On (multi-stage) Proof-of-Work blockchain protocols

P. D'Arco[1] and F. Mogavero[1]

Dipartimento di Informatica
University of Salerno
Via Giovanni Paolo II, 132
I-84084, Fisciano (SA), Italy
pdarco@unisa.it,
francescomogavero@outlook.com

Abstract. In this paper we analyze permissionless blockchain protocols, whose distributed consensus algorithm lies on a Proof-of-Work composed of $k \geq 1$ sequential hash-puzzles. We put our focus on a restricted scenario, widely used in the blockchain literature, in which the number of miners, their hash rates, and the difficulty values of the hash-puzzles are constant throughout time. Our main contribution is a *closed-form* expression for the *mining probability* of a miner, that is, the probability the miner completes the Proof-of-Work of the next block to be added to the blockchain before every other miner does. Our theoretical results can be applied to existing Proof-of-Work based blockchain protocols, such as Bitcoin or Ethereum. We also point out some security issues implied by our findings, which make not trivial at all the design of multi-stage (i.e. $k \geq 2$) Proof-of-Work blockchain protocols.

Keywords: Mining probability · Hypoexponential distribution · Proof-of-Work · Blockchain

1 Introduction

Blockchains. Bitcoin and the blockchain technology came to the fore in 2008, when an enigmatic web navigator published the famous Bitcoin Whitepaper, using the pseudonymous of Satoshi Nakamoto [18]. Bitcoin was presented as a decentralized cryptocurrency working on top of the Bitcoin's blockchain, which, in turn, was proposed as a public, tamper-resistant, distributed, and decentralized transaction ledger, maintained and replicated *entirely* and *consistently* by anonymous, unpermissioned, and trustless nodes, in a weakly synchronized ([25]) peer-to-peer network.

Roughly speaking, the Bitcoin blockchain is a chain of blocks. Each block has a set of block headers. Block headers include a hash pointer to a *Merkle tree* storing some transactions, and a hash pointer to the previous block in the chain [19]. Bitcoin transactions are mainly used for Bitcoin transfers between Bitcoin addresses. The only way to extend the Bitcoin's blockchain, as well as to create new coins, is by *mining* a new block that accommodates transactions not already added to the blockchain. In order to mine a block, it is required to complete a Proof-of-Work. In general, in current permissionless blockchain protocols, the Proof-of-Work consists of solving a single hash-puzzle. The hash-puzzle game works as follows: every *distributed competitor*, called miner, runs on-top of a Bitcoin node and he is required to compose a new block with some of the valid transactions he has heard from the network. Each miner tries to find a *nonce* to add to the block headers of his composed block, in such a way that the hash value of the set of block headers is, in binary, lower than the *target* value of the hash-puzzle [19]. The first miner who finds a valid nonce for his block – the first miner who completes the Proof-of-Work – is the winner. He broadcasts his block to the network, and he gets as a reward an amount of coins, given by the sum of the current *block reward* value and the *transaction fees* of every transaction in the proposed block [1]. At the same time, he starts to mine the next block of the chain, following the one just mined. Any other *non-faulty* miner receives the block, 8 verifies that the block is *valid*, and adds it to his local version of the blockchain. Then, he starts to mine the next block of the chain, following the one just mined.

Outline and contributions. Our contribution is twofold. Firstly, in Section 2, we provide a brief overview of blockchain protocols whose *consensus* mechanism lies on a Proof-of-Work composed of a single hash-puzzle. This Proof-of-Work model is currently used in blockchains as Bitcoin and Ethereum. In Subsection 2.1, we provide a mathematical analysis of this Proof-of-Work model, with respect to the *target* and *difficulty* value of the hash-puzzle, and to the *hash rates* of the miners. In Bitcoin the hash rate indicates the number of trials in the Proof-of-Work game a miner performs per unit of time. If we consider 1 second as a unit of time, then the unit of measure of the hash rate is the number of hash function executions per second (hash/s) [2]. In particular, our contribution formally proves, under some simplifying assumptions widely used in the blockchain literature, the equivalence between the *ratio of the global hash rate* a miner has, that is, the ratio between the hash rate of the miner and the sum of the hash rates of all the miners, and his *mining probability*, that is, the probability the miner completes the Proof-of-Work of the next block to be added to the blockchain before every other miner does. Similar work was conducted by Houy in 2016 [11]. Secondly, in Section 3, we introduce recent related work on alternative blockchain protocols whose Proof-of-Work is composed of $k \geq 1$ sequential hash-puzzles. In Section 4 we extend and generalize the analysis made in Subsection 2.1 (under the same simplifying assumptions) to this new model of blockchain protocols. In these protocols, every miner has to sequentially find k *nonces* to add to the block headers of his proposed block, in order to complete the Proof-of-Work of his block. Each miner can start the hash-puzzle number $s+1$ of the Proof-of-Work of a block *only after* he has found a valid nonce for the hash-puzzle number s. The first miner who completes the last hash-puzzle of the Proof-of-Work of his proposed block is the winner. We provide a *closed-form* expression for the *mining probability*, which is valid in Proof-of-Works composed of a generic $k \geq 1$ number of sequential hash-puzzles. We exhibit an example which shows that, if $k \geq 2$, then the ratio of the global hash rate held by a miner and his mining probability are *not necessarily* equal. Such an example rises up important security issues.

2 Single hash-puzzle Proof-of-Works

In Subsection 2.1, we provide a brief overview of blockchain protocols whose *consensus* mechanism lies on a Proof-of-Work composed of a single hash-puzzle. In particular, we present a mathematical analysis of the mechanics of this Proof-of-Work model, and we formally prove, under some simplifying assumptions widely used in the blockchain literature, the equivalence between the *ratio of the global hash rate* held by a miner and his *mining probability*.

In Subsection 2.2, we complete our overview by presenting some problems that can occur in real-world single-stage Proof-of-Work based blockchain protocols.

2.1 A mathematical analysis of Proof-of-Works composed of a single hash-puzzle

Let $h > 0$ be the current hash rate of a miner, and let d be a positive value, denoting the current *difficulty* of the hash-puzzle. The difficulty value in Bitcoin, d, is $d = 2^{256}/t$, where t is a positive value indicating the current *target* value of the hash-puzzle. The hash-puzzle's output space is $\{0,1\}^{256}$, since the Bitcoin hash-puzzle uses a double SHA-256 hash function circuit.

Roughly speaking, using the *random-oracle* assumption[1] to model the hash function (assumption

[1] We will implicitly use this assumption throughout the entire work.

widely used in the Bitcoin and the blockchain literature [10, 20, 5]), for any distinct input x, the output of the hash function can be considered as *uniformly* and *independently* distributed in the output space. It follows that, under the assumptions of constant hash-puzzle difficulty, every trial the miner makes to complete the Proof-of-Work is an independent experiment, each of them having probability of success[2] $prob = t/2^{256} = 1/d$. The independent experiment is also known as a *Bernoulli trial* [10, 12, 14]. Under the assumptions of constant hash-puzzle difficulty and constant hash rate, the number of successes *per unit of time* in a sequence of *identically distributed* Bernoulli trials is described by a *homogeneous Poisson point process*, having *rate* parameter $\lambda = h/d$ [7, 21]. The rate parameter indicates the average number of successes per unit of time[3]. In our scenario, the *homogeneous Poisson point process* is a stochastic process, which describes the number of Proof-of-Works the miner completes per *interval of time*; a new point found in the homogeneous Poisson point process corresponds bijectively to a new valid *nonce* found – a new Proof-of-Work completed, or, equivalently, a new block mined – by the miner. Formally, the probability of having $k \in \mathbb{N}_0$ successes in the interval of time (a, b) is [7]

$$P\big(\text{Successes(a,b)} = k\big) \overset{\text{def}}{=} \big(\Lambda(b) - \Lambda(a)\big)^k \cdot e^{-\big(\Lambda(b)-\Lambda(a)\big)} \cdot (k!)^{-1} \tag{1}$$

Here $\Lambda(t)$ is $\Lambda(t) \overset{\text{def}}{=} \int_0^t \lambda(t)dt$, $\forall t \geq 0$ [7]. The function $\lambda(t)$ is the aforementioned rate parameter. Given our assumptions on constant hash rate and difficulty parameter, $\lambda(t)$ is constant throughout time and the process is homogeneous. Moreover, as we stated before, we have $\lambda(t) = h/d$, $\forall\, t \geq 0$ [7, 21].

The interarrival time between two successive points in a homogeneous Poisson point process, as well as the time between the time a miner begins to participate in the Proof-of-Work of a block (i.e. "*point 0*" in his Poisson process) and the time the miner finds the first point ("*point 1*" in his Poisson process), are described by a *exponential distribution* with the same λ-parameter of the process [6, 19].

Consider $M \geq 2$ competing miners trying to mine the block with *block height* $\gamma \in \mathbb{N}_0$. The block height value of a block indicates the number of blocks in the blockchain preceding that block. Let X_p be the exponential distribution describing the interarrival time between two successive blocks found by miner p, for each $p \in \{1, \ldots, M\}$. The parameter of X_p is $\lambda_p = h_p/d$, where $h_p > 0$ is the hash rate of miner p, and $d > 0$ denotes the difficulty of the hash-puzzle. Denote by $f_{X_p}(t) = P(X_p = t) = \lambda_p \, \mathrm{e}^{-\lambda_p \cdot t}$ the *probability density function*, and by $R_{X_p}(t) = P(X_p > t) = \mathrm{e}^{-\lambda_p \cdot t}$ the *survival function* relative to X_p. If each miner starts to mine block γ at the same time, then the winner is the miner whose Poisson point process first finds a point. Therefore, for each $p \in \{1, \ldots, M\}$, the probability with which miner p mines block γ is independent from the hash-puzzle's difficulty and target values, and it is equal to the ratio of the global hash rate the miner possesses, as we formally

[2] For completeness, it is worth noting that, in the literature, the difficulty value is also indicated as $d' = (2^{16} - 1) \cdot 2^{208}/t \approx 2^{224}/t$, and, as a consequence, the probability value as $prob = t/2^{256} \approx 1/(d' \cdot 2^{32})$ [17]. The value $(2^{16} - 1) \cdot 2^{208} \approx 2^{224}$ is the maximum allowed *target* value in Bitcoin [17, 19]. Nevertheless, our results are independent from this consideration, as they can be equivalently applied to both notations by setting $d' \approx d/2^{32}$.

[3] In a homogeneous Poisson process the average number of successes in t units of time is $\lambda \cdot t$ [6]. If we let t be a unit of time, i.e. $t = 1$, then the statement holds.

prove below

$$P\left(p \text{ mines block } \gamma\right) = \int_0^\infty f_{X_p}\left(t\right) \prod_{\substack{z=1 \\ z\neq p}}^M R_{X_z}\left(t\right) dt = \int_0^\infty \lambda_p \, e^{-\lambda_p \cdot t} \prod_{\substack{z=1 \\ z\neq p}}^M e^{-\lambda_z \cdot t} \, dt =$$

$$= \int_0^\infty \lambda_p \prod_{z=1}^M e^{-\lambda_z \cdot t} \, dt \; = \int_0^\infty \lambda_p \, e^{-(\sum_{z=1}^M \lambda_z)\cdot t} \, dt \; = \; \lambda_p \int_0^\infty e^{-(\sum_{z=1}^M \lambda_z)\cdot t} \, dt =$$

$$= \lambda_p \, \frac{-1}{\sum_{z=1}^M \lambda_z} \cdot \left[e^{-(\sum_{j=0}^M \lambda_j)\cdot t}\right]_0^\infty = \lambda_p \, \frac{-1}{\sum_{z=1}^M \lambda_z} \cdot (-1) = \frac{\lambda_p}{\sum_{z=1}^M \lambda_z} = \frac{h_p}{\sum_{z=1}^M h_z}$$

On the other hand, a new Proof-of-Work is globally completed every time one of the M miners "finds a point" in his Poisson point process. Thus, the global block mining event can be mathematically represented as the sum of the M homogeneous Poisson point processes. Due to the mutual independence of the processes, the sum of the M homogeneous Poisson point processes is still a homogeneous Poisson point process, with *rate* parameter $\lambda_{glob} = \sum_{p=1}^M \lambda_p = \sum_{p=1}^M h_p/d$. Hence, the waiting time until a new Proof-of-Work is globally completed is given by an exponential distribution with a constant rate parameter λ_{glob}. The expected value of the exponential distribution – the expected time until a new block is mined –, λ_{glob}^{-1}, should be $\lambda_{glob}^{-1} = 10$ minutes [19, 14].

However, despite the aforementioned static model, widely used in theoretical studies on the Bitcoin and blockchain technology [7, 13, 21], in the real Bitcoin network the number of miners involved in the mining game and their hash rates are not constant throughout time. Consequently, the hash-puzzle difficulty needs to be periodically updated. Therefore, the Poisson point processes describing a miner's mining process and the global mining process are not necessarily homogeneous [7]. The Bitcoin's hash-puzzle difficulty parameter is periodically updated every 2016 blocks added to the *longest valid* chain ([19]) of the blockchain, by considering the block arrival times of the last 2016 added blocks [13], in order to let the mean waiting time between two successive blocks mined be ten minutes. Moreover, the value of the current global hash rate, $h_{glob} = \sum_{p=1}^M h_p$, is not public, since the hash rates of every miner are unknown, but it can be implicitly estimated from the block arrival times of the past blocks [7, 13, 3].

2.2 Forks

Due to *faulty* actions, caused by network propagation delays or malicious miners, temporary forks in the Bitcoin blockchain are possible. To cope with this issue, the Bitcoin protocol has adopted the *longest-valid* chain rule [19]. Accordingly, each honest network node selects the longest *valid-chain* of his local version of the blockchain as the relevant chain. A chain is *valid* if it is composed only of *valid* blocks and transactions. A transaction is *valid* if it has been written according to the Bitcoin rules, the digital signature of the coin sender is valid, and the sender has not tried to double-spend a coin that he had already spent in another transaction. A block is *valid* if its Proof-of-Work has been solved, and every transaction the block accommodates is valid [19].

At the same time, forks may also cause a block recently added in a miner's longest chain to suddenly "*disappear*" from the blockchain. Indeed, another miner may broadcast to the network a different longest valid chain. In this case, the most recent blocks of the first miner become *stale* or *orphan* block, and all the *transaction sets* they contain will "*disappear*" with them. To address this issue, the Bitcoin rules have adopted the heuristic of *transaction confirmations.* A transaction is considered to be *permanently* added to the blockchain if a block that contains the transaction has been mined, and six other blocks have been mined on top of it, such that the block is the seventh last block in its branch. The intuition is that, if six successive blocks are mined extending a branch, then, with high probability, the majority of the network nodes have that branch in their local views of the blockchain. At this point, this branch will presumably be the longest valid chain in the long-term.

For completeness, we present a brief overview of what a faulty miner is [26, 16]. Generally, a miner is faulty when he mines a block that does not completely respect the blockchain protocol rules. A faulty behavior may be *accidentally* caused by external factors or voluntarily caused by the miner's actions. As an example regarding the first case, the blockchain local version in possession of the miner may not be fully up-to-date, and thus the miner's last mined block, which extends the miner's longest chain, may not extend other miner's longest chain. In this case, the other miners would reject the block, after receiving it. In the second case, a miner may intentionally misbehave, that is, he may be a *malicious* miner. As an example, he may intentionally mine a new block on top of the blockchain's branch that he prefers, in order to perform an attack such as the *double-spending attack* [22].

3 Related work on Multi-Stage Proof-of-Work Blockchain protocols

In this section, we present an overview of related work on multi-stage Proof-of-Work blockchain protocols. In particular, in Subsection 3.1, we describe a recently proposed multi-stage Proof-of-Work blockchain protocol, while in Subsection 3.2 we list some issues we found on the proposed protocol.

3.1 Multi-Stage Proof-of-Work blockchain

The first work on multi-stage Proof-of-Work blockchain protocols dates back to 2019 [23]. The protocol proposed in [23], which is based on Bitcoin technology, targeted the waiting time for transaction confirmation in Bitcoin. Since in Bitcoin a new block is globally found on average every ten minutes, it is required averagely about an hour until the transactions, carried by the newest mined block, are confirmed. The author proposed an alternative mining game, by dividing the Proof-of-Work in $k \geq 1$ consecutive hash-puzzles, that have to be solved sequentially. He denoted each hash-puzzle with the term *stage*. We use the terms *stage* and *sequential hash-puzzle* interchangeably. Moreover, the author proposed a pipeline-like architecture for block mining, to increase the efficiency of mining. The goal was to get a higher *block throughput rate*, and, consequently, a lower time for transaction confirmations, and a higher *transaction processing rate*[4]. The author pointed out to have obtained, *implicitly*, also the *sharding* ([15]) property. Roughly speaking, the sharding property incentivizes a more *collaborative* mining among distributed miners, thanks to which the network miners divide their work trying to jointly mine a *shared* block composed of an agreed upon set of transactions [15]. In [23], the network miners are free to partition themselves into groups, called *pipelines*. Miners inside a pipeline are furtherly partitioned into subgroups. The sharding property in the blockchain protocol proposed in [23] ensures that miners belonging to different subgroups of a pipeline work *collaboratively* on different pieces, or *shards*, of a job, in order to mine a shared block.

Hardware incompatible hash functions. The protocol employs μ hardware incompatible hash functions, $G_0, \ldots G_{\mu-1}$. Two hash functions are hardware incompatible if any type of ASIC, or other special-purpose hardware, that allows to compute the output of one function faster than general-purpose mining hardware, cannot be easily reconfigured to give an advantage over general-purpose

[4] Presently, Bitcoin processes 7 transactions per second on average [19].

hardware in the computation of the other function [23]. There is no correlation between k and μ. The purpose of using hardware incompatible hash functions is to make it harder for miners to obtain high values for the hash rates in multiple stages. The author suggested the NIST finalists for the SHA-3 competition as a valid set of hardware incompatible hash functions.

Transactions. A transaction is a tuple $(\mathsf{IL}, \mathsf{OL}, \sigma)$ [23], where

1. $\mathsf{IL} = ((\mathsf{pk}_1, \mathsf{c}_1), \ldots, (\mathsf{pk}_s, \mathsf{c}_s))$, given a $s \geq 1$. For each $i \in \{1, \ldots, s\}$, pk_i is a public key, c_i is the amount of coins to be withdrawn from the Bitcoin address $H(\mathsf{pk}_i)$, where H is a hash function defined at protocol level.
2. $\mathsf{OL} = ((\alpha_1, \delta_1), \ldots, (\alpha_t, \delta_t))$, given a $t \geq 1$. For each $j \in \{1, \ldots, t\}$, α_j is a recipient address and δ_j is the amount of coins to send to α_j.
3. $\sum_{i=1}^{s} \mathsf{c}_i \geq \sum_{j=1}^{t} \delta_j$. The difference $\left(\sum_{i=1}^{s} \mathsf{c}_i\right) - \left(\sum_{j=1}^{t} \delta_j\right) \geq 0$ is the value of the sum of the transaction fees in the block.
4. σ is the set of signatures on the pair $(\mathsf{IL}, \mathsf{OL})$, computed with the private keys sk_i corresponding to pk_i, for each $i \in \{1, \ldots, s\}$.

Genesis-blocks composition and their Proof-of-Work. The first k blocks of the blockchain, $B_0, \ldots, B_{k-1}$ are the genesis-blocks. They do not carry any transaction, and they must be mined to bootstrap the cryptocurrency and to mine some initial coins. This way, it becomes possible making transactions, by transacting the existing coins. The structure of the i-th genesis-block, for each $i \in \{0, \ldots, k-1\}$, is the following [23]

$$\boxed{\begin{array}{l} i, \\ \text{bdigest}_i, \\ t_i,\ \eta_i,\ \tau_i,\ \alpha_i,\ \mathsf{c}_i \end{array}}$$

where

$$\begin{cases} i \text{ is the block number, such that } 0 \leq i \leq k-1 \\ \text{bdigest}_0 \ = H_0\left(0, t_0, \eta_0, \tau_0, \alpha_0, c_0\right) \\ \text{bdigest}_i \ = H_i\left(\text{bdigest}_{i-1}, t_i, \eta_i, \tau_i, \alpha_i, c_i\right) \text{ for each } i \in \{1, \ldots, k-1\} \\ t_i \ \text{ is the target value of the hash puzzle of block } \ i, \ \text{ for each } i \in \{0, \ldots, k-1\} \\ \eta_i \ \text{ is the nonce of block } \ i, \ \text{ for each } i \in \{0, \ldots, k-1\} \\ \tau_i \ \text{ is the timestamp of the completion time of block } \ i\text{'s Proof-of-Work}, \ \text{ for each } i \in \{0, \ldots, k-1\} \\ \alpha_i \ \text{ is the address of the recipient of block } i\text{'s reward}, \ \text{ for each } i \in \{0, \ldots, k-1\} \\ \mathsf{c}_i \ \text{ is the reward for mining block } \ i, \ \text{ for each } i \in \{0, \ldots, k-1\} \end{cases}$$

The Proof-of-Work for the genesis-block i is valid if $\text{bdigest}_i < t_i$. Thus, the Proof-of-Work of a genesis-block is almost identical to Bitcoin's.

General-blocks composition and their Proof-of-Work. The Proof-of-Work of a general-block is divided into $k \geq 1$ sequential stages. The hash function of the s-th stage is $H_s = G_{s \bmod \mu}$, for each $s \in \{0, \ldots, k-1\}$. Each stage target and difficulty parameters are set such that, globally, the expected time to solve each stage, denoted by T, is the same. There is no fixed amount of coins given to a single user as *block reward*, since the rewarding system is divided into *stage rewards*. A stage reward is a fixed amount of coins, given to the user that successfully completes a stage of a block. It consists of newly created coins, which will be effectively generated once the block has been fully mined and submitted to the network. Moreover, the winners of different stages of a block also have to divide the transaction fees obtained from the transactions in the block.

The structure of a general-block is the following [23]

$$\boxed{\begin{array}{l} \text{bn}, \\ \text{bdigest}, \\ \mathcal{L}, \\ t_0,\ \eta_0,\ \tau_0,\ \alpha_0,\ \mathsf{c}_0 \\ t_1,\ \eta_1,\ \tau_1,\ \alpha_1,\ \mathsf{c}_1 \\ \vdots \\ t_{k-1},\ \eta_{k-1},\ \tau_{k-1},\ \alpha_{k-1},\ \mathsf{c}_{k-1} \end{array}}$$

where

$$\begin{cases} \text{bn} \geq k \text{ is the block number} \\ \text{bdigest is the digest of the block} \\ \mathcal{L} \ \text{ is the possibly empty hash tree of transactions carried by the block} \\ t_s \ \text{ is the target value of the hash puzzle of stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \eta_s \ \text{ is the nonce of stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \tau_s \ \text{ is the timestamp of the completion time of stage } \ s\text{'s hash-puzzle}, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \alpha_s \ \text{ is the address of the recipient of stage } s\text{'s reward}, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \mathsf{c}_s \ \text{ is the reward for completing stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \end{cases}$$

Consider a general-block B_{i+k}, with $i \geq 0$. Based on the definition of the protocol, the outputs of the stages and the value of bdigest_{i+k} are obtained through the following computation [23]

$$\begin{cases} g_{i+k,0} = H_0 \left(\text{bdigest}_i, i+k, \mathsf{RH}(\mathcal{L}_{i+k}), t_{i+k,0}, \alpha_{i+k,0}, c_{i+k,0}, \tau_{i+k,0}, \eta_{i+k,0}\right) \\ g_{i+k,1} = H_1 \left(\text{bdigest}_{i+1}, g_{i+k,0}, t_{i+k,1}, \alpha_{i+k,1}, c_{i+k,1}, \tau_{i+k,1}, \eta_{i+k,1}\right) \\ \vdots \\ g_{i+k,k-1} = H_{k-1} \big(\text{bdigest}_{i+k-1}, g_{i+k,k-2}, t_{i+k,k-1}, \alpha_{i+k,k-1}, c_{i+k,k-1}, \tau_{i+k,k-1}, \\ \qquad\qquad\qquad \eta_{i+k,k-1}\big) \end{cases}$$

where

$$\begin{cases} \mathsf{RH}(\mathcal{L}_{i+k}) \ \text{ is the root hash of the possibly empty hash tree in which the} \\ \qquad\qquad\quad \text{block's transactions are stored} \\ g_{i+k,s} \text{ is output of stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \\ t_{i+k,s} \ \text{ is the target value of the hash puzzle of stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \eta_{i+k,s} \ \text{ is the nonce of stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \tau_{i+k,s} \ \text{ is the timestamp of the completion time of stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \alpha_{i+k,s} \ \text{ is the address of the recipient of stage } s\text{'s reward}, \ \text{ for each } s \in \{0,\dots,k-1\} \\ \mathsf{c}_{i+k,s} \ \text{ is the reward for completing stage } \ s, \ \text{ for each } s \in \{0,\dots,k-1\} \end{cases}$$

The Proof-of-Work is valid if and only if $g_{i+k,s} < t_{i+k,s}$, for each $s \in \{0,\dots,k-1\}$. Finally, the value of $g_{i+k,k-1}$ is assigned to bdigest_{i+k}. Following the protocol, stage s of block B_{i+k}'s Proof-of-Work requires the output of stage $s-1$ of block B_{i+k}'s Proof-of-Work as input, for each $i \geq 0$ and for each $s \in \{1, \ldots, k-1\}$. Furthermore, stage s of block B_{i+k}'s Proof-of-Work requires $\mathrm{bdigest}_{i+s}$ as input, for each $i \geq 0$, and for each $s \in \{0, \ldots, k-1\}$. The first general-block has the second condition always satisfied, since all of his previous blocks have been already mined by someone in the network to bootstrap the protocol. We will exploit later this property in our security analysis on the mining probability regarding block B_k. Due to the nature of the Proof-of-Work, the author suggested a pipeline-like mining architecture, in which *physical* miners which work in the same pipeline are partitioned into k groups. The architecture may be realized by letting the k groups work in parallel and on different *shards* of the same problem, according to the *sharding* property mentioned earlier in this Section. If the general-blocks already mined in the blockchain are $i \geq 0$, group s works on stage s of block B_{i+k-s}, for each $s \in \{0, \ldots, k-1\}$. This way, miners in the same pipeline belonging to different groups mine *collaboratively* to complete the Proof-of-Work of a block before the other pipelines do, in order to obtain the mining reward. At the same time, miners on the same stage mine *competitively*. Finally, different pipelines compete against each other to mine the blocks. Generally, once a miner in a group completes a stage, he broadcasts to the network all the information necessary to start the successive stage of the same block, and to prove that the stage has been successfully completed. The pipeline-like mining architecture is depicted in Figure 1.

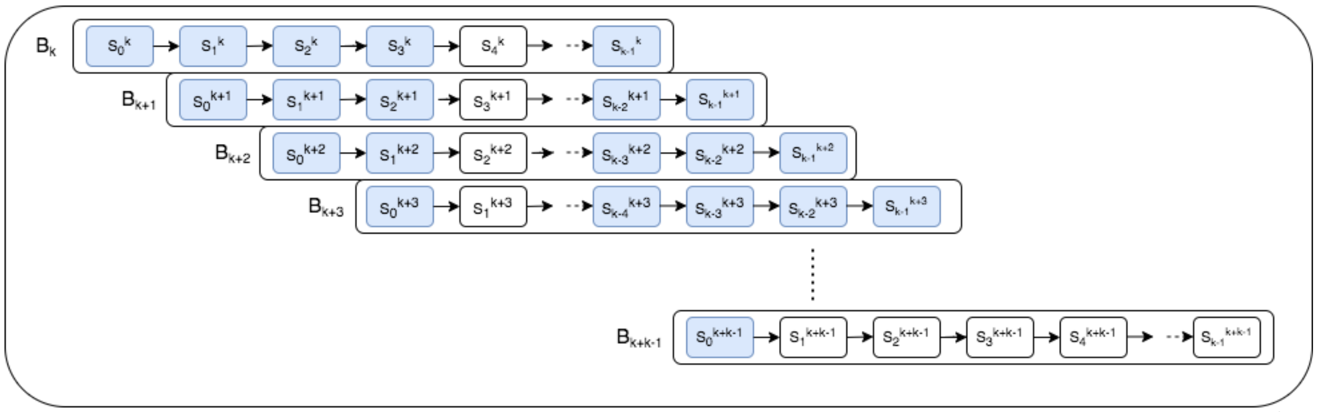
B_k
S_0^k S_1^k S_2^k S_3^k S_4^k S_{k-1}^k
B_{k+1}
S_0^{k+1} S_1^{k+1} S_2^{k+1} S_3^{k+1} S_{k-2}^{k+1} S_{k-1}^{k+1}
B_{k+2}
S_0^{k+2} S_1^{k+2} S_2^{k+2} S_{k-3}^{k+2} S_{k-2}^{k+2} S_{k-1}^{k+2}
B_{k+3}
S_0^{k+3} S_1^{k+3} S_{k-4}^{k+3} S_{k-3}^{k+3} S_{k-2}^{k+3} S_{k-1}^{k+3}
B_{k+k-1}
S_0^{k+k-1} S_1^{k+k-1} S_2^{k+k-1} S_3^{k+k-1} S_4^{k+k-1} S_{k-1}^{k+k-1}

Fig. 1: The pipeline-like mining architecture with $i = 0$. The x axis denotes the time. The Proof-of-Works of at maximum k blocks can be active in the same moment. Following the protocol, the expected time to complete each stage of every block is the same.

3.2 Issues

In this Subsection, we list some issues we found on the proposed protocol.

1. If $k \geq 2$ is the number of stages, then the mining probability of a miner and the ratio of the global hash rate the miner holds are not necessarily equal, as shown in Subsection 4.6.
2. Some *hardware incompatible* hash functions may be present in multiple stages. In this case, the advantage the miner has gained by buying an ASIC for those hash functions may be worthwhile in many stages.
3. A pipeline-like mining architecture cannot be easily constructed, since stage mining is a stochastic event. As a consequence, the pipeline-like block mining architecture can hardly ever be perfectly synchronized among different stages. Considering a single pipeline, it is possible to prove

that the probability that stage s of block B_{i+k} is completed exactly at the same time of completion of stage $s-1$ of block B_{i+k+1} is negligible, for each $i \geq 0$ and for each $s \in \{1, \dots, k-1\}$.

For completeness, other possible security issues regarding this protocol have been proposed in [9, 8].

4 Mining probability

In this Section, we propose a closed-form expression for the mining probability. The expression found is valid in every blockchain protocol whose Proof-of-Work is composed of $k \geq 1$ sequential hash-puzzles, under the assumptions of constant hash rates and hash-puzzles difficulties, and by letting every miner start to mine a block at the same time. Each miner can start the stage $s+1$ of his Proof-of-Work only after he has found a valid nonce in stage s, for all $s \in \{0, \dots k-2\}$.

As described in Subsection 2.1, although difficult to be satisfied in practice, these assumptions are widely used in the Bitcoin and blockchain literature [7, 13, 21]. It is worth remarking that these results may be applied to Proof-of-Work based blockchain protocol proposed in the future. Moreover, as will be stated in Subsection 4.5, we finally prove that, when $k = 1$, these results are equivalent to those valid in existing Proof-of-Work based blockchain protocols, such as Bitcoin or Ethereum, which have been proved in Subsection 2.1.

As announced in Subsection 3.1, in the protocol proposed in [23], the expression is applicable to the Proof-of-Work of the first general block B_k under the aforementioned assumptions. Indeed, all the genesis blocks have already been mined before the Proof-of-Work of block B_k had started. Consequently, as soon as a miner completes stage s of block B_k's Proof-of-Work, he can immediately start stage $s+1$ of the same Proof-of-Work, for each $s \in \{0, \dots, k-2\}$. Conversely, given a block B_{i+k}, such that $i \geq 1$ and $s \in \{0, \dots, k-2\}$, when a miner completes stage s of block B_{i+k}, he may need to wait until someone in the network mines and broadcasts block B_{i+s+1}, and $bdigest_{i+s+1}$ is available. Only at this point, he can start stage $s+1$ of block B_{i+k}. For this reason, we will restrict our analysis to block B_k.

4.1 Notation

We use the following notation:

- the integer $k \geq 1$ denotes the number of sequential hash-puzzles a Proof-of-Work is composed of.
- the integer $M \geq 2$ denoted the number of competing miners involved in the Proof-of-Work of a block.
- $d_0, \dots, d_{k-1}$ are the positive real values representing hash-puzzles difficulties.
- $h_{p,s}$ is a positive real value, denoting the hash rate of miner p on stage s, for all $p \in \{1, \dots, M\}$ and for all $s \in \{0, \dots, k-1\}$.
- $X_{p,s}$ is the exponential distribution describing the time miner p takes to complete the stage s of a block, for all $p \in \{1, \dots, M\}$, and for all $s \in \{0, \dots, k-1\}$.
- $\lambda_{p,s} = h_{p,s}/d_s$ is the positive real parameter of $X_{p,s}$, for all $p \in \{1, \dots, M\}$, and for all $s \in \{0, \dots, k-1\}$.

Let $p \in \{1, \dots, M\}$. The time miner p takes to complete the entire Proof-of-Work of a block is given by the *hypoexponential distribution* $X_{p,+_k} = \sum_{s=0}^{k-1} X_{p,s}$. The distribution $X_{p,+_k}$ is simply the sum of the time miner p takes to complete every stage of his Proof-of-Work. We can represent the time to complete a block's Proof-of-Work this way, since every single stage can be described by its own independent Poisson point process. The Poisson point process related to the first stage starts at the same time the Proof-of-Work of the block does. Completing a stage is equal to finding the first point into its Poisson process. As soon as the first point in a stage's Poisson process is found, the stage's process is not relevant anymore, and the next stage (i.e. the next Poisson process) starts. Therefore, the time to complete every single stage is described by an exponential distribution (as in Bitcoin), while the time to complete the Proof-of-Work is the sum of the individual stages' exponential distributions.

4.2 The hypoexponential distribution

The *probability density function* and the *survival function* of the hypoexponential distribution have been presented by Scheuer [24], and, as *closed-form* expressions, by Amari and Misra [4].

Results on the hypoexponential distribution. As explained in [24], the literature studies on the hypoexponential distribution have been divided into three major subcases, as follows:

1. In the first subcase, $\lambda_{p,s} = \lambda_{p,s'}$, for all $s' \neq s$, and $s, s' \in \{0, \ldots, k-1\}$. The hypoexponential distribution is reduced to an *Erlang distribution* with *shape* parameter k and *rate* parameter λ_p, with $\lambda_p = \lambda_{p,0} = \ldots = \lambda_{p,k-1}$.
2. In the second subcase, $\lambda_{p,s} \neq \lambda_{p,s'}$, for all $s' \neq s$, and $s, s' \in \{0, \ldots, k-1\}$.
3. In the third subcase, $\lambda_{p,s}$ can be either equal or not equal to $\lambda_{p,s'}$, for all $s' \neq s$, and $s, s' \in \{0, \ldots, k-1\}$.

The first two cases have been deeply studied in mathematics over the years, and we will not focus on them. Scheuer, Amari, and Misra have been the first researchers focusing on the third, and most general, scenario. It is worth remarking that the hypoexponential distribution is not *memoryless*, as opposed to the exponential distribution.

4.3 Probability density function and survival function

In this Subsection we recall the Amari and Misra's *closed-form* expressions for the *probability density function* and the *survival function* of a hypoexponential distribution [4], omitting the majority of the low-level technical details that have led to the composition of the two expressions, which are not necessary for our goals. We refer to the original papers for the complete mathematical analysis of the two expressions.

Let $p \in \{1, \ldots, M\}$. Let $f_{X_{p,+_k}}(t) = P(X_{p,+_k} = t)$ be the *probability density function*, and let $R_{X_{p,+_k}}(t) = P(X_{p,+_k} > t)$ be the *survival function* of $X_{p,+_k}$. In order to compute $f_{X_{p,+_k}}(t)$ and $R_{X_{p,+_k}}(t)$, the exponential random variables $X_{p,s}$ and $X_{p,s'}$ (such that $s' \neq s$, and $s, s' \in \{0, \ldots, k-1\}$) which satisfy the condition $\lambda_{p,s} = \lambda_{p,s'}$ must be grouped together. Their common λ value must be denoted with a new parameter. For example, if $k = 4$, $\lambda_{p,0} = \lambda_{p,3}$, and $\lambda_{p,1} = \lambda_{p,2}$, then two new parameters, $\beta_{p,1}$ and $\beta_{p,2}$, can be set as follows:

$$\begin{aligned} \lambda_{p,0} &= \lambda_{p,3} = \beta_{p,1} \\ \lambda_{p,1} &= \lambda_{p,2} = \beta_{p,2} \end{aligned}$$

At the same time, denote by $r_{p,1}$ and $r_{p,2}$ the number of λ-parameters having value equal to $\beta_{p,1}$ and $\beta_{p,2}$, respectively. At this point, if the number of different λ values is a_p, then a_p pairs are obtained:

$(\beta_{p,1}, r_{p,1}), \ldots, (\beta_{p,a_p}, r_{p,a_p})$. It holds that $\sum_{q_p=1}^{a_p} r_{p,q_p} = k$. The probability density function $f_{X_{p,+_k}}(t)$ is ([24], equation 13)

$$f_{X_{p,+_k}}(t) \overset{\text{def}}{=} B_p \sum_{q_p=1}^{a_p} \sum_{l_p=1}^{r_{p,q_p}} \frac{\Phi_{p,q_p,l_p}(-\beta_{p,q_p})}{(r_{p,q_p}-l_p)!\,(l_p-1)!}\, t^{r_{p,q_p}-l_p}\, e^{-\beta_{p,q_p} t} \tag{2}$$

where

$$\begin{cases} B_p = \left(\prod_{q_p=1}^{a_p} (\beta_{p,q_p})^{r_{p,q_p}}\right) \\ \Phi_{p,q_p,l_p}(t) = (-1)^{l_p-1} \cdot (l_p-1)! \cdot \sum_{\Omega_{2_p}(1)} \prod_{\substack{j_p=1 \\ j_p \neq q_p}}^{a_p} \binom{i_{j_p}+r_{p,j_p}-1}{i_{j_p}} \cdot \tau_{j_p} \quad \text{, (see [4], equation 4)} \\ \tau_{j_p} = (\beta_{p,j_p}+t)^{-(r_{p,j_p}+i_{j_p})} \quad \text{, (see [4], between equations 3 and 4)} \\ \Omega_{2_p}(1) = \sum_{\substack{j_p=1 \\ j_p \neq q_p}}^{a_p} i_{j_p} = l_p - 1 : i_{j_p} \in \mathbb{N}_0 \text{ for each } j_p \in \{1,\ldots,a_p\} \quad \text{, (see [4], \textit{Notation} Paragraph)} \end{cases}$$

The survival function is ([4], equation 3)

$$R_{X_{p,+_k}}(t) \overset{\text{def}}{=} B_p \sum_{q_p=1}^{a_p} \sum_{l_p=1}^{r_{p,q_p}} \frac{\Psi_{p,q_p,l_p}(-\beta_{p,q_p})}{(r_{p,q_p}-l_p)!\,(l_p-1)!}\, t^{r_{p,q_p}-l_p}\, e^{-\beta_{p,q_p} t} \tag{3}$$

where B_p is defined as above, and

$$\begin{cases} \Psi_{p,q_p,l_p}(t) = -(-1)^{l_p-1} \cdot (l_p-1)! \cdot \sum_{\Omega_{2_p}(0)} \prod_{\substack{j_p=0 \\ j_p \neq q_p}}^{a_p} \binom{i_{j_p}+r_{p,j_p}-1}{i_{j_p}} \cdot \tau_{j_p} \quad \text{, (see} \\ \qquad \text{[4], equation 5. See [4], \textit{Notation} Paragraph for the "-" sign)} \\ \tau_{j_p} \text{ as above} \\ \Omega_{2_p}(0) = \sum_{\substack{j_p=0 \\ j_p \neq q_p}}^{a_p} i_{j_p} = l_p - 1 : i_{j_p} \in \mathbb{N}_0 \text{ for each } j_p \in \{0,\ldots,a_p\} \quad \text{, (see [4], \textit{Notation} Paragraph)} \\ \beta_{p,0} = 0,\ r_{p,0} = 1 \text{ , (see [4], between equations 2 and 3)} \end{cases}$$

4.4 A *closed-form* expression for the mining probability

The expression is valid under the assumptions described at the beginning of this Section. Actually, given the *memoryless* property of the exponential distribution, the assumption on the common Proof-of-Work starting time can be slightly relaxed. Indeed, the expression of the mining probability is still valid even if we assume a scenario where the M miners start their respective Proof-of-Work of a block at different times. However, it is required that, at some time t^*, every miner has started his Proof-of-Work and is still currently working on the first hash-puzzle. In this case, the mining probability of a miner can be computed with our expression by considering time t^* as the common starting time of the Proof-of-Work of every miner. Similarly, the assumptions of constant hash rates and hash-puzzles difficulties can be slightly relaxed too. Indeed, the only variables involved in the computation

of the mining probability are the λ-parameters of the hypoexponential distributions. Hence, we can simply require that the λ-parameters of the hypoexponential distributions are constant throughout time. Recalling $\lambda_{p,s} = h_{p,s}/d_s$, for each $p \in \{1, \dots, M\}$, and for each $s \in \{0, \dots, k-1\}$, it follows that these assumptions can be replaced with slightly weaker assumptions of constant ratios $h_{p,s}/d_s$, for each $p \in \{1, \dots, M\}$, and for each $s \in \{0, \dots, k-1\}$. In Subsection 4.5 we show that, if $k = 1$, our expression is reduced to the mining probability of permissionless blockchain protocols whose Proof-of-Work is composed of one hash-puzzle. Therefore, the aforementioned relaxations can be applied to the mining probability expression of these blockchains too (discussed in Subsection 2.1).
Let $p \in \{1, \dots, M\}$ be a miner, and γ a non-negative integer. The probability miner p is the first one in completing the Proof-of-Work of block γ is

$$P\left(p \text{ mines block } \gamma\right) = \int_0^\infty f_{X_{p,+k}}(t) \prod_{\substack{z=1\\ z\neq p}}^{M} R_{X_{z,+k}}(t)\ dt \tag{4}$$

By substituting expressions (2) and (3), the integration above is equal to

$$\int_0^\infty \left(B_p \sum_{q_p=1}^{a_p} \sum_{l_p=1}^{r_{p,q_p}} \tfrac{\Phi_{p,q_p,l_p}\left(-\beta_{p,q_p}\right)}{(r_{p,q_p}-l_p)!\,(l_p-1)!}\ t^{r_{p,q_p}-l_p}\ e^{-\beta_{p,q_p} t} \right) \cdot \left(\prod_{\substack{z=1\\ z\neq p}}^{M} B_z \Big(\sum_{q_z=1}^{a_z} \sum_{l_z=1}^{r_{z,q_z}} \tfrac{\Psi_{z,q_z,l_z}\left(-\beta_{z,q_z}\right)}{(r_{z,q_z}-l_z)!\,(l_z-1)!}\ t^{r_{z,q_z}-l_z}\ e^{-\beta_{z,q_z} t} \Big) \right) dt$$

Due to the linearity of integration, the expression above is equal to

$$\Big(\prod_{z=1}^{M} B_z\Big) \int_0^\infty \left(\sum_{q_p=1}^{a_p} \sum_{l_p=1}^{r_{p,q_p}} \tfrac{\Phi_{p,q_p,l_p}\left(-\beta_{p,q_p}\right)}{(r_{p,q_p}-l_p)!\,(l_p-1)!}\ t^{r_{p,q_p}-l_p}\ e^{-\beta_{p,q_p} t} \right) \cdot \left(\prod_{\substack{z=1\\ z\neq p}}^{M} \Big(\sum_{q_z=1}^{a_z} \sum_{l_z=1}^{r_{z,q_z}} \tfrac{\Psi_{z,q_z,l_z}\left(-\beta_{z,q_z}\right)}{(r_{z,q_z}-l_z)!\,(l_z-1)!}\ t^{r_{z,q_z}-l_z}\ e^{-\beta_{z,q_z} t} \Big) \right) dt$$

More concisely, the expression above is

$$\Big(\prod_{z=1}^{M} B_z\Big) \int_0^\infty \left(\prod_{z=1}^{M} \Big(\sum_{q_z=1}^{a_z} \sum_{l_z=1}^{r_{z,q_z}} \tfrac{\Theta_{z,q_z,l_z}\left(-\beta_{z,q_z}\right)}{(r_{z,q_z}-l_z)!\,(l_z-1)!}\ t^{r_{z,q_z}-l_z}\ e^{-\beta_{z,q_z} t} \Big) \right) dt$$

where

$$\Theta_{z,q_z,l_z}\left(-\beta_{z,q_z}\right) = \left\{ \begin{array}{ll} \Phi_{z,q_z,l_z}\left(-\beta_{z,q_z}\right), & \text{iff } z = p \\ \Psi_{z,q_z,l_z}\left(-\beta_{z,q_z}\right), & \text{otherwise} \end{array} \right\}$$

Due to the distributive property of multiplication over addition, the expression above is equal to

$$\Big(\prod_{z=1}^{M} B_z\Big) \int_0^\infty \sum_{q_1=1}^{a_1} \sum_{l_1=1}^{r_{1,q_1}} \cdots \sum_{q_M=1}^{a_M} \sum_{l_M=1}^{r_{M,q_M}} \Big(\prod_{z=1}^{M} \tfrac{\Theta_{z,q_z,l_z}\left(-\beta_{z,q_z}\right)}{(r_{z,q_z}-l_z)!\,(l_z-1)!}\ t^{r_{z,q_z}-l_z}\ e^{-\beta_{z,q_z} t} \Big)\ dt$$

Due to the linearity of integration, the expression above is equal to

$$\Big(\prod_{z=1}^{M} B_z\Big) \sum_{q_1=1}^{a_1} \sum_{l_1=1}^{r_{1,q_1}} \cdots \sum_{q_M=1}^{a_M} \sum_{l_M=1}^{r_{M,q_M}} \Big(\int_0^\infty \Big(\prod_{z=1}^{M} \tfrac{\Theta_{z,q_z,l_z}\left(-\beta_{z,q_z}\right)}{(r_{z,q_z}-l_z)!\,(l_z-1)!}\ t^{r_{z,q_z}-l_z}\ e^{-\beta_{z,q_z} t} \Big)\ dt \Big)$$

and to

$$\Big(\prod_{z=1}^{M} B_z\Big) \cdot \sum_{q_1=1}^{a_1} \sum_{l_1=1}^{r_{1,q_1}} \cdots \sum_{q_M=1}^{a_M} \sum_{l_M=1}^{r_{M,q_M}} \Big(\Big(\prod_{z=1}^{M} \tfrac{\Theta_{z,q_z,l_z}\,(-\,\beta_{z,q_z})}{(r_{z,q_z}-l_z)!\,(l_z-1)!}\Big) \cdot \int_0^{\infty} t^{\sum_{z=1}^{M}(r_{z,q_z}-\,l_z)}\; e^{-\sum_{z=1}^{M}\beta_{z,q_z}\;t}\; dt\Big) \tag{5}$$

Euler's Gamma Function. According to the definition of the Euler's gamma function, if we let $t_2 = \eta t$, then for each $\alpha, \eta \in \mathbb{R}$, $\alpha, \eta \neq 0$, it holds that

$$\int_0^{\infty} t^{\alpha} \cdot e^{-\eta \cdot t} dt \;=\; \int_0^{\infty} \frac{1}{\eta^{\alpha}} {t_2}^{\alpha} \cdot e^{-t_2} \cdot \frac{1}{\eta} dt_2 \;=\; \frac{1}{\eta^{\alpha+1}} \int_0^{\infty} {t_2}^{\alpha} \cdot e^{-\cdot t_2} dt_2 \;=\; \frac{a!}{\eta^{\alpha+1}}$$

If we let $\alpha = \sum_{z=1}^{M}(r_{z,q_z} -\, l_z)$ and $\eta = \sum_{z=1}^{M} \beta_{z,q_z}$, it holds that

$$\int_0^{\infty} t^{\sum_{z=1}^{M}(r_{z,q_z}-\,l_z)}\; e^{-\sum_{z=1}^{M}\beta_{z,q_z}\;t}\; dt \;=\; \big(\textstyle\sum_{z=1}^{M}\beta_{z,q_z}\big)^{-1-\sum_{z=1}^{M}(r_{z,q_z}-\,l_z)} \;\cdot\; \Big(\textstyle\sum_{z=1}^{M}(r_{z,q_z}\,-\,l_z)\Big)!$$

Hence, expression (5) can be equivalently defined as

$$\Big(\prod_{z=1}^{M} B_z\Big) \cdot \sum_{q_1=1}^{a_1} \sum_{l_1=1}^{r_{1,q_1}} \cdots \sum_{q_M=1}^{a_M} \sum_{l_M=1}^{r_{M,q_M}} \Big(\Big(\prod_{z=1}^{M} \tfrac{\Theta_{z,q_z,l_z}\,(-\,\beta_{z,q_z})}{(r_{z,q_z}-l_z)!\,(l_z-1)!}\Big) \cdot \big(\textstyle\sum_{z=1}^{M}\beta_{z,q_z}\big)^{-1-\sum_{z=1}^{M}(r_{z,q_z}-\,l_z)} \cdot$$
$$\cdot \Big(\textstyle\sum_{z=1}^{M}(r_{z,q_z}\,-\,l_z)\Big)!\Big) =$$

$$= \Big(\prod_{z=1}^{M} B_z\Big) \cdot \sum_{q_1=1}^{a_1} \sum_{l_1=1}^{r_{1,q_1}} \cdots \sum_{q_M=1}^{a_M} \sum_{l_M=1}^{r_{M,q_M}} \left(\frac{\Big(\prod_{z=1}^{M} \frac{\Theta_{z,q_z,l_z}\,(-\,\beta_{z,q_z})}{(l_z-1)!}\Big) \cdot Multinomial_{(\sum_{z=1}^{M}(r_{z,q_z}-\,l_z));\,(r_{1,q_1}-l_1),\,\ldots\,,(r_{M,q_M}-l_M)}}{\big(\sum_{z=1}^{M}\beta_{z,q_z}\big)^{1+\sum_{z=1}^{M}(r_{z,q_z}-\,l_z)}}\right)$$

where

$$Multinomial_{(\sum_{z=1}^{M}(r_{z,q_z}-\,l_z));\,(r_{1,q_1}-l_1),\,\ldots\,,(r_{M,q_M}-l_M)} \;=\; \frac{\Big(\sum_{z=1}^{M}(r_{z,q_z}\,-\,l_z)\Big)!}{\prod_{z=1}^{M}\big((r_{z,q_z}\,-\,l_z)!\big)}$$

is a multinomial coefficient. Letting

$$\begin{cases} \Phi'_{z,q_z,l_z}\,(t) \;=\; \frac{\Phi_{z,q_z,l_z}\,(t)}{(l_z-1)!} \;=\; (-1)^{l_z-1} \;\cdot\; \sum_{\Omega_{2_z}(1)} \prod_{j_z} \binom{i_{j_z}+r_{z,j_z}-1}{i_{j_z}} \;\cdot\; \tau_{j_z} \\ \Psi'_{z,q_z,l_z}\,(t) \;=\; \frac{\Psi_{z,q_z,l_z}\,(t)}{(l_z-1)!} \;=\; -\,(-1)^{l_z-1} \;\cdot\; \sum_{\Omega_{2_z}(0)} \prod_{j_z} \binom{i_{j_z}+r_{z,j_z}-1}{i_{j_z}} \;\cdot\; \tau_{j_z} \\ \Theta'_{z,q_z,l_z}\,(-\,\beta_{z,q_z}) = \begin{Bmatrix} \Phi'_{z,q_z,l_z}\,(-\,\beta_{z,q_z}), & \text{iff } z = p \\ \Psi'_{z,q_z,l_z}\,(-\,\beta_{z,q_z}), & \text{otherwise} \end{Bmatrix} \end{cases}$$

the mining probability of miner p is

$$P\,(p \text{ mines block } \gamma) \;=\; \Big(\prod_{z=1}^{M} B_z\Big) \cdot \sum_{q_1=1}^{a_1} \sum_{l_1=1}^{r_{1,q_1}} \cdots \sum_{q_M=1}^{a_M} \sum_{l_M=1}^{r_{M,q_M}}$$
$$\left(\frac{\Big(\prod_{z=1}^{M} \Theta'_{z,q_z,l_z}\,(-\,\beta_{z,q_z})\Big) \cdot Multinomial_{(\sum_{z=1}^{M}(r_{z,q_z}-\,l_z));\,(r_{1,q_1}-l_1),\,\ldots\,,(r_{M,q_M}-l_M)}}{\big(\sum_{z=1}^{M}\beta_{z,q_z}\big)^{1+\sum_{z=1}^{M}(r_{z,q_z}-\,l_z)}}\right) \tag{6}$$

To the best of the authors' knowledge expression (6) is no further simplifiable.

4.5 The *closed-form* expression for the mining probability with $k = 1$

If $k = 1$, expression (6) is reduced to the mining probability of permissionless blockchain protocols whose Proof-of-Work is composed of one hash-puzzle.

Proof. Look at the *closed-form* expression of the mining probability (6). Let $p \in \{1, \dots, M\}$. If $k = 1$, the time miner p takes to complete the Proof-of-Work follows the hypoexponential distribution $X_{p,+1}$ with parameter $\lambda_{p,0} = h_{p,0}/d_0$, where $h_{p,0} > 0$ denotes the hash rate of miner p, and d_0 is the value indicating the difficulty of the hash-puzzle. After grouping operations, it remains a single pair $(\beta_{p,1}, 1)$, with $\beta_{p,1} = \lambda_{p,0}$. Hence, the number of distinct β-values is $a_p = 1$, and the number of occurrences of this single β-value is $r_{p,1} = 1$, for each $p \in \{1, \dots, M\}$.

If $k = 1$, expression (6) consists of a single addend

$$P\left(p \text{ mines block } \gamma\right) = \left(\prod_{z=1}^{M} B_z\right) \cdot \sum_{q_1=1}^{1} \sum_{l_1=1}^{1} \dots \sum_{q_M=1}^{1} \sum_{l_M=1}^{1}$$

$$\left(\frac{\Phi'_{p,q_p,l_p}\left(-\beta_{p,q_p}\right) \cdot \left(\prod_{\substack{z=1 \\ z \neq p}}^{M} \Psi'_{z,q_z,l_z}\left(-\beta_{z,q_z}\right) \right) \cdot \mathit{Multinomial}_{\left(\sum_{z=1}^{M}(r_{z,q_z} - l_z)\right);\, (r_{1,q_1} - l_1),\, \dots\, ,(r_{M,q_M} - l_M)}}{\left(\sum_{z=1}^{M} \beta_{z,q_z}\right)^{1+\sum_{z=1}^{M}(r_{z,q_z} - l_z)}} \right) \tag{7}$$

It holds that

$$\begin{cases} \prod_{z=1}^{M} B_z = \prod_{z=1}^{M} \lambda_{z,0} \quad \text{since } B_z = \lambda_{z,0} \text{ for each } z \in \{1, \dots, M\} \\ \\ \mathit{Multinomial}_{\left(\sum_{z=1}^{M}(r_{z,q_z} - l_z)\right);\, (r_{1,q_1} - l_1),\, \dots\, ,(r_{M,q_M} - l_M)} = 1 \quad \text{since } r_{z,q_z} = l_z = 1 \text{ for each } z \in \{1, \dots, M\} \\ \\ \left(\sum_{z=1}^{M} \beta_{z,q_z}\right)^{1+\sum_{z=1}^{M}(r_{z,q_z} - l_z)} = \sum_{z=1}^{M} \lambda_{z,0} \quad \text{since } r_{z,q_z} = l_z,\ q_z = 1,\ \beta_z = \lambda_{z,0} \text{ for each } z \in \{1, \dots, M\} \end{cases}$$

and

$\Phi'_{p,q_p,l_p}(-\beta_{p,q_p}) = 1.$

Proof. $\Phi'_{p,q_p,l_p}(-\beta_{p,q_p}) = \Phi'_{p,1,1}(-\lambda_{p,0})$ since $q_p = 1$, $l_p = 1$, and $\beta_{p,q_p} = \beta_{p,1} = \lambda_{p,0}$ as defined in Subsection 4.3. Following the definition of Φ', it holds that $\Phi'_{p,1,1}(-\lambda_{p,0}) = \sum_{\Omega_{2_p}(1)} \prod_{\substack{j_p=1 \\ j_p \neq q_p}}^{a_p} \binom{i_{j_p}+r_{p,j_p}-1}{i_{j_p}} \cdot \tau_{j_p}$ since $l_p = 1$.

Focusing on the summation, in particular on the definition of $\Omega_{2_p}(1)$ provided in Subsection 4.3, we have the constraint $\Omega_{2_p}(1) = \sum_{\substack{j_p=1 \\ j_p \neq 1}}^{1} i_{j_p} = 0 : i_{j_p} \in \mathbb{N}_0$ for each $j_p \in \{1\}$, since $q_p = 1$, $a_p = 1$, and $l_p = 1$. It follows that i_1 can take any possible non-negative integer γ to satisfy the constraint. Once the value is arbitrarily assigned, we can focus on the product $\prod_{\substack{j_p=1 \\ j_p \neq q_p}}^{a_p} \binom{i_{j_p}+r_{p,j_p}-1}{i_{j_p}} \cdot \tau_{j_p}$.

Since $q_p = a_p = 1$, and $j_p \neq q_p$ by definition, we have an *empty product.* Therefore $\Phi'_{p,1,1}(-\lambda_{p,0}) = \sum_{i_1=\gamma} 1 = 1$, where γ is a arbitrary and constant non-negative integer.

$\Psi'_{z,q_z,l_z}(-\beta_{z,q_z}) = \frac{1}{\lambda_{z,0}}$ for each $z \in \{1, \dots, M\}, z \neq p$.

Proof. Let $z \in \{1, \dots, M\}, z \neq p$. We have $\Phi'_{z,q_z,l_z}(-\beta_{z,q_z}) = \Phi'_{z,1,1}(-\lambda_{z,0})$ since $q_z = 1$, $l_z = 1$, and $\beta_{z,q_z} = \beta_{z,1} = \lambda_{z,0}$ as defined in Subsection 4.3. Following the definition of Ψ', it holds that $\Psi'_{z,1,1}(-\lambda_{z,0}) = -\sum_{\Omega_{2_z}(0)} \prod_{\substack{j_z=0 \\ j_z \neq q_z}}^{a_z} \binom{i_{j_z}+r_{z,j_z}-1}{i_{j_z}} \cdot \tau_{j_z}$ since $l_z = 1$. Focusing on the summation, in particular on the definition of $\Omega_{2_z}(0)$ provided in Subsection 4.3, we have the constraint $\Omega_{2_z}(0) = \sum_{\substack{j_z=0 \\ j_z \neq 1}}^{1} i_{j_z} = 0 : i_{j_z} \in \mathbb{N}_0$ for each $j_z \in \{0, 1\}$, since $q_z = 1$, $a_z = 1$, and $l_z = 1$. It follows that $i_0 = 0$, while i_1 can take any possible non-negative integer γ to satisfy the constraint. Once the value is arbitrarily assigned, only the product $\prod_{\substack{j_z=0 \\ j_z \neq q_z}}^{a_z} \binom{i_{j_z}+r_{z,j_z}-1}{i_{j_z}} \cdot \tau_{j_z}$ is left.

Putting it all together, we recall $\beta_{z,0} = 0$, $r_{z,0} = 1$, as defined in Subsection 4.3, $q_z = 1$, $a_z = 1$, and $l_z = 1$, and we recall that $\Omega_{2_z}(0)$ can be substituted by $i_0 = 0$ and $i_1 = \gamma$, with $\gamma \in \mathbb{N}_0$. It holds that $\Psi'_{z,1,1}(-\lambda_{z,0}) = -\sum_{\Omega_{2_z}(0)} \prod_{\substack{j_z=0 \\ j_z \neq q_z}}^{a_z} \binom{i_{j_z}+r_{z,j_z}-1}{i_{j_z}} \cdot \tau_{j_z} =$
$= -\sum_{\substack{i_0=0 \\ i_1=\gamma}} \prod_{\substack{j_z=0 \\ j_z \neq 1}}^{1} \binom{i_{j_z}+r_{z,j_z}-1}{i_{j_z}} \cdot \tau_{j_z} = -\sum_{\substack{i_0=0 \\ i_1=\gamma}} \prod_{j_z=0} \binom{i_{j_z}+r_{z,j_z}-1}{i_{j_z}} \cdot \tau_{j_z} =$
$= -\sum_{\substack{i_0=0 \\ i_1=\gamma}} \binom{i_0+r_{z,0}-1}{i_0} \cdot \tau_0 = -\binom{r_{z,0}-1}{0} \cdot \tau_0$. We recall $r_{z,0} = 1$. Hence, what is left is $-\tau_0$.

According to its definition, $\tau_0 = (\beta_{z,0} + t)^{-(r_{z,0}+i_0)} = (-\lambda_{z,0})^{-1}$, since $\beta_{z,0} = 0$, $r_{z,0} = 1$, $i_0 = 0$, while the input of function Ψ, t, is t $= \lambda_{z,0}$. Therefore $\Psi'_{z,1,1}(-\lambda_{z,0}) = -\tau_0 = \frac{1}{\lambda_{z,0}}$.

Putting it all together, what is left in expression (7) is

$$\left(\prod_{z=1}^{M} \lambda_{z,0}\right) \cdot \frac{\prod_{\substack{z=1 \\ z \neq p}}^{M} \frac{1}{\lambda_{z,0}}}{\left(\sum_{z=1}^{M} \lambda_{z,0}\right)} = \frac{\lambda_{p,0}}{\sum_{z=1}^{M} \lambda_{z,0}} = \frac{h_{p,0}}{\sum_{z=1}^{M} h_{z,0}}$$

Q.E.D.

4.6 Ratio of the global hash rate a miner holds and his mining probability

The mining probability in a multi-stage Proof-of-Work blockchain protocol can be practically computed with several tools, such as Matlab or Wolfram Mathematica. We used the *hypoexponential* *distribution* library of Wolfram Mathematica, in order to compute the mining probability through expression (4). Alternatively, we implemented a Mathematica Library, which computes the mining probability directly through expression (6). Benchmarking results prove that, at least in the case in which $M \leq 5$ and $k \leq 5$, our implementation of expression (6) can be faster than the built-in Mathematica library in computing the mining probability value on a standard personal computer[5]. Therefore, one practical application of expression (6) may be a time-efficient computation of the mining probability. The authors believe that more advanced and peculiar implementations of expression (6) may improve the time performances even more[6].

In the following, we describe the relationship between the ratio of the global hash rate a miner holds and his mining probability. In particular, we prove that, if $k \geq 2$, then the ratio of the global hash rate in possession of a miner and his mining probability are *not necessarily* equal.

Example 1. Let $M = 2$, $k = 2$, and $d_0 = 2^8$, $d_1 = 2^{12}$. If $h_{1,0} = 1053.3420821484203$ *hash/s*, $h_{1,1} = 3350.877902092879$ *hash/s*, $h_{2,0} = 388.6077318015238$ *hash/s*, $h_{2,1} = 6217.723708824381$ *hash/s*, then the first miner possesses the 39.99% of the global hash rate[7] and obtains a mining probability of 0.49100464.

Such a case can occur in the Proof-of-Work of the first *general block*, B_k, in the mining architecture proposed in [23]. Indeed, suppose that the second miner is initially the only miner in the blockchain network, and he has just mined the *k genesis blocks* to bootstrap the blockchain protocol. Let the hash functions used in the two stages be *hardware incompatible*. Following the protocol, the difficulties of the hash-puzzles are set and updated at repeated intervals, to let the expected time to complete the two stages be the same. It means that $\mathbb{E}[X_{2,0}] \overset{\text{def}}{=} 1/\lambda_{2,0}$ is equal to $\mathbb{E}[X_{2,1}] \overset{\text{def}}{=} 1/\lambda_{2,1}$. This constraint is satisfied in the example. Right before the second miner starts to mine block B_k, the first miner joins the mining game. Similar cases to the one described in *Example 1* might advantage a clever miner, who may optimally divide his hash rate among the different hash-puzzles, and obtain a mining probability value higher than the ratio of the global hash rate he holds. As a consequence, multi-stage Proof-of-Work blockchain protocols may be more vulnerable than single-stage counterparts to attacks such as the 51% *attack* [19], Indeed, if a "*low hash-rated*" miner is malicious, and his mining probability is greater than 0.50, then the miner would easily become a 51% *attacker*, and he would "*control*" the blockchain protocol [19].

5 Conclusion

In this paper we obtained a *closed-form* expression for the *mining probability*, which is valid, under some assumptions, in permissionless blockchain protocol whose Proof-of-Work is composed of $k \geq 1$ sequential hash-puzzles. Our theoretical results can be applied to existing Proof-of-Work based blockchain protocols, such as Bitcoin or Ethereum. Eventually, we proved that, if $k \geq 2$, then the ratio of the global hash rate held by a miner and his mining probability are *not necessarily* equal. This might advantage a clever miner, who may optimally divide his hash rate among the different hash-puzzles, and obtain a mining probability value higher than the ratio of the global hash rate he holds. Such a possibility opens up critical security issues. Multi-stage Proof-of-Work blockchain protocols deserve further and careful investigations.

[5] The benchmarking was performed on a Notebook HP Pavilion Laptop 15-cs2023nl, equipped with a quad-core CPU Intel Core$^{\text{TM}}$ i7-8565U CPU running at 1.80GHz, 8MB cache, and 16GB (2x8) SO-DIMM SDRAM DDR4 running at 2400MHz.

[6] Our source code, a comprehensive documentation, and the information regarding the benchmark tests done are available on GitHub: https://github.com/FraMog/MiningProbabilityMultiStageProof-of-Work.

[7] The ratio of the global hash rate the first miner possesses is $(h_{1,0} + h_{1,1})/(h_{1,0} + h_{1,1} + h_{2,0} + h_{2,1})$.

References

[1] URL: https://en.bitcoin.it/wiki/Mining#Reward (visited on 01/15/2021).
[2] URL: https://en.bitcoinwiki.org/wiki/Hashrate (visited on 01/15/2021).
[3] URL: https://www.blockchain.com/en/charts/hash-rate (visited on 01/15/2021).
[4] S. V. Amari and R. B. Misra. "Closed-Form Expressions for Distribution of Sum of Exponential Random Variables". In: *IEEE TRANSACTIONS ON RELIABILITY* 46.4 (1997).
[5] I. Bentov, A. Gabizon, and A. Mizrahi. "Cryptocurrencies Without Proof of Work". In: *Financial Cryptography and Data Security*. Springer Berlin Heidelberg, 2016, pp. 142–157.
[6] A. Birnbaum. "Statistical Methods for Poisson Processes and Exponential Populations". In: *Journal of the American Statistical Association* 49.266 (1954), pp. 254–266.
[7] R. Bowden, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor. *Block arrivals in the Bitcoin blockchain*. 2018. arXiv: 1801.07447 [cs.CR].
[8] P. D'Arco and Z. E. Ansaroudi. "Security attacks on Multi-stage Proof-of-Work blockchain". In: *Security, Privacy and Trust in the Internet of Things (SPT-IoT 2021)*. Forthcoming.
[9] C. Donghoon, H. Munawar, and J. Pranav. *Spy Based Analysis of Selfish Mining Attack on Multi-Stage Blockchain*. Cryptology ePrint Archive, Report 2019/1327. https://eprint.iacr.org/2019/1327. Sept. 2019.
[10] J. Garay, A. Kiayias, and N. Leonardos. "The Bitcoin Backbone Protocol: Analysis and Applications". In: *Advances in Cryptology - EUROCRYPT 2015*. Ed. by E. Oswald and M. Fischlin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 281–310.
[11] N. Houy. "The Bitcoin Mining Game". In: *Ledger* 1 (Dec. 2016), pp. 53–68.
[12] G. O. Karame, E. Androulaki, M. Roeschlin, A. Gervais, and S. Čapkun. "Misbehavior in Bitcoin: A Study of Double-Spending and Accountability". In: *ACM Trans. Inf. Syst. Secur.* 18.1 (May 2015).
[13] D. Kraft. "Difficulty control for blockchain-based consensus systems". In: *Peer-to-Peer Networking and Applications* 9 (Apr. 2015).
[14] Y. Lewenberg, Y. Bachrach, Y. Sompolinsky, A. Zohar, and J. S. Rosenschein. *Bitcoin Mining Pools: A Cooperative Game Theoretic Analysis*. Istanbul, Turkey, 2015.
[15] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena. "A Secure Sharding Protocol For Open Blockchains". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS '16. Vienna, Austria: Association for Computing Machinery, 2016, pp. 17–30.
[16] N. A. Lynch. *Distributed Algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
[17] D. Malone and K. J. O'Dwyer. "Bitcoin Mining and its Energy Footprint". In: Jan. 2014, pp. 280–285.
[18] S. Nakamoto. *Bitcoin: A peer-to-peer electronic cash system"*. 2008. URL: http://bitcoin.org/bitcoin.pdf.
[19] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. 1st. Priceton, New Jersey, USA: Princeton University Press, 2016.
[20] R. Pass, L. Seeman, and A. Shelat. "Analysis of the Blockchain Protocol in Asynchronous Networks". In: *Advances in Cryptology – EUROCRYPT 2017*. Springer International Publishing, 2017, pp. 643–673.
[21] M. Rosenfeld. *Analysis of Bitcoin Pooled Mining Reward Systems*. 2011. arXiv: 1112.4980 [cs.DC].
[22] M. Rosenfeld. *Analysis of Hashrate-Based Double Spending*. 2014. arXiv: 1402.2009 [cs.CR].

[23] P. Sarkar. *Multi-Stage Proof-of-Work Blockchain*. Cryptology ePrint Archive, Report 2019/162. https://eprint.iacr.org/2019/162. July 2019.

[24] E. M. Scheur. "Reliability of an m-out of-n system when component failure induces higher failure rates in survivors". In: *IEEE TRANSACTIONS ON RELIABILITY* 37.1 (1988).

[25] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen, and D. I. Kim. "A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks". In: *IEEE Access* 7 (2019), pp. 22328–22370.

[26] Y. Zhang, H. Chen, and M. Guizani. *Cooperative wireless communications*. CRC press, 2009.