considered the killers of decentralization.

The X11 algorithm uses multiple rounds consisting of 11 different hashing algorithms, making it one of the safest and more sophisticated cryptographic hashing algorithms used by modern cryptocurrencies. It was intended to make ASICs much more difficult to create, thus giving the currency ample time to evolve before the centralization of mining became a threat. This approach has been successful to a large extent. To do this, Evan Duffield combined 11 different hashing algorithms (Blake, BMW, Groestl, JH, Keccak, Skein, Luffa, Cubehash, Shavite, Simd, Echo) into a single algorithm [20].

X11 is one of the most energy efficient algorithms currently in existence. Because graphics cards don't need to use a lot of processing power, the algorithm can reduce heat by 30% for GPU miners. It provides faster hashing for CPUs as well as providing cooler GPUs.

It is an algorithm used by many cryptocurrencies such as Dash, Hatch, Pura, SmartCoin, CannabisCoin, Influxcoin, StartCoin, Onix, Sibcoin, Cream, ArcticCoin, Polis, Quebecoin.

7) Lyra2Z and Lyra2REv2

Lyra2 is a password hashing scheme that can also work as a key derivation function (KDF). Lyra2Z and Lyra2REv2 were designed based on Lyra2 [21].

Lyra2Z is a chain algorithm that uses Blake256 for the first round and Lyra2 for the last round. It is the algorithm utilized in the infrastructure of many cryptocurrencies such as Zcoin, GINcoin, Zoin, Criptoreal, Taler.

Lyra2REv2 is a proof-of-work algorithm written for Vertcoin. It is a chain-based algorithm with various hashing algorithms (Blake, Keccak, Cubehash, Lyra2, Skein and BMW). It is the algorithm used in the infrastructure of many

cryptocurrencies such as MonaCoin, Rupee, Straks, Verge, Shield and Galactrum.

IV. CONCLUSION

In the study, which started with the aim of making a detailed examination of the hashing algorithms used in cryptocurrency, all the information obtained about hashing, which is the basis of the hashing algorithms used in cryptocurrency, was presented in a specific way. In addition, modulo operation, which is one of the easy-to-understand hashing functions, is exemplified and the resulting collision situation is schematized. The use of the hash table used with hashing has also been adopted. Since the hashing algorithms used in cryptocurrencies are cryptographic, after analyzing the purpose and structure of these algorithms more clearly and examining the features they should have, it was also determined how an ideal cryptographic hashing algorithm should be. As a result, the basics of the hashing algorithms used in the cryptocurrency that is examined one by one have been fully comprehended. Then, as the most substantial part of the study, the study was concluded by determining the advantages and disadvantages of these algorithms by examining the cryptographic hashing algorithms SHA256, Ethash, Scrypt, Equihash, RandomX, X11, Lyra2Z and Lyra2REv2, which are designed to be used only by a certain cryptocurrency or play a fundamental role in the creation of several cryptocurrencies, their developers, for what purposes they were created, their features, structures, working methods and areas of use. By sharing all the results and information obtained, the planned goal was achieved.

REFERENCES

- [1] H. H. Okuyucu, "Hash Fonksiyonlarının Adli Bilişimde Uygulamaları ve C++ İle Şifreleme Algoritması Tasarımı", PhD Thesis, Karabük University, Karabük, Turkey, 2020.
- [2] Ş. E. Şeker. (2008) Özetleme Fonksiyonları (Hash Function) page on Bilgisayar Kavramları. [Online]. Available: <https://bilgisayarkavramlari.com/2008/05/26/ozetleme-fonksiyonlari-hash-function/>
- [3] E. Yavuz, H. Avunduk, "Tedarik Zinciri Yönetiminde Blok Zincir Teknolojisinin Kullanımı", *Izmir Democracy University Social Sciences Journal*, vol. 4.1, pp. 33-56, 2021.
- [4] R. Sedgewick, *Algorithms in Java*, Parts 1-4, Addison-Wesley Professional, 2002.

- [5] D. E. Knuth, *The art of computer programming*, vol. 3: Searching and sorting. Reading MA: Addison-Wisley, 1973.
- [6] J. Stokes, "Understanding CPU caching and performance", Technical report, Ars Technica, LLC, 2002.
- [7] A. J. Menezes, P. C. Van Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 2018.
- [8] A. Usta and S. Dođantekin, *Blockchain 101*, MediaCat Kitapları, İstanbul, 2017.
- [9] M. Kınacı, "Blok Zinciri Teknolojisi ve Akıllı Sözleşmelerin Yaygınlaşmasının Önündeki Engeller", master's thesis, Bahçeşehir Üniversitesi, İstanbul, Turkey, 2019.
- [10] C. R. Dougherty, "Vulnerability Note VU# 836068 MD5 vulnerable to collision attacks", Vulnerability notes database, CERT Carnegie Mellon University Software Engineering Institute (2008), 2019.
- [11] M. Ciampa, *CompTIA Security+ 2008 in depth*. Course Technology Press, 2008.
- [12] G. Leurent, T. Peyrin, "SHA-1 is a shambles: First chosen-prefix collision on SHA-1 and application to the PGP web of trust", in *Proceedings of the 29th USENIX Conference on Security Symposium*, 2020, p. 1839-1856.
- [13] F. Mendel, N. Pramstaller, C. Rechberger, M. Kontak and J. Szmıdt, "Cryptanalysis of the GOST Hash Function", *Advances in Cryptology-CRYPTO 2008: 28th Annual International Cryptology Conference*, Santa Barbara, CA, USA, Proceedings 28, Springer Berlin Heidelberg, 2008, p. 162-178.
- [14] E. Balcısoy, "Yüksek Performanslı Bitcoin Madenciliđi İçin SHA256 Özet Algoritmasının Eniyilenmesi", master's thesis, TOBB Economy and Technology University, Ankara, Turkey, 2017.
- [15] M. Nofer, P. Gomber, O. Hinz and D. Schiereck, "Blockchain", *Business and Information Systems Engineering*, vol. 59, pp. 183-187, 2017.
- [16] O. Özeltın, M. Ersoy, "Kamu Yönetiminde Blokzincir Kullanımı: D5 Örneđi", *Nevşehir Hacı Bektaş Veli Üniversitesi SBE Dergisi*, vol. 10.2, pp. 746-763, 2020.
- [17] O. Esuruoso, *High Speed FPGA Implementation of Cryptographic Hash Function*, University of Windsor, Canada, 2011.
- [18] *The script password-based key derivation function*, C. Percival and S. Josefsson, 2016.
- [19] A. Biryukov and D. Khovratovich, "Equihash: Asymmetric proof-of-work based on the generalized birthday problem." *Ledger*, vol 2, pp. 1-30, 2017.
- [20] D. LEE, L. Low, "Inclusive fintech: blockchain, cryptocurrency and ICO", *World Scientific*, 2018.
- [21] M. Van Beirendonck, et al. "A Lyra2 FPGA core for Lyra2REv2-based cryptocurrencies", *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, IEEE, 2019, p. 1-5.