

# Generic Blockchain on Generic Human Behavior

Clémentine Gritti<sup>1</sup><sup>a</sup>, Frédéric A. Hayek<sup>2</sup><sup>b</sup> and Pascal Lafourcade<sup>2</sup><sup>c</sup>

<sup>1</sup>University of Canterbury, New Zealand

<sup>2</sup>Université Clermont Auvergne, CNRS, Mines de Saint-Étienne, LIMOS, Clermont-Ferrand, France

**Keywords:** Blockchain, Consensus, Cryptocurrency.

**Abstract:** Blockchain is a type of distributed ledger. A wide range of consensus algorithms exists to reach consensus in a decentralized manner. However, most of them trade energy consumption for a degree of openness. Blockchains are primarily used for tokens and cryptocurrencies. Often the process of minting new tokens depends on actionable real world behaviors. A difficulty persists in securely translating said behavior into a decentralized blockchain. We formalize the generic concept of Proof of Behavior (PoB), and use it to create a consensus mechanism for generic permissionless blockchains.


## 1 INTRODUCTION


Blockchain technologies have attracted a lot of attention, with many possible applications (*e.g.*, cryptocurrency, healthcare, identity verification, *etc.*). However, they are not environmental-friendly. With climate change, it becomes urgent to develop and deploy technologies that act responsibly towards our planet.


A blockchain is a distributed ledger taking the form of a set of blocks “*chained*” together by hash functions. We call *miner* an entity who can and tries to create new blocks in a given blockchain. For creating valid blocks in Bitcoin (Nakamoto, 2008) (which introduced the world to blockchain in 2008 by Nakamoto), miners have to partially invert a cryptographic hash function – *i.e.*, find a nonce to embed in the current block such that the hash of said block is less than a given target. This process is called *Proof of Work* (PoW). Miners are then rewarded with freshly created bitcoins, the cryptocurrency whose transactions are written in the blocks. Mining has become so profitable that a lot of resources have been allocated to solve these inversion problems. Today Bitcoin mining’s energy consumption scales to that of nations (O’Dwyer and Malone, 2014). Many blockchains, such as Bitcoin Cash (BTC, ), Monero (based on Cryptonote (Van Saberhagen, 2013)) as well as Ethereum (Buterin et al., 2014) before the Merge, have also used PoW for blocks creation.

Nevertheless, blockchains by and large are shying away from PoW. Even blockchains that did implement PoW are trying to change their consensus mechanism. The best example is Ethereum’s Merge (*i.e.*, Ethereum 2.0) that replaced PoW with a version of Proof of Stake (PoS). PoS is a family of consensus mechanisms relying on miners’ stake (*i.e.*, wealth). All PoS claim to be energy efficient. It was first introduced by Peercoin’s Proof of Stake-Time (PoST) (King and Nadal, 2012). PoST is basically a PoW with an adjustable target such that the target depends on the coins held by the miner (amount and duration). *In fine* it does not vary significantly from PoW such as to create a paradigm change. All PoS mechanisms, by definition, restrict mining to those who have a stake in the system. Apart from the fact that this could be seen as an infringement on the openness of the system, PoS is generally thought of as either (a) an extra security layer: giving bigger stakeholders more responsibility in ensuring the safety of the system; or (b) allowing less decentralization: wealthy people are more powerful and arguably, very wealthy individuals who do not care about the underlying assets could make very dangerous attackers. Therefore, even if PoS has been introduced as an alternative to energy-intensive PoW, it brings limitations that could question the existence of decentralized ledgers.

PoW, PoST, along with Proof of Authority (PoA) and Proof of Space (PoSp), are considered Nakamoto-style consensus mechanisms, meaning that one miner gets the right to mine a block by doing *something*, and other miners can accept that block or reject it by sim-

<sup>a</sup> <https://orcid.org/0000-0002-0835-8678>

<sup>b</sup> <https://orcid.org/0000-0003-1083-0625>

<sup>c</sup> <https://orcid.org/0000-0002-4459-511X>

ply mining atop it or not. Contrary to Nakamoto-style consensus, some blockchains use Byzantine Fault Tolerant (BFT) consensus mechanisms. One good example of such a blockchain is Algorand (Gilad et al., 2017). However BFT consensus mechanisms cannot operate when the set of miners is unspecified, because their algorithms depend on the total number of miners which must be well defined. This constraint could restrict the variety of blockchain types.

With the recent growing trend in designing and releasing cryptocurrencies, different blockchain types have been presented, namely permissioned/permissionless and private/public blockchains. We define common blockchain types inspired by (Bernabe et al., 2019) as follows:

- **Public blockchains** where any entity can emit operations that can potentially be incorporated into the blockchain.
- **Private blockchains** where only a set of authorized entities can emit operations that can potentially be incorporated into the blockchain.
- **Permissionless blockchains** where any entity can participate in the mining process.
- **Permissioned blockchains** where only a set of authorized entities can participate in mining.

Note that through this definition, PoS blockchains are regarded as *permissioned* since only those who have a stake in the system can actually participate in the mining process. Others define blockchains as permissioned when the current set of miners selects their successors; and in PoS blockchains when somebody sells a stake to another person, the seller is also selling the right to mine; so in a way it is also the set of current miners who selects their successors. This definition enables us to identify *public permissionless* blockchains (e.g., Bitcoin (Nakamoto, 2008), Ethereum (Buterin et al., 2014)), *public permissioned* blockchains (e.g., Ethereum 2.0), *private permissioned* blockchains (such as the ones built with the Hyperledger Fabric framework (Androulaki et al., 2018)) and *private permissionless* (with no examples, as this scenario has almost no use cases). As such, most non-energy consuming blockchains are indeed *permissioned* (though it may be public).

A cryptocurrency is a currency that not only relies on cryptographic primitives, but must also have a certain degree of decentralization and privacy as described in (Lansky, 2018). Hence there is a distinction between regular digital currencies and cryptocurrencies.

Recent years have seen a rise of tokens rewarded to people for specific day-to-day actions such as creating solar-based energy using photovoltaics (e.g., Solarcoin (Johnson et al., 2015), walking and running

(e.g., Sweatcoin (Fomenko, 2022; Derlyatka et al., 2019; Elliott et al., 2019), car-pooling (e.g. and Eco-coin (Van Mensvoort et al., 2015)). Most of such tokens claim to be cryptocurrencies, yet lack the decentralization required to effectively be a cryptocurrency (Lansky, 2018).

Our aim in this paper is to propose a solution to overcome the aforementioned challenges: we propose a generic blockchain mined by using Proof of Behavior (PoB) (Grollemund et al., 2020). PoB aims to replace PoW by a specific human behavior – preferably an ecological one – to avoid consuming energy in hash computations on the one hand, and to further the development of eco-responsible habits on the other.

## 1.1 Contributions

In (Grollemund et al., 2020) the authors just define the concept of PoB, here we go above and beyond. We formally define a generic Proof of Behavior along with all information necessary to verify it and to extract its impact.

We construct a consensus mechanism for blockchains using the PoB along with well defined cryptographic primitives. This blockchain is energy efficient while it promotes the specific type of behavior that the PoB relies upon. We use a Verifiable Delay Function (VDF) (Boneh et al., 2018a) and then reduce the energy consumption to at most a couple of CPUs per miner which is negligible compared to unlimited parallelization in PoW-based consensus.

Furthermore by simply rewarding tokens to the blockchain miners, we are able to reward a specific behavior with the blockchain associated cryptocurrency. Optionally, we could choose to allow miners to also incorporate into the blockchain the PoBs of non-miners and reward them for it. This entails the creation of an ecosystem where all people who make a specific behavior are rewarded with cryptocurrency.

We construct the transaction fees in such a way that the miner is not biased towards bigger transactions nor smaller transactions: certain conditions must be met for this to be non-exploitable by the miner, but the underlying idea is that senders “pay” (in fact destroy) a fee rate in concordance with the size of the transaction, while the miner is rewarded the same fixed amount of coins for whatever transaction. We also propose to add a temporal demurrage (highly praised by many economists such as Silvio Gesell (Gesell, 1958)) on coins such that they lose value with time. We show how to incorporate this with the non-biased transaction construction.

There are many applications to such a cryptocurrency, and many pre-existing projects can benefit

from such a structure. We also detail an application to this “behavior chain” in *EcoMobiCoin*, a cryptocurrency mined by ecological mobility. To our knowledge, this is the first framework for using generic behaviors in a blockchain consensus mechanism and for rewarding generic behaviors with cryptocurrency.

## 1.2 Related Work

We are seeing the rise of projects aiming at creating cryptocurrency as a result of human action. Most of them target human mobility, perceived through the lens of ecology or through the lens of health.

**Ecocoin.** is a currency that aims to reward ecological behaviors (Van Mensvoort et al., 2015). These behaviors are defined by the Ecocoin foundation and include green commuting (such as car-pooling or biking), low home energy expenditure, low carbon footprint foods, etc. Interestingly, ecocoins are backed by trees: to each ecocoin in circulation corresponds a tree put in escrow. In the whitepaper and in public talks, conceivers of Ecocoin claim it as a cryptocurrency, and claim that people shall be rewarded for their actions according to the difficulty of said actions. For example, someone biking in Portugal earns more ecocoins than someone biking in the Netherlands because bikes are used much less often in Portugal. For the moment, Ecocoin is implemented in separate events and organizations independently. Furthermore, Ecocoin relies on *inspectors* to verify users’ behaviors. Inspectors can be people that personally attest of a certain behavior (as done with L’Oréal Paris in the Netherlands) or can be certified vendors such as online Marketplaces that could attest that you ordered low speed shipping. However, neither the whitepaper nor any online resource explicit its underlying distributed ledger. Hence, there are no security guarantees whatsoever. Furthermore, the project seems abandoned since 2019.

**Sweat.** is a cryptocurrency aiming at complementing Sweatcoin (Fomenko, 2022; Derlyatka et al., 2019; Elliott et al., 2019). Sweatcoin is a digital token (not a cryptocurrency) that is awarded to people who do physical efforts, such as walk, bike, swim, etc. Users can then use sweatcoins in affiliated markets. Sweat on the other hand is a cryptocurrency also rewarded to users for their physical activity, however only the person’s first 5’000 steps can earn them Sweat. After that, they can only get sweatcoins. Furthermore, Sweat implements demurrage as a non-activity fee. The Decentralized Autonomous Organization (governed by all holders of Sweat) sets the

activity threshold and the corresponding fees. Sweat is built on the Near blockchain (Polosukhin and Skidanov, 2022). Sweat relies on *Movement Validators* to validate a person’s physical activity. SweatCo Ltd is the only Movement Validator as of launch. Verification is done through the analysis of raw data sent by the recording device. It is intended that in the future, third parties such as activity trackers, fitness equipment makers, and fitness app developers can operate as Movement Validators. However verification of raw data sent by recording devices is prone to attacks, for example by manually manufacturing the data. We define a Proof of Behavior in such a way to require the impossibility of synthesize the data.

**Clinicoin.** is a cryptocurrency on Clinicoin’s public permissioned blockchain<sup>1</sup>. Clinicoin connects three types of people: (i) users who are regular people who log healthy activities; (ii) validators who validate (through what is called a Proof-of-Engagement) a person’s activity, they can be developers of health apps or in-person validators such as exercise instructors or personal trainers; and (iii) healthcare and research providers who are interested in people’s health data. Users can choose whether to share their data with providers or not.

Examples above use (entirely or partially) GPS traces to determine users’ behaviors, which is not necessarily tamper-proof, nor privacy-preserving. Nevertheless, we notice that projects that aim at creating cryptocurrencies as a reward for human behavior tend to restrict mining. This may be for security purposes or for efficiency. Some projects, such as Weward<sup>2</sup>, used to have a blockchain before completely abandoning it in favor of a digital token architecture.

**Solarcoin.** is a cryptocurrency created to reward people for solar energy produced. It is minted at the rate of 1 solarcoin per 1 MWh of solar energy produced. To get solarcoins, one must send proof of one’s solar energy production to the Solarcoin Foundation, which in turn rewards the energy producer with coins on the blockchain. The Solarcoin Foundation used to use its own blockchain, but have now migrated onto the Energy Web Chain (Hartnett et al., 2019) which uses a Proof of Authority type of consensus. Clearly Solarcoin lacks decentralization since it relies on a single authority to verify solar energy production, not to mention the Proof of Authority blockchain.

<sup>1</sup><https://clinicoin.io/en> (accessed on 2023-03-07).

<sup>2</sup><https://en.weward.fr/> (accessed on 2023-03-07).

**Primecoin.** (King, 2013a; King, 2013b) is a cryptocurrency that instead of wasting computing power to do a regular PoW which serves no purpose except choosing the next miner, miners do “*computational work*” that actually is useful. Primecoin’s miners are chosen based on who creates new specific prime-number related sequences. The goal is to replace PoW’s by a useful energy consumption. This takes the form of “working” on finding Cunningham and bi-twin chains, which are interesting to mathematicians.

**Elapsed Time Consensus Protocols.** encompass Proof of Elapsed Time (Chen et al., 2017) and Proof of Luck (Milutinovic et al., 2016), since they share many similarities. Basically, these consensus mechanisms rely on Trusted Execution Environments (TEEs) to wait a certain period of time before emitting a proof that they did so. This proof is what gives a miner mining rights. The miner will thus incorporate it into the block, making it a valid block. In (Bowman et al., 2021), they also propose a way to not rely on TEEs. This relies on a function **z-test** which they modify into a “time-based” z-test to suit their proof. This function allows them to calculate if a given miner has mined too many blocks, in which case it will restrict them from mining for a period of time. It is easy to see that this only works in permissioned blockchains (which they do specify as “private”) since in permissionless blockchains miners can create new identities on the fly, and are thus immune to restrictions from the z-test.

Many blockchains claim to be energy efficient *and* permissionless. However, all PoS blockchain, by definition, are not permissionless, be it regular PoS, PoST, liquid PoS (used in Tezos (Goodman, 2014)). Few consensus mechanisms can claim to be non energy consuming *and* permissionless. One such example is PoSp (Park et al., 2018). PoSp can be implemented in many forms, most interesting of which are Proof of Retrievability (PoR) (Miller et al., 2014) (also known as Proof of Storage) consisting of proving that a certain data is being correctly stored at the time of the proof. However, PoSp constructions require evergrowing memory, and apart from their direct application (storing files in a distributed manner) show no further development.

### 1.3 Road Map

In the following section, we present the cryptographic tools that we need to design a PoB blockchain. In Section 3, we define our eco-friendly Proof of Behavior (PoB). In Section 4, we propose a blockchain us-

ing consensus mechanism based on PoB. In Section 5, we focus on the security of such PoB blockchain. In Section 6, we explain the process of blockchain-based demurrage. In Section 7, we suggest our main application, called *EcoMobiCoin*, based on our PoB blockchain, along with other possible examples. Finally, we conclude our paper in the last section.

## 2 CRYPTOGRAPHIC TOOLS

### 2.1 Hash Functions

Hash functions are an important building block of blockchain technologies. Hash functions take as input data of arbitrary length and output a string of fixed length. Hash values in such context are used to identify blockchain transactions and to generate PoWs.

We require the hash function to be computationally efficient and deterministic. Given an input  $x$ , computing  $H(x) = y$  is efficient and always gives the same output  $y$ . Moreover, we need the hash function to be collision-resistant: it must be infeasible to find two different inputs  $x$  and  $x'$  such that they both yield the same output  $y = H(x) = H(x')$ .

### 2.2 Digital Signatures

Blockchains require the use of digital signatures in order to authenticate users’ transactions. When a user submits a transaction to the network, it must prove that it has enough funds to spend while protecting itself from getting those funds spent by other users. Then the submitted transaction is verified and the new blockchain state is agreed upon by the network.

Let Alice plan to send 1 coin to Bob. Using her private key, Alice signs a transaction spending 1 coin and submits both the transaction and its signature to the network. The miners check the contents of the submitted transaction as well as the validity of the associated signature using the public key of Alice. If the signature is valid, then the transaction is confirmed and included in the upcoming block.

**Definition 1.** A digital signature scheme consists of three probabilistic polynomial time algorithms, denoted as *KeyGen*, *Sign* and *Verify<sub>sign</sub>*, such that:

$KeyGen(\lambda) \rightarrow (pk, sk)$  is a randomized algorithm that takes a security parameter  $\lambda$ , and outputs the pair of public key  $pk$  and private key  $sk$ .

$Sign(sk, m) \rightarrow \sigma$  is a deterministic algorithm taking the private key  $sk$  and the to-be-signed message  $m$  and outputs a signature  $\sigma$ .



$\text{Verify}_{\text{sign}}(pk, m, \sigma) \rightarrow \{\text{Accept}, \text{Reject}\}$  is a deterministic algorithm taking the message  $m$ , its signature  $\sigma$  along with the supposed signing public key  $pk$  and either outputs *Accept* if  $\sigma$  is a valid signature for  $m$  with  $pk$  or *Reject* otherwise.

**Correctness.** If  $(pk, sk) \leftarrow \text{KeyGen}(\lambda)$  and  $\sigma \leftarrow \text{Sign}(m, sk)$  for a given message  $m$ , then  $\text{Verify}_{\text{sign}}(m, \sigma, pk) \rightarrow \text{Accept}$ .

**Unforgeability.** We are interested in digital signature schemes that are existentially unforgeable under a chosen-message attack (Goldwasser et al., 1988). Informally, let an adversary get access to signatures for messages that he chooses. This adversary must not be able to create a valid signature for a new message.

### 2.3 Verifiable Delay Functions

For our consensus mechanism, we rely on Verifiable Delay Functions (VDFs) (Boneh et al., 2018a). A verifiable delay function is a function that cannot be computed in less than a pre-defined number of sequential computations, and it must be “quickly” verifiable.

**Definition 2.** A VDF scheme consists of three algorithms, *Setup*, *Eval* and  $\text{Verify}_{\text{VDF}}$ , such that:

$\text{Setup}(\lambda, t) \rightarrow pp = (ek, vk)$  is a randomized algorithm that takes a security parameter  $\lambda$  and a desired puzzle difficulty  $t$  and produces public parameters  $pp$  that consist of an evaluation key  $ek$  and a verification key  $vk$ . We require *Setup* to be polynomial time in  $\lambda$ . By convention, the public parameters specify an input space  $X$  and an output space  $\mathcal{Y}$ . We assume that  $X$  is efficiently sampleable. *Setup* might need secret randomness, leading to a scheme requiring a trusted setup. For meaningful security, the puzzle difficulty  $t$  is restricted to be sub-exponentially sized in  $\lambda$ .

$\text{Eval}(ek, x) \rightarrow (y, \pi)$  takes as input the public parameter  $ek$  and  $x \in X$  and produces an output  $y \in \mathcal{Y}$  and a (possibly empty) proof  $\pi$ . *Eval* may use random bits to generate the proof  $\pi$  but not to compute  $y$ . For all  $pp$  generated by  $\text{Setup}(\lambda, t)$  and all  $x \in X$ , algorithm  $(\text{eval}, ek, x)$  must run in parallel time  $t$  with  $\text{poly}(\log(t), \lambda)$  processors.

$\text{Verify}_{\text{VDF}}(vk, x, y, \pi) \rightarrow \{\text{Accept}, \text{Reject}\}$  is a deterministic algorithm that takes as inputs the public parameter  $vk$ , an entry  $x \in X$ , the supposed corresponding output  $y \in \mathcal{Y}$  and the supposed corresponding proof  $\pi$  and outputs *Accept* or *Reject*.  $\text{Verify}_{\text{VDF}}$  must run in total time polynomial in  $\log(t)$  and  $\lambda$ .

**Sequentiality.** Any honest user can generate the pair  $(y, \pi) \leftarrow \text{Eval}(pp, x)$  in  $t$  sequential steps, while any parallel-machine adversary with a polynomial number of processors cannot distinguish the output  $y$  from random values in significantly fewer steps.

**Efficient Verifiability.** The algorithm  $\text{Verify}_{\text{VDF}}$  is aimed to be as fast as possible for honest users to run. In particular, the total time should be of the order  $O(\text{polylog}(t))$ .

**Uniqueness.** For every input  $x$ , it is difficult to obtain a value  $y$  such that  $\text{Verify}_{\text{VDF}}(pp, x, y, \pi) \rightarrow \text{Accept}$ , but  $y \neq \text{Eval}(pp, x)$ .

**Constructions.** Two possible constructions are Wesolowski’s construction (Wesolowski, 2019) and Pietrzak’s construction (Pietrzak, 2018). They are both efficient and simple, though the first is a little more efficient and the latter a little simpler. A detailed comparison and analysis of both constructions can be found in (Boneh et al., 2018b).

## 3 PROOF OF BEHAVIOR

Our generic blockchain is mined using a *Proof of Behavior* (PoB) (Grollemund et al., 2020). The blockchain’s genericity consists of the type of behavior the blockchain seeks to promote.

A PoB should contain (i) data about the behavior itself, which allows anyone to verify the behavior; (ii) a timestamp (since a PoB has a limited duration) and (iii) the identity of its producer. All these information should be signed together by a validator binding the producer with the timestamp with the behavior. The validator could be operated by a human actor or by an automatic verification mechanism such as a Trusted Platform Module. It is the signature by the validator that prevents the user from modifying the timestamp.

**Definition 3.** A Proof of Behavior (PoB) is a tuple  $\pi = (b || \sigma_b)$  where  $b$  is the tuple  $b = (pk, ts, data)$  containing  $pk$  the identity of the producer (seen here as a public key),  $ts$  the timestamp of the behavior and  $data$  some auxiliary data about the behavior itself; and  $\sigma_b$  is the signature of  $b$  by a given validator able to verify that the behavior included in  $data$  was done at timestamp  $ts$  by  $pk$ .

Accompanying a PoB is the Quantify function defined as follows:

$\text{Quantify}(\pi) \rightarrow v$  is a deterministic algorithm that takes as input the PoB  $\pi$ , and outputs a non-

negative number  $v \geq 0$  that represents the value of said behavior.

Quantify plays the role of a verification function such that Quantify returns a non-negative number to allow for variation of behavior importance. In certain cases where all valid PoBs are equally important, Quantify outputs only either 0 (invalid) or 1 (valid).

**Example.** Let us consider a behavior as mobility using public transportation where users tap their card on the terminal. Such a terminal can be seen as a proxy of the public transportation organization. When the user taps their card on the terminal, the terminal, acting as the validator, creates a PoB  $\pi = (b||\sigma_b)$  with  $b = (pk, ts, data)$  where  $pk$  is the public key of the user,  $ts$  is a timestamp and  $data$  is data about the journey being travelled. To verify and quantify the proof, we plug it into Quantify which should output 1 if the PoB is valid and 0 otherwise. If in the underlying transportation network it is required to tap one's card at entry *and* exit of stations (which is the case in many places), then the Quantify function outputs a number proportional to the distance covered.

**Quantifiable.** We say a PoB scheme is quantifiable when  $\exists \pi, \text{Quantify}(\pi) \notin \{0, 1\}$ .

**Trust.** In the example above, we must trust the transportation authority. This hardly makes for a universal trust-based system. Instead, it could lead to different cities, states or countries adopting a clone of the same blockchain, since they do not trust each others' transportation authorities. However, the validator does not always need to be a physical or moral person. In other instances, it could simply be a signer of some Trusted Platform Module (TPM), or it could be any of a set of validators. Centralization around validators depends on the instantiation of the blockchain.

**PoB Expiration.** Only "recent" PoBs should ever be incorporated in the blockchain. Let  $exp$  be a time unit. We consider a PoB *expired* when its timestamp is more than  $exp$  older than the newest PoB in the blockchain up to this point. A discussion on expiration implications is given in Section 4.4.

## 4 BLOCKCHAIN AND CRYPTOCURRENCY

We describe a possible generic cryptocurrency that rewards specific human behaviors and its underlying

public permissionless blockchain based on our PoB consensus mechanism.

### 4.1 Mining and Consensus

Our mining mechanism goes as follows: instead of miners competing with raw computational power to gain mining rights, they compete with their behaviors. If PoBs are quantifiable, then bigger behaviors amount to more chance of being the next miner. To mine atop a given block  $\beta$ , the miner  $m$  with a PoB  $\pi_m$  has to compute a VDF with a delay parameter depending on  $\pi_m$ . Specifically, this delay parameter is

$$t = \frac{\delta \cdot H(\beta || \pi_m)}{\text{Quantify}(\pi_m)} \quad (1)$$

where  $\delta$  is a difficulty parameter and  $H(\cdot)$  is a cryptographic hash function. Table 1 summarizes all blockchain parameters (*i.e.*, values that have to be determined with each instantiation of the blockchain).

Table 1: Blockchain parameters.

$exp$	Expiration time unit
$\delta$	Difficulty
$P$	Difficulty adjustment period
$\Delta_t$	Desired average mining time
$fr$	Fee rate
$tr$	Fixed miner transaction reward
$k$	Miner's reward rate for own PoB
$k'$	Non-miners' reward rate for own PoB
$k''$	Miner's reward rate for not own PoB
$min_{cur}$	Minimal subdivision of currency
$d_{block}$	Demurrage rate per block
$d_{day}$	Demurrage rate per day

**Discussion on Mining.** Thanks to the  $\beta$  being one of the inputs of the hash function, the length of the VDF using a given PoB cannot be predicted (and thus the VDF cannot be launched) until the previous block is known. Furthermore, more important PoBs have more chance of inducing a smaller  $t$  yielding a quicker VDF thanks to the division by  $\text{Quantify}(\pi_m)$ . Naturally, when there are more miners, the probability that all of their PoBs yield long VDFs is lower. The variable  $\delta$  should be adjusted every given time period  $P$  (that corresponds to a certain number of blocks). For instance, for a desired average mining time  $\Delta_t$ , every  $P$  blocks, for an average mining time of  $\bar{\Delta}_t$  over the last  $P$  blocks,  $\delta$  is updated as follows:  $\delta \leftarrow \delta \cdot \frac{\bar{\Delta}_t}{\Delta_t}$ . Like is done in Bitcoin (Nakamoto, 2008), we could naturally add minimum and maximum values for  $\delta$ 's update, such as no less than the quarter of the previous value and no more than the quadruple of the previous value.

Furthermore, notice that the VDF does not depend on the current block’s contents. Hence, on the downside the miner could potentially create many valid sibling blocks simultaneously, but on the upside it is impossible for a miner to parallelize mining by computing many VDFs in parallel, each with a different block content. More on the downside of this in Section 5.

Figure 1 summarizes the mining process: people produce PoBs and transactions (denoted by Tx), and put them in a pool. Then miners, using their own PoBs, compute the VDF, and the first miner to complete their computations publishes a new block.

**Main Chain.** As in Bitcoin (Nakamoto, 2008), the main chain is the sub-chain with the highest cumulative difficulty, consisting of the sum of the  $\delta$  of each of its blocks.

## 4.2 Cryptocurrency

In a permissionless blockchain, the miner is rewarded for creating the block as well as the work put in for having mining rights, *i.e.*, in our case this is doing a PoB and the corresponding VDF. The reward  $r$  is proportional to the quantification of the behavior  $r = k \cdot \text{Quantify}(\pi)$  for  $k > 0$ .

We choose to reward all PoBs, and not only the miners’. To do so, the miner can incorporate in their block not only transactions, but PoBs as well. And the producers of those PoBs get rewarded – but to a lesser extent than the miner. The reward  $r'$  for a PoB not used for mining is also proportional to the quantification of the underlying PoB  $\pi'$ :  $r' = k' \cdot \text{Quantify}(\pi')$ , with  $0 < k' < k$ .

To incentivize the miner to incorporate PoBs in the block, the miner gets rewarded for doing so. Specifically, let  $r''$  be that reward for a given PoB  $\pi'$  incorporated by the miner, the reward is proportional to the quantification of said PoB:  $r'' = k'' \cdot \text{Quantify}(\pi')$  with  $0 < k'' < k' < k$ .

## 4.3 Transactions

Coins can only be created by doing PoBs. Every PoB incorporated in the blockchain should be associated with a special transaction that creates coins.

There are three types of transactions:

1. *Regular transactions*: send coins.
2. *Reward transactions*: reward PoB producers.
3. *Coinbase transactions*: reward the miner.

Below we describe those three types of transaction.

### 4.3.1 Regular Transactions

Regular transactions are divided into three parts, the first containing the transaction input, the second containing the fees and the third containing the transaction output, as shown in Figure 2. Each transaction can have as many inputs as the sender wishes. However, they only have one or two outputs (one input for the receiver, and a second one for returning the remainder to the sender), plus one fee.

**Transaction Fees.** We wish to incentivize the miner to incorporate transactions in the blockchain. We also wish to not distinguish between transactions: cryptocurrencies generally favor transactions with big fees, which favor bigger transactions over smaller ones (since the fees could be higher while remaining the same proportion of the amount transacted).

Our solution is to have the sender destroy (“pay”) a certain rate of the transaction’s value, the *fee rate* denoted as  $fr \in ]0, 1[$ , and the miner be always paid the same fixed amount for any transaction, the *transaction reward* denoted as  $tr$ . There must be restrictions on the fee rate, the transaction reward, the minimal amount sent, denoted as  $min_{sent}$ , and the minimal unit of currency, denoted as  $min_{cur}$ , as follows:

$$fr \cdot min_{sent} \geq tr \geq min_{cur} \quad (2)$$

This prohibits the miner, economically, from making many bogus transactions with themselves.

**Valid Regular Transactions.** Let  $n$  be the number of inputs to the transaction, and  $in_i$  denote the  $i^{th}$  input. Let  $m$  denote the number of outputs, and  $out_i$  denote the  $i^{th}$  output. A transaction is *valid* if the next two equations are verified:

$$(1 - fr) \cdot \sum_{i=1}^n in_i \geq \sum_{i=1}^m out_i \quad (3)$$

$$\forall i \in [1, m], \exists j \in \mathbb{N}, out_i = j \cdot min_{cur} \quad (4)$$

Equation 4 ensures that all outputted coins are indeed multiples of the minimal value. Note that no upper limit is imposed on the amount of burned coins. Equation 3 ensures that the output is less or equal to the input minus the fees.

**Example.** Suppose the minimal unit of currency is  $min_{cur} = 1$ , the fee rate  $fr = 0.1$ , and the transaction reward  $tr = 1$ . Firstly, with such values, we must have  $min_{sent} \geq 10$ . If Alice wishes to send 55 coins to Bob, then out of these 55 coins, only  $\lfloor (1 - fr) \cdot 55 \rfloor = \lfloor (1 - 0.1) \cdot 55 \rfloor = \lfloor (0.9) \cdot 55 \rfloor = \lfloor 49.5 \rfloor = 49$  coins actually will be sent to Bob. Moreover, Alice will pay 6 coins as fees, and the miner will receive 1 coin only.

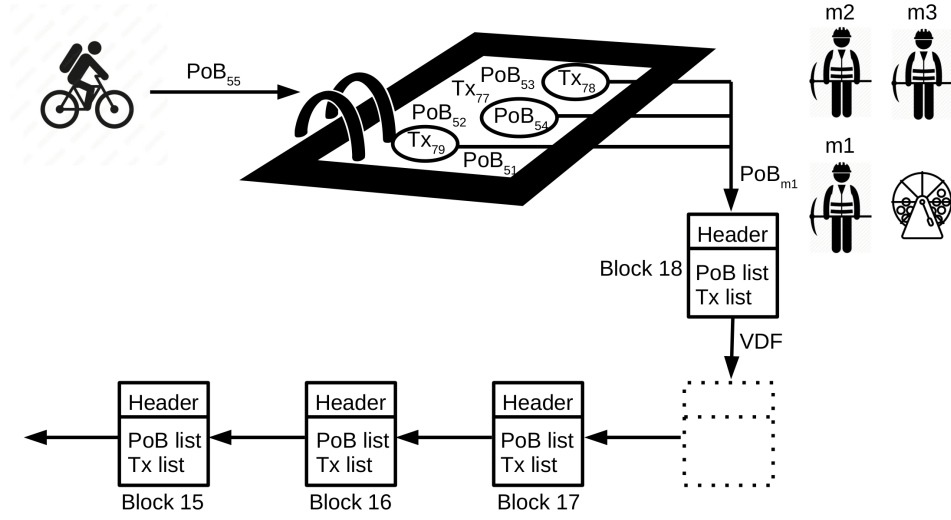


Figure 1: Blockchain construction.

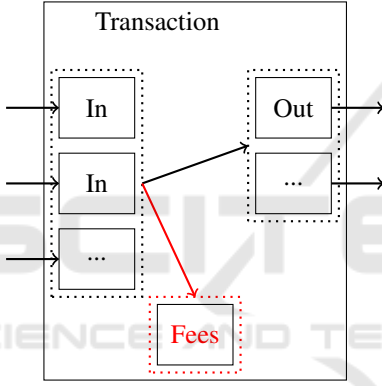


Figure 2: Transaction Structure.

### 4.3.2 Reward Transactions

There are as many reward transactions as there are PoBs. Each reward transaction has one input, which is a PoB, and one output, which rewards its producer.

**Valid Reward Transactions.** A reward transaction is *valid* if and only if, for an input  $\pi$  and an output  $out$ , we have:

$$k' \cdot \text{Quantify}(\pi) \geq out \quad (5)$$

$$out + \text{min}_{cur} \geq k' \cdot \text{Quantify}(\pi) \quad (6)$$

$$\exists j \in \mathbb{N}, out = j \cdot \text{min}_{cur} \quad (7)$$

Equation 5 ensures that the output is no more than the value of the PoB. Equation 6 ensures that the miner does not give the producer of the PoB less than what they are expected to get. Equation 7 ensures that the output is a multiple of the minimal value.

### 4.3.3 Coinbase Transactions

There is only one coinbase transaction per block. It has no input, since it depends on the whole block, and one output  $out$  which rewards the miner.

**Valid Coinbase Transactions.** Let  $n$  be the number of regular transactions in the block. Let  $m$  be the number of reward transactions in the block. Let  $\pi_i$  denote the PoB of the  $i^{\text{th}}$  reward transaction in the block. Let  $\pi_m$  denote the miner's PoB. Let  $Q(\cdot) = \text{Quantify}(\cdot)$  (for space purposes). A coinbase transaction is *valid* if and only if it verifies the following:

$$\exists j \in \mathbb{N}, out = j \cdot \text{min}_{cur} \quad (8)$$

$$k \cdot Q(\pi_m) + n \cdot tr + \sum_{i=1}^m k'' \cdot Q(\pi_i) \geq out \quad (9)$$

Equation 9 ensures that the miner does not reward itself more than he is allowed to: on the left hand side, the first term considers the miner's reward for its own PoB, the second considers the reward for including  $n$  transactions and the third is the rewards for including other people's PoBs. Equation 8 ensures that the output is a multiple of the minimal value.

## 4.4 Blocks

A block is comprised of a VDF output and its associated proof and PoB, the coinbase transaction (the reward of the miner for the work put in), other PoBs and their rewards, regular transactions, and the signature of all aforementioned data by the miner. Concretely, a block  $\beta$  contains  $\beta = (data_\beta, \sigma_{data_\beta})$  with  $data_\beta = (\pi_{PoB_m}, y, \pi_{VDF}, Tx_{miner}, (\pi_{PoB_i}, Tx_i)_{i \in I}, (Tx_j)_{j \in J}, \pi_{PoB_m} = (b || \sigma_b)$  with  $b = (pk_m, ts, data_{PoB})$ .



**Block Size.** The blocks’ size must be bounded. The actual size should depend on the PoBs’ size and their frequency, so this is blockchain dependent. Hence we do not give a specific size in this paper. Also, a certain percentage of the blocks must be allocated to PoBs and another to transactions. Otherwise the miner could fill up the block with whatever is most rewarding – potentially only PoBs or only transactions. The percentage of the block size also depends on the PoBs’ size and frequency.

**Valid Blocks.** For a block to be deemed valid, it must verify the following:

- All transactions must be valid.
- All PoBs (including the miner’s) must be valid, *i.e.*,  $\forall \pi \in PoBlist, Quantify(\pi) > 0$ , as well as non-expired and not already in the blockchain.
- The miner must be the person who has done the VDF’s PoB, *i.e.*, the recipient of  $Tx_{miner}$  is the account associated with  $pk_m$ .
- The block must be signed by the miner, *i.e.*,  $Verify_{sign}(pk_m, data_\beta, \sigma_{data_\beta}) \rightarrow Accept$ .
- The VDF must be valid, *i.e.*, with  $v \leftarrow Quantify(\pi_{PoB_m})$  and  $t \leftarrow \frac{\delta \cdot H(\beta || \pi_{PoB_m})}{v}$  we have  $Setup(\lambda, t) \rightarrow (ek, vk)$  and  $Verify_{VDF}(vk, \beta_{prev}, y, \pi_{VDF}) \rightarrow Accept$ .

**Discussion on Expiration.** For a block to be deemed valid, all of its PoBs must be less than  $exp$  older than the newest PoB in the blockchain. Note that a PoB’s timestamp, which is signed by a validator, is compared only to that of other PoBs (who are already in the blockchain). This avoids any synchrony assumption between miners and users. As a consequence, when checking if a PoB is already in the blockchain, one only needs to check the last couple of blocks. Specifically, to check if a PoB  $\pi$  that is in the current block isn’t already in the blockchain, we start by checking if it is in the previous block, then the one before it, and so on, until we reach a block where the newest PoB is more than  $exp$  older than  $\pi$ . Hence  $\pi$  cannot be before such a block: suppose such a block is valid and  $\pi$  exists in the blockchain before that block; then that block’s PoBs are all more than  $exp$  older than  $\pi$  which itself is at least as old as the oldest PoB in the blockchain so far; then that block cannot be valid because its PoBs are expired.

#### 4.5 Energy Consumption

For the energy consumption we only take into account the consensus mechanism. We do not take into account the required verification of valid transactions

nor the optional quantification of other people’s PoBs, for the former is necessary in all blockchains and the latter is optional. Suppose all miners are mining using the entirety of their non-expired PoBs (less than  $exp$  older than the newest PoB used for mining in the blockchain), then corresponding to each such PoB is a VDF being computed. Since VDFs are non-parallelizable, for each non-expired PoB is one CPU computing a VDF. Unlike other Nakamoto-style consensus, miners here cannot augment their mining power by augmenting computers.

## 5 SECURITY ANALYSIS

Our blockchain achieves the same security properties as Bitcoin’s Backbone Protocol (Garay et al., 2015):

**Persistence (informally).** Once a transaction goes more than  $k$  blocks deep into the blockchain of one honest miner, then it will be included in every honest miner’s blockchain with overwhelming probability, and it will be assigned a permanent position in the ledger.

**Liveness (informally).** All transactions originating from honest account holders will eventually end up at a depth more than  $k$  blocks in an honest miner’s blockchain, and hence the adversary cannot perform a selective denial of service attack against honest account holders.

### Proof Sketch

**Model.** In the Bitcoin Backbone Protocol (Garay et al., 2015), the execution of a protocol  $\Pi$  is driven by an environment program  $\mathcal{Z}$  which may spawn multiple instances running the protocol  $\Pi$ . The programs in question can be thought of as Interactive Turing machines (ITM) that have communication, input and output tapes. The environment performs a round-robin participant execution sequence for a fixed set of parties. The execution driven by  $\mathcal{Z}$  is defined with respect to a protocol  $\Pi$ , an adversary  $\mathcal{A}$  (also an ITM) and a set of parties  $P_1, \dots, P_n$ ; these are hardcoded in a control program  $C$  (also an ITM). The adversary can corrupt at most  $t$  out of  $n$  total parties. Parties have access to two functionalities: the *random oracle* and the *diffusion channel*. The number of queries to the random oracle is bound to at most  $q$  per round per party. Remark that this is a flat-model interpretation of the parties’ computation power where all parties are assumed equal. In the real world, different honest parties may have different hashing power, nevertheless the flat-model does not sacrifice generality since one

can imagine that real honest parties are simply clusters of some arbitrary number of honest flat-model parties. The protocol relies on 4 algorithms:

**Chain Validation.** The first algorithm performs a validation of the structural properties of a given chain  $C$ . For each block of the chain, the algorithm checks that the Proof of Work is properly solved, that the counter  $ctr$  does not exceed  $q$  and that the hash of the previous block is properly included in the block. It also verifies the consistency of the data included in the blocks.

**Chain Comparison.** This algorithm finds the best possible chain when given a set of chains *i.e.*, the longest chain. This algorithm relies on the chain validation algorithm.

**Proof of Work.** This algorithm seeks to find a valid block by producing a PoW. To do so, it repeatedly queries the random oracle hash function with the current block containing different nonces. The nonce is represented with a counter variable  $ctr$  initialized at 1 and incremented by 1 at each try. The total number of queries is bounded by  $q$ . If the algorithm finds a valid block, then the chain is extended and returned; otherwise the chain is returned unaltered. Part of the algorithm is represented below (where  $(ctr, h)$  represents the block and  $T$  the hash target):

```

while  $ctr \leq q$  do
  if  $H(ctr, h) < T$  then
    Append block to blockchain
    return Blockchain
   $ctr \leftarrow ctr + 1$ 
return Blockchain

```

**The Backbone Protocol.** This is the algorithm executed by the miners and which is assumed to run indefinitely. Put simply, the algorithm calls the chain comparison algorithm to get the best possible chain. Then it tries to produce a PoW using the Proof of Work algorithm.

**Assumptions.** The authors in (Garay et al., 2015) assume that the protocol is executed by a fixed number of miners, however the number itself is not necessarily known to the miners. The miners themselves cannot authenticate each other and therefore there is no way to know the source of a message. It is assumed that messages are eventually delivered and all parties in the network are able to synchronize in the course of a round. The hypotheses are: (i) the adversary controls less than half of the total hashing power, (ii) the network synchronizes much faster relative to the PoW solution rate and (iii) digital signatures cannot be forged.

**Parallel with Proof of Behavior.** We propose first an equivalent version of the Proof of Work algorithm, and then draw a parallel between the new algorithm and our blockchain. Let  $\tau$  be the smallest positive counter value such that the hash of the block is less than  $T$ . (Note that it may be that  $\tau > q$ ; this simply means that it takes more than  $q$  hashes to get to the counter realizing a valid block, which means that it takes more than 1 round to realize a PoW.) Initialize the counter at 1. Instead of checking if the hash of the block is less than  $T$ , we check if the counter is equal to  $\tau$ . If it is, then we have found a valid block, if not we increment the counter and repeat, until the counter reaches  $q$ . This yields the following instead:

```

while  $ctr \leq q$  do
  if  $ctr = \tau$  then
    Append block to blockchain
    return Blockchain
   $ctr \leftarrow ctr + 1$ 
return Blockchain

```

This algorithm is equivalent to the one used in the Proof of Work algorithm, yet it can encompass the VDF construction. Indeed, when doing a VDF there is also a fixed number of steps to do before finishing it. The only difference with a PoW is that this number is known in advance, which does not change the algorithms in the slightest. We can thus model our blockchain with exactly the Bitcoin Backbone Protocol. Their hashing power translates to mining power, which in our case corresponds to the number of PoBs per miner and their importance (*i.e.*, quantification). Therefore all the proofs follow from (Garay et al., 2015) (under the same assumptions), and our blockchain is in fact an operational transaction ledger.

## 6 DEMURRAGE

The economist Silvio Gesell was a firm believer in temporal monetary demurrage (Gesell, 1958). Demurrage is money quantity diminishing with time. For example suppose a demurrage rate of 10% per day, then if Alice has 100 units today, she'll have only 90 tomorrow, then 81 the day after, *etc.* The goal is to incentivize spending and prevent hoarding. Demurrage has been replicated a couple of times throughout history, but was difficult to put in practice because of the difficulty to discriminate old bills. The use of periodic stamps on bills was not practical. This would be easy to implement in a digital currency. We can smooth the demurrage over, by making the demurrage take place at every block. However, one can infer the average daily demurrage rate from the block rate and vice-versa. The daily demurrage rate  $d_{day}$  and the

block demurrage rate  $d_{block}$  are related by the formula:  $d_{day} = 1 - (1 - d_{block})^n$  where  $n$  is the average number of blocks mined per day.

**Example.** Suppose a rate of 10% per day. If Alice has 100 coins today, she will be left with 90 coins tomorrow, then 81 coins the day after tomorrow, etc... Suppose there are on average 144 blocks per day (same mining rate as Bitcoin, *i.e.*, one block every 10 minutes (Nakamoto, 2008)). If we aim to have a daily demurrage rate of  $d_{day} = 10\% = 0.1$ , then we must have a block demurrage rate of  $d_{block} = 1 - \sqrt[n]{1 - d_{day}} = 1 - \sqrt[144]{1 - 0.1} = 0.00073 = 0.073\%$ .

Demurrage must be taken into account during transactions. So the input to transactions, minus the demurrage, minus the fees must equal the output (to the next whole integer). For  $n$  a total number of transaction inputs, and for  $i \in [1, n]$  let  $in_i$  denote the  $i^{th}$  input, let  $age_i$  be the age (in number of blocks) of the  $i^{th}$  input. Then a block is deemed valid if and only if:

$$\left[ (1 - fr) \cdot \sum_{i=1}^n (1 - d_{block})^{age_i} \times in_i \right] = \sum_{i=1}^m out_i \quad (10)$$

$$\forall i \in [1, m], \exists k \in \mathbb{N}, out_i = k \cdot min_{cur} \quad (11)$$

The same idea of temporal demurrage was the cornerstone of the Freicoins cryptocurrency<sup>3</sup> This is a Proof of Work based blockchain implementing a temporal demurrage. We explicit how to put the temporal demurrage into work in our blockchain and its transaction system.

## 7 APPLICATIONS

### 7.1 EcoMobiCoin

One application of PoB-based blockchains is ecological mobility. To go back to the example provided in Section 3, we can have PoBs created by a transportation seller for their customers. When customers tap their card upon the terminal, the terminal emits a PoB for them<sup>4</sup>. Customers then have two options: they can either utilize these PoBs and plug them into VDFs to gain mining rights, or simply put it in a mining pool where some miner might take it and incorporate it into their block. The latter option still is less lucrative but still provides them with some coins. Such a cryptocurrency would indeed reward all willing customers with coins while remaining fully distributed.

<sup>3</sup><http://freico.in/> (accessed on 2023-03-15).

<sup>4</sup>The Quantify function could for example quantify PoBs based on the distance travelled by public transports.

### 7.2 Other Applications

Let us try to replace the digital currencies that reward walking and running using our construction. In this case, we have a PoB  $\pi = (b || \sigma_b)$  where  $b = (pk, ts, data)$  with  $pk$  the public key of the user,  $ts$  the timestamp and  $data$  containing for instance the GPS trace of the behavior along with acceleration detection information and gyroscope information. As for  $\sigma_b$ , we require it to be the signature of  $b$  by either the TPM while using virtualization, or by trusted sensors. These are two constructions proposed in (Saroiu and Wolman, 2010) who could play the role of a validator and verify that  $data$  is indeed authentic and has not been tampered with. As for Quantify, it is a public classifier that takes in  $\pi$ , and if the signature checks out, returns a quantification of the mobility. Classifiers like this are already used by Google, Sweatcoin or other applications.

## 8 IMPLEMENTATION

We are currently working on an implementation of this blockchain. The work in progress is applied to the ecological mobility idea. Though it is meant to become generic to any type of behavior.

## 9 CONCLUSION

In this paper, we have put forth a truly distributed generic cryptocurrency that rewards specific human behavior. It is based on Proofs of Behavior (PoB), and uses Verifiable Delay Functions (VDFs) among other cryptographic primitives. Thanks to this construction, the underlying blockchain is completely permissionless, as long as the creation of PoBs is unrestricted. Moreover, the mining process's energy consumption is extremely limited: only one CPU runs per PoB per *exp*. We understand that many consensus mechanisms are even less energy consuming, but they seldom are permissionless. We also propose a non-biased fee system: neither in favor of bigger transactions nor in favor of smaller ones. We have also depicted how to implement demurrage into a cryptocurrency, which we do in a different way than Freicoins. We are working on an implementation of our blockchain. Technical details thus lie as future work.

## REFERENCES

Bitcoin cash. <https://bitcoincash.org/>. Accessed: 2023-02-

- 14.
- Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., et al. (2018). Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the thirteenth EuroSys conference*, pages 1–15.
- Bernabe, J. B., Canovas, J. L., Hernandez-Ramos, J. L., Moreno, R. T., and Skarmeta, A. (2019). Privacy-preserving solutions for blockchain: Review and challenges. *IEEE Access*, 7:164908–164940.
- Boneh, D., Bonneau, J., Bünz, B., and Fisch, B. (2018a). Verifiable delay functions. In *Annual international cryptology conference*, pages 757–788. Springer.
- Boneh, D., Bünz, B., and Fisch, B. (2018b). A survey of two verifiable delay functions. Cryptology ePrint Archive, Paper 2018/712.
- Bowman, M., Das, D., Mandal, A., and Montgomery, H. (2021). On elapsed time consensus protocols. In *International Conference on Cryptology in India*, pages 559–583. Springer.
- Buterin, V. et al. (2014). A next-generation smart contract and decentralized application platform. *white paper*, 3(37):2–1.
- Chen, L., Xu, L., Shah, N., Gao, Z., Lu, Y., and Shi, W. (2017). On security analysis of proof-of-elapsed-time (poet). In *International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pages 282–297. Springer.
- Derlyatka, A., Fomenko, O., Eck, F., Khmelev, E., and Elliott, M. T. (2019). Bright spots, physical activity investments that work: Sweatcoin: a steps generated virtual currency for sustained physical activity behaviour change. *British Journal of Sports Medicine*, 53(18):1195–1196.
- Elliott, M., Eck, F., Khmelev, E., Derlyatka, A., Fomenko, O., et al. (2019). Physical activity behavior change driven by engagement with an incentive-based app: evaluating the impact of sweatcoin. *JMIR mHealth and uHealth*, 7(7):e12445.
- Fomenko, O. (2022). Sweat economy. Technical report, SweatCo Ltd. Accessed: 2023-03-07 :[https://drive.google.com/file/d/1IPklRcEQvgJkCaeYvGh43yjWl-Dj5\\_6i/view/](https://drive.google.com/file/d/1IPklRcEQvgJkCaeYvGh43yjWl-Dj5_6i/view/).
- Garay, J., Kiayias, A., and Leonardos, N. (2015). The bitcoin backbone protocol: Analysis and applications. In *Advances in Cryptology-EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 281–310. Springer.
- Gesell, S. (1958). *The natural economic order*. Owen London.
- Gilad, Y., Hemo, R., Micali, S., Vlachos, G., and Zeldovich, N. (2017). Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th symposium on operating systems principles*, pages 51–68.
- Goldwasser, S., Micali, S., and Rivest, R. L. (1988). A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM Journal on Computing*, 17(2):281–308.
- Goodman, L. (2014). Tezos: A self-amending crypto-ledger position paper. *self-published paper*, 3.
- Grollemund, P.-M., Lafourcade, P., Thiry-Atighehchi, K., and Tichit, A. (2020). Proof of behavior. In *The 2nd Tokenomics Conference on Blockchain Economics, Security and Protocols*. hal-02559573.
- Hartnett, S., Henly, C., Hesse, E., Hildebrandt, T., Jentzch, Christoph and Krämer, K., MacDonald, G., Morris, J., Touati, H., Trbovich, A., and von Waldenfels, J. (2019). The energy web chain: Accelerating the energy transition with an open-source, decentralized blockchain platform. Technical report, Energy Web Foundation.
- Johnson, L., Isam, A., Gogerty, N., and Zitoli, J. (2015). Connecting the blockchain to the sun to save the planet. Available at SSRN 2702639.
- King, S. (2013a). Primecoin: Cryptocurrency with prime number proof-of-work. Technical report, PrimeCoin. Accessed: 2023-02-14 :<https://primecoin.io/>.
- King, S. (2013b). Primecoin: Cryptocurrency with prime number proof-of-work. *July 7th*, 1(6).
- King, S. and Nadal, S. (2012). Ppcoin: Peer-to-peer cryptocurrency with proof-of-stake. *self-published paper*, 19(1).
- Lansky, J. (2018). Possible state approaches to cryptocurrencies. *Journal of Systems integration*, 9(1):19–31.
- Miller, A., Juels, A., Shi, E., Parno, B., and Katz, J. (2014). Permacoin: Repurposing bitcoin work for data preservation. In *2014 IEEE Symposium on Security and Privacy*, pages 475–490. IEEE.
- Milutinovic, M., He, W., Wu, H., and Kanwal, M. (2016). Proof of luck: An efficient blockchain consensus protocol. In *proceedings of the 1st Workshop on System Software for Trusted Execution*, pages 1–6.
- Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*.
- O’Dwyer, K. J. and Malone, D. (2014). Bitcoin mining and its energy footprint. *IET*.
- Park, S., Kwon, A., Fuchsbauer, G., Gaži, P., Alwen, J., and Pietrzak, K. (2018). Spacemint: A cryptocurrency based on proofs of space. In *International Conference on Financial Cryptography and Data Security*, pages 480–499. Springer.
- Pietrzak, K. (2018). Simple verifiable delay functions. In *10th Innovations in Theoretical Computer Science conference (ITCS 2019)*.
- Polosukhin, I. and Skidanov, A. (2022). The near white paper. Technical report, SweatCo Ltd. Accessed: 2023-03-07 :<https://near.org/papers/the-official-near-white-paper/>.
- Saroiu, S. and Wolman, A. (2010). I am a sensor, and i approve this message. In *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*, pages 37–42.
- Van Mensvoort, K., Just, L., Perlin, A., Blezer, K., Baart, R., Rolfe, K., and Sijmons, P. (2015). The eco coin: A cryptocurrency backed by sustainable assets. Technical report, Eco Coin.
- Van Saberhagen, N. (2013). Cryptonote v 2.0.
- Wesolowski, B. (2019). Efficient verifiable delay functions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 379–407. Springer.