Cryptocurrency Mining through Stochastic Lenses

ABSTRACT

In the consensus protocol used in most cryptocurrencies, participants are expected to perform actions for which they receive a reward. These actions are colloquially known as mining, and protocols must ensure that mining is reward-compatible: participants are expected to behave correctly if they want to maximise their gains. Yet studying incentives behind mining actions in cryptocurrencies has proven a difficult task, leaving us without a clear understanding of optimal miner behaviour.

Our goal is to understand how do optimal mining strategies look like, and particularly under which circumstances are miners incentivised to fork the blockchain by mining on top of blocks which are not at the tail of the current chain. We model mining in cryptocurrencies as a stochastic game in which players always try to produce new blocks, and are rewarded for it. The probability of a player producing a new block depends on the hash power of the player, that is, the proportion of the computational power he controls compared to the power of the entire network. Players then look to maximise their utility in the long run, and their best strategy depends on their hash power and the discount applied to the reward of each new block being mined. We show that if no discount is offered, then miners do not have any incentive to fork, no matter how high their hash power is. On the other hand, when working with discounts similar to those in Bitcoin, miners do have an incentive to fork; in fact, it is profitable to do so even when they own less than half of the network's hash power.

1 INTRODUCTION

The Bitcoin Protocol [18–20], or Nakamoto Protocol, introduces a novel decentralized network-consensus mechanism that is trustless and open for anyone connected to the Internet. To support such an open and dynamic topology, the protocol requires an underlying currency (a so-called *cryptocurrency* [20]) to encourage/discourage participants to/from taking certain actions. The largest network running this protocol at the time of writing is the Bitcoin network, and its underlying cryptocurrency is Bitcoin (BTC). Following the success of Bitcoin, several other cryptocurrencies have been created. Some of them are simple replicas of Bitcoin with slight modifications of the protocol parameters (e.g. Litecoin [32] or Bitcoin Cash [30]), while some of them introduce interesting new modifications on top of the protocol to provide further functionalities (e.g. Ethereum [26, 31] or Monero [33]).

Due to the success and popularity of cryptocurrencies, there has been a growing body of research on the topic [1, 4, 7, 9, 10, 12–16, 21, 22, 25, 34], giving us better clarity of their underlying protocols. However, due to the fact that the multitude of actors and incentives involved in cryptocurrencies make them rather hard to formalize and study rigorously, we are still lacking the complete understanding of how they work. Therefore, our goal in this paper is to construct a realistic model of a cryptocurrency protocol, and study its dynamics. In doing so, we focus on the action of cryptocurrency *mining*, as this will allow us to measure the impact of incentives on the actors participating in the protocol. We model cryptocurrency mining as an infinite stochastic game, and we study what are the best mining strategies based on different choices for mining rewards. The model itself tries to stay as close as possible to reality, and consider fundamental properties such as the deflationary character of mining rewards in cryptocurrencies, and the economical discounts that different strategies incur for monetary gains that happen far in the future. To provide context for our contributions, we first briefly explain how the Bitcoin protocol works.

The Bitcoin protocol. The objective of the Bitcoin protocol is to generate consensus on a data structure that is replicated amongst all nodes in a trustless and decentralized peer-to-peer network. The data structure used in the protocol is an append-only record of transactions. New transactions are spread across the network using peer-to-peer communication until one node is *allowed* (we will explain this in detail later) to assemble a group of transactions into a new *block*, and present this block as a candidate to extend the data structure. Apart from the set of transactions, the block will contain a pointer to some previous block. The newly formed block is then spread throughout the network, and the whole process is repeated. Since every block points to a previous block, a tree of blocks is naturally formed. The consensus data structure is generally defined as the longest branch of such a tree, also known as the *blockchain*.

To provide an incentive for the nodes in the network to keep generating new blocks (and to check that transactions they include in a block are correct according to the protocol rules), the protocol requires an underlying currency. This currency is then used to reward generation of new blocks: whenever a participant forms a new block, she receives a certain amount of currency. This is known as the block reward, and is the way in which new currency is created. In Bitcoin, this amount was originally 50BTC and halves approximately every four years (the current reward is 12.5BTC). The currency generated in a block is considered valid only if the block belongs to the blockchain. This is important, because block rewards may cease to be valid currency if the block is no longer part of the blockchain, even if the block was part of it when it was created. Therefore, whenever a node forms a new block, it is encouraged to place this block in a part of the tree with a high probability of becoming part of the blockchain. Actually, the protocol states that new blocks should always be appended on top of a block with maximal distance to the root of the tree, although participants are not obliged to follow this rule.

Since blocks give a reward, nodes will naturally want to generate blocks[1]. If we expect the currency to have any value, generating new blocks must then be hard. Under the proof-of-work framework, participants generating new blocks are required to solve some computationally hard problem per each new block. In Bitcoin, a block is *valid* in the protocol if its hash value, when interpreted

[1]In order to encourage including all the transactions that are correct according to the cryptocurrency rules, the miners also receive a small *transaction fee* for each transaction they include in a block. As in currently used cryptocurrencies fees rarely exceed 10% of the mining reward [28, 29], we focus on block rewards when gauging the miner's economic motivation for participating in the protocol.

as a number, is less than a certain threshold. Since hash functions are unpredictable, the only way to generate a valid block is to try with several different blocks, until one of them has a hash value below the established threshold. This is known as *mining*, and the number of (valid and invalid) blocks per second that a miner can hash is referred to as her *hash power*. Nodes who participate in the generation of blocks are called *miners*.

Mining rewards introduce a competition for generating new blocks, and to ensure that one's blocks form part of the blockchain. This can be naturally viewed as a non-cooperative game in which the miners take certain actions in order to maximize their benefit, thus defining different strategies that describe their behaviour. A natural question that a miner would ask in this scenario is what is her optimal strategy, given that she controls some fixed amount of hash power. On the other hand, the protocol designers might wonder how to set the parameters of the protocol in such a way that the default behaviour defines a Nash equilibrium (i.e. a set of strategies where no miner has an incentive to perform a different action). Studying these types of questions is the main objective of this paper.

Contributions. The first contribution of our work is a realistic model of cryptocurrency mining. We model mining as an infinite stochastic game in which miners are expected to maximize their long-term utility. One of the benefits of our approach is that using a few basic design parameters we can in fact accommodate different cryptocurrencies, and not focus solely on e.g. Bitcoin. These parameters also allow us to account for fundamental factors such as deflation, or discount in the block reward. Another benefit of our proposal is the fact that we can reason about all possible mining strategies, without the need to focus on a specific subclass.

The second contribution of our work is a set of results about optimal behaviour of miners in different cryptocurrencies. In particular, we study mining under the assumption that block rewards are constant (as it will eventually be in cryptocurrencies with tail-emission such as Monero or Ethereum), or when they decrease over time (e.g. Bitcoin). In the first scenario, we show that the default scenario of always mining on the latest block of the blockchain is indeed a Nash equilibrium. However, this is not the case for the second scenario, and in fact we prove that strategies that involve mining on a block that is not at the tail of the blockchain (known as *forking*) can indeed give higher utility in some cases. Thus we study what is the best strategy for miners when assuming everyone else is playing the default strategy. The choice of strategy depends on the hash power, the rate at which block rewards decrease over time, and the usual financial discount rate. We show that our model confirms the commonly held belief that players should start deviating from the default strategy when they approach 50% of the network's hash power (also known as 51% attack). However, in this case the strategy may not be as simple as "always try to fork", but rather be a combination of appropriately choosing when to fork, and when to keep on extending the current blockchain.

Related Work. There are a number of studies that approach mining from a game-theoretical point of view [3, 4, 7, 12, 13, 22]. The main difference with our work is in the choice of the reward function [3, 4, 7, 13, 22] these papers use, or restricting the space of viable mining strategies [7, 13, 22]. Perhaps the study which is closest in spirit to ours is the one by Kiayias et al. in [12]. Here the authors also focus on the hash power thresholds for which the default strategy is not an optimal strategy any more. Albeit the authors consider a range of strategies, their model is still somewhat more restrictive than ours. Overall, the underlying characteristic of our model is that the block emission cadence is discrete and not continuous, that the reward function can be fine tuned depending on the discounts and deflationary parameters, and that we do not restrict the space of strategies that the miners are allowed to use.

There are also recent work regarding miners' strategies in multi-cryptocurrency markets [6, 23]. The main difference with our work is that we focus on a single cryptocurrency in a closed world setting (e.g. we do not reason about the exchange rate of the studied cryptocurrency with the US dollar or another cryptocurrency). Finally, there are a number of papers on network properties of the Bitcoin protocol, as well as technical considerations regarding its security and privacy (see e.g. the survey by Conti et al. [5]). One interesting result here is that the network's specificity of the protocol could give participants an incentive to deviate from default behaviour [2, 11, 27].

Organization of the paper. We present our game-theoretic formalization of cryptocurrency mining in Section 2. Our results for constant and decreasing rewards are presented in Sections 3 and 4, respectively. Finally, some concluding remarks are given in Section 5. We provide the main details of the proofs of the results in the paper; however, due to the lack of space, complete proofs of all results are provided in the full version of this paper, which can be found at https://anonymous.4open.science/repository/08eff11e-78b1-4836-837a-cff08348a8c8. Besides, independent Python and C++ scripts used in the paper for the utility evaluation and plots generation can also be found in this repository.

2 A GAME-THEORETIC FORMALIZATION OF CRYPTOCURRENCY MINING

The mining game is played by a set $\mathbf{P} = \{0, 1, , \dots, m-1\}$ of players, with $m \geq 2$. In this game, each player has some reward depending on the number of blocks she owns. Every block must point to a previous block, except for the first block which is called the *genesis block*. Thus, the game defines a tree of blocks. Each block is put by one player, called the *owner* of this block. Each such tree is called a *state of the game*, or just *state*, and it represents the knowledge that each player has about the blocks that have been mined thus far.

The key question for each player is, then, where do I put my next block? In bitcoin, miners are only allowed to spend their reward if their blocks belongs to the *blockchain*, which is simply the longest chain of blocks in the current state. Thus, players face essentially two possibilities: to put their blocks right after the end of the blockchain, or try to *fork*, betting that a smaller chain will eventually become the blockchain. As the likelihood of mining the next block is directly related to the comparative hash power of a player, it makes sense to model mining as an infinite stochastic game, in which the probability of executing the action of a player p is given by her comparative hash power.

In what follows, we first define the components of the infinite stochastic games considered in this paper, and the notions of strategy, utility and equilibrium for such games. Our formalization is similar to others presented in the literature [12], except for the way in which miners are rewarded and the way in which these rewards are accumulated in the utility function. Hence, we analyse them in more detail in Section 2.2, emphasizing the properties of these two elements that are fundamental for our formalization.

2.1 The definition of the game

2.1.1 Blocks, states and the notion of blockchain. In a game played by m players, a block is defined as a string b over the alphabet $\{0, 1, \ldots, m-1\}$. We denote by $\mathbf{B}$ the set of all blocks, that is, $\mathbf{B} = \{0, 1, \ldots, m-1\}^*$. Each block apart from ε has a unique owner, defined by the function owner : $(\mathbf{B} \smallsetminus \{\varepsilon\}) \to \{0, 1, \ldots, m-1\}$ such that owner(b) is equal to the last symbol of b. As in [12], a state of the game is defined as a tree of blocks. More precisely, a state of the game, or just state, is a finite and nonempty set of blocks $q \subseteq \mathbf{B}$ that is prefix closed. That is, q is a set of strings over the alphabet $\{0, 1, \ldots, m-1\}$ such that if $b \in q$, then every prefix of b (including the empty word ε) also belongs to q. Note that a prefix closed subset of $\mathbf{B}$ uniquely defines a tree with ε as the root. The intuition here is that each element of q corresponds to a block that was put into the state q by some player. The genesis block corresponds to ε. When a player p decides to mine on top of a block b, she puts another block into the state defined by the string $b \cdot p$, where we use notation $b_1 \cdot b_2$ for the concatenation of two strings b_1 and b_2. Notice that with this terminology, given $b_1, b_2 \in q$, we have that b_2 is a descendant of b_1 in q if b_1 is a prefix of b_2, which is denoted by $b_1 \preceq b_2$. Moreover, a path in q is a nonempty set π of blocks from q for which there exist blocks b_1, b_2 such that $\pi = \{b \mid b_1 \preceq b \text{ and } b \preceq b_2\}$; in particular, b_2 is a descendant of b_1 and π is said to be a path from b_1 to b_2. Finally, let $\mathbf{Q}$ be the set of all possible states in a game played by m players, and for a state $q \in \mathbf{Q}$, let $|q|$ be its size, measured as the cardinality of the set q of strings (or blocks).

The *blockchain* of a state q, denoted by bc(q), is the path π in q of largest length, in the case that this path is unique. If two or more different paths in q are tied for the longest, then we say that the blockchain in q does not exist, and we assume that bc(q) is not defined (so that bc($\cdot$) is a partial function). Notice that if bc(q) $= \pi$, then π has to be a path in q from the genesis block ε to a leaf of q.

Example 2.1. The following picture shows a state q of the game assuming that $\mathbf{P} = \{0, 1\}$:

$$\begin{tikzcd}
 & 0 & & & & \\
\varepsilon \arrow[ru] \arrow[rd] & & & 110 & & \\
 & 1 \arrow[r] & 11 \arrow[ru] \arrow[rd] & & & \\
 & & & 111 \arrow[r] & 1111 \arrow[r] & 11110
\end{tikzcd}$$

In this case, we have that $q = \{\varepsilon, 0, 1, 11, 110, 111, 1111, 11110\}$, so q is a finite and prefix-closed subset of $\mathbf{B} = \{0, 1\}^*$. The owner of each block $b \in q \smallsetminus \{\varepsilon\}$ is given by the the last symbol of b; for instance, we have that owner(11) = 1 and owner(11110) = 0. Moreover, the longest path in q is $\pi = \{\varepsilon, 1, 11, 111, 1111, 11110\}$, so that the blockchain of q is π (in symbols, bc(q) $= \pi$). Finally, $|q| = 8$, as q is a set consisting of eight blocks (including the genesis block ε).

Assume now that q' is the following state of the game:

$$\begin{tikzcd}
 & 0 & & \\
\varepsilon \arrow[ru] \arrow[rd] & & & 110 \\
 & 1 \arrow[r] & 11 \arrow[ru] \arrow[rd] & \\
 & & & 111
\end{tikzcd}$$

We have that bc(q') is not defined since the paths $\pi_1 = \{\varepsilon, 1, 11, 110\}$ and $\pi_2 = \{\varepsilon, 1, 11, 111\}$ are tied for the longest path in q'. □

Real-life Bitcoin blocks also contain transactions that indicate movement of money in the system, and thus there are several different blocks that a player p can use to extend the current state when mining upon a block b (e.g, depending on the ordering of transactions, or the nonce being used to announce the block). Since we are interested in miners behaviour, we just focus on the owner of the block following b, and do not consider the possibility of two different blocks belonging to p being added on top of b.

2.1.2 Actions of a miner. On each step, each miner chooses a block in the current state, and attempts to mine on top of this block. Thus, in each turn, each of the players race to place the next block in the state, and only one of them succeeds. The probability of succeeding is directly related to the comparative amount of hash power available to this player, the more hash power the likely it is that she will mine the next block before the rest of the players. Once a player places a block, this block is added to the current state, obtaining a different state from which the game continues.

Let $p \in \mathbf{P}$. Given a block $b \in \mathbf{B}$ and a state $q \in \mathbf{Q}$, we denote by mine(p, b, q) an action played in the mining game in which player p mines on top of block b. Such an action mine(p, b, q) is considered to be valid if $b \in q$ and $b \cdot p \notin q$. The set of valid actions for player p is collected in the set:

$$\mathbf{A}_p \;=\; \{\text{mine}(p, b, q) \mid b \in \mathbf{B}, q \in \mathbf{Q} \text{ and } \\ \text{mine}(p, b, q) \text{ is a valid action}\}.$$

Moreover, given $a \in \mathbf{A}_p$ with $a = \text{mine}(p, b, q)$, the result of applying a to q, denoted by $a(q)$, is defined as the state $q \cup \{b \cdot p\}$. Finally, we denote by $\mathbf{A}$ the set of combined actions for the m players, that is, $\mathbf{A} = \mathbf{A}_0 \times \mathbf{A}_1 \times \cdots \times \mathbf{A}_{m-1}$.

2.1.3 The pay-off of a miner. Most cryptocurrencies based on blockchain follow this miner's payment rules: (1) Miners receive a one-time reward per each block they mine. For example, in Bitcoin, each miner will receive a pre-set reward when she mines a block (12.5BTC at the time of writing). (2) The only blocks that are valid are those in the blockchain; if a block is not in the blockchain then the reward given for mining this block cannot be spent. To prevent drastic changes on what are valid blocks, the Bitcoin protocol enforces that a block reward can only be spent when there are more than one hundred blocks on top of it.

So how should the reward function look like? The first naive idea is to give miners the block reward as soon as they put a block at the top of the current blockchain. This provides some incentive to extend the blockchain, however, it does not completely protect the system from forks that (potentially malicious) miners might want to initiate. To illustrate this, consider the state q' in Example 2.1, where we have two blocks (110 and 111) competing to be in the blockchain, and consider 111 to be the last block added to reach q'. If player 0 already cashed in the reward for the block 110 (since this was awarded immediately), she has no further incentive to mine on top of this block, as the block 1110 would give her an equal reward. This can then allow a malicious miner (1 in this example) to do a local fork (e.g. the block 111), where for instance some transactions are being blocked, and does not provide an incentive for other miners to protect the other branch competing to be the blockchain. The way that the Bitcoin protocol goes around this is by paying for blocks which are buried deep into the blockchain. This now gives a strong incentive to player 0 to mine on top of the block 110, since she cannot cash in her reward immediately after mining this block, but only once it has been extended many times.

In Section 2.2, we explain how our pay-off model mimics the rules and incentives mentioned in the previous paragraphs. For now we assume, for each player $p \in \mathbf{P}$, the existence of a reward function $r_p : \mathbf{Q} \to \mathbb{R}$ such that the reward of p in a state q is given by $r_p(q)$. Moreover, the combined reward function of the game is $\mathbf{R} = (r_0, r_1, \dots, r_{m-1})$.

2.1.4 Transition probability function. As a last component of the game, we assume that $\mathbf{Pr} : \mathbf{Q} \times \mathbf{A} \times \mathbf{Q} \to [0, 1]$ is a transition probability function satisfying that for every state $q \in \mathbf{Q}$ and combined action $\mathbf{a} = (a_0, a_1, \dots, a_{m-1})$ in $\mathbf{A}$:

$$\sum_{p=0}^{m-1} \mathbf{Pr}(q, \mathbf{a}, a_p(q)) \quad = \quad 1.$$

Notice that if p_1 and p_2 are two different players, then for every action $a_1 \in \mathbf{A}_{p_1}$, every action $a_2 \in \mathbf{A}_{p_2}$ and every state $q \in \mathbf{Q}$, it holds that $a_1(q) \neq a_2(q)$. Thus, we can think of $\mathbf{Pr}(q, \mathbf{a}, a_p(q))$ as the probability that player p places the next block, which will generate the state $a_p(q)$. As we have mentioned, such a probability is directly related with the hash power of player p, the more hash power the likely it is that action a_p is executed and p mines the next block before the rest of the players. In what follows, we assume that the hash power of each player does not change during the mining game, which is captured by the following condition: for each player $p \in \mathbf{P}$, we have that $\mathbf{Pr}(q, \mathbf{a}, a_p(q)) = h_p$ for every $q \in \mathbf{Q}$ and $\mathbf{a} \in \mathbf{A}$ with $\mathbf{a} = (a_0, a_1, \dots, a_{m-1})$. We refer to such a fixed value h_p as the hash power of player p. Moreover, we assume that $h_p > 0$ for every player $p \in \mathbf{P}$, as if this not the case then p can just be removed from the mining game.

2.1.5 The mining game: definition, strategy, utility and equilibrium. Putting together the components defined in the previous sections, we have that an infinite stochastic game is a tuple $(\mathbf{P}, \mathbf{Q}, \mathbf{A}, \mathbf{R}, \mathbf{Pr})$, where $\mathbf{P}$ is the set of players, $\mathbf{Q}$ is the set of states, $\mathbf{A}$ is the set of combined actions, $\mathbf{R}$ is the combined pay-off function and $\mathbf{Pr}$ is the transition probability function.

We have defined a game that can capture miners' interactions. A fundamental component of such a game is the strategy that each player decides to take, which combined determine the utility of each player. In particular, miners take actions and decide about strategies trying to maximize their utility. In this sense, a equilibrium of the game is a fundamental piece of information about miners' behavior, because such an equilibrium is a combination of players' strategies where no miner has an incentive to perform a different action. The notions of strategy, utility and stationary equilibrium are the last components of our game-theoretical characterization of mining in Bitcoin, and they are defined next.

A Markov strategy (or just strategy) for a player $p \in \mathbf{P}$ is a function $s : \mathbf{Q} \to \mathbf{A}_p$. We define $\mathbf{S}_p$ as the set of all strategies for player p, and $\mathbf{S} = \mathbf{S}_0 \times \mathbf{S}_1 \times \cdots \times \mathbf{S}_{m-1}$ as the set of combined strategies for the game (recall that $\mathbf{P} = \{0, \dots, m-1\}$ is the set of players). To define the notions of utility and equilibrium, we need some additional notation. Let $\mathbf{s} = (s_0, \dots, s_{m-1})$ be a combined strategy. Then given $q \in \mathbf{Q}$, define $\mathbf{s}(q)$ as the combined action $(s_0(q), \dots, s_{m-1}(q))$. Moreover, given an initial state $q_0 \in \mathbf{Q}$, the probability of reaching state $q \in \mathbf{Q}$, denoted by $\mathbf{Pr}^{\mathbf{s}}(q \mid q_0)$, is defined as 0 if $q_0 \not\subseteq q$ (that is, if q is not reachable from q_0), and otherwise it is recursively defined as follows: if $q = q_0$, then $\mathbf{Pr}^{\mathbf{s}}(q \mid q_0) = 1$; otherwise, we have that $|q| - |q_0| = k$, with $k \geq 1$, and

$$\mathbf{Pr}^{\mathbf{s}}(q \mid q_0) = \sum_{\substack{q' \in \mathbf{Q}:\\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k-1}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0) \cdot \mathbf{Pr}(q', \mathbf{s}(q'), q).$$

In this definition, if for a player p we have that $s_p(q') = a$ and $a(q') = q$, then $\mathbf{Pr}(q', \mathbf{s}(q'), q) = h_p$. Otherwise, we have that $\mathbf{Pr}(q', \mathbf{s}(q'), q) = 0$. Notice that this is well defined, since there can be at most one player p whose action in the state q' leads us to the state q. For the sake of presentation, the probability of reaching a state q from the initial state $\{\varepsilon\}$ (consisting only of the genesis block ε) is denoted by $\mathbf{Pr}^{\mathbf{s}}(q)$, instead of $\mathbf{Pr}^{\mathbf{s}}(q \mid \{\varepsilon\})$. The framework just described corresponds to a Markov chain; in particular, the probability of reaching a state from an initial state is defined in the standard way for Markov chains [17]. However, we are not interested in the steady distribution for such a Markov chain and, thus, we do not explore this connection in this paper.

We finally have all the necessary ingredients to define the utility of a player in a mining game given a particular strategy. As is common when looking at personal utilities, we define it as the summation of the expected rewards, and choose to impose a discount for future rewards using a factor $\beta \in (0, 1)$.

Definition 2.1. *The β–discounted utility of player p for the strategy $\mathbf{s}$ from the state q_0 in the mining game, denoted by $u_p(\mathbf{s} \mid q_0)$, is defined as:*

$$u_p(\mathbf{s} \mid q_0) \quad = \quad (1-\beta) \cdot \sum_{q \in \mathbf{Q}\,:\, q_0 \subseteq q} \beta^{|q|-|q_0|} \cdot r_p(q) \cdot \mathbf{Pr}^{\mathbf{s}}(q \mid q_0).$$

Notice that the value $u_p(\mathbf{s} \mid q_0)$ may not be defined if the series $\sum_{q \in \mathbf{Q}\,:\, q_0 \subseteq q} \beta^{|q|-|q_0|} \cdot r_p(q) \cdot \mathbf{Pr}^{\mathbf{s}}(q \mid q_0)$ diverges. To avoid this problem, from now on we assume that for every pay-off function $\mathbf{R} = (r_0, \dots, r_{m-1})$, there exists a polynomial P such that $|r_p(q)| \leq P(|q|)$ for every player $p \in \mathbf{P}$ and state $q \in \mathbf{Q}$. Under this simple yet general condition, which is satisfied by the pay-off functions considered in this paper and in other game-theoretical formalizations of Bitcoin mining [12], it is possible to prove that $u_p(\mathbf{s} \mid q_0)$ is a real number. Moreover, as for the definition of the probability of reaching a state from the initial state $\{\varepsilon\}$, we use notation $u_p(\mathbf{s})$ for the β–discounted utility of player p for the strategy $\mathbf{s}$ from $\{\varepsilon\}$, instead of $u_p(\mathbf{s} \mid \{\varepsilon\})$.

As a last ingredient in our formalization, we need to introduce the notion stationary equilibrium. Given a player $p \in \mathbf{P}$, a combined strategy $\mathbf{s} \in \mathbf{S}$, with $\mathbf{s} = (s_0, s_1, \ldots, s_{m-1})$, and a strategy s for player p ($s \in \mathbf{S}_p$), we denote by $(\mathbf{s}_{-p}, s)$ the strategy $(s_0, s_1, \ldots s_{p-1}, s, s_{p+1}, \ldots, s_{m-1})$.

Definition 2.2. *A strategy* $\mathbf{s}$ *is a* β*–discounted stationary equilibrium from the state* q_0 *in the mining game if for every player* $p \in \mathbf{P}$ *and every strategy* s *for player* p ($s \in \mathbf{S}_p$)*, it holds that:*

$$u_p(\mathbf{s} \mid q_0) \quad \geq \quad u_p((\mathbf{s}_{-p}, s) \mid q_0).$$

2.2 On the pay-off and utility of a miner

We design our pay-off model to mimic the incentives of the payment system of Bitcoin and other cryptocurrencies. More precisely, given a player p and a state q, for every block $b \in q$ assume that the reward obtained by p for the block b in q is given by $r_p(b, q)$, so that $r_p(q) = \sum_{b \in q} r_p(b, q)$. This decomposition can be done in a natural and straightforward way for the pay-off functions considered in this paper and in other game-theoretical formalizations of Bitcoin mining [12]. Then to enforce the fact that the block reward for the block b is not granted immediately, we pay in Definition 2.1 a portion of $r_p(b, q)$, for each state q where b is in. In other words, if a miner owns a block, then she will be rewarded for this block in every state where this block is part of the blockchain, in which case $r_p(b, q) > 0$.

This means that we might pay the miner infinitely many times for a single block. A natural question is then whether we overpay for the blocks. This is where the discount factors in our definition of utility come into play. More precisely, we pay a portion of block b's reward each time it is included in the current blockchain. In other words, when a player mines a new block, she will receive the full amount for this block only if she manages to maintain the block in the blockchain up to infinity. Otherwise, if this block ceases to be in the blockchain, we only pay a fraction of the full amount. Formally, given a combined strategy $\mathbf{s}$, we can define the utility of a block b for a player p, denoted by $u_p^b(\mathbf{s})$, as follows:

$$u_p^b(\mathbf{s}) \quad = \quad (1 - \beta) \cdot \sum_{q \in \mathbf{Q}\,:\, b \in \mathrm{bc}(q)} \beta^{|q|-1} \cdot r_p(q, b) \cdot \mathbf{Pr}^{\mathbf{s}}(q).$$

For simplicity, here we assume that the game starts in the genesis block ε, and not in an arbitrary state q_0. The discount factor in this case is $\beta^{|q|-1}$, since $|\{\varepsilon\}| = 1$.

To see that we pay the correct amount for each block, assume that there is a maximum value for the reward of a block b for player p, which is denoted by $M_p(b)$. Thus, we have that there exists $q_1 \in \mathbf{Q}$ such that $b \in q_1$ and $M_p(b) = r_p(b, q_1)$, and for every $q_2 \in \mathbf{Q}$ such that $b \in q_2$, it holds that $r_p(b, q_2) \leq M_p(b)$. Again, such an assumption is satisfied by most currently circulating cryptocurrencies, by the pay-off functions considered in this paper, and by other game-theoretical formalizations of Bitcoin mining [12]. Then we have that:

Proposition 2.3. *For every player* $p \in \mathbf{P}$*, block* $b \in \mathbf{B}$ *and combined strategy* $\mathbf{s} \in \mathbf{S}$*, it holds that:* $u_p^b(\mathbf{s}) \leq \beta^{|b|} \cdot M_p(b)$.

Thus, the utility obtained by player p for a block b is at most $\beta^{|b|} \cdot M_p(b)$, that is, the maximum reward that she can obtained for the block b in a state multiplied by the discount factor $\beta^{|b|}$, where $|b|$ is the minimum number of steps that has to be performed to reach a state containing b from the initial state $\{\varepsilon\}$. Moreover, a miner can only aspire to get the maximum utility for a block b if once b is included in the blockchain, it stays in the blockchain in every future state. This tell us that our framework puts a strong incentive for each player in maintaining her blocks in the blockchain.

$$\begin{tikzcd}
 & & & & & 10011 \\
\varepsilon \arrow[r] & 1 \arrow[r] & 10 \arrow[r] & 100 \arrow[r] & 1001 \arrow[ru] \arrow[rd] & \\
 & & & & & 10010
\end{tikzcd}$$

Figure 1: Although two paths are competing to become the blockchain, the blocks up to 1001 will contribute to the reward in both paths.

3 EQUILIBRIA WITH CONSTANT REWARD

The first version of the game we analyse is when the reward function $r_p(q)$ pays each block in the blockchain the same amount c. While this may not reflect many cryptocurrencies' protocols, where the reward diminishes over time [18–20, 30, 32, 33], this simpler case serves as a good baseline for future results, and it establishes the main techniques we will use. Furthermore, as argued in this section, the results obtained here could serve as a good recommendation for cryptocurrencies' protocols to enforce fair behaviour of the miners when their income consist only of transaction fees.

3.1 Defining constant reward

When considering the constant reward c for each block, $r_p(q)$ will equal c times the number of blocks owned by p in the blockchain $\mathrm{bc}(q)$ of q, when the latter is defined. On the other hand, when $\mathrm{bc}(q)$ is not defined it might seem tempting to simply define $r_p(q) = 0$. However, even if there is more than one longest path from the root of q to its leaves, it might be the case that all such paths share a common subpath. In fact, this often happens in the Bitcoin's network, when two blocks were mined on top of the same block (with a small time delay). While in this situation the blockchain is not defined, the miners know that they will at least be able to collect their reward on the portion of the state these two paths agree on. Figure 1 illustrates this situation.

In order to model the aforementioned scenario, we need to introduce some notation. Recall that a block b is a string over the alphabet $\mathbf{P}$, and we use notation $|b|$ for the length of b as a string. Moreover, given blocks b_1, b_2, we use notation $b_1 \preceq b_2$ to indicate that b_1 is a prefix of b_2 when considered as strings. Then we define:

$$\begin{aligned}
\mathrm{longest}(q) \quad &= \quad \{b \in q \mid \text{for every } b' \in q : |b'| \leq |b|\} \\
\mathrm{meet}(q) \quad &= \quad \{b \in q \mid \text{for every } b' \in \mathrm{longest}(q) : b \preceq b'\}.
\end{aligned}$$

Intuitively, longest(q) contains the leaves of all paths in the state q that are currently competing for the blockchain, and meet(q) is the path from the genesis block to the last block for which all these paths agree on. For instance, if q is the state from Figure 1, then we have that longest$(q) = \{10011, 10010\}$, and meet$(q) = \{\varepsilon, 1, 10, 100, 1001\}$. Notice that meet$(q)$ is well defined as $\preceq$ is a linear order on the finite and non-empty set $\{b \in q \mid$ for every $b' \in$

longest$(q) : b \preceq b'\}$. Also notice that meet(q) = bc(q), whenever bc(q) is defined.

Now we have the ingredients to introduce a reward function that pays a player according to her blocks in meet(q). More precisely, the reward function considered in this section, which is called **constant reward**, is defined as follows for a player p:

$$r_p(q) \quad = \quad c \cdot \sum_{b \in \text{meet}(q)} \chi_p(b),$$

where c is a positive real number, $\chi_p(b) = 1$ if owner$(b) = p$, and $\chi(p) = 0$ otherwise. Notice that this function is well defined since meet(q) always exists. Moreover, if q has a blockchain, then we have that meet(q) = bc(q) and, hence, the reward function is defined for the blockchain of q.

3.2 The default strategy maximizes the utility

Let us start with analysing the most obvious strategy for all players: regardless of what everyone else does, keep mining on the blockchain, which is called the *default* strategy. More precisely, a player following the default strategy tries to mine upon the final block that appears in the blockchain of a state q. If the blockchain in q does not exist, meaning that there are al least two longest paths from the genesis block, then the player tries to mine on the final block of one of these paths according to her rewards in them; she chooses the one that maximizes her reward, which in the case of constant reward means the path that contains the largest number of blocks belonging to her (if there is more than one of these paths, then between the final blocks of these paths she chooses the first according to a lexicographic order on the strings in $\{0, \dots, m-1\}^*$). Notice that this is called the default strategy as it reflects the desired behaviour of the miners participating in the Bitcoin network. For a player p, let us denote this strategy by DF$_p$, and consider the combined strategy $\mathbf{DF} = (\text{DF}_0, \text{DF}_1, \dots, \text{DF}_{m-1})$.

We can now easily calculate the utility of player p under **DF**. Intuitively, a player p will receive a fraction h_p of the next block that is being placed in the blockchain, corresponding to her hash power. Therefore, at stage i of the mining game, the blockchain defined by the game will have i blocks, and the expected amount of blocks owned by the player p will be $h_p \cdot i$. This means that the total utility for player p amounts to

$$u_p(\mathbf{DF}) \;=\; (1-\beta) \cdot h_p \cdot c \cdot \sum_{i=0}^{\infty} i \cdot \beta^i \;=\; h_p \cdot c \cdot \frac{\beta}{(1-\beta)}.$$

The question then is: can any player do better? As we show in the following theorem, the answer is no as the default strategy maximizes the utility.

Theorem 3.1. *Let p be a player, β be a discount factor in $(0, 1)$ and u_p be the utility function defined in terms of β. Then for every combined strategy* $\mathbf{s}$*: $u_p(\mathbf{s}) \leq u_p(\mathbf{DF})$.*

Proof. Let $\mathbf{s} = (s_0, \dots, s_{m-1})$ be an arbitrary combined strategy, and define $Q_\mathbf{s} = \{q \in \mathbf{Q} \mid \mathbf{Pr}^\mathbf{s}(q) > 0\}$. Thus, $Q_\mathbf{s}$ is the set of all states that can be reached from the genesis block using the combined strategy $\mathbf{s}$. For example, we have that $Q_\mathbf{DF}$ is the set of states q such that q consists of a single path from the genesis block to the final block in bc(q). Moreover, define a mapping $\sigma : Q_\mathbf{s} \to 2^{Q_\mathbf{DF}}$ as follows. Given two states q_1, q_2, we say that q_2 can be reached from q_1 in one step according to the strategy $\mathbf{s}$ if $q_2 = a(q_1)$, where $a = s_{p'}(q_1)$ for some $p' \in \mathbf{P}$; that is, we have that q_2 can be reached from q_1 in one step according to $\mathbf{s}$ if q_2 is the result of applying the action $s_{p'}(q_1)$ to q_1 for some player p'. Then for each state $q \in Q_\mathbf{s}$, consider all distinct sequences $\rho = q_0, \dots, q_n$ such that q_{i+1} can be reached from q_i in one step according to $\mathbf{s}$ ($i \in \{1, \dots, n-1\}$), $q_0 = \varepsilon$ and $q_n = q$. To each such a sequence $\rho = q_0, \dots, q_n$, associate a block b_ρ of length n as follows. For every $i \in \{1, \dots, n\}$, if for a player $p' \in \mathbf{P}$, it holds that $s_{p'}(q_{i-1}) = a_{p'}$ and $q_i = a_{p'}(q_{i-1})$, then i-th symbol of b_ρ is p'. Notice that the i-th symbol of b_ρ is well defined as $q \in Q_\mathbf{s}$ and the sets of actions for two distinct players are disjoint. Finally, define $\sigma(q)$ as the set of all states $q' \in Q_\mathbf{DF}$ consisting of a single path whose final block is a block b_ρ associated to a sequence ρ for q; formally, we have that:

$$\sigma(q) \;=\; \big\{q' \in Q_\mathbf{DF} \mid \text{there exists a sequence } \rho \text{ for } q \text{ such that } q' = \{b \in \mathbf{B} \mid b \preceq b_\rho\}\big\}.$$

Intuitively, if ρ is a sequence for q, then the i-th symbol of b_ρ tells us which player won the i-th round of the mining game. The state $q' \in \sigma(q)$ associated with b_ρ simply recreates what would have happened if this was the order in which the players were mining the blocks when they use **DF** as their strategy. In this proof, we need the following property of the mapping σ.

Claim 1. *For every pair of distinct states q, q' in $Q_\mathbf{s}$, the sets $\sigma(q)$ and $\sigma(q')$ are disjoint.*

Recall that the utility of player p using combined strategy **DF** is defined as:

$$u_p(\mathbf{DF}) \quad = \quad (1-\beta) \cdot \sum_{q \in \mathbf{Q}} \beta^{|q|-1} \cdot r_p(q) \cdot \mathbf{Pr}^\mathbf{DF}(q).$$

If we choose to sum only over the states in the images under σ of the states of $Q_\mathbf{s}$, then by Claim 1 we have that:

$$u_p(\mathbf{DF}) \quad \geq \quad (1-\beta) \cdot \sum_{q \in \sigma(q^*) \,:\, q^* \in Q_\mathbf{s}} \beta^{|q|-1} \cdot r_p(q) \cdot \mathbf{Pr}^\mathbf{DF}(q).$$

Rearranging the term in the right-hand side, we obtain:

$$u_p(\mathbf{DF}) \quad \geq \quad (1-\beta) \cdot \sum_{q^* \in Q_\mathbf{s}} \sum_{q \in \sigma(q^*)} \beta^{|q|-1} \cdot r_p(q) \cdot \mathbf{Pr}^\mathbf{DF}(q).$$

For each state $q^* \in Q_\mathbf{s}$, notice that if $q \in \sigma(q^*)$, then $|q| = |q^*|$ and the number blocks owned by p in q is the same as the number of blocks owned by p in q^*. Thus, we have that $r_p(q) \geq r_p(q^*)$ since $q \in Q_\mathbf{DF}$ and, therefore, every block owned by p in q is in bc(q) and bc(q) = meet(q). Notice that it could be the case that $r_p(q) > r_p(q^*)$, as some blocks owned by p in q^* may not be in meet(q^*). We conclude that:

$$\sum_{q \in \sigma(q^*)} \beta^{|q|-1} \cdot r_p(q) \cdot \mathbf{Pr}^\mathbf{DF}(q) \geq \sum_{q \in \sigma(q^*)} \beta^{|q^*|-1} \cdot r_p(q^*) \cdot \mathbf{Pr}^\mathbf{DF}(q)$$
$$= \beta^{|q^*|-1} \cdot r_p(q^*) \sum_{q \in \sigma(q^*)} \mathbf{Pr}^\mathbf{DF}(q).$$

Moreover, by definition of $\mathbf{Pr}^\mathbf{DF}$ and $\mathbf{Pr}^\mathbf{s}$, we have that:

$$\sum_{q \in \sigma(q^*)} \mathbf{Pr}^\mathbf{DF}(q) \quad = \quad \mathbf{Pr}^\mathbf{s}(q^*).$$

Combining the previous results and considering that $\mathbf{Pr}^{\mathbf{s}}(q^*) = 0$ for every $q^* \in \mathbf{Q} \setminus Q_{\mathbf{s}}$, we conclude that:

$$\begin{aligned} u_p(\mathbf{DF}) &\geq (1-\beta) \cdot \sum_{q^* \in Q_{\mathbf{s}}} \left(\beta^{|q^*|-1} \cdot r_p(q^*) \sum_{q \in \sigma(q^*)} \mathbf{Pr}^{\mathbf{DF}}(q) \right) \\ &= (1-\beta) \cdot \sum_{q^* \in Q_{\mathbf{s}}} \beta^{|q^*|-1} \cdot r_p(q^*) \cdot \mathbf{Pr}^{\mathbf{s}}(q^*) \\ &= (1-\beta) \cdot \sum_{q^* \in \mathbf{Q}} \beta^{|q^*|-1} \cdot r_p(q^*) \cdot \mathbf{Pr}^{\mathbf{s}}(q^*) \\ &= u_p(\mathbf{s}), \end{aligned}$$

which was to be shown. □

As a corollary of Theorem 3.1, we obtain that:

Corollary 3.2. *For every $\beta \in (0,1)$, the strategy* **DF** *is a β-discounted stationary equilibrium.*

While constant-block reward does not faithfully model the reality of most cryptocurrencies, we would like to argue why Theorem 3.1 could serve as a recommendation on how to enforce good behaviour of miners when the block reward consists only of transaction fees. If a cryptocurrency protocol imposes a transaction fee proportional to the size of the transaction and a maximal size of a block, then the blocks would regularly achieve the maximal reward assuming that the volume of transactions is high. This would make the block reward constant, so Theorem 3.1 tells miners that the best strategy is to mine on top of the existing blockchain, as this will maximize their utility in the long run. Thus, such constraint on transaction fees would ensure the neutrality of the protocol. Note however that if the market value of the cryptocurrency is too volatile, we could reach a point where the incentive to mine is really low, or transaction fees are too expensive.

4 DECREASING REWARD

Miner's fees in many cryptocurrencies, including Bitcoin, are not constant, but diminish over time [18, 30, 32, 33]. We model such fees as a constant factor that is lowered after every new block in the blockchain. That is, we use the following reward function r_p for all players $p \in \mathbf{P}$, denoted as the α-**discounted reward**:

$$r_p(q) = c \cdot \sum_{b \in \text{meet}(q)} \alpha^{|b|} \cdot \chi_p(b),$$

In this section, we show that when miners' fees decrease over time, forking can be a good strategy. In fact, we explore in Sections 4.1 and 4.2 several strategies based on the maximum length of the fight the miner is willing to carry out when doing a fork, and how far back the miner is willing to do a fork. Interestingly, for such strategies we confirm the folklore fact that it is profitable to fork with more than half of the hash power. Moreover, our exploration in Section 4.2 gives us a concrete strategy that with less than half of the hash power can be used to defeat the default strategy.

4.1 When is forking a good strategy?

To calculate when forking is a viable option, we consider a scenario when one of our m players decides to deviate from the default strategy, while the remaining players all follow the default strategy. The first observation is that in this case we can reduce the m player game to a two player game, where all the players following the default strategy are represented by a single player behaving as the protocol dictates, with the combined hash power of all these players. Therefore in this section we will consider that the mining game is played by two players 0 and 1, where 0 represents the miners behaving according to the default strategy, and 1 the miner trying to determine whether forking is economically more viable than mining on the existing blockchain. We always assume that player 1 has hash power h, while player 0 has hash power $1 - h$.

$$\begin{tikzcd} & & & & 1111 \arrow[r] & 11111 \\ & 1 \arrow[r] & 11 \arrow[r] & 111 \arrow[ur, dashed] \arrow[r] & 1110 & \\ \varepsilon \arrow[ur, dashed] \arrow[dr] & & & & & \\ & 0 & & & & \end{tikzcd}$$

Figure 2: Dashed arrows are used to indicate when player 1 does a fork. The first block (block 0) is mined by player 0. At this point, player 1 decides to fork (mining the block 1), and successfully mines the blocks 11 and 111 on this branch. When player 0 mines the block 1110, player 1 decides to fork again, mining the blocks 1111 and 11111.

In order to determine whether there is an incentive for player 1 to fork, we first need to determine her utility when she is playing according to the default strategy $\mathbf{DF} = (\text{DF}_0, \text{DF}_1)$.

Lemma 4.1. *If h is the hash power of player 1, then:*

$$u_1(\mathbf{DF}) = h \cdot c \cdot \frac{\alpha \cdot \beta}{(1 - \alpha \cdot \beta)}.$$

As in the case of constant reward, this corresponds to h times the utility of winning all the blocks in the single blockchain generated by the default strategy.

Now suppose that player 1 deviates from the default strategy, and considers a strategy based on forking the blockchain once player 0 mines a block. How would this new strategy look? To consider a realistic scenario, in this section we consider the strategy AF (for *always fork*), where player 1 forks as soon as player 0 mines a block in the blockchain, and she continues mining on the new branch until it becomes the blockchain. Here player 1 is willing to fork every time player 0 produces a block in the blockchain. In other words, in AF, player 1 tries to have all the blocks in the blockchain. This strategy is depicted in Figure 2.

4.1.1 *The utility of always forking.*

The question we want to answer is twofold. On the one hand, we want to know whether AF is a better strategy than DF_1 for player 1, under the assumption that player 0 uses DF_0, and under some specific values of α, β and h. On the other hand, and perhaps more interestingly, we can also answer a more analytical question: given realistic values of α and β, how much hash power does player 1 need to consider following AF instead of DF_1? Answering both questions requires us to compute the utility for the strategy $\mathbf{AF} = (\text{DF}_0, \text{AF})$.

Theorem 4.2. *If h is the hash power of player 1, then:*

$$u_1(\mathbf{AF}) = \frac{\Phi}{1 - \Gamma}, \text{ where}$$

$$\Phi = \frac{\alpha \cdot \beta \cdot h \cdot c}{(1-\alpha)} \cdot \left[\mathrm{C}(\beta^2 \cdot h \cdot (1-h)) - \alpha \cdot \mathrm{C}(\alpha \cdot \beta^2 \cdot h \cdot (1-h))\right],$$
$$\Gamma = \alpha \cdot \beta \cdot h \cdot \mathrm{C}(\alpha \cdot \beta^2 \cdot h \cdot (1-h)),$$

and $\mathrm{C}(x) = \frac{1-\sqrt{1-4x}}{2x}$ *is the generating function of Catalan numbers.*

Before presenting the proof of Theorem 4.2, let us give some intuition about it. Player 1, adopting the AF strategy, will always start the game mining on ε, regardless of how many blocks player 0 manages to append, and continues until her branch is the longest. Therefore, the only states that contribute to player 1's utility are those in where she made at least one successful fork (all others states give zero reward to her). Having player 1 achieved the longest branch once, say, at block b, both players will now mine on b and the situation repeats as if b were ε, with proper shifting in the reward and β-discount. In other words, we have $u_1(\mathbf{AF}) = \Phi + \Gamma \cdot u_1(\mathbf{AF})$, where Φ is the contribution of a single successful fork, and Γ is the shifting factor, from which we obtain the expression for $u_1(\mathbf{AF})$ given in Theorem 4.2.

Proof. Let $Q_{\mathbf{AF}} = \{q \in \mathbf{Q} \mid \mathbf{Pr}^{\mathbf{AF}}(q) > 0\}$ be the set of all states that can be reached from the genesis block using the strategy **AF**. Recall from the proof of Theorem 3.1 the definition of sequence ρ for a state q, and the construction of string b_ρ from such a sequence ρ. By using these elements, we define $\tau : Q_{\mathbf{AF}} \to 2^{\{0,1\}^*}$ as follows: $\tau(q) = \{b_\rho \mid \rho \text{ is a sequence for } q\}$. Intuitively, $\tau(q)$ is the set of all moves that players 0 and 1 can do in $|q| - 1$ steps according to **AF** that lead them to the state q when starting in the genesis block. As such, they are coded as sequences of zeros and ones that tell us which player puts a block at the stage i of the game, for $i \in \{1, \ldots, |q| - 1\}$. It is straightforward to verify the following:

Claim 2. *For every $q, q' \in Q_{\mathbf{AF}}$, it holds that:*

(a) *If $q \neq q'$, then $\tau(q)$ is disjoint from $\tau(q')$.*
(b) $\mathbf{Pr}^{\mathbf{AF}}(q) = \sum_{w \in \tau(q)} \mathbf{Pr}(w)$*, where $\mathbf{Pr}(w)$ for a word w with n_0 zeroes and n_1 ones is defined as $h^{n_1}(1-h)^{n_0}$.*

In particular, Claim 2 (a) can be proved exactly in the same way Claim 1 is proved. Notice that Claim 2 (a) tells us that a sequence of actions of players 0 and 1 uniquely determines a state of the game. Moreover, Claim 2 (b) tells us that the probability of a state q is the sum of probabilities of all the sequences of actions of players 0 and 1 that end up in q when started in the genesis block. Observe that since the actions of players 0 and 1 are independent trials, with probabilities $1-h$ and h, respectively, the probability of a state where player 0 wins n_0 rounds and player 1 wins n_1 rounds is $h^{n_1}(1-h)^{n_0}$, as stated in the claim.

For every $w \in \{0,1\}^*$, there exists a unique state $q \in Q_{\mathbf{AF}}$ such that $w \in \tau(q)$. Given Claim 2 (a), to prove this claim we only need to prove the existence of such a state q. If $w = \varepsilon$, then $q = \{\varepsilon\}$. On the other hand, if $w = p_1 \cdots p_n$ with $n \geq 1$ and each $p_i \in \{0,1\}$, then $q = q_n$ in a sequence $q_0, \ldots, q_n$ of states defined by the rules: (1) $q_0 = \varepsilon$; and (2) for every $i \in \{1, \ldots, n\}$, it holds that $q_i = a_i(q_{i-1})$, where $a_i = \mathrm{DF}_0(q_{i-1})$ if $p_i = 0$, and $a_i = \mathrm{AF}(q_{i-1})$ if $p_i = 1$. Thus, we conclude that the utility of player 1 can be rewritten as follows:

$$\begin{aligned}
u_1(\mathbf{AF}) &= (1-\beta) \cdot \sum_{q \in \mathbf{Q_{AF}}} \beta^{|q|-1} \cdot r_1(q) \cdot \mathbf{Pr}^{\mathbf{AF}}(q) \\
&= (1-\beta) \cdot \sum_{q \in \mathbf{Q_{AF}}} \beta^{|q|-1} \cdot r_1(q) \cdot \Bigg(\sum_{w \in \tau(q)} \mathbf{Pr}(w)\Bigg) \\
&= (1-\beta) \cdot \sum_{q \in \mathbf{Q_{AF}}} \sum_{w \in \tau(q)} \beta^{|q|-1} \cdot r_1(q) \cdot \mathbf{Pr}(w) \\
&= (1-\beta) \cdot \sum_{q \in \mathbf{Q_{AF}}} \sum_{w \in \tau(q)} \beta^{|w|} \cdot r_1(w) \cdot \mathbf{Pr}(w) \\
&= (1-\beta) \cdot \sum_{w \in \{0,1\}^*} \beta^{|w|} \cdot r_1(w) \cdot \mathbf{Pr}(w),
\end{aligned}$$

$$\begin{tikzcd} \varepsilon \arrow[r] & 1 \end{tikzcd} \qquad \begin{tikzcd} & 1 \arrow[r] & 11 \\ \varepsilon \arrow[ur] \arrow[dr] & & \\ & 0 & \end{tikzcd} \qquad \begin{tikzcd} & 1 \arrow[r] & 11 \arrow[r] & 111 \\ \varepsilon \arrow[ur] \arrow[dr] & & & \\ & 0 \arrow[r] & 00 & \end{tikzcd}$$

(a) (b) (c)

Figure 3: States in a game played according to strategy AF.

given that $|w| = |q| - 1$ for every $w \in \tau(q)$, and assuming that $r_1(w)$ is defined as $r_1(q)$ for the only state q such that $w \in \tau(q)$.

We now describe all the states in which player 1 receives a non-zero reward in terms of words. For this, let us consider the set S of all words $w \in \{0,1\}^*$ that represent states q (via τ) in which player1 owns at least one block in the blockchain for the *first time*. The smallest of them is $w = 1$, which represents the state in Figure 3 (a). This state is created when player 1 wins the first move of the game, successfully mining upon the genesis block. Next is the word 011, representing the state in Figure 3 (b). To arrive at this state player 0 must have mined the first block, player 1 forked, and then player 1 won the following block (on her forking branch). The next words in S are 00111 and 01011, both representing the state in Figure 3 (c). In general, the words in the set S have the form $d \cdot 1$, where d is a *Dyck word*: a word with the same number of 0s and 1s, but such that no prefix of d has more 1s than 0s (this intuitively means that at no point player 1 had more blocks than player 0). Note that the only Dyck word of length 0 is ε, the next Dyck word by length is 01, and then 0011 and 0101, etc. As it turns out, the number of Dyck words of length $2m$ is the m-th Catalan number [24]. We denote by $\mathcal{D}_{2m}$ the set of Dyck words of length $2m$, and by $\mathcal{D}$ the set of all Dyck words.

Since all states where player 1 receives a reward involve putting a block in the blockchain, all words w with $r_1(w) > 0$ are therefore of the form $d \cdot 1 \cdot w'$ with $d \in \mathcal{D}$. Now let q be the only state such that $d \cdot 1 \cdot w' \in \tau(q)$. State q can be seen as a tree with two branches: one only with blocks earned by player 0, and the other one with at least $\frac{|d|}{2} + 1$ blocks owned by player 1 (plus maybe more, depending on w'). We can then calculate the reward for q as $r_1(q) = r_1(d \cdot 1) + \alpha^{\frac{|d|}{2}+1} \cdot r_1(w')$, obtaining that:

$$u_1(\mathbf{AF}) = (1-\beta) \cdot \sum_{d \in \mathcal{D}} \sum_{w \in \{0,1\}^*} \Bigg(\beta^{|d|+1+|w|} \cdot \left[r_1(d \cdot 1) + \alpha^{\frac{|d|}{2}+1} \cdot r_1(w)\right] \cdot \mathbf{Pr}(d \cdot 1 \cdot w)\Bigg).$$

Splitting up the summation, we get that $u_1(\mathbf{AF})$ is equal to:

$$(1-\beta)\cdot\sum_{d\in\mathcal{D}}\sum_{w\in\{0,1\}^*}\beta^{|d|+1+|w|}\cdot r_1(d\cdot 1)\cdot\mathbf{Pr}(d\cdot 1\cdot w)+$$
$$(1-\beta)\cdot\sum_{d\in\mathcal{D}}\sum_{w\in\{0,1\}^*}\beta^{|d|+1+|w|}\cdot\alpha^{\frac{|d|}{2}+1}\cdot r_1(w)\cdot\mathbf{Pr}(d\cdot 1\cdot w).$$

We denote the first term in the equation above by Φ. By definition of the probability of a word, we have that $\mathbf{Pr}(d\cdot 1\cdot w)=\mathbf{Pr}(d\cdot 1)\cdot\mathbf{Pr}(w)$. Next, we use this fact in the expression for $u_1(\mathbf{AF})$ to split the second term into the elements that depend only on d, and the ones that depend only on w:

$$u_1(\mathbf{AF}) = \Phi + \left(\sum_{d\in\mathcal{D}}\beta^{|d|+1}\cdot\alpha^{\frac{|d|}{2}+1}\cdot\mathbf{Pr}(d\cdot 1)\right)\cdot \left((1-\beta)\cdot\sum_{w\in\{0,1\}^*}\beta^{|w|}\cdot r_1(w)\cdot\mathbf{Pr}(w)\right).$$

Since the term $(1-\beta)\cdot\sum_{w\in\{0,1\}^*}\beta^{|w|}\cdot r_1(w)\cdot\mathbf{Pr}(w)$ is precisely $u_1(\mathbf{AF})$, we have that:

$$u_1(\mathbf{AF}) = \Phi+\left(\sum_{d\in\mathcal{D}}\beta^{|d|+1}\cdot\alpha^{\frac{|d|}{2}+1}\cdot\mathbf{Pr}(d\cdot 1)\right)\cdot u_1(\mathbf{AF}).$$

By denoting with Γ the term $\sum_{d\in\mathcal{D}}\beta^{|d|+1}\cdot\alpha^{\frac{|d|}{2}+1}\cdot\mathbf{Pr}(d\cdot 1)$, we get the equation $u_1(\mathbf{AF})=\frac{\Phi}{1-\Gamma}$. Let us now find a closed form for Γ:

$$\begin{aligned}\Gamma &= \sum_{d\in\mathcal{D}}\beta^{|d|+1}\cdot\alpha^{\frac{|d|}{2}+1}\cdot\mathbf{Pr}(d\cdot 1)\\ &= \alpha\cdot\beta\cdot\sum_{\ell=0}^{\infty}\sum_{d\in\mathcal{D}_{2\ell}}(\alpha\cdot\beta^2)^{\ell}\cdot h^{\ell}\cdot(1-h)^{\ell}\cdot h\\ &= \alpha\cdot\beta\cdot h\cdot\sum_{\ell=0}^{\infty}|\mathcal{D}_{2\ell}|\cdot(\alpha\cdot\beta^2\cdot h\cdot(1-h))^{\ell}\\ &= \alpha\cdot\beta\cdot h\cdot\mathbf{C}(\alpha\cdot\beta^2\cdot h\cdot(1-h)).\end{aligned}$$

The final equality is obtained by recalling the fact that $|\mathcal{D}_{2\ell}|$ is the ℓ-th Catalan number, so that the summation in the previous line defines the generating function of these numbers.

The closed form for Φ is computed analogously, yielding the claimed expression. □

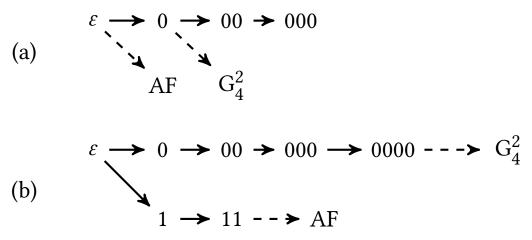
4.1.2 *When is* AF *better than* DF*?*

With the closed forms for $u_1(\mathbf{DF})$ and $u_1(\mathbf{AF})$, we can compare the utilities of these strategies for player 1 for fixed values of α and β, but varying her hash power. In particular, we consider realistic values of α and β as follows. For α we calculate the compound version of the discount in Bitcoin, that is, a value of α that would divide the reward by half every 210.000 blocks, i.e. $\alpha = 0.9999966993$. For β we calculate the 10-minute rate that is equivalent to the US real interest rate in the last few years, which is approximately 2%.[2] This gives us a value of $\beta = 0.9999996156$.

Figure 5(a) shows the value of the utility of player 1 for the combined strategies **AF** and **DF** (this figure also includes two other strategies that will be explained in the next section). The plot data was generated using GMP C++ multi precision library [8]. The point where the utility for **AF** and **DF** meet is $h = 0.499805 \pm 0.000001$, which means that player 1 should use **AF** as soon as she controls more than this proportion of the hash power. On the other hand, the more her hash power is below this threshold, the more she should adopt the default strategy. This follows since in this case she will indefinitely be involved in fights she cannot win.

It is interesting to note that this value h is strictly less than 0.5 (a similar result was obtained in [12], although in a model without discounted reward). This captures the fact that even with less than half of the hash power, player 1 can still win a finite number of initial blocks, which receive a significant reward due to having a low discount factor. It can be shown that the hash power where **AF** and **DF** meet tends to 0.5 when α and β tend to 1.

[2]https://data.worldbank.org/indicator/FR.INR.RINR?locations=US



$$\begin{tikzcd} \varepsilon \arrow[r] \arrow[dr, dashed] & 0 \arrow[r] \arrow[dr, dashed] & 00 \arrow[r] & 000 \\ & \mathrm{AF} & \mathrm{G}^2_4 & \end{tikzcd}$$

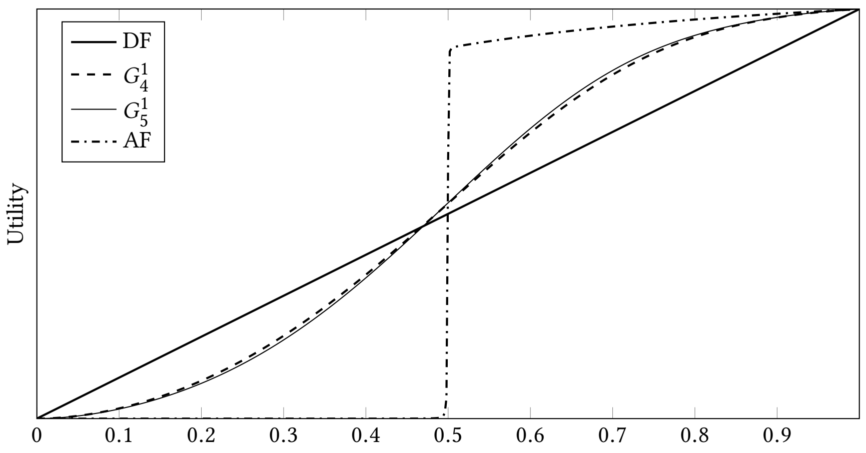
$$\begin{tikzcd} \varepsilon \arrow[r] \arrow[dr] & 0 \arrow[r] & 00 \arrow[r] & 000 \arrow[r] & 0000 \arrow[r, dashed] & \mathrm{G}^2_4 \\ & 1 \arrow[r] & 11 \arrow[r, dashed] & \mathrm{AF} & & \end{tikzcd}$$

Figure 4: Difference between AF **and** G^2_4 **in terms of actions of these strategies in two states.**

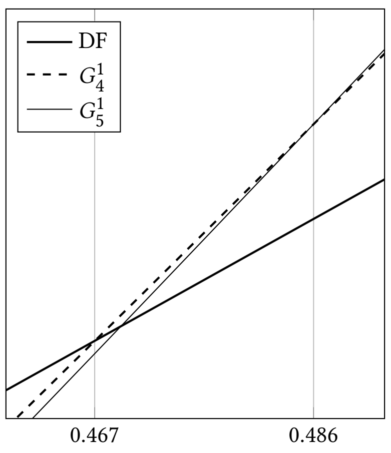
4.2 Giving up for more utility

By adding a little more flexibility to the strategy of always forking, we can identify approaches that make a fork profitable with less hash power. The families of strategies that we study in this section involve two parameters. The first parameter, denoted by k, regulates how far back the miner will fork, when confronted with a chain of blocks she does not own. The second parameter, called the *give-up* time, and denoted by ℓ, tells us the maximum number of blocks that the player's opponent is allowed to extend the current blockchain with before the player gives up mining on the forking branch. If the player does not manage to transform her fork into the new blockchain before her opponent mines ℓ blocks, she will restart the strategy treating the tail of the current blockchain as the new genesis block. We denote these strategies by G^k_ℓ.

Example 4.3. Let us consider how G^2_4 would be different from AF. Since $k = 2$, both strategies take the same action when the state is $\{\varepsilon\}$, $\{\varepsilon, 0\}$, and $\{\varepsilon, 0, 00\}$, namely, mining at ε. Hence, assume that both strategies are facing a chain of three blocks owned by player 0, as shown in Figure 4(a). In this case, AF would again try to do a fork from the genesis block as no block belongs to player 1. On the other hand, G^2_4 would try to do a fork instead on the dotted line, that is, the second block that does not belong to her. The second difference is provided by the give-up time, which is shown in Figure 4(b). Normally, AF is willing to continue forking regardless of the hope of winning, therefore the move for the state in Figure 4(b) would still be to mine upon her own block 11. On the other hand, G^2_4 has already seen 4 blocks from the start of the fork, so with this strategy player 1 instead gives up and tries to mine upon 0000, rebooting the strategy as if 0000 was the genesis block. Note that in fact AF = G^∞_∞, that is, the strategy where player 1 is willing to keep fighting for a



(a) Utilities for player 1 of combined strategies, **DF**, $\mathbf{G}_4^1$, $\mathbf{G}_5^1$ and **AF**



(b) Zoom in around the values of the hash power where **DF** intersects with $\mathbf{G}_4^1$ and $\mathbf{G}_5^1$.

Figure 5: Comparing the utilities of three forking strategies against the default strategy

fork of any length, and to do a fork at the beginning of any chain of blocks she does not own. □

Define $\mathbf{G}_\ell^k$ as the combined strategy $(\mathrm{DF}_0, \mathrm{G}_\ell^k)$. To compute an analytical form for $u_1(\mathbf{G}_\ell^k)$, we proceed as in the proof of Theorem 4.2. In this case, the set of paths leading to winning states has a more complex combinatorial nature, as expected when taking into account the parameters k and ℓ. We obtain the following.

Theorem 4.4. *For every pair of positive integers ℓ, k such that $k < \ell$, we have that:*

$$u_1(\mathbf{G}_\ell^k) = \frac{\Phi_{\ell,k}}{1 - \Gamma_{\ell,k}},$$

where $\Phi_{\ell,k}$ and $\Gamma_{\ell,k}$ are rational functions of α, β and h (that is, divisions of polynomials of α, β and h).

In other words, given specific values of k and ℓ, we can still compute the utility of these strategies. We use this theorem to analyse these strategies, plotting them, as we did before, for $\alpha = 0.9999966993$ and $\beta = 0.9999996156$. Figures 5(a) and 5(b) give interesting information about the advantages of these strategies. We fix $k = 1$, and plot in Figure 5(a) the utilities of combined strategies **DF**, $\mathbf{G}_4^1$, $\mathbf{G}_5^1$ and **AF**. In Figure 5(b), we zoom in around the values of the hash power where **DF** intersects with $\mathbf{G}_4^1$ and $\mathbf{G}_5^1$. As we see in the figures, for a fork window $k = 1$, the optimal amount time player 1 should be willing to fight for a branch before giving up depends on the hash power. With little hash power the likelihood of winning a branch is small, so player 1 should give up as early as possible. However, the more hash power she obtains, the better it is to wait more. Interestingly, with more than 46.7% of the hash power, player 1 already should start using strategy G_4^1 to defeat the default strategy, and with more than 48.6% hash power, she should adopt G_5^1. We know that player 1 should use AF not before around $h = 0.499805$, so this gives us a lot of extra room to look for optimal strategies if we are willing to fork (especially considering that every percentage of hash power in popular cryptocurrencies may cost millions).

Plots for strategies with $k > 1$ present a similar behaviour: the more hash power we have, the more we should be willing to fight for our forks. However, we decided to include only combined strategies $\mathbf{G}_4^1$ and $\mathbf{G}_5^1$ in Figures 5(a) and 5(b), as we have found that G_4^1 is the strategy that beats the default strategy under the least amount of hash power amongst any combination of values for k and ℓ with $k < \ell \leq 100$. The comparison is much less straightforward when looking at varying values of both k and ℓ, but in general, the more hash power the bigger the window of blocks one should aim to do a fork, and the more time one should wait before giving up. We remark that the analysis we can do through 2- or 3-dimensional plots is only one part of the picture, and that a game-theoretical analysis of these strategies is an interesting area for future work.

5 CONCLUDING REMARKS

Our model of mining via a stochastic game allows for an intuitive representation of miners' actions as strategies, and gives us a way of understanding the incentive of miners. One of the advantages of our model is its generality: it can be adapted to specify more complex actions, study other forms of reward and include cooperation between miners. We are currently looking at strategies that involve withholding a mined block to the rest of the network, for which we just needed a slight extension of the notion of actions. We would also like to study incentives under different models of cooperation between miners, and also other forms of equilibria in a dynamic setting. Lastly, we are interested in studying how to modify the Bitcoin protocol in order to ensure that default behaviour is a stationary equilibirum for any hash-power repartition.

REFERENCES

[1] Nicola Atzei, Massimo Bartoletti, Stefano Lande, and Roberto Zunino. 2017. A formal model of Bitcoin transactions. *IACR Cryptology ePrint Archive* 2017 (2017), 1124.

[2] Lear Bahack. 2013. Theoretical Bitcoin Attacks with less than Half of the Computational Power (draft). (2013). http://arxiv.org/abs/1312.7013

[3] Bruno Biais, Christophe Bisiere, Matthieu Bouvard, and Catherine Casamatta. 2018. The blockchain folk theorem. (2018).

[4] Miles Carlsten, Harry Kalodner, S. Matthew Weinberg, and Arvind Narayanan. 2016. On the Instability of Bitcoin Without the Block Reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*.

[5] Mauro Conti, Sandeep Kumar, Chhagan Lal, and Sushmita Ruj. 2018. A survey on security and privacy issues of bitcoin. *IEEE Communications Surveys & Tutorials* (2018).

[6] Swapnil Dhamal, Tijani Chahed, Walid Ben-Ameur, Eitan Altman, Albert Sunny, and Sudheer Poojary. 2018. A Stochastic Game Framework for Analyzing Computational Investment Strategies in Distributed Computing with Application to Blockchain Mining. *arXiv preprint arXiv:1809.03143* (2018).

[7] Ittay Eyal and Emin Gün Sirer. 2014. Majority Is Not Enough: Bitcoin Mining Is Vulnerable. In *Financial Cryptography and Data Security*.

[8] GNU. 2018. The GNU Multiple Precision Arithmetic Library (v. 6.1.2). (2018). https://gmplib.org/

[9] Ethan Heilman. 2014. One Weird Trick to Stop Selfish Miners: Fresh Bitcoins, A Solution for the Honest Miner (Poster Abstract). In *Financial Cryptography and Data Security*.

[10] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. 2015. Eclipse Attacks on Bitcoin's Peer-to-Peer Network. In *24th USENIX Security Symposium, USENIX Security 15*. 129–144.

[11] Benjamin Johnson, Aron Laszka, Jens Grossklags, Marie Vasek, and Tyler Moore. 2014. Game-Theoretic Analysis of DDoS Attacks Against Bitcoin Mining Pools. In *Financial Cryptography and Data Security*.

[12] Aggelos Kiayias, Elias Koutsoupias, Maria Kyropoulou, and Yiannis Tselekounis. 2016. Blockchain Mining Games. In *Proceedings of the 2016 ACM Conference on Economics and Computation (EC '16)*. 365–382.

[13] Joshua A. Kroll, Ian C. Davey, and Edward W. Felten. 2013. The Economics of Bitcoin Mining, or Bitcoin in the Presence of Adversaries. In *The Twelfth Workshop on the Economics of Information Security (WEIS 2013)*.

[14] Aron Laszka, Benjamin Johnson, and Jens Grossklags. 2015. When Bitcoin Mining Pools Run Dry - A Game-Theoretic Analysis of the Long-Term Impact of Attacks Between Mining Pools. In *Financial Cryptography Workshops (Lecture Notes in Computer Science)*, Vol. 8976. 63–77.

[15] Yoad Lewenberg, Yoram Bachrach, Yonatan Sompolinsky, Aviv Zohar, and Jeffrey S. Rosenschein. 2015. Bitcoin Mining Pools: A Cooperative Game Theoretic Analysis. In *AAMAS*. ACM, 919–927.

[16] Yuval Marcus, Ethan Heilman, and Sharon Goldberg. 2018. Low-Resource Eclipse Attacks on Ethereum's Peer-to-Peer Network. *IACR Cryptology ePrint Archive* 2018 (2018), 236.

[17] Michael Mitzenmacher and Eli Upfal. 2005. *Probability and computing - randomized algorithms and probabilistic analysis*. Cambridge University Press.

[18] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf.

[19] Arvind Narayanan, Joseph Bonneau, Edward W. Felten, Andrew Miller, and Steven Goldfeder. 2016. *Bitcoin and Cryptocurrency Technologies - A Comprehensive Introduction*. Princeton University Press.

[20] Arvind Narayanan and Jeremy Clark. 2017. Bitcoin's academic pedigree. *Commun. ACM* 60, 12 (2017), 36–45.

[21] Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine Shi. 2016. Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016*. 305–320.

[22] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. 2017. Optimal Selfish Mining Strategies in Bitcoin. In *Financial Cryptography and Data Security*. 515–532.

[23] Alexander Spiegelman, Idit Keidar, and Moshe Tennenholtz. 2018. Game of Coins. *arXiv preprint arXiv:1805.08979* (2018).

[24] R.P. Stanley. 2015. *Catalan Numbers*. Cambridge University Press. 2–24 pages.

[25] I. Stewart, D. Ilie, Alexei Zamyatin, Sam Werner, M. F. Torshizi, and William J. Knottenbelt. 2018. Committing to Quantum Resistance: A Slow Defence for Bitcoin against a Fast Quantum Computing Attack. *IACR Cryptology ePrint Archive* 2018 (2018), 213.

[26] Christopher P Thompson. 2017. *Ethereum - Distributed Consensus (A Concise Ethereum History Book)*. CreateSpace Independent Publishing Platform.

[27] Marie Vasek, Micah Thornton, and Tyler Moore. 2014. Empirical Analysis of Denial-of-Service Attacks in the Bitcoin Ecosystem. In *Financial Cryptography and Data Security*.

[28] Blockchain.com Website. 2018. Bitcoin total mining revenue. https://www.blockchain.com/charts/miners-revenue.

[29] Blockchain.com Website. 2018. Bitcoin total transaction fees. https://www.blockchain.com/charts/transaction-fees-usd.

[30] Bitcoin Cash Website. 2018. https://www.bitcoincash.org.

[31] Ethereum Website. 2018. https://ethereum.org.

[32] Litecoin Website. 2018. https://litecoin.org.

[33] Monero Website. 2018. https://getmonero.org.

[34] Yevhen Zolotavkin, Julián García, and Carsten Rudolph. 2017. Incentive Compatibility of Pay Per Last N Shares in Bitcoin Mining Pools. In *GameSec (Lecture Notes in Computer Science)*, Vol. 10575. 21–39.

A PROOFS AND INTERMEDIATE RESULTS

A.1 Convergence of the utility function

To ensure that the utility function $u_p(\mathbf{s} \mid q_0)$ is well defined, we impose the restriction that for every payoff function $\mathbf{R} = (r_0, \ldots, r_{m-1})$, there exists a polynomial P such that $|r_p(q)| \leq P(|q|)$ for every player $p \in \mathbf{P}$ and state $q \in \mathbf{Q}$. In this section, we prove that this is indeed a sufficient condition for $u_p(\mathbf{s} \mid q_0)$ to be a real number, for which we first need a technical lemma.

Lemma A.1. *Let $q_0 \in \mathbf{Q}$ and $\mathbf{s}$ be a combined strategy. Then for every $k \geq 0$, it holds that*

$$\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=k}} \mathbf{Pr}^{\mathbf{s}}(q \mid q_0) \quad = \quad 1.$$

Proof. We prove the lemma by induction on k. For $k = 0$ the property trivially holds since $\mathbf{Pr}^{\mathbf{s}}(q_0 \mid q_0) = 1$. Thus, assuming that the property holds for k, we need to prove that it holds for $k + 1$. We have that:

$$\begin{aligned}
&\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=k+1}} \mathbf{Pr}^{\mathbf{s}}(q \mid q_0) \ = \\
&\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=k+1}} \left(\sum_{\substack{q' \in \mathbf{Q}: \\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0) \cdot \mathbf{Pr}(q', \mathbf{s}(q'), q) \right) \ = \\
&\sum_{\substack{q' \in \mathbf{Q}: \\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k}} \left(\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=k+1}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0) \cdot \mathbf{Pr}(q', \mathbf{s}(q'), q) \right) \ = \\
&\sum_{\substack{q' \in \mathbf{Q}: \\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0) \cdot \left(\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=k+1}} \mathbf{Pr}(q', \mathbf{s}(q'), q) \right) \ = \\
&\sum_{\substack{q' \in \mathbf{Q}: \\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0) \cdot \left(\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q,\ |q|-|q_0|=k+1,\ \mathbf{s}(q')=(a_0,\ldots,a_{m-1}) \text{ and} \\ \text{there exists } p \in \{0,\ldots,m-1\} \text{ such that } q=a_p(q')}} \mathbf{Pr}(q', \mathbf{s}(q'), q) \right) \ = \\
&\sum_{\substack{q' \in \mathbf{Q}: \\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0) \cdot \left(\sum_{\substack{p \in \{0,\ldots,m-1\}: \\ \mathbf{s}(q)=(a_0,\ldots,a_{m-1})}} \mathbf{Pr}(q', \mathbf{s}(q'), a_p(q')) \right) \ = \\
&\sum_{\substack{q' \in \mathbf{Q}: \\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0).
\end{aligned}$$

Hence, given that

$$\sum_{\substack{q' \in \mathbf{Q}: \\ q_0 \subseteq q' \text{ and } |q'|-|q_0|=k}} \mathbf{Pr}^{\mathbf{s}}(q' \mid q_0) \quad = \quad 1$$

by induction hypothesis, we conclude that

$$\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=k+1}} \mathbf{Pr}^{\mathbf{s}}(q \mid q_0) \quad = \quad 1.$$

□

Proposition A.2. *Let $p \in \{0, \ldots, m-1\}$, $q_0 \in \mathbf{Q}$ and $\mathbf{s}$ be a combined strategy. If there exist a polynomial P such that $|r_p(q)| \leq P(|q|)$ for every $q \in \mathbf{Q}$, then $u_p(\mathbf{s} \mid q_0)$ is a real number.*

Proof. Notice that if P is a zero polynomial, then the property trivially holds. Thus, we assume that P is a nonzero polynomial. Then we have that:

$$u_p(\mathbf{s} \mid q_0) \quad = \quad (1-\beta) \cdot \sum_{q \in \mathbf{Q} \,:\, q_0 \subseteq q} \beta^{|q|-|q_0|} \cdot r_p(q) \cdot \mathbf{Pr}^{\mathbf{s}}(q \mid q_0)$$

$$= \quad (1-\beta) \cdot \sum_{n=0}^{\infty} \Bigg(\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} \beta^{|q|-|q_0|} \cdot r_p(q) \cdot \mathbf{Pr^s}(q \mid q_0) \Bigg)$$

$$= \quad (1-\beta) \cdot \sum_{n=0}^{\infty} \beta^n \cdot \Bigg(\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} r_p(q) \cdot \mathbf{Pr^s}(q \mid q_0) \Bigg). \tag{1}$$

Let $f : \mathbb{N} \to \mathbb{R}$ be a function defined as:

$$f(n) \quad = \quad \sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} r_p(q) \cdot \mathbf{Pr^s}(q \mid q_0).$$

Notice that this function is well-defined as there exists a finite number of states $q \in \mathbf{Q}$ such that $|q| - |q_0| = n$. Then by equation (1), we have that:

$$u_p(\mathbf{s} \mid q_0) \quad = \quad (1-\beta) \cdot \sum_{n=0}^{\infty} \beta^n \cdot f(n).$$

Therefore, to show that $u_p(\mathbf{s} \mid q_0)$ is a real number, we need to show that the series $\sum_{n=0}^{\infty} \beta^n \cdot f(n)$ converges, for which we prove that the series $\sum_{n=0}^{\infty} |\beta^n \cdot f(n)|$ converges (that is, we show that $\sum_{n=0}^{\infty} \beta^n \cdot f(n)$ converges absolutely, which is known to imply that this series is convergent). By definition of function f, we have that:

$$\begin{aligned}
|f(n)| \quad &= \quad \Bigg| \sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} r_p(q) \cdot \mathbf{Pr^s}(q \mid q_0) \Bigg| \\
&\leq \quad \sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} |r_p(q)| \cdot \mathbf{Pr^s}(q \mid q_0) \\
&\leq \quad \sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} P(n) \cdot \mathbf{Pr^s}(q \mid q_0) \\
&= \quad P(n) \cdot \Bigg(\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} \mathbf{Pr^s}(q \mid q_0) \Bigg).
\end{aligned} \tag{2}$$

We have by Lemma A.1 that

$$\sum_{\substack{q \in \mathbf{Q}: \\ q_0 \subseteq q \text{ and } |q|-|q_0|=n}} \mathbf{Pr^s}(q \mid q_0) \quad = \quad 1.$$

Hence, we conclude by equation (2) that:

$$|f(n)| \quad \leq \quad P(n).$$

Thus, we have that:

$$\sum_{n=0}^{\infty} |\beta^n \cdot f(n)| \;=\; \sum_{n=0}^{\infty} \beta^n \cdot |f(n)| \;\leq\; \sum_{n=0}^{\infty} \beta^n \cdot P(n). \tag{3}$$

Given that every term in the series $\sum_{n=0}^{\infty} |\beta^n \cdot f(n)|$ is non-negative, to show that this series converges it is enough to prove that it is bound by a (non-negative) real number. Thus, by equation (3), to finish the proof we need to show that the series $\sum_{n=0}^{\infty} \beta^n \cdot P(n)$ converges. By this can be easily established by using the Ratio Test, as we have that $\beta \in (0,1)$ and

$$\lim_{n \to \infty} \frac{\beta^{n+1} \cdot P(n+1)}{\beta^n \cdot P(n)} \;=\; \beta \cdot \lim_{n \to \infty} \frac{P(n+1)}{P(n)} \;=\; \beta,$$

since $\lim_{n \to \infty} \frac{P(n+1)}{P(n)} = 1$ as P is a nonzero polynomial. This concludes the proof of the proposition. □

A.2 Proof of Proposition 2.3

We have that:

$$\begin{aligned}
u_p(\mathbf{s}) &= (1-\beta)\cdot \sum_{q\in \mathbf{Q}\,:\, b\in q} \beta^{|q|-1}\cdot r_p(b,q)\cdot \mathbf{Pr}^{\mathbf{s}}(q)\\
&\leq (1-\beta)\cdot \sum_{q\in \mathbf{Q}\,:\, b\in q} \beta^{|q|-1}\cdot M_p(b)\cdot \mathbf{Pr}^{\mathbf{s}}(q)\\
&= (1-\beta)\cdot M_p(b)\cdot \sum_{q\in \mathbf{Q}\,:\, b\in q} \beta^{|q|-1}\cdot \mathbf{Pr}^{\mathbf{s}}(q)\\
&= (1-\beta)\cdot M_p(b)\cdot \sum_{i=|b|+1}^{\infty}\Bigg(\sum_{q\in \mathbf{Q}\,:\, b\in q \text{ and } |q|=i} \beta^{|q|-1}\cdot \mathbf{Pr}^{\mathbf{s}}(q)\Bigg)\\
&= (1-\beta)\cdot M_p(b)\cdot \sum_{i=|b|+1}^{\infty}\Bigg(\beta^{i-1}\cdot\sum_{q\in \mathbf{Q}\,:\, b\in q \text{ and } |q|=i} \mathbf{Pr}^{\mathbf{s}}(q)\Bigg)\\
&\leq (1-\beta)\cdot M_p(b)\cdot \sum_{i=|b|+1}^{\infty}\Bigg(\beta^{i-1}\cdot\sum_{q\in \mathbf{Q}\,:\, |q|=i} \mathbf{Pr}^{\mathbf{s}}(q)\Bigg).
\end{aligned}$$

By lemma A.1, we have that $\sum_{q\in \mathbf{Q}\,:\,|q|=i} \mathbf{Pr}^{\mathbf{s}}(q) = 1$. Hence, we conclude that:

$$\begin{aligned}
u_p(\mathbf{s}) &\leq (1-\beta)\cdot M_p(b)\cdot \sum_{i=|b|+1}^{\infty}\Bigg(\beta^{i-1}\cdot\sum_{q\in \mathbf{Q}\,:\, |q|=i} \mathbf{Pr}^{\mathbf{s}}(q)\Bigg)\\
&= (1-\beta)\cdot M_p(b)\cdot \sum_{i=|b|+1}^{\infty}\beta^{i-1}\\
&= (1-\beta)\cdot \beta^{|b|}\cdot M_p(b)\cdot \sum_{i=|b|+1}^{\infty}\beta^{i-1-|b|}\\
&= (1-\beta)\cdot \beta^{|b|}\cdot M_p(b)\cdot \sum_{j=0}^{\infty}\beta^{j}\\
&= (1-\beta)\cdot \beta^{|b|}\cdot M_p(b)\cdot \frac{1}{1-\beta}\\
&= \beta^{|b|}\cdot M_p(b),
\end{aligned}$$

which was to be shown.

Proof of Claim 1

For the sake of contradiction, assume that q, q' are two distinct states in $Q_{\mathbf{S}}$ such that both $\sigma(q)$ and $\sigma(q')$ contain a state $q^* \in Q_{\mathbf{DF}}$. By definition of $Q_{\mathbf{DF}}$, there exists a block b^* such that $q^* = \{b \in \mathbf{B} \mid b \preceq b^*\}$. By definition of mapping σ, there exist a sequence $\rho = q_0, \ldots, q_n$ for q and a sequence $\rho' = q'_0, \ldots, q'_n$ for q' such that $b^* = b_\rho$ and $b^* = b_{\rho'}$. If $\rho = \rho'$, then $q = q'$ as $q = q_n$ and $q' = q'_n$. Hence, we have that $\rho \neq \rho'$. Let i be the first position where ρ and ρ' differ, so that sequences $q_0, \ldots, q_{i-1}$ and $q_0, \ldots, q'_{i-1}$ are the same and $q_i \neq q'_i$ (notice that $i \in \{1, \ldots, n\}$ since $q_0 = q'_0 = \varepsilon$). Then both q_i and q'_i are reachable from q_{i-1} in one step. Therefore, it follows that $q_i = a_{p_1}(q_{i-1})$ and $q'_i = a_{p_2}(q_{i-1})$, where $a_{p_1} = s_{p_1}(q_{i-1})$, $a_{p_2} = s_{p_2}(q_{i-1})$ and $p_1 \neq p_2$. Hence, we have that the symbols in the i-th positions of b_ρ and $b_{\rho'}$ are different, from which we conclude that $b_\rho \neq b_{\rho'}$, and reach a contradiction since $b^* = b_\rho$ and $b^* = b_{\rho'}$.

Proof of Lemma 4.1

By the definition of utility we have:

$$u_1(\mathbf{DF}) = (1-\beta)\cdot \sum_{q\in \mathbf{Q}} \beta^{|q|-1}\cdot r(q)\cdot \mathbf{Pr}^{\mathbf{DF}}(q).$$

Separating the sum by the state size, we can write:

$$u_1(\mathbf{DF}) = (1-\beta)\cdot \sum_{i=1}^{\infty}\beta^{i-1}\cdot\Bigg(\sum_{q\in \mathbf{Q}\,:\,|q|=i} r_1(q)\cdot \mathbf{Pr}^{\mathbf{DF}}(q)\Bigg).$$

By encoding each state $q \in \mathbf{Q}$ as a binary string $w \in \{0,1\}^*$ (as in the proof of Theorem 4.2) we can compute the utility as follows:

$$u_1(\mathbf{DF}) \quad = \quad (1-\beta)\cdot c\cdot\sum_{i=0}^{\infty}\beta^i\cdot\Bigg(\sum_{w\in\{0,1\}^i}\Big(\sum_{j=1}^{i} w[j]\cdot\alpha^j\Big)\cdot\mathbf{Pr}^{\mathbf{DF}}(q_w)\Bigg),$$

where $w[j]$ is the j-th symbol of the string w and $q_w = \{b \in \mathbf{B} \mid b \preceq w\}$. Notice that in the equation above, we use the fact that when playing **DF** each state contains a single blockchain (and nothing else), thus implying that for every word $w \in \{0,1\}^*$, it holds that $\text{meet}(q_w) = \text{bc}(q_w)$ and $\chi_1(b) = \text{owner}(b) = w[j]$, for every block $b \in q_w$ such that $|b| = j \geq 1$. By rearranging the order of the summation we obtain:

$$u_1(\mathbf{DF}) \quad = \quad (1-\beta)\cdot c\cdot\sum_{i=0}^{\infty}\beta^i\cdot\Bigg(\sum_{j=1}^{i}\alpha^j\cdot\Big(\sum_{w\in\{0,1\}^i} w[j]\cdot\mathbf{Pr}^{\mathbf{DF}}(q_w)\Big)\Bigg)$$

Using the fact that that mining any block for player 1 is an independent Bernoulli trial with probability of success h, and the fact that $\mathbf{Pr}^{\mathbf{DF}}(\{q_w \mid w \in \{0,1\}^i \text{ and } w[j]=1\}) = h$ and $\mathbf{Pr}^{\mathbf{DF}}(\{q_w \mid w \in \{0,1\}^i \text{ and } w[j]=0\}) = (1-h)$, for all $i \geq 1$ and $j \in \{1,\dots,i\}$, we can conclude that $\sum_{w\in\{0,1\}^i} w[j]\cdot\mathbf{Pr}^{\mathbf{DF}}(q_w) = \mathbf{E}(w[j]) = h$, thus yielding:

$$u_1(\mathbf{DF}) \;=\; (1-\beta)\cdot c\cdot\sum_{i=0}^{\infty}\beta^i\cdot\Big(\sum_{j=1}^{i}\alpha^j\cdot\mathbf{E}(w[j])\Big) \;=\; (1-\beta)\cdot c\cdot h\cdot\sum_{i=0}^{\infty}\beta^i\cdot\Big(\sum_{j=1}^{i}\alpha^j\Big).$$

Computing the final summation, we get:

$$\begin{aligned} u_1(\mathbf{DF}) \quad &= \quad (1-\beta)\cdot c\cdot h\cdot\sum_{i=0}^{\infty}\beta^i\cdot\frac{\alpha\cdot(1-\alpha^i)}{1-\alpha} \\ &= \quad (1-\beta)\cdot c\cdot h\cdot\frac{\alpha}{1-\alpha}\cdot\Big(\sum_{i=0}^{\infty}\beta^i-\sum_{i=0}^{\infty}(\alpha\cdot\beta)^i\Big). \end{aligned}$$

Using the fact that $\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$ for $x \in (0,1)$, we obtain the desired result:

$$u_1(\mathbf{DF}) \quad = \quad h\cdot c\cdot\frac{\alpha\cdot\beta}{(1-\alpha\cdot\beta)}.$$

A.3 Proof of Theorem 4.2

Let $Q_{\mathbf{AF}} = \{q \in \mathbf{Q} \mid \mathbf{Pr}^{\mathbf{AF}}(q) > 0\}$ be the set of all states that can be reached from the genesis block using the strategy **AF**, and from the proof of Theorem 3.1 recall the definition of sequence ρ for a state q, and recall the construction of string b_ρ from such a sequence ρ. By using these elements, we define $\tau : Q_{\mathbf{AF}} \mapsto 2^{\{0,1\}^*}$ as follows:

$$\tau(q) \quad = \quad \{b_\rho \mid \rho \text{ is a sequence for } q\}.$$

Intuitively, $\tau(q)$ is the set of all moves that players 0 and 1 can do in $|q|-1$ steps according to **AF** that lead them to the state q when starting in the genesis block. As such, they are coded as sequences of zeros and ones that tell us which player puts a block at the stage i of the game, for $i \in \{1,\dots,|q|-1\}$. It is straightforward to verify the following:

Claim 3. *For every $q, q' \in Q_{\mathbf{AF}}$, it holds that:*

(a) *If $q \neq q'$, then $\tau(q)$ is disjoint from $\tau(q')$.*
(b) $\mathbf{Pr}^{\mathbf{AF}}(q) = \sum_{w\in\tau(q)}\mathbf{Pr}(w)$*, where* $\mathbf{Pr}(w)$ *for a word w with n_0 zeroes and n_1 ones is defined as $h^{n_1}(1-h)^{n_0}$.*

In particular, Claim 3 (a) can be proved exactly in the same way Claim 1 is proved. Notice that Claim 3 (a) tells us that a sequence of actions of players 0 and 1 uniquely determines a state of the game. Moreover, Claim 3 (b) tells us that the probability of a state q is the sum of probabilities of all the sequences of actions of players 0 and 1 that end up in q when started in the genesis block. Observe that since the actions of players 0 and 1 are independent trials, with probabilities $1-h$ and h, respectively, the probability of a state where player 0 wins n_0 rounds and player 1 wins n_1 rounds is $h^{n_1}(1-h)^{n_0}$, as stated in the claim.

For every $w \in \{0,1\}^*$, there exists a unique state $q \in Q_{\mathbf{AF}}$ such that $w \in \tau(q)$. Given Claim 2 (a), to prove this claim we only need to prove the existence of such a state q. If $w = \varepsilon$, then $q = \{\varepsilon\}$. On the other hand, if $w = p_1\cdots p_n$ with $n \geq 1$ and each $p_i \in \{0,1\}$, then $q = q_n$ in a sequence $q_0,\dots,q_n$ of states defined by the rules: (1) $q_0 = \varepsilon$; and (2) for every $i \in \{1,\dots,n\}$, it holds that $q_i = a_i(q_{i-1})$, where $a_i = \text{DF}_0(q_{i-1})$ if $p_i = 0$, and $a_i = \text{AF}(q_{i-1})$ if $p_i = 1$. Thus, we conclude that the utility of player 1 can be rewritten as follows:

$$\begin{aligned} u_1(\mathbf{AF}) \quad &= \quad (1-\beta)\cdot\sum_{q\in\mathbf{Q}}\beta^{|q|-1}\cdot r_1(q)\cdot\mathbf{Pr}^{\mathbf{AF}}(q) \\ &= \quad (1-\beta)\cdot\sum_{q\in\mathbf{Q}_{\mathbf{AF}}}\beta^{|q|-1}\cdot r_1(q)\cdot\mathbf{Pr}^{\mathbf{AF}}(q) \end{aligned}$$

$$\begin{tikzcd}
\varepsilon \arrow[r] & 1
\end{tikzcd}$$

(a)

$$\begin{tikzcd}
& 1 \arrow[r] & 11 \\
\varepsilon \arrow[ur] \arrow[dr] & & \\
& 0 &
\end{tikzcd}$$

(b)

$$\begin{tikzcd}
& 1 \arrow[r] & 11 \arrow[r] & 111 \\
\varepsilon \arrow[ur] \arrow[dr] & & & \\
& 0 \arrow[r] & 00 &
\end{tikzcd}$$

(c)

Figure 6: States in a game played according to strategy AF.

$$\begin{aligned}
&= (1-\beta) \cdot \sum_{q \in \mathbf{Q_{AF}}} \beta^{|q|-1} \cdot r_1(q) \cdot \Bigg(\sum_{w \in \tau(q)} \mathbf{Pr}(w) \Bigg) \\
&= (1-\beta) \cdot \sum_{q \in \mathbf{Q_{AF}}} \sum_{w \in \tau(q)} \beta^{|q|-1} \cdot r_1(q) \cdot \mathbf{Pr}(w) \\
&= (1-\beta) \cdot \sum_{q \in \mathbf{Q_{AF}}} \sum_{w \in \tau(q)} \beta^{|w|} \cdot r_1(w) \cdot \mathbf{Pr}(w) \\
&= (1-\beta) \cdot \sum_{w \in \{0,1\}^*} \beta^{|w|} \cdot r_1(w) \cdot \mathbf{Pr}(w),
\end{aligned}$$

given that $|w| = |q| - 1$ for every $w \in \tau(q)$, and assuming that $r_1(w)$ is defined as $r_1(q)$ for the only state q such that $w \in \tau(q)$.

We now describe all the states in which player 1 receives a non-zero reward in terms of words. For this, let us consider the set S of all words $w \in \{0,1\}^*$ that represent states q (via τ) in which player 1 owns at least one block in the blockchain for the *first time*. The smallest of them is $w = 1$, which represents the state in Figure 6 (a). This state is created when player q wins the first move of the game, successfully mining upon the genesis block. Next is the word 011, representing the state in Figure 6 (b). To arrive at this state player 0 must have mined the first block, player 1 forked, and then player 1 won the following block (on her forking branch). The next words in S are 00111 and 01011, both representing the state in Figure 6 (c). In general, the words in the set S have the form $d \cdot 1$, where d is a *Dyck word* [24]: a word with the same number of 0s and 1s, but such that no prefix of d has more 1s than 0s (this intuitively means that at no point player 1 has more blocks than player 0). Note that the only Dyck word of length 0 is ε, the next Dyck word by length is 01, and then 0011 and 0101, etc. As it turns out, the number of Dyck words of length $2m$ is the m-th Catalan number [24]. We use $\mathcal{D}$ to denote the set of all Dyck words. Notice that by definition all elements of $\mathcal{D}$ are of even length.

Since all states where player 1 receives a reward involve putting a block in the blockchain, all words w with $r_1(w) > 0$ are therefore of the form $d \cdot 1 \cdot w'$ with $d \in \mathcal{D}$. Now let q be the only state such that $d \cdot 1 \cdot w' \in \tau(q)$. State q can be seen as a tree with two branches: one only with blocks earned by player 0, and the other one with at least $\frac{|d|}{2} + 1$ blocks owned by player 1 (plus maybe more, depending on w'). We can then calculate the reward for q as:

$$r_1(q) \quad = \quad r_1(d \cdot 1) + \alpha^{\frac{|d|}{2}+1} \cdot r_1(w').$$

Hence, we obtain $u_1(\mathbf{AF})$ is equal to:

$$(1-\beta) \cdot \sum_{d \in \mathcal{D}} \sum_{w \in \{0,1\}^*} \beta^{|d|+1+|w|} \cdot \left[r_1(d \cdot 1) + \alpha^{\frac{|d|}{2}+1} \cdot r_1(w) \right] \cdot \mathbf{Pr}(d \cdot 1 \cdot w).$$

Splitting up the summation we get that $u_1(\mathbf{AF})$ is equal to:

$$(1-\beta) \cdot \sum_{d \in \mathcal{D}} \sum_{w \in \{0,1\}^*} \beta^{|d|+1+|w|} \cdot r_1(d \cdot 1) \cdot \mathbf{Pr}(d \cdot 1 \cdot w) + (1-\beta) \cdot \sum_{d \in \mathcal{D}} \sum_{w \in \{0,1\}^*} \beta^{|d|+1+|w|} \cdot \alpha^{\frac{|d|}{2}+1} \cdot r_1(w) \cdot \mathbf{Pr}(d \cdot 1 \cdot w).$$

We denote the first term in the equation above by Φ. By definition of the probability of a word, we have that $\mathbf{Pr}(d \cdot 1 \cdot w) = \mathbf{Pr}(d \cdot 1) \cdot \mathbf{Pr}(w)$. Next, we use this fact in the expression for $u_1(\mathbf{AF})$ to split the second term into the elements that depend only on d, and the ones that depend only on w:

$$u_1(\mathbf{AF}) \quad = \quad \Phi + \Bigg(\sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot \alpha^{\frac{|d|}{2}+1} \cdot \mathbf{Pr}(d \cdot 1) \Bigg) \cdot \Bigg((1-\beta) \cdot \sum_{w \in \{0,1\}^*} \beta^{|w|} \cdot r_1(w) \cdot \mathbf{Pr}(w) \Bigg).$$

Since the term $(1-\beta) \cdot \sum_{w \in \{0,1\}^*} \beta^{|w|} \cdot r_1(w) \cdot \mathbf{Pr}(w)$ is precisely $u_1(\mathbf{AF})$, we have that:

$$u_1(\mathbf{AF}) \quad = \quad \Phi + \Bigg(\sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot \alpha^{\frac{|d|}{2}+1} \cdot \mathbf{Pr}(d \cdot 1) \Bigg) \cdot u_1(\mathbf{AF}).$$

By denoting with Γ the term $\sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot \alpha^{\frac{|d|}{2}+1} \cdot \mathbf{Pr}(d \cdot 1)$, we get the equation:

$$u_1(\mathbf{AF}) \quad = \quad \frac{\Phi}{1-\Gamma}.$$

Let us now find a closed form for Γ and Φ, starting with Γ. In what follows, we use $\mathcal{D}_{2\ell}$ to denote the set of all Dyck words of length 2ℓ (recall that all Dyck words are of even length):

$$\begin{aligned}
\Gamma &= \sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot \alpha^{\frac{|d|}{2}+1} \cdot \mathbf{Pr}(d \cdot 1) \\
&= \alpha \cdot \beta \cdot \sum_{d \in \mathcal{D}} \beta^{|d|} \cdot \alpha^{\frac{|d|}{2}} \cdot \mathbf{Pr}(d \cdot 1) \\
&= \alpha \cdot \beta \cdot \sum_{\ell=0}^{\infty} \sum_{d \in \mathcal{D}_{2\ell}} (\alpha \cdot \beta^2)^{\ell} \cdot h^{\ell} \cdot (1-h)^{\ell} \cdot h \\
&= \alpha \cdot \beta \cdot \sum_{\ell=0}^{\infty} |\mathcal{D}_{2\ell}| \cdot (\alpha \cdot \beta^2)^{\ell} \cdot h^{\ell} \cdot (1-h)^{\ell} \cdot h \\
&= \alpha \cdot \beta \cdot h \cdot \sum_{\ell=0}^{\infty} |\mathcal{D}_{2\ell}| \cdot (\alpha \cdot \beta^2 \cdot h \cdot (1-h))^{\ell} \\
&= \alpha \cdot \beta \cdot h \cdot \mathbf{C}(\alpha \cdot \beta^2 \cdot h \cdot (1-h)).
\end{aligned}$$

The final equality is obtained by recalling the fact that $|\mathcal{D}_{2\ell}|$ is the ℓ-th Catalan number, so that the summation in the previous line defines the generating function of these numbers. Notice that function $\mathbf{C}(x)$ is defined and continuous for $x \in (0, \frac{1}{4}]$, and that $\alpha \cdot \beta^2 \cdot h \cdot (1-h) \in (0, \frac{1}{4}]$ since $\alpha \in (0, 1]$, $\beta \in (0, 1)$ and $h \cdot (1-h) \in (0, \frac{1}{4})$ for every $h \in (0, 1)$.

Finally, we compute a closed form for Φ. First, recall that:

$$\begin{aligned}
\Phi &= (1-\beta) \cdot \sum_{d \in \mathcal{D}} \sum_{w \in \{0,1\}^*} \beta^{|d|+1+|w|} \cdot r_1(d \cdot 1) \cdot \mathbf{Pr}(d \cdot 1 \cdot w) \\
&= (1-\beta) \cdot \sum_{d \in \mathcal{D}} \sum_{w \in \{0,1\}^*} \beta^{|d|+1+|w|} \cdot r_1(d \cdot 1) \cdot \mathbf{Pr}(d \cdot 1) \cdot \mathbf{Pr}(w)
\end{aligned}$$

Splitting the part that depends on d and the part that depends on w, we get:

$$\Phi \quad = \quad (1-\beta) \cdot \Bigg(\sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot r_1(d \cdot 1) \cdot \mathbf{Pr}(d \cdot 1) \Bigg) \cdot \Bigg(\sum_{w \in \{0,1\}^*} \beta^{|w|} \cdot \mathbf{Pr}(w) \Bigg).$$

To calculate $\sum_{w \in \{0,1\}^*} \beta^{|w|} \cdot \mathbf{Pr}(w)$, observe that for all w of some fixed length ℓ, we are adding only a single factor $\beta^{|w|}$ to the entire sum, or more formally, $\sum_{w \in \{0,1\}^{\ell}} \beta^{|w|} \cdot \mathbf{Pr}(w) = \beta^{\ell}$. Therefore:

$$\begin{aligned}
\Phi &= (1-\beta) \cdot \Bigg(\sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot r_1(d \cdot 1) \cdot \mathbf{Pr}(d \cdot 1) \Bigg) \cdot \Bigg(\sum_{\ell=0}^{\infty} \beta^{\ell} \Bigg) \\
&= (1-\beta) \cdot \Bigg(\sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot r_1(d \cdot 1) \cdot \mathbf{Pr}(d \cdot 1) \Bigg) \cdot \Bigg(\frac{1}{1-\beta} \Bigg) \\
&= \sum_{d \in \mathcal{D}} \beta^{|d|+1} \cdot r_1(d \cdot 1) \cdot \mathbf{Pr}(d \cdot 1).
\end{aligned}$$

Calculating $\mathbf{Pr}(d \cdot 1)$ and removing the extra β factor, we now get:

$$\Phi \quad = \quad \beta \cdot \sum_{d \in \mathcal{D}} \beta^{|d|} \cdot h^{\frac{|d|}{2}+1} \cdot (1-h)^{\frac{|d|}{2}} \cdot r_1(d \cdot 1).$$

By calculating $r_1(d \cdot 1)$ explicitly, we obtain:

$$\Phi \quad = \quad \beta \cdot \sum_{d \in \mathcal{D}} \beta^{|d|} \cdot h^{\frac{|d|}{2}+1} \cdot (1-h)^{\frac{|d|}{2}} \cdot \Bigg(\sum_{i=1}^{\frac{|d|}{2}+1} \alpha^i \cdot c \Bigg).$$

By representing all Dyck words via their lengths, we obtain:

$$\begin{aligned}\Phi &= \beta \cdot \sum_{\ell=0}^{\infty} \sum_{d \in \mathcal{D}_{2\ell}} \beta^{2\ell} \cdot h^{\ell+1} \cdot (1-h)^{\ell} \cdot \left(\sum_{i=1}^{\ell+1} \alpha^{i} \cdot c \right) \\ &= \alpha \cdot \beta \cdot h \cdot c \cdot \sum_{\ell=0}^{\infty} \sum_{d \in \mathcal{D}_{2\ell}} (\beta^2 \cdot h \cdot (1-h))^{\ell} \cdot \left(\sum_{i=0}^{\ell} \alpha^{i} \right).\end{aligned}$$

Considering that $\sum_{i=0}^{\ell} \alpha^i = \frac{1-\alpha^{\ell+1}}{1-\alpha}$, we obtain:

$$\Phi = \alpha \cdot \beta \cdot h \cdot c \cdot \sum_{\ell=0}^{\infty} \sum_{d \in \mathcal{D}_{2\ell}} (\beta^2 \cdot h \cdot (1-h))^{\ell} \cdot \left(\frac{1-\alpha^{\ell+1}}{1-\alpha} \right).$$

Since none of the terms of the summation depends on the specific word d, we get:

$$\Phi = \frac{\alpha \cdot \beta \cdot h \cdot c}{(1-\alpha)} \cdot \sum_{\ell=0}^{\infty} |\mathcal{D}_{2\ell}| \cdot (\beta^2 \cdot h \cdot (1-h))^{\ell} \cdot (1-\alpha^{\ell+1}).$$

Therefore, we have that:

$$\begin{aligned}\Phi &= \frac{\alpha \cdot \beta \cdot h \cdot c}{(1-\alpha)} \cdot \left[\left(\sum_{\ell=0}^{\infty} |\mathcal{D}_{2\ell}| \cdot (\beta^2 \cdot h \cdot (1-h))^{\ell} \right) - \left(\sum_{\ell=0}^{\infty} |\mathcal{D}_{2\ell}| \cdot (\beta^2 \cdot h \cdot (1-h))^{\ell} \cdot \alpha^{\ell+1} \right) \right] \\ &= \frac{\alpha \cdot \beta \cdot h \cdot c}{(1-\alpha)} \cdot \left[\left(\sum_{\ell=0}^{\infty} |\mathcal{D}_{2\ell}| \cdot (\beta^2 \cdot h \cdot (1-h))^{\ell} \right) - \left(\alpha \cdot \sum_{\ell=0}^{\infty} |\mathcal{D}_{2\ell}| \cdot (\beta^2 \cdot h \cdot (1-h) \cdot \alpha)^{\ell} \right) \right]\end{aligned}$$

Hence, by using the definition of the generating function for Catalan numbers, we finally conclude that:

$$\Phi = \frac{\alpha \cdot \beta \cdot h \cdot c}{(1-\alpha)} \cdot \left[\mathrm{C}(\beta^2 \cdot h \cdot (1-h)) - \alpha \cdot \mathrm{C}(\alpha \cdot \beta^2 \cdot h \cdot (1-h)) \right].$$

A.4 Proof of Theorem 4.4

In this section, we develop analytical expressions for the utilities when considering the strategy G_{ℓ}^{k} against a default player. In fact, it is possible to express $u(\mathrm{G}_{\ell}^{k})$ as a rational function (i.e., a quotient of two polynomials). In particular, these functions can be evaluated to any degree of precision, which allowed us to establish the results of section 4.

There are two miners (or, more generally, pools of miners), labelled Miner 0 and Miner 1. Miner 0 follows the **DF** strategy, that is, they will always mine on top of the longest chain, regardless of where their past blocks are. On the other hand, Miner 1 plays with the G_{ℓ}^{k} strategy: given a portion of at most k blocks not owned by her at the end of the blockchain, she forks at the beginning of this chain (we refer to k as the disadvantage). When forking, she establishes a give-up length ℓ such that she will give up the fork if the main branch achieves to append ℓ blocks, orphaning all mined blocks on the fork. This scenario captures the fact that, with reasonable hash power, a pool does not want to fork too far behind, and will give up the fork if it does not prove successful, before reaching a hopeless situation.

Theorem A.3. *Miner 1, with hash power h, fixes a disadvantage k, a give-up length ℓ, and plays with* G_{ℓ}^{k}. *Miner 0 plays with* **DF**. *The utility of miner 1 is given by*

$$u_1(\mathrm{G}_{\ell}^{k})(h) = \frac{\Phi_{k,\ell}(h)}{1-\Gamma_{k,\ell}(h)},$$

where $\Phi_{k,\ell}, \Gamma_{k,\ell}$ are polynomials in h (with coefficients depending on k, ℓ, α and β).

In the proof of this theorem, we develop precise expressions for $\Phi_{k,\ell}, \Gamma_{k,\ell}$. These are combinations of two sets of polynomials we call Trapezoidal and Pentagonal Dyck polynomials ($T_{n,m}, P_{n,m}$, see A.5). In what follows, let $x = \beta^2 \cdot h \cdot (1-h)$ for the ease of notation.

Proof. The game begins with both players mining on ε, and assume that Miner 0 first appends j blocks. Here, five types of states can arise:

(0) (Case $j = 0$) Miner 1 immediately appends a block b on top of ε and now both players will mine on top of b.
(a) After miner 0 appends $j \leq k$ blocks on top of ε, miner 1 appends one block on top of ε and starts the fork. Miner 1 wins the fork, gaining at most $j + \ell + 1$ blocks.
(b) After miner 0 appends $j > k$ blocks on top of ε, miner 1 appends one block at position $j - k$. Miner 1 wins the fork, gaining at most $k + \ell + 1$ blocks.
(c) After miner 0 appends $j \leq k$ blocks on top of ε, miner 1 appends one block on top of ε and starts the fork. Miner 0's branch achieves length $\ell + 1$ and miner 1 gives up.

(d) After miner 0 appends $j > k$ blocks on top of ε, miner 1 appends one block at position $j - k$. Miner 0's branch achieves length $\ell + 1$ and miner 1 gives up.

A key observation is that after one of this cases, both players will mine over the same block, and reboot the strategies as if this block were ε. This recursive behaviour allows us to compute the utility of single forks in the same spirit as in 4.2, and then use proper shifting to establish an equation in $u_1(\mathrm{G}_\ell^k)(h)$. In other words, we have

$$u_1(\mathrm{G}_\ell^k)(h) = u_0(h) + \sum_{j=1}^{k}(u_{a,j}(h) + u_{c,j}(h)) + \sum_{j=k+1}^{\infty}(u_{b,j}(h) + u_{d,j}(h)),$$

where each term corresponds to the total utility contributed by the cases above. In what follows, we develop expressions for each of these terms.

A.4.1 Case (0). Every state q in this case is of the form $1 \cdot w$, where $w \in \{0,1\}^*$, and thus the utility amounts to

$$\begin{aligned}u_0(h) &= (1-\beta) \cdot \sum_{w\in\{0,1\}^*} \beta^{1+|w|}(\alpha + \alpha \cdot r_1(w)) \cdot h \cdot \mathbf{Pr}(w).\\ &= (1-\beta)\frac{\alpha\cdot\beta\cdot h}{1-\beta} + \alpha\cdot\beta\cdot h\cdot(1-\beta)\sum_{w\in\{0,1\}^*}\beta^{|w|}r_1(w)\cdot\mathbf{Pr}(w)\\ &= \alpha\cdot\beta\cdot h + \alpha\cdot\beta\cdot h\cdot u_1(\mathrm{G}_\ell^k)(h),\end{aligned}$$

A.4.2 Case (a). These states are all of the form

$$q = 0^j \cdot 1 \cdot d \cdot 1 \cdot w, d \in \mathcal{D}_{j-1}, |d| + j \le 2\ell, w \in \{0,1\}^*,$$

where $\mathcal{D}_{j-1}$ are Dyck words with disadvantage $j - 1$.

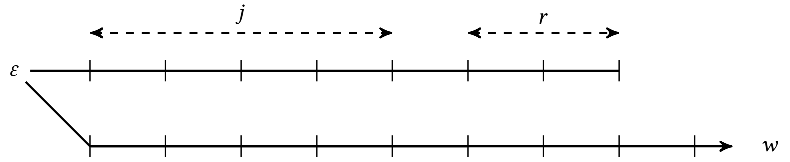


Figure 7: (Case a) Miner 1 forked at ε successfully, since $r + j \le \ell$.

We have

$$u_{a,j}(h) = (1-\beta)\cdot\sum_{w\in\{0,1\}^*, d\in\mathcal{D}_{j-1}, |d|\le 2\ell-j}\beta^{|d|+j+1+|w|}\cdot\left(\sum_{i=0}^{(|d|+j)/2}\alpha^{i+1} + \alpha^{(|d|+j)/2+1}r(w)\right)\cdot h^{(|d|+j)/2+1}\cdot(1-h)^{(|d|+j)/2}\cdot\mathbf{Pr}(w).$$

Simplifying this expression yields

$$\begin{aligned}u_{a,j}(h) = &\frac{\alpha\cdot\beta\cdot h}{1-\alpha}\cdot x^j\cdot\left(\sum_{d\in\mathcal{D}_{j-1},|d|\le 2l-j}x^{(|d|-j)/2}\cdot(1-\alpha^{(|d|+j)/2+1})\right)\\ &+\alpha\cdot\beta\cdot h\cdot(\alpha\cdot x)^j\cdot\left(\sum_{d\in\mathcal{D}_{j-1},|d|\le 2l-j}(\alpha\cdot x)^{(|d|-j)/2}\right)\cdot\left(\sum_{w\in\{0,1\}^*}\beta^{|w|}\cdot r_1(w)\cdot\mathbf{Pr}(w)\right)\\ =&\frac{\alpha\cdot\beta\cdot h}{1-\alpha}\cdot x^j(T_{j-1,\ell-j}(x) - \alpha^{j+1}\cdot T_{j-1,\ell-j}(\alpha\cdot x)) + \alpha\cdot\beta\cdot h\cdot(\alpha\cdot x)^j\cdot T_{j-1,\ell-j}(\alpha\cdot x)\cdot u_1(\mathrm{G}_\ell^k)(h),\end{aligned}$$

where $T_{n,m}$ is defined on section A.5.

A.4.3 Case (b). These states are of the form

$$q = 0^j \cdot 1 \cdot d \cdot 1 \cdot w, d \in \mathcal{D}_{k-1}, |d| + k \le 2\ell, w \in \{0,1\}^*.$$

Analogously as before, we have

$$u_{b,j}(h) = (1-\beta)\cdot\sum_{\substack{w\in\{0,1\}^*\\ d\in\mathcal{D}_{k-1}\\ |d|\le 2\ell-k}}\beta^{j+|d|+k+1+|w|}\cdot\left(\alpha^{j-k}\cdot\left(\sum_{i=0}^{(|d|+k)/2}\alpha^{i+1}\right)+\alpha^{(|d|-k)/2+j+1}\cdot r(w)\right)\cdot h^{(|d|+k)/2+1}\cdot(1-h)^{(|d|-k)/2+j}\cdot\mathbf{Pr}(w).$$

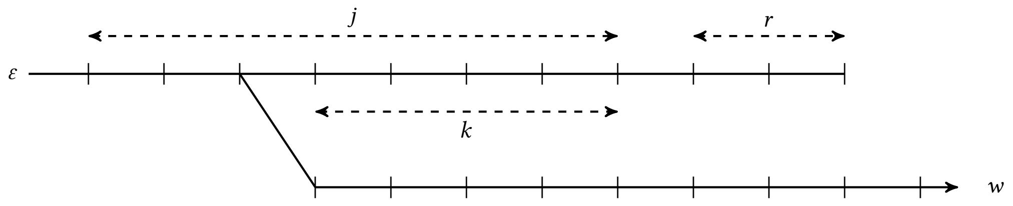


Figure 8: (Case b) Miner 1 forked successfully, since $r + k \le \ell$.

Developing and simplifying this expression as the latter case yields

$$\begin{aligned} u_{b,j}(h) = {} & \frac{\alpha \cdot \beta \cdot h}{1-\alpha} \cdot \alpha^{j-k} \cdot \beta^{j+k} \cdot h^k \cdot (1-h)^j \left(T_{k-1,\ell-k}(x) - \alpha^{k+1} \cdot T_{k-1,\ell-k}(\alpha \cdot x)\right) \\ & + \alpha \cdot \beta \cdot h \cdot \alpha^j \cdot \beta^{j+k} \cdot h^k \cdot (1-h)^j \cdot T_{k-1,\ell-k}(\alpha \cdot x) \cdot u_1(\mathrm{G}_\ell^k)(h). \end{aligned}$$

A.4.4 Case (c). These states are of the form

$$q = 0^j \cdot 1 \cdot e \cdot w, e \in \mathcal{E}_{j-1,\ell-j+1}, w \in \{0,1\}^*,$$

where $\mathcal{E}_{n,m}$ are all the binary strings with m zeroes such that in every prefix, the number of ones minus the number of zeroes does not exceed n. The number of such strings is analyzed in section A.5. Let us denote by $H(s)$ the Hamming weight of a binary string s.

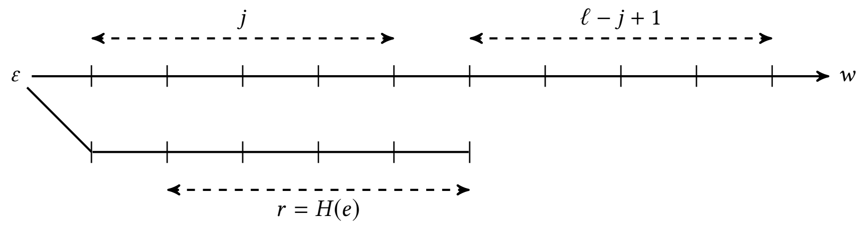


Figure 9: (Case c) Miner 1 forked at ε but gave up, since the upper branch attained ℓ blocks.

We have

$$u_{c,j}(h) = (1-\beta) \cdot \sum_{w \in \{0,1\}^*, e \in \mathcal{E}_{j-1,\ell-j-1}} \beta^{\ell+H(e)+|w|+2} \cdot \alpha^{\ell+1} \cdot r_1(w) \cdot h^{H(e)+1} \cdot (1-h)^{\ell+1} \cdot \mathbf{Pr}(w).$$

This yields the following:

$$\begin{aligned} u_{c,j}(h) &= \alpha \cdot x \cdot (\alpha \cdot \beta \cdot (1-h))^\ell \cdot u_1(\mathrm{G}_\ell^k)(h) \cdot \sum_{e \in \mathcal{E}_{j-1,\ell-j-1}} (\beta \cdot h)^{H(e)} \\ &= \alpha \cdot x \cdot (\alpha \cdot \beta \cdot (1-h))^\ell \cdot P_{\ell,j}(\beta \cdot h) \cdot u_1(\mathrm{G}_\ell^k)(h). \end{aligned}$$

A.4.5 Case (d). These states are of the form $q = 0^j \cdot 1 \cdot e \cdot w, e \in \mathcal{E}_{k-1,\ell-k+1}, w \in \{0,1\}^*$.

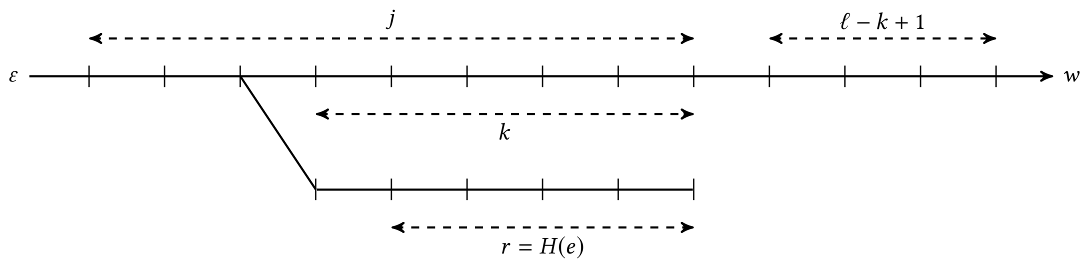


Figure 10: (Case d) Miner 1 forked at some block, but gave up, since the upper branch attained ℓ blocks.

As in case (c), we compute

$$u_{d,j}(h) = (1-\beta) \cdot \sum_{w\in\{0,1\}^*, e\in\mathcal{E}_{k-1,\ell-k-1}} \beta^{\ell+j-k+H(e)+|w|+2} \cdot \alpha^{\ell+j-k+1} \cdot r_1(w) \cdot h^{H(e)+1} \cdot (1-h)^{\ell+j-k+1} \cdot \mathbf{Pr}(w)$$
$$= \alpha \cdot x \cdot (\alpha \cdot \beta \cdot (1-h))^{j+\ell-k} \cdot P_{\ell,k}(\beta \cdot h) \cdot u_1(\mathrm{G}^k_\ell)(h).$$

Putting all of these expressions together and summing over j gives an equation for $u_1(\mathrm{G}^k_\ell)(h)$, that solves to the claimed rational function. More precisely, we have

$$u_1(\mathrm{G}^k_\ell)(h) = u_0 + \sum_{j=1}^{k}(u_{a,j} + u_{c,j}) + \sum_{j=k+1}^{\infty}(u_{b,j} + u_{d,j}) = \Phi_{k,\ell} + \Gamma_{k,\ell} \cdot u_1(\mathrm{G}^k_\ell)(h),$$

where Φ is the total contribution of one fork (successful or not) and Γ is the total shifting factor. First note that $u_{b,j} + u_{d,j}$ is a geometric sequence in j with a ratio in $[0,1)$, therefore the summation converges. In particular, $\Phi_{k,\ell}$ and $\Gamma_{k,\ell}$ are polynomials in h whose coefficients depend in α, β, ℓ, k. □

A.5 Trapezoidal and Pentagonal Dyck Polynomials

In this section, we define two sets of polynomials that control the combinatorial nature of our game. They arise as generating polynomials of the sequences of states of fixed length.

Recall that $\mathcal{D}_{a,2b}$ is the set of binary strings of length $2b + a$ such that the number of ones equals the number of zeroes plus a, and such that for every prefix, the number of ones minus the number of zeroes is at most a. Now, define $\sigma_{a,b} := |\mathcal{D}_{a,2b}|$. Interpreting each 0 as a unitary $\uparrow$ step in $\mathbb{Z}^2$ and each 1 as a unitary $\rightarrow$ step, we have that $\sigma_{a,b}$ is the total number of $\{\uparrow, \rightarrow\}$ paths from $(0,0)$ to $(a, a+b)$ that lie within the trapezoid $(0,0), (a,0), (a,a+b), (0,b)$. We enumerate them in proposition A.7.

Also, from section A.4, the set $\mathcal{E}_{a,b}$ consists of all binary strings with b zeroes such that in every prefix, the number of ones minus the number of zeroes does not exceed a. Also, define $\mathcal{E}^r_{a,b}$ the number of such strings which have exactly r ones, and define $\theta^r_{a,b} := |\mathcal{E}^r_{a,b}|$. As before, note that $\theta^r_{a,b}$ is the total number of $\{\uparrow, \rightarrow\}$ paths in $\mathbb{Z}^2$ from $(0,0)$ to (r,b) that lie within the pentagon $\{(0,0),(0,a),(r,r-a),(r,b),(0,b)\}$ (this is a rectangle if $r < a$). We develop a linear recurrence for $\theta^r_{a,b}$ that solves in terms of $\sigma_{a,b}$ in proposition A.9.

Definition A.4. *For positive integers n, m, let*

$$\begin{cases} T_{n,m}(x) &= \sum_{i=0}^{m} \sigma_{n,i} \cdot x^i \\ P_{n,m}(x) &= \sum_{i=0}^{m-1} \theta^i_{n-1,m-n+1} \cdot x^i \end{cases}$$

This polynomials arise when evaluating utility for states with two branches. These states can be enumerated by length of the fork, and sums like the following arise (recall that $\mathcal{D}_{n,2m}$ are the Dyck draws with disadvantage $n + 2m$):

$$\sum_{d\in\mathcal{D}_n} x^{(|d|-n)/2} = \sum_{i=0}^{m}\sum_{d\in\mathcal{D}_{n,2i}} x^{(|d|-n)/2} = \sum_{i=0}^{m} x^i \cdot \left(\sum_{d\in\mathcal{D}_{n,2i}} 1\right) = \sum_{i=0}^{m} x^i \cdot |\mathcal{D}_{n,2i}| = T_{n,m}(x).$$

For the sake of completeness, we compute closed forms for $\sigma_{a,b}$ and $\theta^r_{a,b}$ in the following section.

A.6 Trapezoidal and Pentagonal Dyck paths

A.6.1 Counting paths in a trapezoid. For nonnegative integers a, b, let $\mathcal{L}_{a,b}$ be the trapezoid in $\mathbb{Z}^2$ whose vertices are $\{(0,0),(a,0),(a+b,b),(0,b)\}$. Also, let $\mathcal{T}_{a,b}$ be set of one-step north-east paths from $(0,0)$ to $(a+b,b)$ that stay inside $\mathcal{L}_{a,b}$, and $\sigma_{a,b}$ the amount of such paths. Finally, let C_n denote the n-th Catalan number.

Proposition A.5. *The sequence $\sigma : \mathbb{N}^2 \rightarrow \mathbb{N}$ verifies the following recurrence:*

$$\begin{cases} \sigma_{a,b} = \sum_{i=0}^{b} \sigma_{a-1,i} \cdot C_{b-i} & \textit{for } a \geq 1, b \in \mathbb{N}, \\ \sigma_{x,0} = 1 & \textit{for } x \in \mathbb{N}, \\ \sigma_{0,y} = C_y & \textit{for } y \in \mathbb{N}. \end{cases}$$

Proof. To prove this, we write $\mathcal{T}_{a,b}$ as a union of disjoint sets. As a first remark, note that every path in $\mathcal{T}_{a,b}$ touches the line $l = \overline{(a,0)(a+b,b)}$ at least once. For $i \in \{0,\ldots,b\}$, let P_i be the point $(a+i,i) \in l$ and $\mathcal{T}^{(i)}_{a,b}$ all paths in $\mathcal{T}_{a,b}$ that touch the line l for the first time at P_i. We have the disjoint union

$$\mathcal{T}_{a,b} = \bigcup_{i=0}^{b} \mathcal{T}^{(i)}_{a,b}.$$

Also, note that a path in $\mathcal{T}_{a,b}$ touches l for the first time in P_i if and only if it passed through the point $P_i - (1,0)$ without exiting $\mathcal{T}_{a-1,i}$, followed by an east step and any path from P_i to $(a+b,b)$. This yields

$$\sigma_{a,b} = \sum_{i=0}^{b} (\text{number of paths from } (0,0) \text{ to } P_i - (0,1)) \cdot (\text{number of paths from } P_i \text{ to } (a+b,b))$$

Note that the amount of paths from P_i to $(a+b,b)$ is C_{b-i}, since both points belong to l, proving that $\sigma_{a,b}$ verifies the recurrence equation. The border cases $\sigma_{0,\cdot}, \sigma_{\cdot,0}$ are straightforward to prove. □

Now let us define the sequence of generating functions with respect to the second variable of $\sigma_{\cdot,\cdot}$ as follows:

$$\begin{aligned} \phi_a : \quad & \mathbb{R} \to \mathbb{R} \\ & x \mapsto \sum_{j=0}^{\infty} \sigma_{a,j} \cdot x^j \end{aligned}$$

Proposition A.6. *For $x \in [0, 1/4]$ and $a \in \mathbb{N}$*

$$\phi_a(x) = c(x)^{a+1},$$

where $c(x) := \frac{1-\sqrt{1-4x}}{2x}$ is the generating function of the Catalan numbers.

Proof. We have $\sigma_{a,\cdot} = \sigma_{a-1,\cdot} \star C_\cdot$, where $\star$ is the convolution operator, therefore $\phi_a(x) = \phi_{a-1}(x) \cdot c(x)$. The result follows noting that $\phi_0(x) = c(x)$. □

Extracting the sequence from the Taylor series of $\phi_a(x)$ around 0 and proving by induction gives

Proposition A.7. *For $(a,b) \in \mathbb{N}^2$,*

$$\sigma_{a,b} = \frac{(a+b) \cdot (a+2b)!}{b! \cdot (a+b+1)!} = \frac{a+1}{a+b+1} \cdot \binom{a+2b}{a+b}.$$

A.6.2 Counting paths in a pentagon. Let $\mathcal{P}^r_{a,b}$ be the set of north-east unitary-step paths from $(0,0)$ to (r,b) that do not exit the pentagon $(0,0), (a,0), (r, r-a), (r,b), (0,b)$, and $\theta^r_{a,b}$ the amount of such paths. Also, let λ be the diagonal line $\overline{(a,0)(r,r-a)}$.

Proposition A.8. *The sequence $\theta : \mathbb{N}^3 \longrightarrow \mathbb{N}$ verifies the following recurrence relation*

$$\theta^r_{a,b} = \theta^r_{a-1,b} + \sum_{i=0}^{r-a} \sigma_{a-1,i} \cdot \sigma_{a+b-r,r-a-i}.$$

Proof. As in the proof of A.5, we write $\mathcal{P}^r_{a,b}$ as a disjoint union:

$$\begin{aligned} \mathcal{P}^r_{a,b} &= (\text{paths that do not touch } \lambda) \cup (\text{paths that touch } \lambda) \\ &= (\text{paths that do not touch } \lambda) \cup \bigcup_{y \in \lambda} (\text{paths that touch } \lambda \text{ for the first time at } y \in \lambda) \\ &= (\text{paths that do not touch } \lambda) \cup \bigcup_{y \in \lambda} (\text{paths from } (0,0) \text{ to } y - (1,0)) \cdot (1,0) \cdot (\text{paths from } y \text{ to } (r,b)) \\ &= (\text{paths that do not touch } \lambda) \cup \bigcup_{i=0}^{r-a} (\text{paths from } (0,0) \text{ to } (a+i,i) - (1,0)) \cdot (1,0) \cdot (\text{paths from } (a+i,i) \text{ to } (r,b)), \end{aligned}$$

where in the last equation, the symbol $\cdot$ stands for concatenation of paths. The recurrence follows noting that the paths that do not touch λ are exactly $\mathcal{P}^r_{a-1,b}$, the paths from $(0,0)$ to $(a+i-1,i)$ are $\mathcal{T}_{a-1,i}$ and the paths from $(a+i,i)$ to (r,b) are exactly $\mathcal{T}_{a+b-r,r-a-i}$ □

Proposition A.9. *For positive integers a, b, r it holds that*

$$\begin{cases} \text{If } r \le a, \quad \theta^r_{a,b} = \binom{r+b}{b}, \\ \text{If } a < r, \quad \theta^r_{a,b} = \sigma_{|b-r|,\min(r,b)} + \displaystyle\sum_{i=\max(1,r-b+1)}^{a} \sum_{j=0}^{r-i} \sigma_{i-1,j} \cdot \sigma_{i+b-r,r-i-j}. \end{cases}$$

Proof. If $r \le a$, the polygon is simply a rectangle from $(0,0)$ to (r,b). If $a < r \le b$, then the result yields adding the recurrence relation from the base case $a = 0$ and noting that $\theta^r_{0,b} = \sigma_{b-r,r}$. If $b < r$, then the base case is $a = r-b$ and $\theta^r_{r-b,b} = \sigma_{r-b,b}$. □