UNIVERSITY OF THESSALY

SCHOOL OF SCIENCE

INFORMATICS AND COMPUTATIONAL BIOMEDICINE

Implementing A Church–Turing–Deutsch Principle Machine on a Blockchain.

Konstantinos Sgantzos.

Master thesis

Supervising Professor:
Athanasios Kakarountas, PhD

Lamia

2017-2018

Abstract

Genetic Algorithms are the elementary particles of a brand-new world of computing. In recent years, technology has evolved exponentially in terms of Hardware, but not in terms of Software. Genetic Algorithms (GAs) are already filling this gap in fields like Big Data mining, Protein Folding predictions, Finance, etc. In this paper we present the possibility of using an "Unbounded Single Taped Turing" medium like a Blockchain to store a Genetic Algorithm that will be able to provide Turing Complete results on any mathematically given problem.

Keywords: Genetic Algorithm, Cellular Automaton, Church-Turing thesis, Universal Computing Device, Blockchain, Bitcoin, Consensus, Moran Model, Decider, Two Stack Push Down Automaton, Ackerman Function Simulation, External Oracles, Small World Network, Hilbert Curves, AGI, Gödel Expansions.

I. Introduction

A Genetic Algorithm is broadly considered as a “black box” of software; it literally emulates the way Natural Evolution has used billions of years to evolve. Starting from a Prokaryote, given the time and the proper evolutionary conditions, Natural Evolution managed to produce Eukaryotes through DNA replication, Cell Division, Random Mutations and trillion replications and recombination [1], [2]. The goal of such an initiative would be no other than reach what we now describe as “Artificial Intelligence”.

There is a distinct difference between Genetic Algorithms and Classical Algorithms in two key points. Classical Algorithms generate only one instance that has a specific goal of solving a problem by approaching the optimal solution, while using deterministic computation. On the other hand, Genetic Algorithms create a population of instances with each iteration. It is the swarm intelligence of those instances that approach the solution of the problem on the best possible way. Genetic Algorithms are not using deterministic computation, but a computation based on Random Number Generators [3]. One of the most common implementation of Genetic Algorithms is the Cellular Automaton.

The concept of a Cellular Automaton (plural: Cellular Automata) was first introduced by John von Neumann in the Hixon Symposium in 1951 [4]. It was described as a discrete model that consists of a simple two-state, one dimensional grid of cells that can be either on or off. Later, in the 1970s a two-state, two-dimensional cellular automaton named “Game of Life” by John Conway, became widely known [5] but it wasn’t until the 1980s with the work of Stephen Wolfram when a systematic study of two-state, one-dimension of Cellular Automata was done [6], presenting the implementation of a Cellular Automaton based on specific set of rules. Wolfram named those “Elementary Cellular Automata” and his research assistant Mathew Cook showed that one of these rules is Turing-Complete. Their work has been published in 2002 in the bestselling book “A New Kind of Science” [7].

In computability theory [8], a Cellular Automaton can be Turing Complete, if it can be used to simulate any single-taped Turing Machine. The term was named after the Computer Scientist and Mathematician Alan Turing. A typical example of such an implementation is the Lambda Calculus which was introduced in 1930 by Alonzo Church [9].

As an extension to the above notion, if such an Automaton is formed, then a swarm of Cellular Automata of similar origin could possibly form what is described as a Church-Turing thesis [10]. Furthermore, above a certain point of computational evolution, they could form what is described by the Church–Turing–Deutsch principle. The principle states that a **Universal Computing Device** can simulate every physical process [11], [12].

In this paper we present that it is theoretically possible for a Turing-Complete algorithm, like a Cellular Automaton based on rule 110, to be implemented on an Unbounded Single Taped Turing Medium such as a Blockchain.

II. CONTEXT

Blockchain, was first introduced in Bitcoin [13], as a universal, fully shared Ledger that would be globally visible to all parties when a transaction was recorded on it without any presence of a trusted central authority. In each transaction, the previous owner signs - using the secret signing key corresponding to his public key a hash of the transaction in which he received the bitcoins (in practice, a SHA-256 hash) and the public key of the next owner [14]. Blockchain is a set of sequential blocks, where each block embeds verified transactions. Each transaction (Tx) which is processed and verified by the network is then added to the blockchain. Transactions at each block are hashed, paired and hashed again until a single hash is obtained, which is the Merkle root [51]. Merkle root is stored in block header. Each block also includes hash of previous block header, which results in a chain of blocks. The basic structure of blockchain is given in Fig. 1. [52]

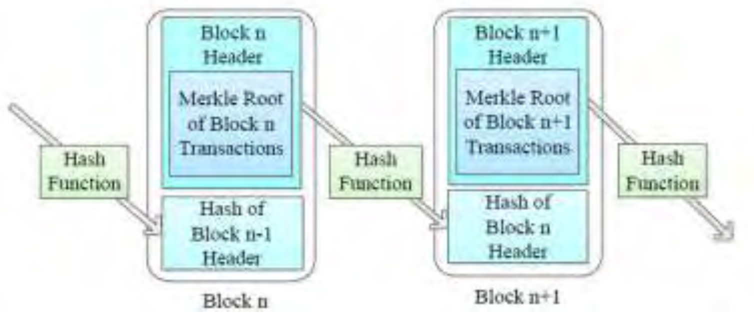


Fig.1 [52]

The concept of **PoW (Proof of Work)** introduced a reward mechanism to the solvers of a random SHA256 puzzle. In the block header, a nonce value is included, which is used as the proof-of-work for creating the block. It is generally considered that Proof-of-work is a system to prevent Denial-of-Service (DoS) attacks and other system misuses [52]. A hash puzzle is a set of mathematical problems which is solved by creating a hash that conforms to a specific requirement. Solving the puzzle is computationally difficult. Bitcoin uses the SHA-256 hash function [53]. Unless the (cryptographic) hash function used for calculating the block hashes is broken, the only fruitful strategy is to try different nonces until a solution is found [54]. In theory, those puzzles can be mathematically represented in the form of a random generated matrix where the difficulty plays a significant part in its creation. In Bitcoin it is a difficulty requirement to have the hash be lower than a specific threshold. It must be noted here that as with many cryptographic puzzles such as Elliptic Curve Cryptography [15], calculating the SHA256 can be rendered as a matrix calculation or a simple linear algebra problem. A classical paradigm is the Hill Cypher [16].

The miners can be seen as solving a Linear system of the form AX <= B where A, X and B are N x N dense matrices. The Eigenvalue of the matrix could verify the solution of the puzzle. It is noteworthy to point out that the determination of the eigenvalues and eigenvectors of a linear system of equations presents an extremely important subject in physics and engineering. We can perceive it as the analogous of matrix diagonalization and can be used in fields as stability analysis, the physics of rotating bodies, and small oscillations of vibrating systems, to name only a few [41]. The computational economics of such a task have been calculated in detail in "Floating Point Operations in Matrix-Vector Calculus" Technical Report (Hunger, 2007) and a detailed listing of the calculation cost in FLOPs (Floating Point Operations) is presented in Appendix 1 [42]. This representation fits the later use of GPU, FPGA and ASIC computing. Finally, if the answer is found, the fastest participant is rewarded. The current reward stands at 12.5 Bitcoins at the time of writing this paper [17].

A certain state called "**Consensus**" is reached by following the chain with the most proof of work (ie: most difficulty behind it) which is what everyone trusts and agree to be the most valid one. This way, every participant agrees over a certain block of transactions, so that there are no conflicts of any given transaction on each block or a previous one [18]. The above notion can be described as "The two Generals" problem and it is considered fundamental in Computer Science.

Abstractly speaking, we could perceive a Blockchain as a representation of a recording cylinder, similar to Edison Standard Phonograph (Fig. 1) where the transactions are being recorded in blocks, forming a spiral of ellipses, from left to right, whereas the recording is permanent and the cylinder virtually unbounded. This concept describes perfectly the theoretical representation of an Unbounded Single Taped Turing Medium.

Fig. 1: Edison Standard Phonograph

In such a medium, it is technically possible to store the evolutionary swarm of a Genetic Algorithm, that is mathematically proven to already be Turing-Complete. Such implementations are documented in the instances of Elementary Cellular Automaton based on Rule 110 [19], or Rule 30 [6].

A Basic Genetic Algorithm, can be represented in the following graph (Fig. 2)

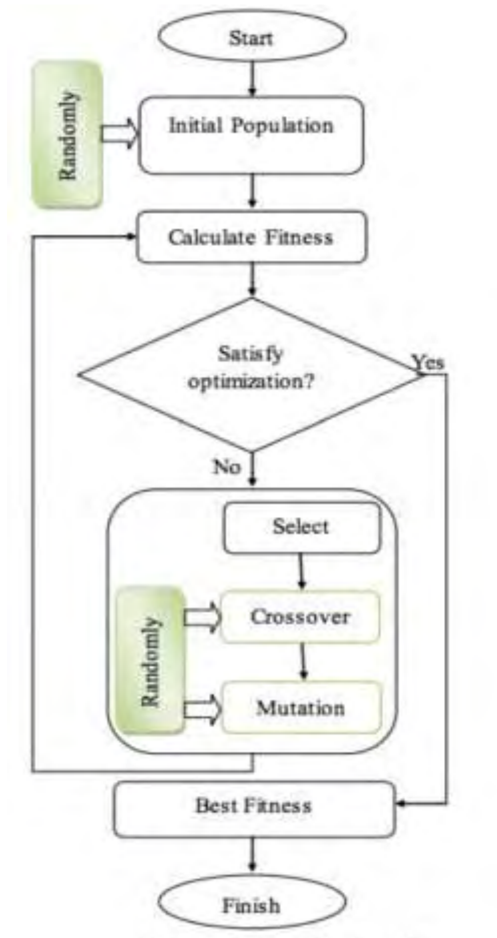


Fig. 2: Basic Genetic Algorithm Flowchart

The initial population and the iteration process include two Randomizations that cannot exceed the range from 0.02% to 2% of the total mutation of the Genetic Algorithm. The randomizations are provided by a computational Random Number Generator in both cases. It is clearly observable that the key point to the fitness procedure relies on the Random element. In the recent years, several scientists tried to implement alterations to the evolutionary process that could derive better results [20].

A Chaotic Genetic Algorithm. Based on an idea of I.G. Tsoulos [21], Reza Ebrahimzadeh and Mahdi Jampour [22] introduced a Chaotic random number mutational variable where the classical Genetic Algorithm used computational Randomization and they observed significant time optimizations of the fitting process.

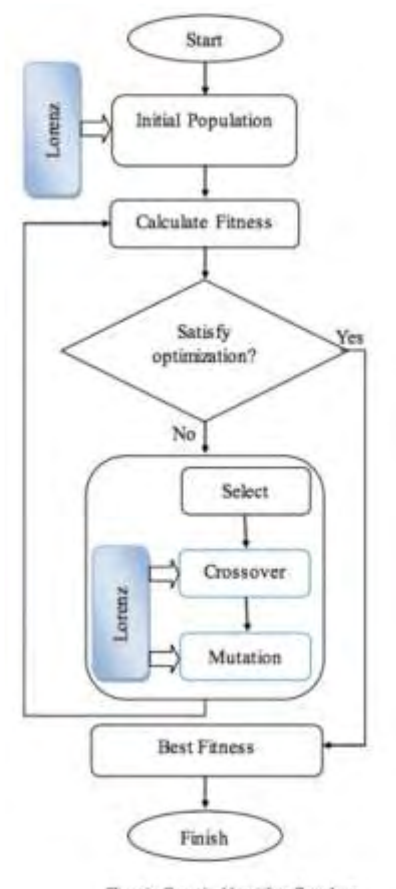


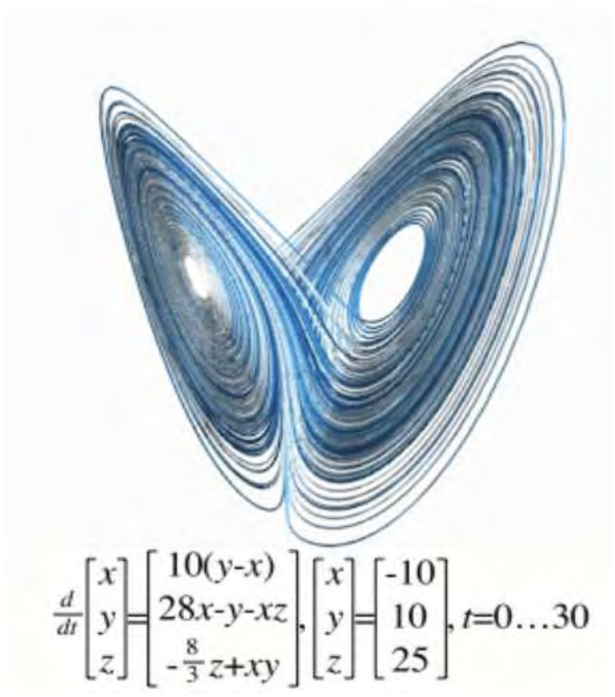
Fig. 3: Chaotic Genetic Algorithm Flowchart

Deterministic Chaos systems vs Natural Random systems. The definition of a deterministic chaos (or simply chaos) system was given by E. Lorenz [23]:

Chaos: When the present determines the future, but the approximate present does not approximately determine the future.

Chaotic systems were mathematically described through a truncated Navier-Stokes partial derivation system, that qualified the representation of chaotic phenomena such as the topology of the water molecules within a storm, or the movement of a double compound pendulum, by altering the triplet of numeric output of the three equations. It is a graphical representation is now known as a "Chaotic Attractor" (Fig. 4)

The most important observation within the Natural Evolutionary procedure throughout the years, seems to be the randomization factor. Random mutations led to new genetic characteristics and Natural Selection decided if the characteristic will remain dominant to the next generations. Whenever the randomization exceeded a certain level, the subject would die out and could not survive. The mutations had to be minimal and the iteration process naturally selectable. On a philosophical perspective side note, one could say that whoever designed the Universe, has carefully designed a Randomization Engine as a prerequisite to its existence.



$$\frac{d}{dt}\begin{bmatrix}x\\y\\z\end{bmatrix}=\begin{bmatrix}10(y-x)\\28x-y-xz\\-\frac{8}{3}z+xy\end{bmatrix},\begin{bmatrix}x\\y\\z\end{bmatrix}=\begin{bmatrix}-10\\10\\25\end{bmatrix},t=0\ldots30$$

Fig. 4: Chaotic Attractor
(source: Wikipedia)

The implementation of a Natural Random Number Generation System, seems like an elusive target: *Since the introduction of "middle square" method by John von Neumann for the production of "pseudo-random" numbers in about 1949, hundreds of other methods have been introduced. While each may have some virtue a single uniformly superior method has not emerged. The problems of cyclical repetition and the need to pass statistical tests for randomness still leave the issue unresolved* [24].

The idea of implementing a Natural Random Number Generation System to the evolutionary process of a Genetic Algorithm is not new. The first element had to be a very carefully chosen Random Number Generation engine. A Chaotic Attractor could be used, but the procedure had to produce repetitions after a specific sequence of Chaotic numbers had been collected.

Rule 110 (and possibly Rule 30) Cellular Automaton, appears to provide the best option, since it qualifies the non-repetition pattern, produces non predictable numbers, while its iteration procedure follows the minimal mutational factor synthesizing at the same time a proven Turing-Complete algorithm (Fig. 5).

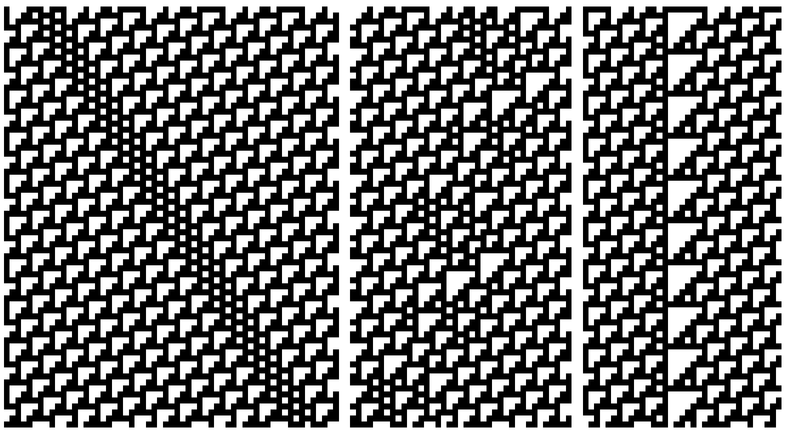


Fig. 5: Rule 110 Cellular Automaton [25]

Such an implementation, after a given time to evolve could theoretically form a Church–Turing–Deutsch principle machine. Blockchains such as Bitcoin present themselves as a suitable medium for such a task, since they form an Unbounded Single Taped Turing Medium. The materialization doesn't necessarily demand heavy processing power, although such power could help the evolutionary procedure as a side effect of the Proof of Work process, which is the source of good quality random numbers that are then available to feed a Moran Evolutionary Model.

Evolutionary Process. The model of the evolutionary process of a set of Genetic Algorithms is following the Moran Model [26]. The model is named after Patrick Moran and describes a simple stochastic process to address the growth of finite populations. It was originally created to mathematically represent the evolution process of a swarm of cells (or individuals) that is maintained to have a constant population of a size of N.

In order for a population to evolve under the Moran model certain rules must be obeyed:
Rule 1: The population must maintain a constant size of N
Rule 2: Overlapping Generations are allowed.
Rule 3: At discrete time intervals, two members A and B of the population are randomly selected (A and B can be the same or different) and while A is allowed to be reproduced, B dies (or vice-versa).
The dynamics of the organization of the sub-populations seems to be the main factor of the evolutionary process in terms of their probability of fitness of the Genetic Algorithms involved and when that happens. The bi-allelic Moran model (Fig. 6) can be defined as a Markov chain [27].

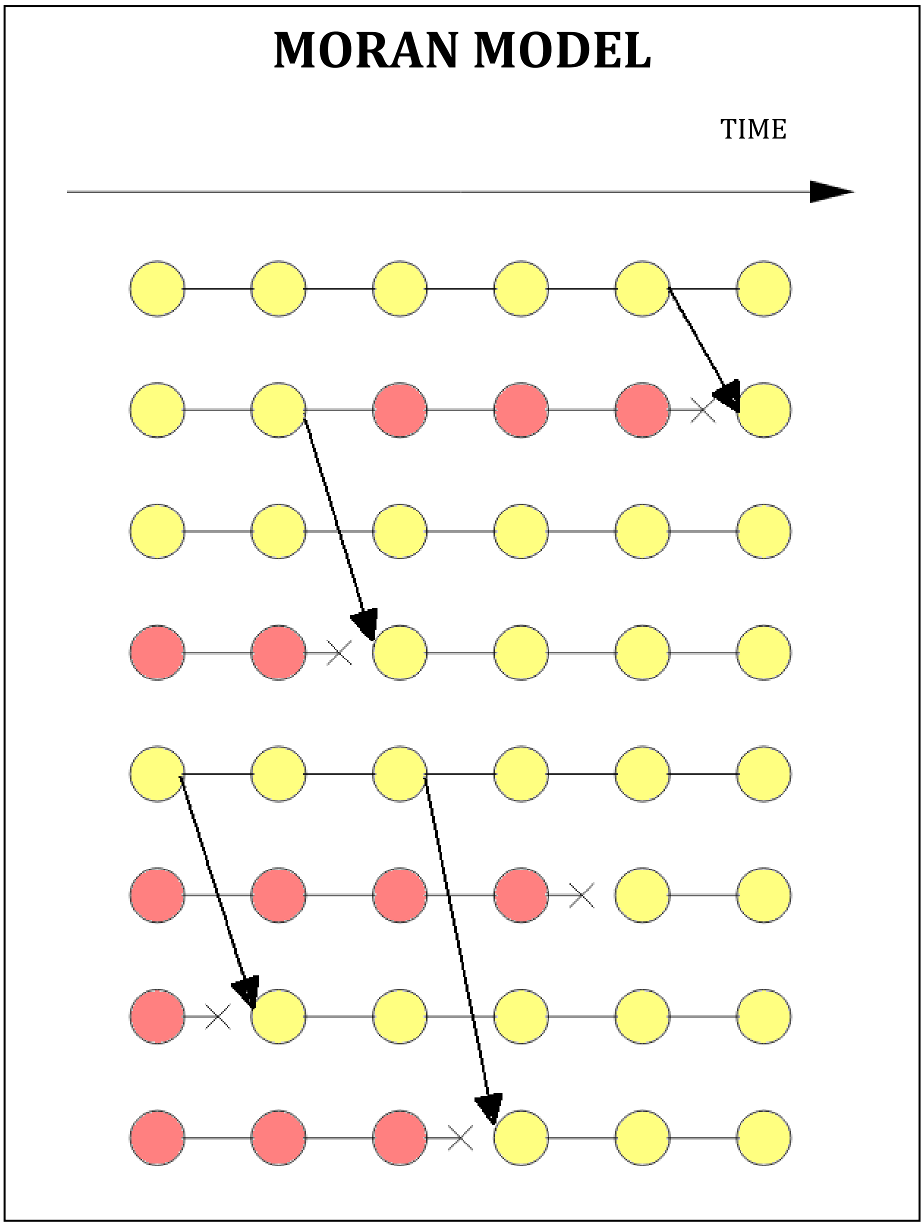


Fig. 6: Moran Model
(Generations through time) [27]

As an example hypothesis, we suppose that of all resident individuals are identical and a new mutant is introduced. We assume that the new mutant has relative fitness *r*, as compared to the residents, whose fitness is 1. The fixation probability of the mutant is then given by:

$$R_1 = (1\text{-}1/r)/(1\text{-}1/r^N).$$

The Moran process consists by two absorbing states: First, the population consists of all residents and second, the population consists by all mutants. No other stable equilibrium state is possible.

Whenever an absorbing state is reached, mutants (residents) are said to have reached fixation (Fig.7).

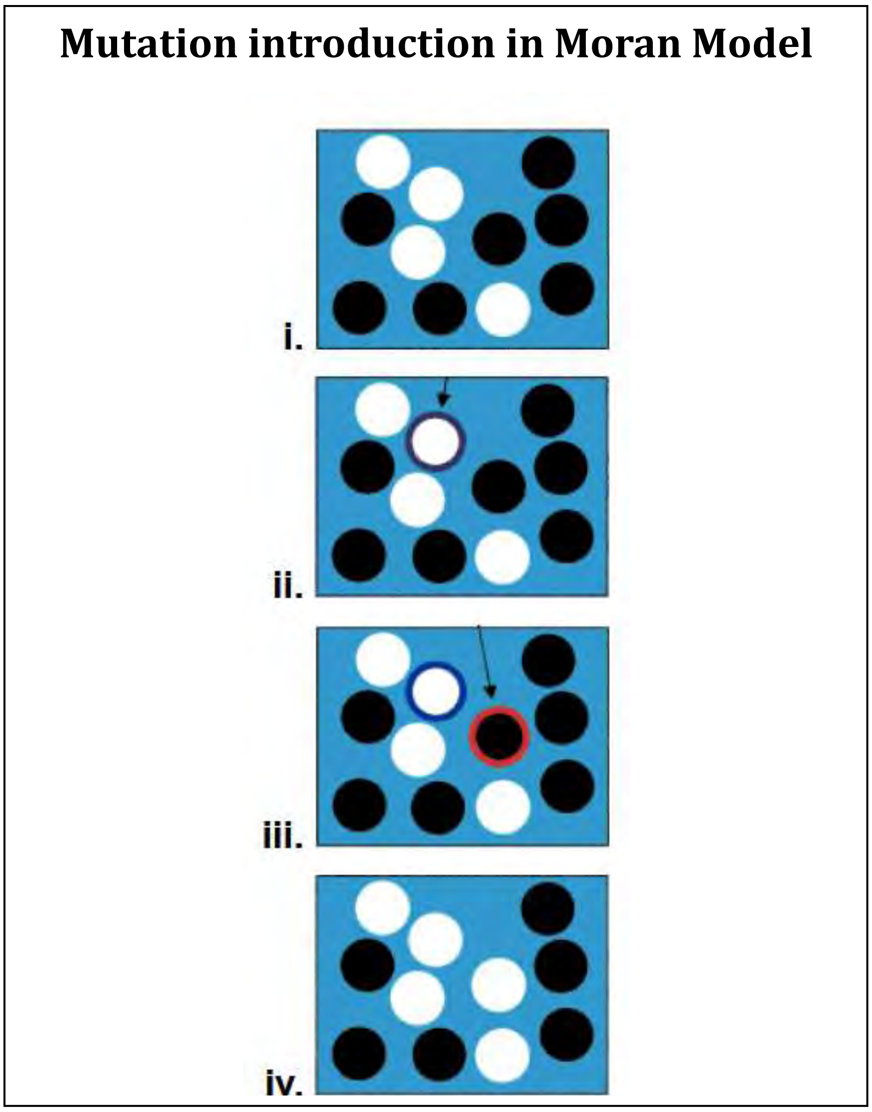


Fig. 7: Moran Model
(Mutation introduction) [28]

The above notion reveals a specific balance between selection and drift: advantageous mutations have a certain chance - but no guarantee - of fixation, whereas disadvantageous mutants are likely - but again, no guarantee - to become extinct [28].
Finally, although on the surface a frivolous use case, CryptoKitties on the Ethereum Blockchain [29] is at heart a sophisticated genetic algorithm that simulates evolution through two random 'kitties' propagating a third as a child. The arousal of such games provides support for the notion that blockchains are a suitable medium for GAs.

Usability. The utilization of such an algorithmic entity is literally limitless. Big Data Mining, Monte Carlo based Predictions, HMM predictions, Expansion of Human Knowledge, Protein Folding Prediction, Epigenetic Analysis, Future Mutations in Human Genome, Intrusion Detection of websites, are a subset of the fields that could potentially benefit. The interaction with the entity could be done initially via scripting language, later with pseudo-code and finally even in natural language when the evolution matures to higher state.

It is trivial to state that commands need to be interpreted via respected Blockchain tokens (ie: Bitcoins, Ethereum, EOS etc) that would be used to get the questions into a Blockchain for the entity to process. The output will be given in the form of unspent transaction outputs (UTXO) on a Blockchain (Fig. 8). Practically, whoever owns the tokens will be able to utilize the processing power of the machine.

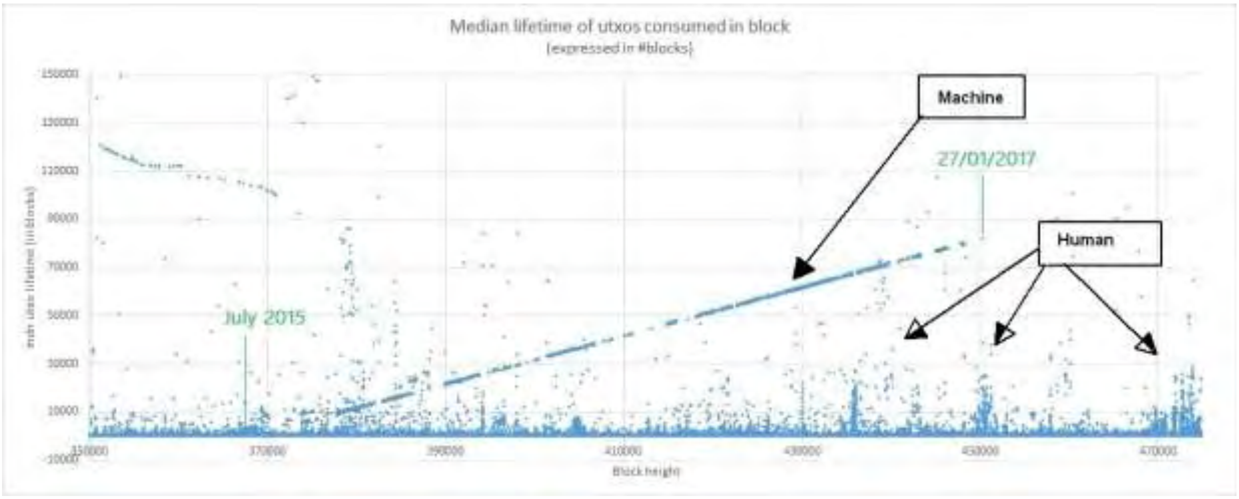


Fig. 8: UTXO graphical representation
(source: OXT.me)

Case Study. Apart from the theoretical approach, there are evidential data, that such an entity is implemented on Bitcoin's Blockchain. Specifically, Bitcoin itself forms a "**Decider**" or a "**Two Stack Push Down Automaton**" (2pda) or, more formally, a special case of a Probabilistic Total Turing Machine, that is controllable via scripting language. The first theoretical representation of such a computational entity was first introduced by Hao Wang in 1954, in his **Wang-B Machine** [48].
The most prominent evidential data is the graph presented in Fig. 8 which represents the Median lifetime of UTXOs consumed in blocks. Bitcoin UTXO Lifespan Prediction as shown by Robert Konrad & Stephen Pinto in 2015 is impossible to be modeled mathematically, since it is purely chaotic. Nevertheless, in the above graph there's a distinct linear formation within the phenomenal chaos [30].

At the same time period the Mempool transaction volume size had increased to up to 150Mb which adds an extra bit of confidence that the incidental linear formation could be the result of computable work (Fig.9)

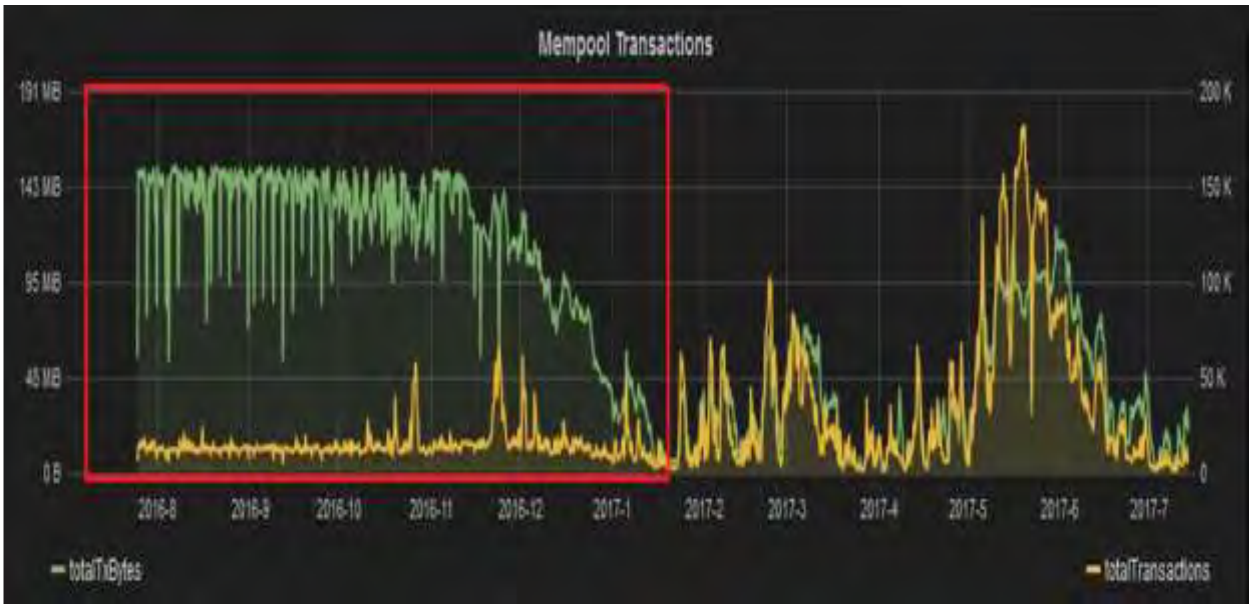


Fig. 9: Mempool transaction volume in Mb
(source: OXT.me)

The Bitcoin scripting language is able to deploy a “two-argument Ackerman Function” which is now considered the simplest example of a well-defined total function which can be computable but not primitive recursive, providing a counterexample to the belief in the early 1900s that every computable function was also primitive recursive [31].

This is achieved via simulation of a “for loop” by the programming technique of “unrolling the loop” [32] and this system demonstrates that Bitcoin can incorporate total computable functions that are simply "recursive" as well as primitive recursive. One of the most interesting parts of this implementation is that a Bitcoin script can be constructed to simulate any decidable function. The added benefit to this comes from computational feasibility. It is sufficient for the alt stack to be a memory register in order to implement looping.
Loops can be unwound this way and operate linearly [33].

Bitcoin script forms a non-synchronous language in the following way:
- The notion of physical time is replaced with the notion of order.
- Only the simultaneity and precedence of events are considered. This means that the physical time does not play any special role. This is defined as multiform notion of time.
- A false is not wrong, it is a script object that has been denied access to the sequence because for example it was beaten by another script to a critical resource. In this way it is possible for a large amount of scripts running with various inputs.
- The first one to end as True is the one that is written to the blockchain.

- The other threads are removed.

This tactic is similarly implemented in CuDa and the threaded CuDa programs [34].

So to summarize, one can perceive the **2pda** as a distributed computer. This computer is the sum of all the full nodes. Every full node incorporates a full copy of blockchain (about 150GB at the time of writing) and its respective memory pool. This entity, as a distributed computer processes transactions (Tx), which have the OP_CODEs embedded. Two commands worth mentioning are OP_TOALTSTACK and OP_FROMALTSTACK. An obvious usage for the aforementioned OP_CODEs is similar to the memory function, in the push-pull sense; as the first one moves data from the top of the main stack to the top of the alt stack and the second moves data from the top of alt stack to the top of main stack. This effectively allows data to be stored and then calculated through the alt stack as simple computational tasks through scripting language.

The **OP_TOALTSTACK**: The opcode that follows this command defines a new function to be called. If this function name is already taken, the transaction is marked as invalid. Within the transaction, the function can be called simply as FunctionName until the process completes and returns the respective result.

The **OP_FROMALTSTACK** : This ends a function and returns. So, from these two OP_CODE commands, you can derive simple recursive functions [33]. There are more OP_CODE commands that are enumerated via the enum opcodetype command [35] but only nineteen (19) of them are correlated to the 2pda (List 1).

A detailed presentation of Turing Complete Computation in Bitcoin was given by Clemens Ley in Satoshi's Vision Conference [50].

```
// stack ops
OP_TOALTSTACK = 0x6b,
OP_FROMALTSTACK = 0x6c,
OP_2DROP = 0x6d,
OP_2DUP = 0x6e,
OP_3DUP = 0x6f,
OP_2OVER = 0x70,
OP_2ROT = 0x71,
OP_2SWAP = 0x72,
OP_IFDUP = 0x73,
OP_DEPTH = 0x74,
OP_DROP = 0x75,
OP_DUP = 0x76,
OP_NIP = 0x77,
OP_OVER = 0x78,
OP_PICK = 0x79,
OP_ROLL = 0x7a,
OP_ROT = 0x7b,
OP_SWAP = 0x7c,
OP_TUCK = 0x7d,
```

List 1: OPCODE list correlated to 2PDA [35]

Turing Completeness within Smart Contracts. Even though the notion of Turing Completeness is usually associated with the method of unrolling the loop [32], Alexander Chepurnoy, Vasily Kharin and Dmitry Meshkov showed that Turing-completeness of a blockchain system can be achieved through unwinding a set of recursive calls between multiple transactions and several blocks instead of using a single block to do it [55].

The proof included a CA110 implementation algorithm, a control script to ensure that the CA110 transformation keeps the same rules during future iterations and a validation script **(Algorithm 3)** for the output representing the single bit, and the unbound grid **(Fig. 10)**

Algorithm 3 Validation script for the output representing the single bit, and the unbound grid

```
function VERIFY(in, out)                ▷ "in" and "out" are lists of inputs and outputs
    function OUTCORRECT(out, script)                           ▷ output structure check
        scriptCorrect ← out[0].script = script
        isCopy1 ← out[1] = out[0].copy(mid← true)
        isCopy2 ← out[2] = out[0].copy(mid← false)
        return (¬out[0].mid) ∧ scriptCorrect ∧ isCopy1 ∧ isCopy2
    end function
    function CORRECTPAYLOAD(in, out)                             ▷ output payload check
        ▷ mid flag is only set for the middle input
        inMidCorrect ← in[1].mid ∧ ¬(in[0].mid ∨ in[2].mid)
        ▷ input positions are correct; n is the index of leftmost column
        inYCorrect ← (in[1].n = in[0].n) ∧ (in[2].n = in[0].n)
        inXCorrect ← (in[1].x = in[0].x+1) ∧ (in[2].x = in[1].x+1)
        ▷ bits satisfy Rule 110
        inValCorrect ← out[0].val=CALCBIT(in[0].val, in[1].val, in[2].val)
        ▷ output position matches the input one
        outPosCorrect ← out[0].x = in[1].x ∧ (out[0].n = in[0].n−1)
        return inValCorrect ∧ inXCorrect ∧ inYCorrect ∧
                inMidCorrect ∧ outPosCorrect ∧ in.size=out.size=3
    end function
    if in[0].x=in[0].n ∧ in.size=1 then          ▷ leftmost — add 2 zeros to the left
        middle ← in[0].copy(x←in[0].n−1, val←0, mid← true)
        left ← in[0].copy(x←in[0].n−2, val←0, mid← false)
        realIn ← left ++ middle ++ in
    else if in[0].x=in[0].n ∧ in.size=2 then ▷ next to leftmost — add 0 to the left
        left ← in[0].copy(x←in[0].n−1, val←0, mid← false)
        realIn ← left ++ in
    else if in[0].x=−1 ∧ in.size=2 then          ▷ rightmost — add 0 to the right
        right ← in[0].copy(x← 1, val←0, mid← false)
        realIn ← in ++ right
    else                                                               ▷ normal cell
        realIn ← in
    end if
    return CORRECTPAYLOAD(realIn, out) ∧ OUTCORRECT(out, in[0].script)
end function
```

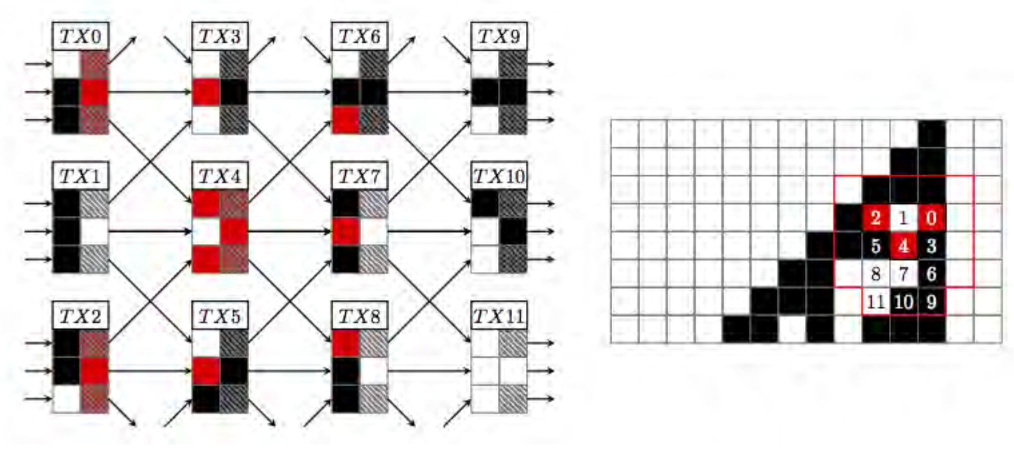


Fig. 10: Evolution of the CA110 described in Alg. 3 [55]

Their approach can be utilized as a method of proving Turing Completeness of various smart contract languages while the code is freely available online [56],[57]

Ackerman Function as a security procedure. Bitcoin as a network presents redundant durability since its creation in 2009. This is due to two main factors. The first and most important is the design of the network nodes and clients forming a "Small World" network **(fig.11)** [33].

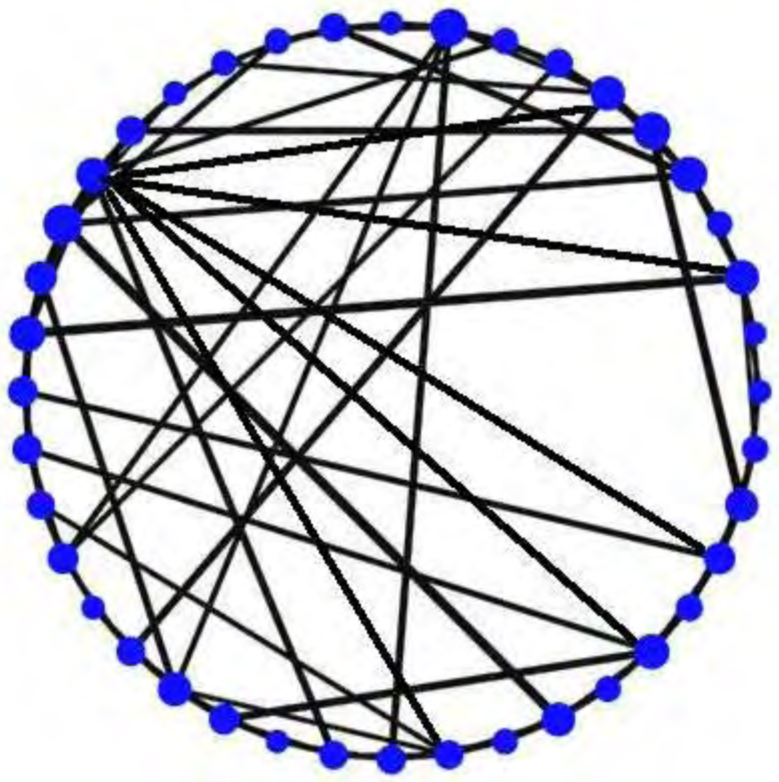


Fig. 11: Small World Network Graph

(Source: Stanford University)

The second factor is that there is a constant network security computational procedure based on the "Self-healing networks" method after a Node leaves the network and a link is broken [45] (fig.12)

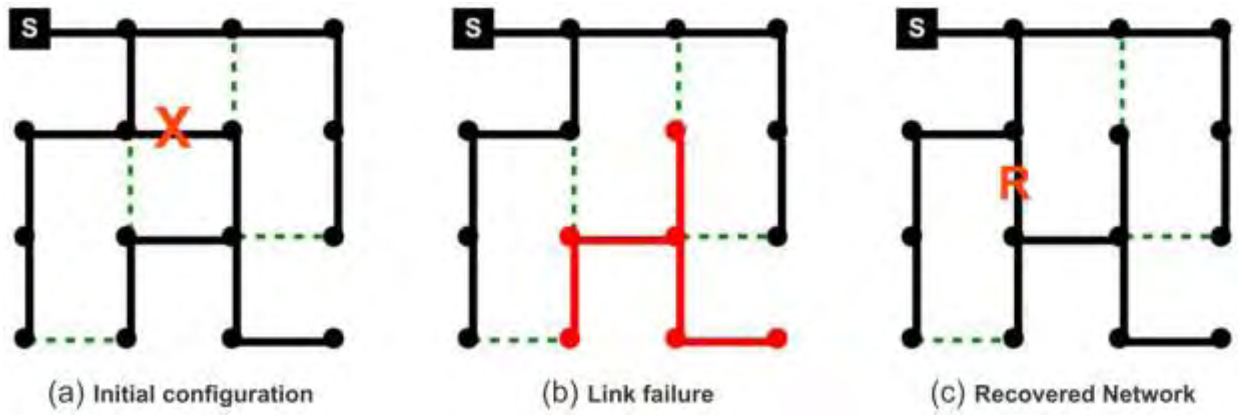


Fig. 12: Example of healing after single link failure. [45]

The resemblance with the Hilbert Space-Filling Curve [46] in the above procedure, also as, the possible computational representation of a Lindenmayer system [47] that such a system requires to be constructed and solved, can also be theoretically restated and computated by the diagonalization of an Ackerman Function representation as an infinite matrix (Appendix 2).

Opportunities and Future Uses. Bitcoin's creation in 2009 was a revolutionary idea in the financial world. It is considered as the digital cash of the new age. Secure, non-centralized, can provide the world with "honest", non-inflatable money. Game theory is utilized into maintaining consensus, without the need of any central authority, while Gresham's Law in effect would eliminate the "bad money" over the "good"; or if you prefer, the "strong" money.

Implementing a Church–Turing–Deutsch principle machine on a Blockchain, could in turn, open a whole new World of applications for a better humanity from Computer Assisted Governance to Extinction Level Events predictions. With emergent technologies like Human-Machine interface, such an entity could provide extensive knowledge in many fields of Science, which was previously impossible to acquire. Using Deep Machine Learning techniques, the evolution level of the algorithm could reach unprecedented levels exponentially, by utilizing the big data acquired by Smart Contracts, everyday transactions, weather conditions, IOT, or stored literature on a Blockchain.

III. Conclusions

In this paper we showed that it is theoretically possible for a Turing-Complete algorithm, like a Cellular Automaton based on rule 110, to be implemented on an Unbounded Single Taped Turing Medium such as a Blockchain.

The implementation could be achieved by authoring a Genetic Algorithm that would evolve, by utilizing an, as close as possible, Naturally Random Generated Mutational System. The iteration process would be based on a Blockchain transaction system, and each entity could store itself when the maximum fitness level was due. We showed that the specific Algorithm should be Turing-Complete in order; as a Swarm Intelligence, to evolve to a Church–Turing–Deutsch principle machine.

The interaction with such an entity, could be achieved via interpreted commands using the transaction system. For this, Blockchain Tokens (ie: coins) will be used as a means of transaction. At the first stages of evolution the system would provide low-level programming support, but could be educated through Machine Learning to accept natural language interaction.

The advantage of implementing such an entity on a blockchain is primarily that it provides a theoretical representation of an Unbounded Single Taped Turing Medium. Secondly, that such a medium can be designed to provide fast iteration mechanism through recorded Tx. Some existing blockchains have the ability of materialize up to thousands Tx per second [18].

The disadvantage is that the GA fitness procedure should be done externally (**External Oracles**) and Application Specific Integrated Circuits (ASICs) are mandatory for the task [36], [37].

IV. Discussion

The implications of such a hypothesis are enormous. Ray Kurzweil, has predicted that by the end of 2029 the world will possibly have one AI that matches human intelligence [38]. What we showed in this paper verifies this claim, and endorses the possibility this could happen much earlier. It must be stated that preliminary forms of such entities are already in existence [39], so it is not a matter of if, rather when this happens. The evolutionary process, from a certain point forth, follows an exponential curve. Hence, when the critical point of reaching human intelligence is met, then it is a matter of months or even days before it expands to much higher levels.

The materialization of such an entity on a Blockchain provides many pros and cons that we tried to describe in this paper. The encryption procedure together with the mandatory token usage, certifies that such an entity won't be able to interact without a cost. This is both good and bad.

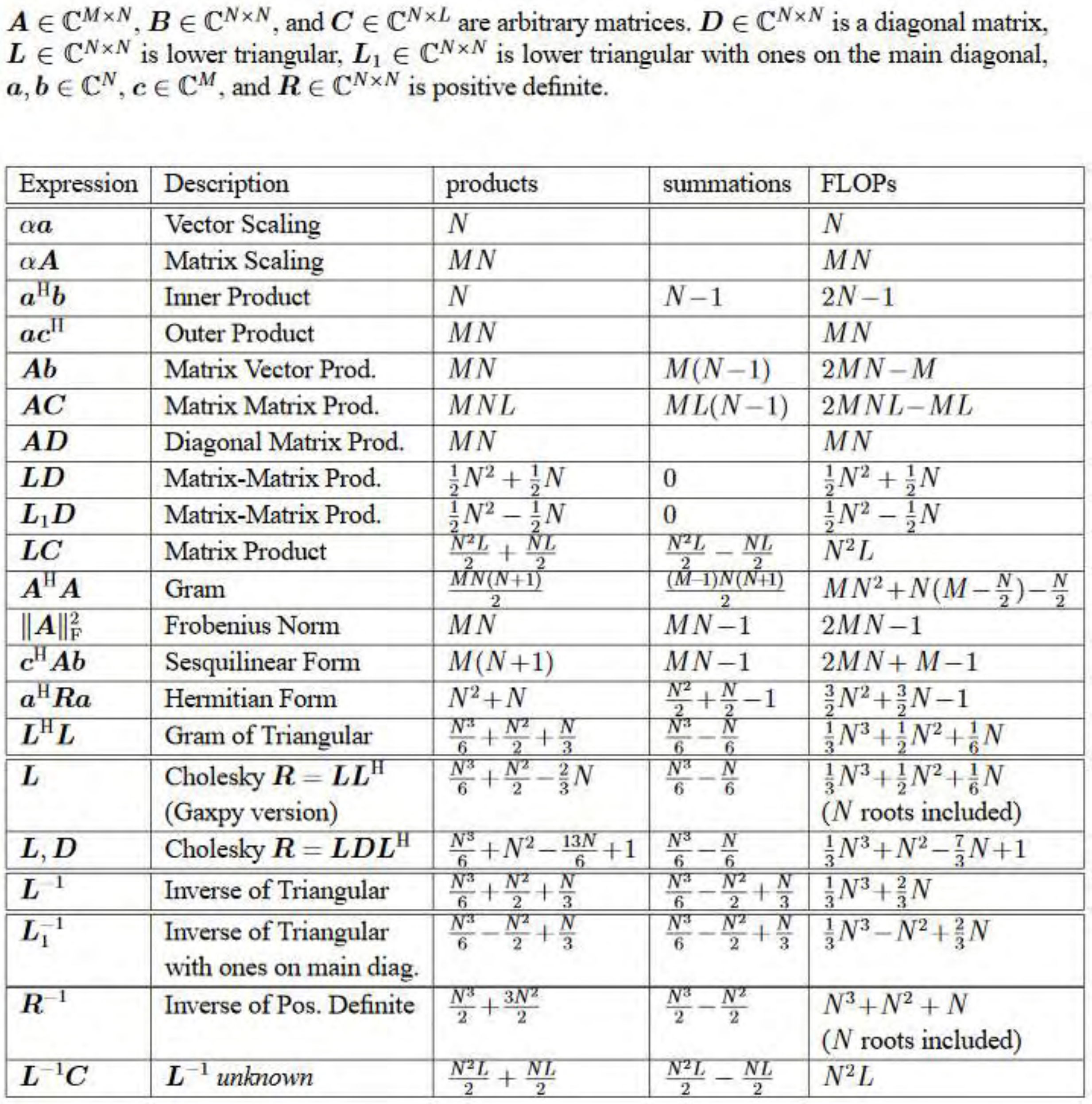
Several times in the past, the scientific world has witnessed an inherent limitation of every formal axiomatic system. The system could contain problems that were impossible to be solved from the theory itself. The incompleteness theorem of Kurt Gödel [49] describes this incapability and predicts that a new theory needs to be invented in order for the old theory to be "expanded" so that it'll be able to produce solutions to insolvable problems. **Gödel Expansions** are the path to innovation, the "secret" ingredient for new science. It happened with Riemannian Geometry and Relativity Theory, with the Parabolic and Euclidian Geometry, with the Information Technology and Physics, Biology, Mathematics and now with an Artificial Intelligence of a Generic form (**AGI**).

Finally, a point of discussion could be about “what happens next”? At this point, a reference to the great text of Isaac Asimov, “The Last Question” [40] is needed:

Can this chaos not be reversed into the Universe once more? Can that not be done?

Appendix 1

An overview of “Floating Point Operations in Matrix-Vector Calculus Report” which presents a detailed listing of the calculation cost in FLOPs (Floating Point Operations) [41]



$\boldsymbol{A} \in \mathbb{C}^{M\times N}$, $\boldsymbol{B} \in \mathbb{C}^{N\times N}$, and $\boldsymbol{C} \in \mathbb{C}^{N\times L}$ are arbitrary matrices. $\boldsymbol{D} \in \mathbb{C}^{N\times N}$ is a diagonal matrix, $\boldsymbol{L} \in \mathbb{C}^{N\times N}$ is lower triangular, $\boldsymbol{L}_1 \in \mathbb{C}^{N\times N}$ is lower triangular with ones on the main diagonal, $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{C}^{N}$, $\boldsymbol{c} \in \mathbb{C}^{M}$, and $\boldsymbol{R} \in \mathbb{C}^{N\times N}$ is positive definite.

Expression	Description	products	summations	FLOPs
$\alpha\boldsymbol{a}$	Vector Scaling	N		N
$\alpha\boldsymbol{A}$	Matrix Scaling	MN		MN
$\boldsymbol{a}^{\mathrm{H}}\boldsymbol{b}$	Inner Product	N	$N-1$	$2N-1$
$\boldsymbol{a}\boldsymbol{c}^{\mathrm{H}}$	Outer Product	MN		MN
$\boldsymbol{A}\boldsymbol{b}$	Matrix Vector Prod.	MN	$M(N-1)$	$2MN-M$
$\boldsymbol{A}\boldsymbol{C}$	Matrix Matrix Prod.	MNL	$ML(N-1)$	$2MNL-ML$
$\boldsymbol{A}\boldsymbol{D}$	Diagonal Matrix Prod.	MN		MN
$\boldsymbol{L}\boldsymbol{D}$	Matrix-Matrix Prod.	$\frac{1}{2}N^2+\frac{1}{2}N$	0	$\frac{1}{2}N^2+\frac{1}{2}N$
$\boldsymbol{L}_1\boldsymbol{D}$	Matrix-Matrix Prod.	$\frac{1}{2}N^2-\frac{1}{2}N$	0	$\frac{1}{2}N^2-\frac{1}{2}N$
$\boldsymbol{L}\boldsymbol{C}$	Matrix Product	$\frac{N^2L}{2}+\frac{NL}{2}$	$\frac{N^2L}{2}-\frac{NL}{2}$	N^2L
$\boldsymbol{A}^{\mathrm{H}}\boldsymbol{A}$	Gram	$\frac{MN(N+1)}{2}$	$\frac{(M-1)N(N+1)}{2}$	$MN^2+N(M-\frac{N}{2})-\frac{N}{2}$
$\Vert\boldsymbol{A}\Vert_{\mathrm{F}}^2$	Frobenius Norm	MN	$MN-1$	$2MN-1$
$\boldsymbol{c}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{b}$	Sesquilinear Form	$M(N+1)$	$MN-1$	$2MN+M-1$
$\boldsymbol{a}^{\mathrm{H}}\boldsymbol{R}\boldsymbol{a}$	Hermitian Form	N^2+N	$\frac{N^2}{2}+\frac{N}{2}-1$	$\frac{3}{2}N^2+\frac{3}{2}N-1$
$\boldsymbol{L}^{\mathrm{H}}\boldsymbol{L}$	Gram of Triangular	$\frac{N^3}{6}+\frac{N^2}{2}+\frac{N}{3}$	$\frac{N^3}{6}-\frac{N}{6}$	$\frac{1}{3}N^3+\frac{1}{2}N^2+\frac{1}{6}N$
$\boldsymbol{L}$	Cholesky $\boldsymbol{R}=\boldsymbol{L}\boldsymbol{L}^{\mathrm{H}}$ (Gaxpy version)	$\frac{N^3}{6}+\frac{N^2}{2}-\frac{2}{3}N$	$\frac{N^3}{6}-\frac{N}{6}$	$\frac{1}{3}N^3+\frac{1}{2}N^2+\frac{1}{6}N$ (N roots included)
$\boldsymbol{L},\boldsymbol{D}$	Cholesky $\boldsymbol{R}=\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{H}}$	$\frac{N^3}{6}+N^2-\frac{13N}{6}+1$	$\frac{N^3}{6}-\frac{N}{6}$	$\frac{1}{3}N^3+N^2-\frac{7}{3}N+1$
$\boldsymbol{L}^{-1}$	Inverse of Triangular	$\frac{N^3}{6}+\frac{N^2}{2}+\frac{N}{3}$	$\frac{N^3}{6}-\frac{N^2}{2}+\frac{N}{3}$	$\frac{1}{3}N^3+\frac{2}{3}N$
$\boldsymbol{L}_1^{-1}$	Inverse of Triangular with ones on main diag.	$\frac{N^3}{6}-\frac{N^2}{2}+\frac{N}{3}$	$\frac{N^3}{6}-\frac{N^2}{2}+\frac{N}{3}$	$\frac{1}{3}N^3-N^2+\frac{2}{3}N$
$\boldsymbol{R}^{-1}$	Inverse of Pos. Definite	$\frac{N^3}{2}+\frac{3N^2}{2}$	$\frac{N^3}{2}-\frac{N^2}{2}$	N^3+N^2+N (N roots included)
$\boldsymbol{L}^{-1}\boldsymbol{C}$	$\boldsymbol{L}^{-1}$ *unknown*	$\frac{N^2L}{2}+\frac{NL}{2}$	$\frac{N^2L}{2}-\frac{NL}{2}$	N^2L

Appendix 2

Computing the Ackermann function can be restated in terms of an infinite table where the natural numbers are represented along the top row. To determine a number in the table, we must take the number immediately to the left, then look up the required number in the previous row, at the position given by the number just taken. If there is no number to its left, simply look at the column headed "1" in the previous row. Despite the large values occurring in this early section of the table, some even larger numbers have been defined, such as Graham's number, which cannot be written with any small number of Knuth arrows. This number is constructed with a technique similar to applying the Ackermann function to itself recursively. Here is a small upper-left portion of the table but with the values replaced by the relevant expression from the function definition to show the pattern clearly [43], [44]:

Values of $A(m, n)$

$m \backslash n$	**0**	**1**	**2**	**3**	**4**	**n**
0	0+1	1+1	2+1	3+1	4+1	
1	A(0, 1)	A(0, A(1, 0)) = A(0, 2)	A(0, A(1, 1)) = A(0, 3)	A(0, A(1, 2)) = A(0, 4)	A(0, A(1, 3)) = A(0, 5)	A(0, A(1, *n*-1))
2	A(1, 1)	A(1, A(2, 0)) = A(1, 3)	A(1, A(2, 1)) = A(1, 5)	A(1, A(2, 2)) = A(1, 7)	A(1, A(2, 3)) = A(1, 9)	A(1, A(2, *n*-1))
3	A(2, 1)	A(2, A(3, 0)) = A(2, 5)	A(2, A(3, 1)) = A(2, 13)	A(2, A(3, 2)) = A(2, 29)	A(2, A(3, 3)) = A(2, 61)	A(2, A(3, *n*-1))
4	A(3, 1)	A(3, A(4, 0)) = A(3, 13)	A(3, A(4, 1)) = A(3, 65533)	A(3, A(4, 2))	A(3, A(4, 3))	A(3, A(4, *n*-1))
5	A(4, 1)	A(4, A(5, 0))	A(4, A(5, 1))	A(4, A(5, 2))	A(4, A(5, 3))	A(4, A(5, *n*-1))
6	A(5, 1)	A(5, A(6, 0))	A(5, A(6, 1))	A(5, A(6, 2))	A(5, A(6, 3))	A(5, A(6, *n*-1))

Source: Wikipedia

Acknowledgements:

Athanasios P. Kakarountas, Craig S. Wright, Ian Grigg, George P. Papageorgiou, Joseph VaughnPerling, Uyen Thuc Nguyen, Vassilis P. Plagiannakos.
The author is indebted to the following people who indirectly inspired him to conduct the present study:
Pantelis Bagos, Harilaos Sandalidis, Konstantinos Delimpasis, Ioannis Anagnostopoulos, Evripides Markou, Spathoulas Georgios and Braliou Georgia. Without their deep academic knowledge, their support and endorsement, this paper would never have been possible.

References

[1] Alberts B, Johnson A, Lewis J, Raff M, Roberts K, Walter P (2002). Molecular Biology of the Cell. Garland Science. ISBN 0-8153-3218-1. Chapter 5: DNA Replication Mechanisms

[2] Article: “What is DNA Replication?". yourgenome.org. Welcome Genome Campus. Retrieved 24 February 2017.

[3] Article: “Genetic Algorithms”, Mathworks, 2017.

[4] John von Neumann, "The general and logical theory of automata," in L.A. Jeffress, ed., Cerebral Mechanisms in Behavior – The Hixon Symposium, John Wiley & Sons, New York, 1951, pp. 1–31.

[5] Gardner, Martin (1970). "Mathematical Games: The fantastic combinations of John Conway's new solitaire game "life"". Scientific American (223): 120–123.

[6] Wolfram, Stephen (1983). "Statistical Mechanics of Cellular Automata". Reviews of Modern Physics. 55 (3): 601–644.

[7] Wolfram, Stephen (2002). “A New Kind of Science”. ISBN 1-57955-008-8

[8] Michael Sipser (1997). Introduction to the Theory of Computation. PWS Publishing. ISBN 0-534-94728-X. Part Two: Computability Theory, Chapters 3–6, pp. 123–222.

[9] Church, A. (1932). "A set of postulates for the foundation of logic". Annals of Mathematics. Series 2. 33 (2): 346–366. JSTOR 1968337. doi:10.2307/1968337.

[10] Rabin, Michael O. (June 2012). Turing, Church, Gödel, Computability, Complexity and Randomization: A Personal View.

[11] Nielsen, Michael. "Interesting problems: The Church–Turing–Deutsch Principle". Retrieved 10 May 2014.

[12] Deutsch, D. (1985). "Quantum theory, the Church–Turing principle and the universal quantum computer" (PDF). Proceedings of the Royal Society. London. 400: 97–117. doi:10.1098/rspa.1985.0070.

[13] Satoshi Nakamoto, (2008) "Bitcoin: A Peer-to-Peer Electronic Cash System " (Whitepaper)

[14] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker Stefan Savage (2013): A fistful of bitcoins: characterizing payments among men with no names (University of California, San Diego - George Mason University†)

[15] D.M. Schinianakis, A.P. Fournaris, H.E. Michail, A.P. Kakarountas and T. Stouraitis, "An RNS Architecture of an Fp Elliptic Curve Point Multiplier", IEEE Transactions on Circuits and Systems I, vol.56, no.6, pp. 1202-1213, June 2009.

[16] Lester S. Hill, Cryptography in an Algebraic Alphabet, The American Mathematical Monthly Vol.36, June–July 1929, pp. 306–312.

[17] Papageorgiou, George P. (2013) Bitness: Bitcoin usage in an enterprise environment, opportunities and challenges (MBA Thesis, University of Greenwich)

[18] Ian Grigg (2017). EOS, An Introduction. (Whitepaper) iang.org/papers/EOS_An_Introduction.pdf

[19] Cook, Matthew (2004). "Universality in Elementary Cellular Automata". Complex Systems. 15 (1). ISSN 0891-2513. Retrieved 24 June 2015.

[20] M.G. Epitropakis, D.K. Tasoulis, N.G. Pavlidis, V.P. Plagianakos, and M.N. Vrahatis, (2011): Enhancing Differential Evolution Utilizing Proximity-based Mutation Operators IEEE Transactions on Evolutionary Computation, Vol. 15, 99-119.

[21] Tsoulos, Ioannis. (2009). Tsoulos, I.G.: Solving constrained optimization problems using a novel genetic algorithm. Appl. Math. Comput. 208, 273-283. Applied Mathematics and Computation. 208. 273-283. 10.1016/j.amc.2008.12.002.

[22] Ebrahimzadeh, Reza & Jampour, Mahdi. (2013). Chaotic Genetic Algorithm based on Lorenz Chaotic System for Optimization Problems. International Journal of Intelligent Systems and Applications. 5. 19-24. 10.5815/ijisa.2013.05.03.

[23] Danforth, Christopher M. (April 2013). "Chaos in an Atmosphere Hanging on a Wall". Mathematics of Planet Earth 2013. Retrieved 4 April 2013.

[24] Yadolah Dodge (Dec., 1996), A Natural Random Number Generator

International Statistical Review / Revue Internationale de Statistique Vol. 64, No. 3 pp. 329-344

[25] By JohnnyNyquist - http://commons.wikimedia.org/wiki/File:Ca110-structures.png, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=18642040

[26] Moran, P. (1958). Random processes in genetics. Mathematical Proceedings of the Cambridge Philosophical Society, 54(1), 60-71. doi:10.1017/S0305004100033193

[27] Xavier Didelot, Statistical population genetics: Lecture 3, The Moran Model, slide 40. University of Oxford. (http://www.stats.ox.ac.uk/~didelot/popgen/lecture3.pdf). Retrieved 27.12.2017

[28] Lieberman, E., Hauert, Ch. & Nowak, M. (2005) Evolutionary Dynamics on Graphs. Nature 433, 312-316

[29] Cryptokitties (2017), https://www.cryptokitties.co/ https://www.dropbox.com/s/a5h3zso545wuqkm/CryptoKitties_WhitePapurr_V2.pdf?dl=0

[30] Robert Konrad & Stephen Pinto, (Dec. 2015) “Bitcoin UTXO Lifespan Prediction”

[31] Dötzel, G. (1991) "A Function to End All Functions." Algorithm: Recreational Programming 2.4, 16-17

[32] K. Aisopos, A.P. Kakarountas, H. Michail, C.E. Goutis, “High throughput implementation of the new Secure Hash Algorithm through partial unrolling”, in Proc. of IEEE 2005 International Workshop on Signal Processing Systems (SiPS’05), Athens, Greece, pp. 99-103, Nov. 2-4, 2005.

[33] Wright, Craig, (2017): “Bitcoin, A Total Turing Machine” (Whitepaper).

[34] CUDA® is a parallel computing platform and programming model invented by NVIDIA.

[35] Opcode List in Bitcoin Core (Version 14.0) https://github.com/bitcoin/bitcoin/blob/v0.14.0/src/script/script.h#L46L187

[36] A. Milidonis, N. Alachiotis, V. Porpodas, H. Michail, G. Panagiotakopoulos, A.P. Kakarountas, C.E. Goutis, “Decoupled Processors Architecture for Accelerating Data Intensive Applications using Scratch-Pad Memory Hierarchy”, Journal of Signal Processing Systems. Springer Science + Business Media, LLC, vol. 59, no.3, pp. 281-296, 2010.

[37] H.E. Michail, A.P. Kakarountas, A.S. Milidonis, G.A. Panagiotakopoulos, V.N. Thanasoulis, C. E.Goutis, “Temporal and System Level Modifications for High Speed VLSI Implementations of Cryptographic Core” in Proc. of the IEEE 2006 International Conference on Electronics, Circuits and Systems (ICECS’06), Nice, France, pp. 1180-1183, Dec. 2006.

[38] Ray Kurzweil (Dec 19, 2014), Don’t Fear Artificial Intelligence. http://time.com/3641921/dont-fear-artificial-intelligence/

[39] Watson, AI for business (2016, IBM). https://www.ibm.com/watson/

[40] Asimov, Isaac (November 1956). "The Last Question". Science Fiction Quarterly.

[41] Wolfram Research Notebook (2017). “Eigenvalue, Eigenvector” http://mathworld.wolfram.com/Eigenvalue.html

[42] Raphael Hunger (2007) “Floating Point Operations in Matrix-Vector Calculus” Technical Report. https://mediatum.ub.tum.de/doc/625604/625604.pdf

[43] Raphael M. Robinson (1948). "Recursion and Double Recursion". Bulletin of the American Mathematical Society. 54 (10): 987–93. doi:10.1090/S0002-9904-1948-09121-2.

[44] Ackerman Function Representation as an infinite table: https://en.wikipedia.org/wiki/Ackermann_function (retrieved April 2018)

[45] Quattrociocchi W, Caldarelli G, Scala A (2014) Self-Healing Networks: Redundancy and Structure. PLoS ONE 9(2): e87986. https://doi.org/10.1371/journal.pone.0087986

[46] D. Hilbert: Über die stetige Abbildung einer Linie auf ein Flächenstück. Mathematische Annalen 38 (1891), 459–460.

[47] Alber, J.; Niedermeier, R. (2000). "On multidimensional curves with Hilbert property". Theory of Computing Systems. 33 (4): 295–312. doi:10.1007/s002240010003

[48] Hao Wang (1957), A Variant to Turing's Theory of Computing Machines, JACM (Journal of the Association for Computing Machinery) 4; 63-92. Presented at the meeting of the Association, June 23-25, 1954.

[49] Kurt Gödel, (1931), "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I", Monatshefte für Mathematik und Physik, v. 38 n. 1, pp. 173–198.

[50] Clemens Ley, (March 25, 2018) “Turing Complete Computation in Bitcoin”, Satoshi’s Vision Conference, Tokyo, Japan. https://www.satoshisvisionconference.com

[51] R. C. Merkle, (1988) "A digital signature based on a conventional Encryption function," Advances in Cryptology — CRYPTO ’87, Lecture Notes in Computer Science, Springer, vol. 293, pp. 369–378.

[52] Merve Can Kus Khalilov and Albert Levi, (2017) “A Survey on Anonymity and Privacy in Bitcoin-like Digital Cash Systems”, DOI 10.1109/COMST.2018.2818623, IEEE Communications Surveys & Tutorials.

[53] D. Eastlake III and T. Hansen. (2011, May). US Secure Hash Algorithms (SHA and SHA-Based HMAC and HKDF), RFC 6234 (Informational), Internet Engineering Task Force [Online]. Available: http://www.ietf.org/rfc/rfc6234.txt

[54] Florian Tschorsch and Björn Scheuermann (2016), Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. DOI 10.1109/COMST.2016.2535718, IEEE Communications Surveys & Tutorials.

[55] Alexander Chepurnoy, Vasily Kharin, Dmitry Meshkov (2018), Self-Reproducing Coins as Universal Turing Machine. https://arxiv.org/pdf/1806.10116.pdf

[56] Alexander Chepurnoy, Vasily Kharin, Dmitry Meshkov (2018), One bit per output rule 110 implementation in σ-state smart contract language, https://git.io/vj6rX

[57] Alexander Chepurnoy, Vasily Kharin, Dmitry Meshkov (2018), One layer per output rule 110 implementation in σ-state smart contract language, https://git.io/vj6sw

Λαμία, 24/8/2018

«Υπεύθυνη Δήλωση μη λογοκλοπής και ανάληψης προσωπικής ευθύνης»

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, και γνωρίζοντας τις συνέπειες της λογοκλοπής, δηλώνω υπεύθυνα και ενυπογράφως ότι η παρούσα εργασία με τίτλο «**Implementing A Church–Turing–Deutsch Principle Machine on a Blockchain**» αποτελεί προϊόν αυστηρά προσωπικής εργασίας και όλες οι πηγές από τις οποίες χρησιμοποίησα δεδομένα, ιδέες, φράσεις, προτάσεις ή λέξεις, είτε επακριβώς (όπως υπάρχουν στο πρωτότυπο ή μεταφρασμένες) είτε με παράφραση, έχουν δηλωθεί κατάλληλα και ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο ΔΗΛΩΝ

Κωνσταντίνος Σγάντζος

Ημερομηνία
24/8/2018

Υπογραφή

Implementing A Church–Turing–Deutsch Principle Machine on a Blockchain

Κωνσταντίνος Σγάντζος