Colordag: An Incentive-Compatible Blockchain

Ittai Abraham
Intel Labs, Israel

Danny Dolev
The Hebrew University of Jerusalem, Israel

Ittay Eyal
Technion, Israel

Joseph Y. Halpern
Cornell University, NY, USA

Abstract

We present *Colordag*, a blockchain protocol where following the prescribed strategy is, with high probability, a best response as long as all miners have less than 1/2 of the mining power. We prove the correctness of Colordag even if there is an extremely powerful adversary who knows future actions of the scheduler: specifically, when agents will generate blocks and when messages will arrive. The state-of-the-art protocol, Fruitchain, is an ε-Nash equilibrium as long as all miners have less than 1/2 of the mining power. However, there is a simple deviation that guarantees that deviators are never worse off than they would be by following Fruitchain, and can sometimes do better. Thus, agents are motivated to deviate. Colordag implements a solution concept that we call *ε-sure Nash equilibrium* and does not suffer from this problem. Because it is an ε-sure Nash equilibrium, Colordag is an ε-Nash equilibrium **and** with probability $1 - \varepsilon$ is a best response.

2012 ACM Subject Classification Computing methodologies → Distributed computing methodologies; Theory of computation → Solution concepts in game theory; Security and privacy → Distributed systems security

Keywords and phrases Game theory, incentives, blockchain

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding Dolev's work was supported by the Federmann Cyber-Security Center in conjunction with the Israel National Cyber Directorate. Eyal's work was supported by the Israel Science Foundation (grant No. 1641/18), Avalanche Foundation, and IC3. Halpern's work was supported in part by AFOSR grant FA23862114029, MURI grant W911NF-19-1-0217, ARO grant W911NF-22-1-0061, and a grant from the Algorand Centers of Excellence program managed by the Algorand Foundation.

Acknowledgements We thank Roi Bar-Zur for comments on an early version of this manuscript, and the reviewers of the paper for their helpful comments.

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

At the heart of Bitcoin [15] is the Nakamoto consensus protocol, which is based on proof-of-work [7, 12, 1]. The system participants, called *miners*, maintain a *ledger* that records all *transactions*—payments or so-called smart-contract operations. The transactions are batched into *blocks*; a miner can publish a block only by expending computational power, at a rate proportional to her computational power in the system. This rate is called *mining power*.

The Nakamoto consensus protocol achieves desirable ledger properties even against an adversary that controls $\alpha < 1/2$ of the mining power [10, 17, 13]. That is, as long as miners that control a majority of the cmoputing power follow the Nakamoto consensus protocol, security is guaranteed. But Nakamoto's protocol relies on *incentives*: The blocks form a tree, and each miner is rewarded for each block it generated that is included in the longest path (blockchain) in the tree. Unfortunately, following the Nakamoto consensus protocol is *not* a best response for miners that control a large fraction (but less than 1/2) of the total computational power [8, 16, 19]. For example, under some minimal modeling assumptions, even a coalition that controls 1/4 of the computational power can increase its reward by deviating from the Nakamoto Consensus protocol.[1] Stated differently, the Nakamoto consensus protocol is not a coalition-resistant equilibrium if there are coalitions that control more than 1/4 of the mining power.

Pass and Shi [18] make major progress with their Fruitchain protocol. In Fruitchain, the blocks form a dag (rather than a tree) with the longest chain determining rewards. However, miners are rewarded for a special type of block, called *fruit*. Each fruit block c is the child of a regular block b_1, and its miner is rewarded if a subsequent block b_2 points to the fruit, both blocks b_1 and b_2 are on the longest chain, and the path between them is shorter than some constant. If the longest chain is sufficiently long that the fruit c does not provide a reward, then c is called *stale*. Fruitchain is an *ε-Nash Equilibrium (NE)*, that is, a miner, even with mining power arbitrarily close to 1/2, can improve her revenue by only a negligible amount by deviating from the protocol. Like Bitcoin [17], Fruitchains is provably correct except with negligible probability in executions of length polynomial in the system's security parameter.

However, Fruitchain allows for a simple deviation by which any coalition can increase its utility without taking any risk: Specifically, a miner points only to its own fruit when generating blocks, ignoring fruit generated by others. This simple deviation dominates the prescribed protocol, as it creates a small probability that the ignored fruit will become stale, increasing the miner's relative revenue. While the probability increase is negligible in the staleness parameter, there is no risk to the miner. Moreover, if all agents are small and play this simple deviation, then the probability that any of them can point to its own fruit before it becomes stale is small; this results in a violation of the ledger properties, as progress becomes arbitrarily slow. Our conclusion is that ε-NE is an inappropriate solution concept in our setting; agents might still be incentivized to deviate from a ε-NE, although the benefit is small.

We present a more robust solution concept that we call *ε-sure NE*. A protocol is an ε-sure NE if, for any player, playing the prescribed protocol is a best response except for some set of runs (executions) that has probability at most ε. If utilities are bounded (as they are in our case), a ε-sure NE is an ε-NE, but the converse is not the case in general.

Our main contribution is the *Colordag* protocol, a PoW-based protocol that is an ε-sure NE, provided that each player controls less than half the total computational power. Like

[1] Under the most optimistic assumptions about the underlying network, this bound increases to only 1/3.

various solutions, starting from Lewenberg et al. [14, 22], Colordag constructs a directed acyclic graph rather than a tree. This graph is used for reward calculation; the ledger consists of a subset of blocks on the graph.

To achieve the required properties, Colordag makes use of three key ideas.

1. Due to the distributed nature of the system, two miners might generate a block before hearing of each others' blocks. The result is a *fork* where two blocks point to the same parent. This gives an advantage to the attacker, as the two blocks only extend the longest chain by one. To deal with forks that occur naturally, Colordag colors blocks randomly, and calculates the reward by looking at the graphs generated by the nodes of each color (technically, the *graph minors* of each color) separately. Adding more colors allows us to keep the original rate of block production, while mitigating the effects of forking: the fact that there are fewer blocks of a given color reduces the probability of forks in the minors. Previous work [10, 2, 24] randomly attributed properties to blocks for performance or resilience. In contrast, here coloring is used only for calculating the reward.
2. Colordag guarantees that, with high probability, malicious behavior (indeed, any deviation from the strategy) will not result in a higher reward for the deviating agent. The basic idea is that honest blocks of a given color will almost always be *acceptable*: they are on a chain that is almost the longest in its minor. Unacceptable blocks get no reward and do not affect the rewards of others. The approach is similar to Sliwinski and Wattenhofer's *block staling*; it is guaranteed to work as long as there is no agent has a majority of mining power, even if players know in advance the order in which they are scheduled.
3. To disincentivize deviation, Colordag penalizes forking: Considering the graph minors of each color separately, if there is more than one acceptable block of a given depth T in a minor, then all blocks of depth T get reward 0. Since each miner i aims to maximize its *relative* revenue (i.e., the ratio between i's revenue and the total reward received by miners while i is active, just as is the case in, e.g., [8, 19, 17, 11]), and (by assumption) deviators have less power than honest agents (i.e., agents that follow the prescribed protocol), a symmetric penalty to a deviator and an honest agent results in the deviator suffering more than the honest agents. Sliwinski and Wattenhofer [21] also use symmetric penalties in a blockdag for all blocks that are not connected by a directed path; each block in a set X of such blocks is penalized by $|X|c$ (for some constant c). However, with their approach, an adversary can harm honest agents. For example, if $c = 3$ and there is a benign honest fork, the attacker can add a third forked block, resulting in a total penalty of $6c$ for honest agents ($3c$ per block) while suffering only $3c$ itself, so deviation is worthwhile for a sufficiently large minority miner. In fact, their threshold is smaller than $1/2$, and their protocol is only an ε-NE, like Fruitchain.

The rest of the paper is organized as follows. In Section 2, we describe an abstract model of a PoW system, similar to models used in previous work, and discuss the bitcoin desiderata. In Section 3, we formalize mining as a game, so that we can make notions like incentive compatibility and best response precise. In Section 4, we formally describe the Colordag mechanism: the Colordag protocol and the revenue scheme that we use. We then prove in Section 5 that Colordag satisfies the ledger desiderata and is an ε-sure equilibrium in the face of coalitions with less than $1/2$ of the computational power, and even if the coalition knows what the scheduler does in advance. Specifically, we show that, for the appropriate choice of parameters, in all but a negligible fraction of histories, miners do not gain if they deviate from the Colordag protocol. Finally, in Section 6, we discuss the values of the Colordag parameters when dealing with a weaker adversary than we assume here and the path to a practical implementation.

2 Model and Desiderata

Blockchain protocols operate by propagating data structures called *blocks* over a reliable peer-to-peer network. We abstract this layer away and describe our model (see Section 2.1), which is similar to previous work. The goal of the protocol is to implement a distributed *ledger* (see Section 2.2), roughly speaking, a commonly-agreed upon record of transactions.

2.1 Model

The system proceeds in rounds in a synchronous fashion, as is common in many other analyses (e.g., [8, 10, 17, 18]). A *history* h is a complete description of what happens to the system over time. Formally, h is a function from rounds to a description of what has happened in the system up to round t (which blocks were generated, which were made public, which agents are in the system, and so on). We denote by $h(t)$ the prefix of h up to time t. There is a possibly unbounded number of agents, called *miners*, named $1, 2, \ldots$. We take the miners to represent coalitions of agents, so we do not talk about coalitions of miners (and will later assume that each miner controls less than 1/2 of the computational power). For each history h and miner i, there exist rounds $T_1^{h,i}$ and $T_2^{h,i}$ such that i is *active* between $T_1^{h,i}$ and $T_2^{h,i}$.

Some previous analyses (e.g., [15, 8, 19, 5, 9]) focused on average rewards, and did not consider adversarial attacks that could lead to a violation of the ledger properties, although in an infinite execution such attacks may succeed with probability one. We aim to prove, with high probability, both that Colordag is incentive compatible (i.e., no agent can increase its utility by deviating from the protocol) and that, if all but at most one agent follow the protocol, then the ledger properties hold. So, like previous work (e.g., [17, 18, 13]), we assume that the system runs for a bounded time, up to some large $T_{\max}$. Without this assumption, even events with arbitrarily small frequency happen with probability one.

Let $Ag(h, t)$ be the set of active miners in the system at round t of history h, that is, all miners i such that $T_1^{h,i} \leq t \leq T_2^{h,i}$. For any given history and time, the set $Ag(h, t)$ is finite. Each miner i has so-called *mining power*, a positive value representing her computational power. The *power* of a miner i at time t, denoted $Pow_t^h(i)$, is her fraction of the mining power at time t in history h. Let $Pow^h(i) = \sup_t Pow_t^h(i)$, and let $Pow(i) = \sup_h Pow^h(i)$. We will be interested in the case that, for all miners i, there exists some $\alpha < 1/2$ such that $Pow(i) \leq \alpha$.

We assume that a scheduler determines which miners are active, which miners move in each round, and how long it takes a message to arrive. To simplify the discussion of the scheduler, we assume (as is the case for Colordag and all other blockchain algorithms) that each miner builds a local version of a directed acyclic graph called a *blockdag*. We refer to each node and its incoming edges in the graph as a *block*. Our hope is that miners have an "almost-common" view of the blockdag. Following the standard convention, we assume that the blockdag has a commonly-agreed-upon root that we refer to as the *genesis block*. The *depth* of a blockdag G, $d(G)$, is the length of a longest path in G. The *depth* of a block b in G, denoted $d(G, b)$, is the length of a longest path in G from the genesis to b.[2]

In every round, the scheduler chooses one miner at random among the miners that are active in that round (a miner i being chosen represents it having solved a computational

[2] We follow standard graph-theoretic terminology here. In the blockchain literature, what we are calling the depth of a node is sometimes called its height.

puzzle), with probability proportional to its power (as in, e.g., [8, 19, 17]); that is, miner i is chosen in round t with probability proportional to $Pow_t(i)$. If the scheduler chooses a miner i in round t, then i either selects some set P of the nodes currently in its blockdag, with the constraint that no node in P can be the ancestor of another node in P, and adds a new vertex v to the blockdag with P as its parents or does nothing. If i adds (P, v), then i can either broadcast this fact or save it for possible later broadcast. Note that a miner cannot send (P, v) to a strict subset of miners; it is either broadcast to all miners or sent to none of them (as in, e.g., [8, 10, 2] and deployed systems [15, 23]). Miners can also broadcast pairs that they saved earlier. If P violates the constraint that no node in P can be the ancestor of another node in P, the message (P, v) is ignored. We assume in the rest of the paper that this does not occur, as the outcome is indistinguishable from simply not generating a block.

Denote by $G^{h(t)}$ the blockdag including all blocks published at or before round t in execution h. Let $G_i^{h(t)}$ denote i's view of $G^{h(t)}$; this is the blockdag at round t of history h according to i. For example, i may not be aware at round t that j created block b, so block b will be in $G^{h(t)}$ but not in $G_i^{h(t)}$. Note that blocks that node i has generated but not published are not included in $G_i^{h(t)}$ (although, of course, i is aware of them); however, if a block $b \in G_i^{h(t)}$ refers to a block b' (i.e., b is a child of b', since we assume that the message broadcast by the miner that created block b has a hash of all the parents of b), then we take b' to have been published, and include it in $G_i^{h(t)}$. We omit the h if it is clear from context or if we are making a probabilistic statement; that is, if we say that a certain property of the graph holds at time t with probability p, then we mean that the set of histories h for which the property of $G^{h(t)}$ holds has probability p.

We assume that there is an upper bound $\Delta \geq 1$ on the number of rounds that it takes for a message to arrive. The arrival time of each message may be different for different miners; that is, if miner i broadcasts (P, v) at round t, miners j and j' might receive (P, v) in different rounds. Messages may also be reordered (subject to the bound on message delivery time).

Note that although there is a bound on message delivery time, miners do not know the publication time of a block. Thus, there is no way that a miner can tell if a block was withheld for a long period of time. Interestingly, in Colordag, agents can tell to some extent from the blockdag topology if a block was withheld for a long period of time; such blocks do not get any reward.

In summary, this is how the scheduler works: (1) it chooses, for each agent i, in which interval i is active and its power; (2) it chooses which agent generates a block in each round (randomly, in proportional to their power); and, finally, (3) it chooses a message-delivery function (i.e., a function that, given a history up to round m, decides how long it will take each round m message to be delivered, subject to the synchrony bound). We assume that the adversary knows the scheduler's choices.

The scheduler's protocol, including the choice of when agents are active and the random choice of which agents generate a block in each round, and the strategies used by the miners together determine a probability on the set of histories of the system. While we have specified that all messages must be delivered within Δ rounds, we have not specified a probability over message delivery times, block-generation times, or when agents are active. Our results hold whatever the probability is over message-delivery times (subject to it being at most Δ) and on when agents are active (subject to no agent having power greater than α). Thus, when we talk about a probability on histories, it is a probability determined by the strategies of the miners and a scheduler that satisfies the constraints above.

2.2 Desiderata

A ledger function $\mathcal{L}$ takes a blockdag G and returns a sequence $\mathcal{L}(G)$ of blocks in G; the kth element in the sequence is denoted $\mathcal{L}_k(G)$. The length of the ledger is denoted $|\mathcal{L}(G)|$. We want the ledgers that arise from the blockdags created by Colordag to satisfy certain properties [10, 17, 13].

The first property requires that once a block allocation is set, its position in the ledger remains the same in the view of all miners.

▶ **Definition 1** (Ledger Consistency). *There exists a constant K such that, for all miners i and j, if $k \leq |\mathcal{L}(G_i^{h(t)})| - K$ and $t \leq t'$, then $\mathcal{L}_k(G_i^{h(t)}) = \mathcal{L}_k(G_j^{h(t')})$.*

The next desideratum is that the length of the ledger should increase at a linear rate. Let $|\mathcal{L}(G)|$ denote the number of elements in the sequence $\mathcal{L}(G)$.

▶ **Definition 2** (Ledger Growth). *There exists a constant g such that, for all rounds $t < t'$ and all miners i, if $t' - t > g$, then $|\mathcal{L}(G_i^{h(t')})| \geq |\mathcal{L}(G_i^{h(t)})| + 1$.*

The final ledger desideratum says that the fraction of the total number of blocks in the ledger that are generated by honest miners should be larger than a positive constant.

▶ **Definition 3** (Ledger quality). *There exist constants $D > 0$ and $\mu \in (0, 1)$ such that for all rounds t and t' such that $t' - t \geq D$, the fraction of blocks mined by honest miners placed on the ledger between round t and t' is at least μ.*

Note that this common requirement is fairly weak. As we will see, Colordag miners will be rewarded, on average, proportionally to their efforts. Indeed, to motivate miners to mine, the system rewards miners for essentially all the blocks they generate (not just the ones on the ledger). The revenue from each block is determined by the *revenue scheme*. Formally, a revenue scheme r is a function that associates with each block b and labeled blockdag G a nonnegative real number $r(G, b)$, which we think of as the revenue associated with block b in the blockdag G. Our final desideratum requires that revenue stabilizes.

▶ **Definition 4** (Revenue Consistency). *There exists a constant K such that, for all miners i and j and times t, t', and t'' such that $t', t'' > t + K$, if b is published at time t in history h, then $r(G_i^{h(t')}, b) = r(G_j^{h(t'')}, b)$.*

Most previous work (e.g., [15, 23, 18]) did not state this requirement explicitly. There, it follows from ledger consistency, since all and only blocks in the ledger get revenue.[3] In contrast, with Colordag, a miner might get revenue for a block even if it is not on the ledger, and may not get revenue for some blocks that are on the ledger. We thus need to separately require that the revenue that a miner gets from a block eventually stabilizes.

3 Revenue Scheme and ε-Sure NE

It is not hard to design protocols that satisfy the blockdag desiderata. However, there is no guarantee that the miners will actually use those protocols. We assume that miners are rational, so our goal is to have a protocol that is *incentive-compatible*: it is in the miners' best interests (appropriately understood) to follow the protocol. Before describing our protocol, we need to explain how the miners get utility in our setting.

[3] Ethereum's *uncle blocks* [23] are off-chain but rewarded; however, their rewards are explicitly placed in the ledger after a small number of blocks, therefore revenue consistency for Ethereum also follows almost trivially from ledger consistency.

3.1 Revenue Scheme

A miner's utility in a blockdag is determined by the miner's *revenue*. We denote by $B_i^{h(t)}$ the blocks generated by miner i in history $h(t)$. Given a revenue scheme r, for each miner i, history h, and round t, we can calculate the revenue $r(G_i^{h(t)}, b)$ for every block $b \in B_i^{h(t)}$.

Given a revenue scheme r, miner i's total revenue at round t according to r in history h of a protocol is the sum $\sum_{b \in B_i^{h(t)}} r(G_i^{h(t)}, b)$ of the revenue obtained for each block b generated by i while it is active in history h. For example, in Bitcoin [15], the revenue of a miner is the number of blocks it generated that are on the so-called main chain. Finally, i's *utility* according to revenue scheme r at round t in history h is i's normalized share of the total revenue while it is active. Taking *time*(b) to be the time that block b was published, for $t \geq T_1^{h,i}$, we define:

$$u_i^r(h,t) = \frac{\sum_{b \in B_i^{h(t)}} r(G_i^{h(t)}, b)}{\sum_{\{b : T_1^{h,i} \leq time(b) \leq \min(t, T_2^{h,i})\}} r(G_i^{h(t)}, b)} \ . \tag{1}$$

This way of determining a miner's utility from a revenue function is common (see, e.g., [8, 19, 18, 11, 5, 4]). Intuitively, the utility is normalized because the value to a miner of holding a unit of currency depends on the total amount of currency that has been generated. A miner is interested in its utility during the time that it is active. Although miner i's utility may change over time, for a protocol that has the revenue consistency property (as Colordag does), in every history, i's utility eventually stabilizes (since the set of blocks that are published between $T_1^{h,i}$ and $T_2^{h,i}$ for which each miner gets revenue and the revenue that the miners get for these blocks eventually stabilize). When we talk about i's utility in history h, we mean the utility after all the revenue up to $T_2^{h,i}$ has stabilized.

3.2 ε-sure NE

As we said in the introduction, we are interested in strategy profiles that form a ε-sure Nash Equilibrium (NE), a strengthening of ε-NE as long as utility is bounded. We now define these notions carefully.

In the definition of ε-sure NE, we are interested in the probability that a history in a set H of histories occurs, denoted $\Pr(H)$. (Note that a history corresponds to a path in the game tree.) In general, the probability of a history depends on the strategies used by the miners. We are interested in sets of histories that have probability at least $(1 - \varepsilon)$, independent of the strategies used by the miners. To ensure that this is the case, we take H to be a set of histories determined by the scheduler's behavior. The scheduler is a probabilistic algorithm. It chooses miners for block generation with probability $Pow_i(t)$, and chooses network propagation time arbitrarily, bounded by a constant Δ. The probabilities of the different histories are then defined by the probabilities of the scheduler's random coins. For example, suppose that there are 10 agents, all with the same computational power, and we consider histories where agent 1 is scheduled first, followed by agent 2. This set of histories has probability 1/100, independent of the agents' strategies.

We denote the strategy of each miner i by σ_i, a strategy profile by $\sigma = (\sigma_1, \ldots, \sigma_n)$, and the profile excluding the strategy of i by σ_{-i}. The profile with miner i's strategy replaced by σ'_i is (σ'_i, σ_{-i}).

▶ **Definition 5** (ε-sure NE)**.** *A strategy profile $\sigma = (\sigma_1, \ldots, \sigma_n)$ is an ε-sure NE if, for each agent i, there exists a set H_i of histories with probability at least $1 - \varepsilon$ such that, conditional* *on H_i, σ_i is a best response to σ_{-i}; that is, for all strategies $\sigma_i' \neq \sigma_i$ of agent i:*

$$u_i(\sigma \mid H_i) \geq u_i((\sigma_i', \sigma_{-i}) \mid H_i).$$

Of course, if, for each agent i, we take H_i to consist of all histories; then we just get back NE, so all Nash equilibria are ε-sure NE for all ε. As the next result shows, if all utilities are in the interval $[m, M]$ then every ε-sure NE strategy profile is an $(M-m)\varepsilon$-NE. Since in our setting, the utility of a miner i is the fraction of total revenue that i obtains while i is active, the utility is in $[0, 1]$, so is clearly bounded.

▶ **Lemma 6.** *If a strategy profile σ is an ε-sure NE and all players' utilities are bounded in the range $[m, M]$, then σ is an $(M-m)\varepsilon$-Nash Equilibrium.*

Proof. For a player i, there is a set of histories H_i with probability $\Pr(H_i) > 1 - \varepsilon$ where σ_i is a best response. In histories not in H_i, denoted $\overline{H}_i$, player i might improve her utility by up to $(M - m)$. The probability of $\overline{H_i}$ is bounded by ε. Therefore, the utility increase of a player by switching her strategy is at most $0(1-\varepsilon) + (M-m)\varepsilon = (M-m)\varepsilon$. Thus, σ is an $(M-m)\varepsilon$-NE. ◀

However, there are ε-NE that are not ε'-sure NE for any $\varepsilon' < 1$. For example, consider a game where a player chooses 0 or 1. She gets utility 0 for choosing 0 and utility ε for choosing 1. Choosing 0 is ε-NE but is not ε'-sure NE for any ε' as choosing 1 strictly increases her utility in all histories. Thus, ε-sure NE is a solution that lies strictly between ε-Nash and Nash equilibrium when utility is bounded, as it is in our case.

We will show that, for all ε, we can choose parameter settings to make Colordag an ε-sure NE. In addition, it satisfies the ledger desiderata.

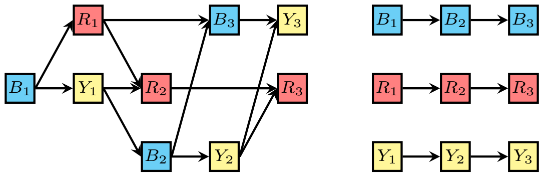
4 Colordag

The Colordag mechanism consists of a recommended strategy that we want participants to follow and a revenue scheme. The strategy, denoted σ^{cd} (*cd* stands for Colordag) is extremely simple: If chosen at round t in history h, miner i takes P to consist of the leaves of $G_i^{h(t)}$. It thus generates a block labeled b with parents P and broadcasts (P, b), adding it to its local view $G_i^{h(t)}$.

The reward function is more involved. Before describing it formally, we give some intuition for it. Suppose that we give all blocks reward 1. It is easy to see that σ^{cd} is a Nash equilibrium. But, with this reward function, so is every strategy profile where miners always publish the blocks they generate at some point. For example, miners can hang blocks off the genesis; this is also a best response. But if all miners choose to do this, it would be impossible to define a ledger that preserves consistency.

There is a simple fix to the second problem: if there is more than one block of the same depth, all blocks of that depth get reward 0. This stops hanging blocks off the genesis from being a best response. But now we have a new problem – we lose reward consistency. At any point, an adversary can penalize an arbitrary block b by adding a new block with the same depth as b. To obtain reward consistency, we would want to call the adversary's block in such cases *unacceptable*, and completely ignore it. Intuitively, we want blocks that hang off a block of depth T to be viewed as unacceptable if they are added after the blockdag has height sufficiently greater than T. This motivates our notion of unacceptability.

Roughly speaking, our reward function gives a reward of 1 to all blocks except those that are unacceptable or those that are forked; these get reward 0. The mechanism thus relies



(a) A colored dag. **(b)** Graph minors.

Figure 1 Coloring a dag.

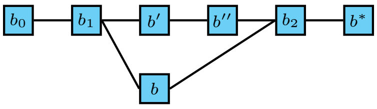


Figure 2 An unacceptable block.

on a rational miner not being able to form a longer chain privately than the honest miners can form. (If a dishonest miner could form a longer chain privately than the honest miners can form, it could then publish that chain and make all the blocks that the honest miners formed during that time unacceptable.) However, forks can happen naturally, due to network latency, meaning honest miners' chain-extension rate is less than their block-generation rate, whereas the rational miner's rate is unimpaired. To mitigate the effect of forking, we color the nodes, effectively partitioning the blockdag into disjoint *graph minors* [6] (one minor for each color); we determine forking (and acceptability) in these graph minors. We can make the amount of forking as small as we want by using enough colors. We now present the key components needed for the reward function, and then give the actual function.

Coloring nodes

Because messages may take up to Δ rounds to arrive, two honest miners can both extend a given block b, because neither has heard of the other's extension at the point when it is doing its own extension. To make our results as strong as possible, following the literature [10, 13, 21], we assume that a deviating miner is able to avoid forking with its own blocks. Thus, a deviator can extend paths in the blockdag faster than would be indicated by her relative power. In particular, a deviator with power less than (but close to) $1/2$ may be able to (with high probability) build paths longer than the honest miners can build, due to forking.

To deal with this problem, Colordag assigns each block a color chosen at random from a sufficiently large set of N_C colors; that is, it assigns each block a number in $\{1, \ldots, N_C\}$ (which we view as a color). In practice, this would be done by taking the color to be the hash of the contents of the block mod N_C. This ensures that, except with negligible probability (1) all colors are equally likely, (2) the color of a block b is learned by the miner that generates b only after b is generated, and (3) colors are commonly known (every miner can compute the color of every block, just knowing its content). In our model, this is like having the scheduler allocate a random color when it chooses a miner in a round. Figure 1a shows a blockdag where the nodes are colored either blue (B), red (R), or yellow (Y).

After coloring each node in the graph G, we consider the graph minor G_c corresponding to color c: The nodes in this graph minor are just the nodes of color c in G; node b' is a child of b in G_c iff b' is a descendant of b in G and there is no path in G from b to b' with an intermediate node (i.e., one strictly between b and b') of color c. Figure 1b shows the minors resulting from our example.

The key point is that, by taking N_C sufficiently large, we make the probability of a fork among the blocks generated by honest miners in G_c arbitrarily small. The reasoning is simple: Suppose that b and b' are generated by honest miners at times t_b and $t_{b'}$, respectively, where $t_{b'} > t_b$. If b and b' have the same color and there are enough colors, then with high probability, $t_{b'} > t_b + \Delta$, so b' is a descendant of b in G, and hence also in G_c. In other words, if two honest blocks are neither an ancestor nor a descendant of one another in G, they are unlikely to have the same color.

Acceptable blocks

We now define what it means for a block to be acceptable. We want it to be the case that a block is unacceptable if it has depth T but was added after the depth of the blockdag is considerably greater than T. The way we capture this is by requiring acceptable blocks to be on paths that are almost the same as a particular longest path in the graph.

Given a dag G_c, we "close off" G_c so that it has a unique initial node and a unique final node (whether or not it already had them), by adding special vertices b^0 and b^*, where b^0 is the parent of all the roots of G_c (essentially we consider b^0 to be the genesis, belonging to all minors) and b^* is the child of all leaves in G_c. We refer to this graph as G_c^+. We denote by $|Q|$ the length of a path Q, which is the number of edges in Q, and hence one less than the number of vertices in Q.

Given a graph G, for each color c, we choose one particular longest path in G_c^+ from b^0 to b^*. If there is more than one longest path, we use a canonical tie-breaking rule, which we now define, as it will be useful later. Intuitively, if there are several paths of maximal length, we order the paths by considering the point where they first differ, and choose using some fixed tie-breaking rule that depends only on the contents of the blocks where they first differ.

▶ **Definition 7** (Canonical path). *Given a blockdag, the canonical path starts at the genesis and continues as all longest paths do up to the first point where some longest paths diverge (this could already happen at the genesis). At this point, we choose some tie-breaking rule to decide which longest paths to follow.*[4] *The canonical path continues as all these longest paths until the next point of divergence. Again, at this point we use the tie-breaking rule to decide which longest paths to follow. We apply this procedure each time longest paths diverge.*

The key point is that all these tie-breaking rules are local. The decisions made are the same (if all the prefixes of these paths exist) in all the graphs we consider.

▶ **Definition 8** (Acceptable Block). *A path P in G_c^+ from block b^0 to block b^* is N_ℓ-*almost-optimal *if the symmetric difference between P and the canonical longest path P^* (i.e., the set of blocks in exactly one of the paths P and P^*) has fewer than N_ℓ blocks. A block b of color c is N_ℓ-*acceptable *iff it is on an N_ℓ-almost-optimal path P of color c. The path P is said to be a* witness *to the acceptability of b.*

We need one more definition before we can define the revenue scheme.

▶ **Definition 9** (Forked Block). *An N_ℓ-acceptable block b in blockdag G is N_ℓ-*forked *if there is another N_ℓ-acceptable block b' with the same color as b, say c, such that $d(G_c, b) = d(G_c, b')$.*

We can now make Colordag's revenue scheme precise. As we said, a block of color c gets reward 1 unless it is unacceptable or it is forked in G_c. The revenue scheme takes N_ℓ as a parameter, so we denote it $r^{cd}_{N_\ell}$.

▶ **Definition 10** (Colordag Revenue Scheme). *A node b is N_ℓ-*compensated *if b is N_ℓ-acceptable in G_c and is not N_ℓ-forked; $r^{cd}_{N_\ell}(G, b) = 1$ if b is N_ℓ-compensated; otherwise, $r^{cd}_{N_\ell}(G, b) = 0$.*

[4] For example, in practice this could be the smallest hash of the block contents.

Colordag Ledger Function

We present here a ledger function that makes the analysis easier, and satisfies all the ledger properties. This function is somewhat inefficient, since not all blocks are a part of the ledger. In Section 6, we show how a small modification of this approach lets us include in the ledger the transactions that appear in all acceptable blocks in the blockdag.

The ledger function of Colordag chooses a fixed color $\hat{c}$, and given graph G, chooses the canonical path in the subgraph of G of color $\hat{c}$. The ledger is defined by the blocks on this path. For example, given the blockdag in Figure 1a, and assuming $\hat{c}$ is yellow, the ledger is the sequence of blocks (Y_1, Y_2, Y_3).

▶ **Definition 11** (Colordag Ledger Function). *Given a blockdag G and a fixed color $\hat{c}$, Colordag's ledger function $\mathcal{L}^{cd}$ returns a sequence consisting of the blocks on the canonical path in $G_{\hat{c}}$.*

Reward Calculation

Since following the protocol is the miners' best response, in practice they will generate a single chain of each color and get rewarded per block. As we now show, the reward calculation can be done in polynomial time, even if miners deviate. Given N_ℓ, a graph G, and a block b of color c, we want to calculate $r^{cd}_{N_\ell}(G, b)$. The first task is to construct the graph minor G_c of color c; this clearly can be done in time polynomial in $|G|$. The next step is to determine the canonical longest path P^* in G_c. We can do this quickly, since it is well known that longest paths in dags can be calculated in linear time [20]. (Indeed, it is straightforward to keep a table of lengths of longest paths and update it as G_c grows over time.) Finally, using depth-first search, we can quickly compute the block b_2 of least depth on P^* that is a descendant of b (which is b itself if b is on P^*) and the block of greatest depth b_1 on P^* that is an ancestor of b. By construction there is a path from b_1 to b_2 that includes b. It is easy to see that b is acceptable iff the number of nodes on the path fromn b_1 to b_2 that includes b (not including b_1 and b_2) and the number of nodes on the canonical path from b_1 to b_2 (again, not including b_1 and b_2) is less than N_ℓ. If b is forked, then similar arguments allow us to check whether a block forking b is acceptable. If b is acceptable and no block forking b is acceptable, then $r^{cd}_{N_\ell}(G, b) = 1$; otherwise, $r^{cd}_{N_\ell}(G, b) = 0$.

5 Analysis

In this section, we show that Colordag satisfies all the blockdag desiderata and is an ε-sure NE (and thus also an ε-NE). Note that it follows directly from the utility definition (Equation 1) that if all agents follow the Colordag protocol, the expected utility of each miner is its relative power. We do the analysis under the assumption that we have a very strong adversary, one who knows the scheduler's protocol. This means that the adversary knows when agents will join and leave the system, when agents will generate blocks, and when messages will arrive. To get this strong guarantee, we may need the parameters N_C and N_ℓ to be large (in general, the choice of N_C and N_ℓ depend on $T_{\max}$). We believe that in practice much smaller parameters will suffice. We return briefly to this issue in the conclusion.

The first step in doing this is to identify a set of "reasonable" histories that has probability at least $1 - \varepsilon$. One of the things that makes a history reasonable is that there is little forking. The whole point of coloring is that we can make the probability of forking arbitrarily small in the graphs of color c, by choosing enough colors.

▶ **Definition 12.** *A pair (b_1, b_2) of blocks is a* natural c-fork *in a history h if b_1 and b_2 both have color c, they are both generated within a window of Δ rounds, and neither is an ancestor of the other in G^h. An interval $[t_1, t_2]$* suffers at most δ-c-forking loss *if, the set of blocks b_1 generated in $[t_1, t_2|$ for which there exists a block b_2 such that (b_1, b_2) is a natural c-fork is a fraction less than δ of the total number of blocks of color c generated in $[t_1, t_2]$.*

We now consider histories that satisfy three properties that will turn out to be key to our arguments.

▶ **Definition 13** (Safe history). *A history is $(N_C, N_\ell, \delta, \delta_C, T_{\max})$-*safe *if, for all miners i, and all colors c,*

SH1. for every subinterval $[t'_1, t'_2]$ of $[0, T_{max}]$, such that at least N_ℓ blocks of color c are generated in the interval $[t'_1, t'_2]$, miner i generates less than $1/2 - \delta$ of them;

SH2. every subinterval $[t'_1, t'_2]$ of $[0, T_{max}]$ such that $t'_2 - t'_1 \geq N_\ell$ suffers at most δ-c-forking loss; and

SH3. for every subinterval $[t'_1, t'_2]$ of $[0, T_{max}]$ such that $t'_2 - t'_1 \geq N_\ell$, there are at least $\delta_C(t'_2 - t'_1)$ blocks of color c generated in $[t'_1, t'_2]$.

Let $H^{N_C,N_\ell,\delta,\delta_C,T_{max}}$ denote the set of histories that are $(N_C, N_\ell, \delta, \delta_C, T_{max})$-safe.

▶ **Proposition 14.** *Suppose that for all miners i, $Pow(i) \leq \alpha < 1/2$. Then for all $\varepsilon > 0$, there exists a positive integer T^*_{max} such that for all $T_{max} \geq T^*_{max}$, there exist N_C, $N_\ell < T_{max}$, $\delta \in (0, 1/2)$, and $\delta_C \in (0, 1)$ such that $\Pr(H^{N_C,N_\ell,\delta,\delta_C,T_{max}}) \geq 1 - \varepsilon$.*

To prove the proposition, we use Hoeffding's inequality to find conditions on the parameters on N_C, N_ℓ, δ, and δ_C for the conditions SH1-SH3 to hold given α and $T_{\max}$ with probability $1-\varepsilon/3$. If all conditions are satisfied, then SH1-SH3 hold with probability at least $1 - \varepsilon$. Finally, we show that such conditions can be found for all sufficiently large $T_{\max}$ values. The proof is deferred to Appendix A.

We say that $(N_C, N_\ell, \delta, \delta_C, T_{\max})$ is *suitable* for ε and α if $\Pr(H^{N_C,N_\ell,\delta,\delta_C,T_{\max}}) \geq 1 - \varepsilon$. We show that $(N_C, N_\ell, \delta, \delta_C, T_{\max})$-safe histories are "good" (in systems where $(N_C, N_\ell, \delta, \delta_C, T_{\max})$ is suitable for the desired ε, and $\alpha < 1/2$). The following propositions show that good things happen in $H^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$. The first one shows that all of blocks generated by honest miners are acceptable.

▶ **Proposition 15.** *For all histories $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{max}}$ and all colors c, there exists a path P from b^0 to b^* in $G_c^{h(t)}$ that contains all blocks of honest miners of color c that are not naturally c-forked. Moreover, every block on P is acceptable.*

Proof. Fix a color c. If b and b' are blocks of honest miners in $G_c^{h(t)}$ that are not naturally forked, then either b is an ancestor of b' or b' is an ancestor of b in $G_c^{h(t)}$. Thus, there is a path P from b^0 to b^* that contains all the blocks of honest miners that are not naturally c-forked (see Figure 3).

Now consider any block b on P. If b is on the canonical longest path P^*, then it is acceptable by definition. Suppose that b is not on P^*. Let b_1 be the last node on P preceding b that is on P^*, and let b_2 be the first node on P following b that is on P^*. Let Q (resp., Q^*) be the subpath of P (resp., P^*) from b_1 to b_2. If the total number of nodes on Q and Q^*, not counting b_1 and b_2, is less that N_ℓ, then the path P' that is identical to P^* up to b_1, continues from b_1 to b_2 along P, and then continues along P^* again, is an N_ℓ-almost optimal path that contains b, showing that b is acceptable.

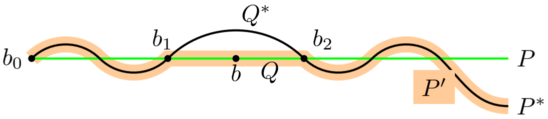


Figure 3 Honest (and hence acceptable) blocks on the path containing all non-forked honest.

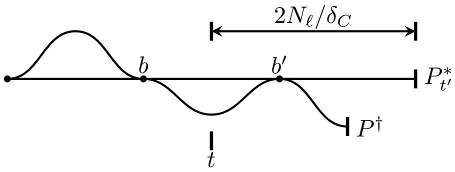


Figure 4 The situation if b' is the only honest block generated after b.

It thus suffices to show that there cannot be more than N_ℓ nodes on Q and Q^*, not counting b_1 and b_2. Suppose, by way of contradiction, that there are. Further suppose that b_1 is generated at time t_1 and b_2 in generated at time t_2. That means that all the blocks on Q and Q^* other than b_1 and b_2 are generated in the interval $[t_1+1, t_2-1]$. Thus, at least N_ℓ blocks are generated in this interval. Since P^* is a longest path, Q^* must be at least as long as Q (otherwise going from b_1 to b_2 along Q would give a longer path). But by Proposition 14, at least a fraction $1/2+\delta$ in the interval $[t_1+1, t_2-1]$ are generated by honest miners. Since there is at most δ-c forking loss, it follows that the majority of the c-colored blocks in this interval are generated by honest miners and are not naturally forked. These blocks must all be on Q. Thus, Q must have a majority of the blocks in this interval, giving us the desired contradiction. ◀

We are now ready to prove that $\mathcal{L}^{\mathrm{cd}}$ satisfies the ledger desiderata (in safe histories) with the Colordag protocol. Note that since we view a miner as representing a coalition of agents, the fact that all but at most one miner is honest means that we allow a coalition with power up to $\alpha < 1/2$ to deviate. The proofs are deferred to Appendix B.

▶ **Proposition 16** (Colordag ledger consistency). *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$ then for all miners i, j and all histories $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, if all but at most one miner is honest in h, $t \le t'$, and $k \le |\mathcal{L}(G_i^{h(t)})| - N_\ell$, then $\mathcal{L}_k(G_i^{h(t)}) = \mathcal{L}_k(G_j^{h(t')})$.*

▶ **Proposition 17** (Colordag ledger growth). *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$, then for all rounds t and t' such that $t' - t \ge N_\ell/\delta_C$, if all but at most one miner is honest in $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, then $|\mathcal{L}^{cd}(G_i^{h(t')})| \ge |\mathcal{L}^{cd}(G_i^{h(t)})| + 1$.*

▶ **Proposition 18** (Colordag ledger quality). *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$ then for all rounds t and t' such that $t' - t \ge 2N_\ell/\delta_C$, and all $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, at least two of the blocks of color $\hat{c}$ added to $\mathcal{L}(G_i^{h(t')})$ in the interval $[t, t']$ are generated by honest miners.*

▶ Note 19. In Propositions 17 and 18, we explicitly assume that we are given an acceptable tuple. Of course, if N_ℓ and δ_C in the tuple are such that $N_\ell/\delta_C > T_{\max}$, then the propositions are essentially vacuous, since there are no times $t, t' < T_{\max}$ such that $t' - t > N_\ell/\delta_C$. Put another way, although it is true that if the system runs for at least N_ℓ/δ_C steps then the ledger is guaranteed to increase in length by 1, given that the system runs for only $T_{\max}$ steps, this is not terribly interesting if $N_\ell/\delta_C > T_{\max}$. Similar comments apply to Proposition 18. The good news is that even for stringent choices of ε and α, there exist suitable tuples that make Propositions 17 and 18 non-vacuous. For example, if $\alpha = .49$ and $\varepsilon = 10^{-7}$, and we assume that $\Delta = 5$, then we can take $T_{\max} = 10^{11}$, $N_\ell = 10^4$, $N_C = 10$, $\delta = .005$, and $\delta_C = 0.04$, to get a suitable tuple, even with the crude analysis in the proof of Proposition 14. In this case, $N_\ell/\delta = 2 \times 10^6$, which is much less than $T_{\max} = 10^{11}$. A more careful analysis should give better numbers, but these suffice to make the point. (As we hinted earlier, with a more realistic adversary, who does not have perfect knowledge of the future, we would also expect far better numbers.) We also note that although Fruitchain does not seem to have an explicit bound $T_{\max}$ on how long the system runs, that bound does arise from the polynomial bound of the p.p.t. environment Z ([18] Section 2.1, *Constraints on (A, Z)*).

The next proposition essentially shows that Colordag is an ε-sure NE.

▶ **Proposition 20.** *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε, $\alpha < 1/2$, $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, and $t_2^i - t_1^i > N_\ell$, then i does not benefit by deviating if all other miners are honest, given revenue scheme $r_{cd}^{N_\ell}$.*

Proof. By Proposition 15, all honest blocks are acceptable in h, no matter what i does. Obviously i can make her own blocks unacceptable, but this would only affect her own revenue and decrease her utility.

It remains to show that i decreases her utility by creating forks. Suppose that M blocks generated in h in the interval $[t_1^i, t_2^i]$ by miners other than i and M' blocks are generated by i. We must have $M > M'$ (SH1). If i does not deviate, then all these blocks are compensated, so i's utility is $\frac{M'}{M+M'}$. If i deviates, i can decrease the utility of the other miners only by forking blocks (since there is nothing that i can do to make a block unacceptable, as we mentioned above). It is easy to see that every block of the other miners that is forked by i comes at a cost of i forking one of his own blocks. Thus, if i deviates so as to fork M'' blocks, then i's utility is $\frac{M'-M''}{M+M'-2M''}$. Since $M'' \leq M' < M$, simple algebra shows that $\frac{M'}{M+M'} > \frac{M'-M''}{M+M'-2M''}$, so this deviation results in the deviator losing utility.

Note that since we assume the deviator knows the history, it can deterministically deviate without affecting the blockdag structure. Hence the equilibrium is not strict. ◀

▶ **Corollary 21.** *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$, then Colordag with this choice of parameters is an ε-sure NE.*

Proof. This is immediate from Proposition 20, since if $(N_C, N_\ell, \delta, \delta_C, T_{\max})$ is suitable for ε and $\alpha < 1/2$, then $\Pr(H^{N_C,N_\ell,\delta,\delta_C,T_{\max}}) \geq 1 - \varepsilon$. ◀

Finally, we prove that the Colordag revenue scheme satisfies revenue consistency. We begin by showing that once a block is deep enough, its revenue is set and does not change.

▶ **Lemma 22.** *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$, then for all miners i, j, all histories $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, all blocks b, and all colors c, if $d(G_{i,c}^{h(t)}, b) \leq d(G_{i,c}^{h(t)}) - 2N_\ell$ and $t \leq t'$, then $r_{N_\ell}^{cd}(G_i^{h(t)}, b) = r_{N_\ell}^{cd}(G_j^{h(t')}, b)$.*

Proof. As in the proof of Proposition 16, let $P_{t'}^*$ be the canonical longest path in $G_{j,c}^{h(t')}$, let P_t be its prefix in $G_{i,c}^{h(t)}$, let P_t^* be the canonical longest path in $G_{i,c}^{h(t)}$, and let b' be the last common block on P_t^* and P_t. As in the proof of Proposition 16, P_t^* and P_t are identical up to b', and we can derive a contradiction if $d(G_{i,c}^{h(t)}, b') \leq d(G_{i,c}^{h(t)}) - N_\ell$, so

$$d(G_{i,c}^{h(t)}, b') > d(G_{i,c}^{h(t)}) - N_\ell. \tag{2}$$

Suppose that b is acceptable in $G_i^{h(t)}$. That means that it is on some N_ℓ-almost optimal path P in $G_{i,c}^{h(t)}$. Let b_1 be the first block on P_t^* that is an ancestor of b, and let b_2 be the first block on P_t^* that is a descendant of b. Perhaps $b_1 = b'$ and perhaps $b_2 = b^*$ (the final block added at the end of the graph). Let Q be the subpath of P from b_1 to b_2, and let Q' be the subpath of P_t^* from b_1 to b_2. Since P is N_ℓ-almost optimal in $G_i^{h(t)}$, it must be the case that $|Q| + |Q'| - 2 < N_\ell$. Since the depth of b is at least N_ℓ less than that of b' (from the proposition statement and from Equation 2), it follows that b_2 must precede b'. Since P_t^* and P_t agree up to b', this argument also shows that $P_{t'}^*$ with Q instead of Q' between b_1 and b_2 is N_ℓ-almost optimal in $G_{j,k}^{h(t')}$, hence that b is acceptable in $G_{j,k}^{h(t')}$. Just changing the roles of $G_i^{h(t)}$ and $G_j^{h(t')}$, this argument shows that if b is acceptable in $G_j^{h(t')}$, then it is also acceptable in $G_i^{h(t)}$.

It is now almost immediate that b is not forked by an acceptable block in $G_i^{h(t)}$ iff it is not forked by an acceptable block in $G_j^{h(t')}$.

In conclusion, block b is acceptable and not forked by an acceptable block in $G_i^{h(t)}$ iff it is acceptable and not forked by an acceptable block in $G_j^{h(t')}$. That is, by the definition of $r_{N_\ell}^{cd}$, it is compensated in $G_i^{h(t)}$ iff it is compensated in $G_j^{h(t')}$. ◀

The next proposition shows that Colordag satisfies revenue consistency.

▶ **Proposition 23** (Colordag Revenue Consistency). *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$, then for all miners i and j and times t, t', and t'' such that $t', t'' > t + 4N_\ell N_C/(\delta_C(1-\delta))$, if b is published at time t in history $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, then $r(G_i^{h(t')}, b) = r(G_j^{h(t'')}, b)$.*

Proof. Suppose that block b is published at time t and has color c. By SH3, within $2N_\ell N_C/(\delta_C(1-\delta))$ rounds, at least $2N_\ell N_C/(1-\delta)$ blocks of color c are generated. By SH1, at least $N_\ell N_C/(1-\delta)$ are honest. By SH2, a fraction $(1-\delta)$ of these are not forked. This means at least $N_\ell N_C$ blocks are not forked, so the depth of G_c has increased by at least $N_\ell N_C$ after $2N_\ell N_C/(\delta_C(1-\delta))$ rounds. Now, for any pair of times $t', t'' > t + 4N_\ell N_C/(\delta_C \delta)$, the depth of the graph is larger by at least $2N_\ell$ than b's depth, therefore, by Lemma 22, the reward for b is the same in both $G_i^{h(t')}$ and $G_j^{h(t'')}$. ◀

6 Conclusion

We present Colordag, a protocol that incentivizes correct behavior of PoW blockchain miners up to 50%, and is an ε-sure equilibrium. That is, unlike previous solutions, the desired behavior is a best response in all but a set of histories of negligible probability. As long as a majority of the participants follow the behavior prescribed by Colordag, the ledger desiderata, as well as reward consistency, all hold.

We prove the properties of Colordag when playing against an extremely strong adversary, one that knows before deviating when agents will generate blocks and when messages will arrive. Intuitively, to benefit from a deviation, a deviator must produce an acceptable path longer than N_ℓ and longer than the honest path. Knowing in advance what order messages can arrive in and whether there is forking means that a deviator knows in advance whether the deviation can succeed. Our analysis shows that, even with this knowledge, a deviation can succeed with only low probability. Unfortunately, to get such a strong guarantee, we may need the parameters N_C and N_ℓ to be quite large Moreover, our ledger is quite inefficient, in that it does not include transactions in blocks that are not on the canonical path in $G_{\hat{c}}$. In practice, we believe that both problems can be dealt with.

We start with the second problem. To improve throughput, we can use ideas that have also appeared in previous work (e.g., [14, 2]): Suppose that b and b' are consecutive blocks on the ledger (which thus both have color $\hat{c}$). When we add b' to the ledger, we also add to the ledger not just the transactions in b', but all the transactions of the acceptable predecessors of b' (of all colors) that were not already included in the ledger. These additional transactions are ordered by the depth of the block they appear in, using color as a tiebreaker, and hash as a second tiebreaker. For example, given the blockdag of Figure 1b, if $\hat{c}$ is blue, then the ledger function includes the transactions in the blocks B_1, Y_1, B_2, R_1, B_3 (in that order); if $\hat{c}$ is red, the ledger includes the transactions in the blocks $B_1, R_1, Y_1, R_2, B_2, Y_2, R_3$ (in that order). It is not hard to check that, with this approach, all transactions in honest blocks of honest agents will be included in the ledger, so our throughput is quite high.

We next consider the fact that we require N_C and N_ℓ to be quite large. This is due to our assumption that a deviator knows what order messages can arrive in and whether there is forking. In practice, a potential deviator will not have this information. For such a weaker adversary, the parameters can be significantly smaller than those required to obtain the bounds presented here. Without this a priori knowledge, the probability that a deviation succeeds drops quickly with N_ℓ. Therefore, the cost of failed attempts grows with N_ℓ, while their overall benefit drops. An analysis of this kind can be done using deep reinforcement learning, which is helpful when the state and action spaces are too rich for an exact solution [11, 3, 4]. This is beyond the scope of this paper, but preliminary experiments suggest that under practical assumptions, with this more limited adversary, Colordag can perform well with reasonable parameter choices. We hope to report on this work in the future.

A The Probability of a Safe History

We prove that a safe history has overwhelming probability.

▶ **Proposition 14.** *Suppose that for all miners i, $Pow(i) \leq \alpha < 1/2$. Then for all $\varepsilon > 0$, there exists a positive integer T^*_{max} such that for all $T_{max} \geq T^*_{max}$, there exist N_C, $N_\ell < T_{max}$, $\delta \in (0, 1/2)$, and $\delta_C \in (0, 1)$ such that $\Pr(H^{N_C, N_\ell, \delta, \delta_C, T_{max}}) \geq 1 - \varepsilon$.*

Proof. We show that there exist constraints on $T_{\max}$, N_C, N_ℓ, and δ_C such that, if the constraints are satisfied, then the probability for the set of histories that have property SH1 (resp., SH2; SH3) is at least $1 - \varepsilon/3$. We then show that these constraints are satisfiable. The result then follows from the union bound.

We start with SH2. Fix a color c, and suppose that there are N_C colors. The probability that a block b has color c is $1/N_C$. To simplify notation in the rest of this proof, we take $\gamma = 1/N_C$. For b to be the earlier of two blocks that are naturally c-forked, there must be another block of color c that is generated within an interval of less than Δ after b is generated. Suppose that b is generated in round r. The probability that a block b generated in round r has color c is γ. The probability that none of the blocks generated in rounds $r + 1, \ldots, r + \Delta - 1$ has color c is $(1 - \gamma)^{\Delta - 1}$, so the probability b is not naturally c-forked is at least $(1 - \gamma)^{\Delta - 1}$.

Fix an interval $[t'_1, t'_2]$. The probability that that $[t'_1, t'_2]$ suffers greater than δ-c-forking loss is exactly the probability that there are fewer than $(1 - \delta)(t'_2 - t'_1)$ blocks of some color c that are naturally forked by a later block. For a fixed color c, by Hoeffding's inequality, this probability is at most $e^{-2(t_2'-t_1')[(t_2'-t_1')((1-\gamma)^{\Delta-1}-\delta)]^2}$. Since we are interested only in the case that $t_2'-t_1' \geq N_\ell$, there are N_C colors, $\gamma = 1/N_C$ and there are at most $\binom{T_{\max}}{2} \leq T_{\max}^2$ possible choices of t_1' and t_2', SH2 holds with probaiblity at least $1-\varepsilon/3$ if

$$N_C T_{\max}^2 e^{-2N_\ell^3((\frac{N_C-1}{N_C})^{\Delta-1}-\delta)^2} < \varepsilon/3. \tag{3}$$

Equation (3) is thus the constraint that needs to be satisfied for SH2.

For SH3, again, fix a color c, and suppose that there are N_C colors. Then the expected number of blocks of color c in an interval $[t_1', t_2']$ is $\gamma(t_2'-t_1')$, so by Hoeffding's inequality, the probability of there being fewer than $\delta_C(t_2'-t_1')$ blocks of color c in the interval $[t_1', t_2']$ is at most $e^{-2(t_2'-t_1')[(t_2'-t_1')(\gamma-\delta_C)]^2}$. Much as in the argument for SH2, it follows that SH3 holds with probability at least $1-\varepsilon/3$ if

$$N_C T_{\max}^2 e^{-2N_\ell^3(\frac{1}{N_C}-\delta_C)^2} < \varepsilon/3. \tag{4}$$

Equation (4) is thus the constraint that needs needs to be satisfied for SH3.

Finally, for SH1, fix $M \geq N_\ell$, K such that $N_\ell \leq K \leq M$, a round t, an miner i, and a color c, and let N_C be the number of colors and $\overline{\alpha}_{i,t,M}$ be i's average power in the interval $[t, t+M]$. Take

$$\delta = (1/2-\alpha)/2. \tag{5}$$

Let $\mathcal{H}_{t,M,K,i}$ consist of all histories where, in the subinterval $[t, t+M]$ of $[0, T_{\max}]$, there are exactly $K \geq N_\ell$ blocks of color c, at least a fraction $1/2-\delta$ of them are generated by miner i. The probability of there being exactly K blocks of color c in the interval is $\binom{M}{K}\gamma^K(1-\gamma)^{M-K}$. Applying Hoeffding's inequality, the probability of being at least $\delta+\alpha$ away from the mean $\overline{\alpha}_{i,t,M}$ is $e^{-2(\delta+\alpha-\overline{\alpha}_{i,t,M})^2K}$. It follows that $\Pr(\mathcal{H}_{t,M,K,i}) \leq \binom{M}{K}\gamma^K(1-\gamma)^{t_2'-K}e^{-2(\delta+\alpha-\overline{\alpha}_{i,t,M})^2K}$.

Let $\mathcal{H}_{t,M,K}$ consist of all histories where, in the interval $[t, t+M]$, there are exactly $K \geq N_\ell$ blocks of color c, and of these, greater than $1/2-\delta$ were generated by some miner i. Thus, $\mathcal{H}_{t,M,K} = \cup_i \mathcal{H}_{t,M,K,i}$, so

$$\Pr(\mathcal{H}_{t,M,K}) \leq \sum_i \Pr(\mathcal{H}_{t,M,K,i}) \leq \sum_i \binom{M}{K}\gamma^K(1-\gamma)^{M-K}e^{-2(\delta+\alpha-\overline{\alpha}_{i,t,M})^2K}.$$

Suppose that

$$N_\ell \geq 4/\delta^2. \tag{6}$$

Then we show that

$$\sum_i e^{-2(\delta+\alpha-\overline{\alpha}_{i,t,M}))^2K} \leq \lceil 1/\alpha \rceil e^{-2\delta^2 K}. \tag{7}$$

To see this, recall that, by assumption, $\overline{\alpha}_{i,t,M} \leq \alpha$, and $\sum_i \overline{\alpha}_{i,t,M} = 1$. Straightforward calculus (details given below) shows that if $\alpha \geq x+z$, $z \leq y \leq x$, and $N > 1/4\delta^2$, then

$$e^{-2(\delta+\alpha-x-z)^2K} + e^{-2(\delta+\alpha-y+z)^2K} \geq e^{-2(\delta+\alpha-x)^2K} + e^{-2(\delta+\alpha-y)^2K}. \tag{8}$$

That is, if $x \geq y$, shifting a little of the weight from y to x increases the sum. It easily follows from this that the sum is maximized if we have as many miners as possible with weight α, and one miner with whatever weight remains. Given that the sum of the weights is 1, we will have roughly $1/\alpha$ miners with weight α. The desired inequality (7) easily follows. Thus,

$$\Pr(\mathcal{H}_{t,M,k}) \leq \binom{M}{K}\gamma^K(1-\gamma)^{M-K}\lceil 1/\alpha\rceil e^{-2\delta^2 K}.$$

Here are the details of the calculation for (8): It's clear that the two sides of the inequality are equal if $z = 0$, So we want to show that the left-hand side increases as z increases. Taking the derivative, it suffices to show that $4(\delta+\alpha-x-z)Ke^{-2(\delta+\alpha-x-z)^2K} - 4(\delta+\alpha-y+z)Ke^{-2(\delta+\alpha-y+z)^2K} \geq 0$ if $z \geq 0$, or equivalently, that $f(z) = (\delta+\alpha-x-z)e^{-2(\delta+(\alpha-x-z)^2K} - (\delta+\alpha-y+z)e^{-2(\delta+\alpha-y+z)^2K} \geq 0$ if $z \geq 0$. We first consider what happens if $z = 0$. We must show that $(\delta+\alpha-x)e^{-2(\delta+\alpha-x)^2K} \geq (\delta+\alpha-y)Ke^{-2(\delta+\alpha-y)^2K}$ if $x \geq y$. The two sides are equal if $x = y$. Taking the derivative with respect to x, it suffices to show that $-e^{-2(\delta+\alpha-x)^2K} + 4(\delta+\alpha-x)^2Ke^{-2(\delta+\alpha-x)^2K} \geq 0$, or equivalently, that $4(\delta+\alpha-x)^2K - 1 \geq 0$. Since $K \geq N_\ell > 1/4\delta^2$ by (5) and $\delta < 1/4$, we have that $f(0) > 0$. Next note that $f'(z) = -e^{-2(\delta+\alpha-x-z)^2K} + 4(\delta+\alpha-x-z)^2Ke^{-2(\delta+\alpha-x-z)^2K} + e^{-2(\delta+\alpha-y+z)^2K} - 4(\delta+\alpha-y+z)^2Ke^{-2(\delta+\alpha-y+z)^2K}$. If $K > 1/4\delta^2$, then $f'(z) = \eta_1 e^{-2(\delta+\alpha-x-z)^2K} - \eta_2 e^{-2((\delta+\alpha-y+z)^2K}$, where $\eta_1 > 0$ and $\eta_2 < 0$. Thus, $f'(z) > 0$, as desired.

Note that $\cup_{\{t,M,K:\ N_\ell \leq K \leq M \leq T_{\max},\ t \leq T_{\max}-M\}}\mathcal{H}_{t,M,K}$ consists of all histories where there are at least N_ℓ blocks of color c and, of these, at least $1/2-\delta$ are generated by some miner i.

$$\begin{array}{ll} & \Pr(\cup_{\{t,M,K:\ N_\ell \leq K \leq M \leq T_{\max},\ t \leq T_{\max}-M\}}\mathcal{H}_{t,M,K}) \\ \leq & \sum_{\{M:\ N_\ell \leq M \leq T_{\max}\}}(T_{\max}-M)\lceil 1/\alpha\rceil \sum_{\{K:\ N_\ell \leq K \leq M\}} \binom{M}{K}\gamma^K(1-\gamma)^{M-K}e^{-2(\delta/2)^2K} \\ \leq & \sum_{\{M:\ N_\ell \leq M \leq T_{\max}\}} T_{\max}\lceil 1/\alpha\rceil e^{-2(\delta/2)^2N_\ell} \sum_K \binom{M}{K}\gamma^K(1-\gamma)^{M-K} \\ \leq & T_{\max}^2\lceil 1/\alpha\rceil e^{-2(\delta/2)^2N_\ell}. \end{array}$$

Since SH1 must holds for all colors c, SH1 holds with probability greater than $1-\varepsilon/3$ if

$$N_C T_{\max}^2\lceil 1/\alpha\rceil e^{-2(\delta/2)^2N_\ell} < \varepsilon/3. \tag{9}$$

To get all of SH1, SH2, and SH3 to hold with probability at least $1-\varepsilon$, we must choose N_ℓ, N_C, $T_{\max}$, δ, and δ_C so that constraints (3), (4), (5), (6), and (9) all hold. Given α, (5) determines δ. We take it to have this value. Recall that $\delta < 1/4$. Given Δ, we next choose N_C sufficiently large such that $(\frac{N_C-1}{N_C})^{\Delta-1} > \frac{1}{2}$. We then choose $\delta_C < \frac{1}{2N_C}$. Finally, for reasons that will become clear shortly, we replace $T_{\max}$ in the equations by N_ℓ^2. (We could equally well have used N_ℓ^k for $k > 2$.) With this replacement and the choices above, we can simplify (3), (4), and (9) to

$$\begin{aligned} N_C N_\ell^4 e^{-2N_\ell^3/16} &< \varepsilon/3 \\ N_C N_\ell^4 e^{-2N_\ell^3(\delta_C/2)^2} &< \varepsilon/ \text{ and} \\ N_C N_\ell^4\lceil 1/\alpha\rceil e^{-2(\delta/2)^2N_\ell} &< \varepsilon/3. \end{aligned} \tag{10}$$

Given N_C, δ, δ_C as determined above, we can clearly choose N_ℓ^* sufficiently large to ensure that these inequalities, together with (6), hold for all $N_\ell > N_\ell^*$. Take $T_{\max}^* = (N_\ell^*)^2$. It follows that for all $T_{\max} \geq T_{\max}^*$, for $\sqrt{T_{\max}} < N_\ell < T_{\max}$, all the constraints hold. This completes the proof. ◀

B Verifying the Colordag Ledger Properties

We prove the three ledger properties.

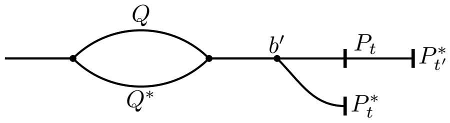


Figure 5 Paths P_t (and $P^*_{t'}$) that are identical to P^*_t up to b'.

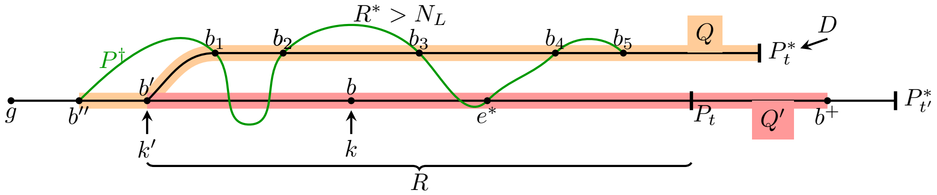


Figure 6 Ledgers in $G^{h(t)}_{i,\hat{c}}$ and $G^{h(t')}_{j,\hat{c}}$ that are identical except for their suffixes.

▶ **Proposition 16** (Colordag ledger consistency). *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$ then for all miners i, j and all histories $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, if all but at most one miner is honest in h, $t \le t'$, and $k \le |\mathcal{L}(G_i^{h(t)})| - N_\ell$, then $\mathcal{L}_k(G_i^{h(t)}) = \mathcal{L}_k(G_j^{h(t')})$.*

Proof. Suppose that $\mathcal{L}_k(G_j^{h(t')}) = b$ and $k \le |\mathcal{L}(G_i^{h(t)})| - N_\ell$. Let $P^*_{t'}$ be the canonical longest path in $G^{h(t')}_{j,\hat{c}}$. Let P_t be its prefix in $G^{h(t)}_{i,\hat{c}}$ and let P^*_t be the canonical longest path in $G^{h(t)}_{i,\hat{c}}$ (see Figure 5).

Let b' be the last common block on P^*_t and P_t. We claim that P^*_t and P_t must be identical up to b'. For if they diverge before b', there must be subpaths Q^* and Q of P^*_t and P_t, respectively, that are disjoint except for their first and last nodes. Since P^*_t and $P^*_{t'}$ are longest paths, we must have $|Q^*| = |Q|$ (if, for example, $|Q^*| > |Q|$, then we can find a path longer that $P^*_{t'}$ by replacing the Q segment by Q^*). The canonical choice will be the same for P^*_t and $P^*_{t'}$, providing the desired contradiction, so the prefixes are the same up to b'.

Let $D = |\mathcal{L}(G_i^{h(t)})|$ (see Figure 6). Since P^*_t is a longest path in $G^{h(t)}_{i,\hat{c}}$, its length is D. Suppose, by way of contradiction, that b is not on P^*_t. Both blocks b and b' are on $P^*_{t'}$, and block b cannot precede b' on its prefix P_t, otherwise it would be on P^*_t. Thus, b' precedes b, and we must have $b' = \mathcal{L}_{k'}(G_i^{h(t)})$, where $k' < D - N_\ell$. Since $|\mathcal{L}(G_i^{h(t)})| = d(G^{h(t)}_{i,\hat{c}})$, it follows that $d(G^{h(t)}_{i,\hat{c}}, b') < D - N_\ell$. (We note for future reference, since it is used in the proof of Proposition 23, that the contradiction comes from this fact.) It follows that the segment R^* of P^*_t from b' to the end must have length greater than N_ℓ. Moreover, if R is the segment of P_t from b' to the end, then R and R^* must be disjoint except for their initial block b'.

We now get a contradiction by considering a path $P^\dagger$ that includes all the honest blocks in $G^{h(t')}_{i,\hat{c}}$ that are not naturally forked. Let b'' be the last block at or preceding b' that is honest and not naturally forked. (If b' is honest and not naturally forked, then $b'' = b'$.) Consider the subpath going from b'' to b' followed by R^*. Call this path Q (highlighted in Figure 6). $P^\dagger$ must intersect Q. For if not, there must be at least as many blocks on Q as there are on $P^\dagger$ generated at or before time t (since P^*_t is the canonical longest path), but none of the blocks on Q other than b'' is an honest block that is not naturally forked. Suppose that b'' is generated at time t''. It follows that in the interval $[t''+1, t]$, fewer honest blocks that are not naturally forked are generated than dishonest blocks, contradicting the assumption that $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$.

Without loss of generality, suppose that, starting at b'', $P^\dagger$ intersects with R^* after it intersects with R. (If $P^\dagger$ does not intersect with R at all, we take R to be the path it intersects with later. The argument is the same if $P^\dagger$ intersects with R after it intersects with R^*.) Let $b_1, b_2, \ldots, b_k$ be the blocks on $P^\dagger$ that are also on R^*, in the order that they appear. For convenience, we take $b_k = b^*$ (the virtual final block). For each pair e, e' of consecutive blocks in $b_1, \ldots, b_k$, the path from e to e' on Q must be at least as long as the path from e to e' on $P^\dagger$ (if $e' = b^*$, we take the path from e to e' on $P^\dagger$ to be the subpath of $P^\dagger$ starting from e and including all the blocks generated at or before time t). It follows that there are at least as many blocks on Q that are not on $P^\dagger$ as there are blocks on $P^\dagger$ that are generated after b'' and at or before time t and are not on Q. We can repeat this process with R to show, roughly speaking, that there are at least as many blocks on R that are not on $P^\dagger$ as there are on $P^\dagger$ that are generated after b'' and at or before time t that are not on R. Suppose that b'' is generated at time t''. It follows that there are at least as many blocks that are either not honest or naturally forked generated between time t'' and t as there are honest blocks that are not naturally forked. This contradicts the assumption that $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$.

The reason that we said "roughly speaking" above is that this argument does not work in one special case. Suppose that the final block on R that is also on $P^\dagger$ is e^*. Further suppose that there are blocks on $P^\dagger$ that are generated at or before time t but after e^*. We cannot conclude that the path from e^* to b^* on R is at least as long as the subpath of $P^\dagger$ consisting of blocks generated after e^* and at or before time t, since R is not necessarily a longest path up to time t.

We deal with this as follows. Let Q' (highlighted in Figure 6) be the segment of $P^*_{t'}$ starting at b' and ending with the first honest block that is not naturally forked that is generated after time t. Call this block b^+. Note that R is a prefix of Q'. Moreover, the subpath of Q' from c to b^+ is indeed at least as long as the subpath of $P^\dagger_{t'}$ from c to b^+. The upshot of this argument is that there are more blocks on Q and R (or Q') that are not on $P^\dagger$ than there are blocks on $P^\dagger$ after b'' that are generated at or before time t (or up to b^+, if we consider Q'). As before, this gives a contradiction to the fact that $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$.

Therefore, our initial assumption was wrong and we conclude that b is on P^*_t. Therefore, it precedes the last common block b' on both P^*_t and $P^*_{t'}$. Since we have shown the two paths coincide until b', it follows that $\mathcal{L}_k(G_i^{h(t)}) = \mathcal{L}_k(G_j^{h(t')})$. ◀

▶ **Proposition 17** (Colordag ledger growth). *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$, then for all rounds t and t' such that $t' - t \geq N_\ell/\delta_C$, if all but at most one miner is honest in $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, then $|\mathcal{L}^{cd}(G_i^{h(t')})| \geq |\mathcal{L}^{cd}(G_i^{h(t)})| + 1$.*

Proof. Suppose that $h \in H^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$. Consider rounds t and t' such that $t' - t \geq 2N_\ell/\delta_C$. Since $t' - t \geq 2N_\ell/\delta_C$ and $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$ there are $K \geq 2N_\ell$ blocks of color $\hat{c}$ generated in this interval. Because h is safe, more than $K/2 \geq N_\ell$ of these blocks are honest and not naturally forked. Let $P^\dagger$ be a path that includes all of these blocks. Let P^*_t denote the canonical longest path of color $\hat{c}$ up to time t. Let b be the last block on P^*_t that is on $P^\dagger$. Let M_0 be the length of P^*_t up to and including b. Suppose that there are M blocks on P^*_t following b, and M' blocks on $P^\dagger$ following b that are generated before time t. Thus, the length of P^*_t is $M_0 + M$. Note that $M \geq M'$ (since P^*_t is a longest path) and

$$M + M' < N_\ell \implies M < N_\ell \tag{11}$$

(otherwise, fewer than half the blocks generated between the time that b was generated and t are honest and not naturally forked, despite the fact that at least N_ℓ blocks are generated in that interval). Now the subpath of $P^\dagger$ up to time t' has length greater than $M_0 + M' + K/2 \geq M_0 + M' + N_\ell$, so the canonical path up to time t' must have at least this length. Thus, for the canonical path up to time t' we have

$$|\mathcal{L}^{\text{cd}}(G_i^{h(t')})| \geq M_0 + M' + N_\ell \geq M_0 + N_\ell \overset{\text{Eq. 11}}{>} M_0 + M = |\mathcal{L}^{\text{cd}}(G_i^{h(t)})| \quad ◀$$

▶ **Proposition 18** (Colordag ledger quality). *If $(N_C, N_\ell, \delta, \delta_C, T_{max})$ is suitable for ε and $\alpha < 1/2$ then for all rounds t and t' such that $t' - t \geq 2N_\ell/\delta_C$, and all $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{max}}$, at least two of the blocks of color $\hat{c}$ added to $\mathcal{L}(G_i^{h(t')})$ in the interval $[t, t']$ are generated by honest miners.*

Proof. As we argued in the proof of Proposition 17, since $t' - t \geq 2N_\ell/\delta_C$, there are at least $2N_\ell$ blocks of color $\hat{c}$ in the interval (by SH3), so we must have at least N_ℓ blocks that are honest and not naturally forked (by SH1 and SH2). Let $P^*_{t'}$ be the canonical longest path up to time t' and let $P^\dagger$ be a path that includes all the honest blocks of color $\hat{c}$ that are not naturally forked up to time t'. Let b be the last honest block that is not naturally forked on $P^*_{t'}$ that is generated prior to time t (b is the genesis block if no other honest blocks on $P^*_{t'}$ are generated prior to time t). We claim that there must be at least two honest blocks that are not naturally forked on $P^*_{t'}$ that come after b. First suppose that there are none. Then there are at least as many blocks on $P^*_{t'}$ that are generated after b as there are on $P^\dagger$ that are generated after b, so, as before, we get a contradiction to the fact that $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$.

Next suppose that there is only one block, say b', on $P^*_{t'}$ that is generated after b that is honest and not naturally forked (see Figure 4). Note that there are more than N_ℓ blocks on $P^\dagger$ after b and hence more than N_ℓ on $P^*_{t'}$ after b (since $P^*_{t'}$ is a longest path). Consider the subpath of $P^*_{t'}$ strictly between b and b' and the subpath of $P^\dagger$ strictly between b and b'. If the total number of blocks on these subpaths is at least N_ℓ, then property SH1 does not hold and we have a contradiction to $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$. If not, then the total number of blocks on the subpath of $P^\dagger$ strictly after b and the subpath of $P^*_{t'}$ strictly after b must be at least N_ℓ, so again we get a contradiction to $h \in H_i^{N_C,N_\ell,\delta,\delta_C,T_{\max}}$. ◀

References

1 Adam Back. Hashcash – a denial of service counter-measure. http://www.cypherspace.org/hashcash/hashcash.pdf, 2002.

2 Vivek Bagaria, Sreeram Kannan, David Tse, Giulia Fanti, and Pramod Viswanath. Prism: Deconstructing the blockchain to approach physical limits. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 585–602, 2019.

3 Roi Bar-Zur, Ameer Abu-Hanna, Ittay Eyal, and Aviv Tamar. Werlman: To tackle whale (transactions), go deep (RL). In *IEEE Symposium on Security and Privacy (SP)*, 2022.

4 Roi Bar-Zur, Danielle Dori, Sharon Vardi, Ittay Eyal, and Aviv Tamar. Deep bribe: Predicting the rise of bribery in blockchain mining with deep RL. In *6th workshop on Deep Learning Security and Privacy (DLSP)*, 2023.

5 Roi Bar Zur, Ittay Eyal, and Aviv Tamar. Efficient MDP analysis for selfish-mining in blockchains. In *2nd ACM Conference on Advances in Financial Technologies (AFT)*, 2020.

6 Reinhard Diestel. *Graph Theory.* Springer Graduate Texts in Mathematics. Springer-Verlag, 5th edition, 2017.

7 Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Proceedings CRYPTO '92: 12th International Cryptology Conference*, pages 139–147. Springer, 1992.

8 Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security*, 2014.

9 Matheus V. X. Ferreira and S. Matthew Weinberg. Proof-of-stake mining games with perfect randomness. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 433–453, 2021.

10 Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin backbone protocol: Analysis and applications. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 281–310, 2015.

11 Charlie Hou, Mingxun Zhou, Yan Ji, Phil Daian, Florian Tramer, Giulia Fanti, and Ari Juels. Squirrl: Automating attack discovery on blockchain incentive mechanisms with deep reinforcement learning. *arXiv:1912.01798*, 2019.

12 Markus Jakobsson and Ari Juels. Proofs of work and bread pudding protocols. In *Secure Information Networks*, pages 258–272. Springer, 1999.

13 Lucianna Kiffer, Rajmohan Rajaraman, and Abhi Shelat. A better method to analyze blockchain consistency. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 729–744, 2018.

14 Yoad Lewenberg, Yonatan Sompolinsky, and Aviv Zohar. Inclusive block chain protocols. In *Financial Cryptography*, Puerto Rico, 2015.

15 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. `http://www.bitcoin.org/bitcoin.pdf`, 2008.

16 Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. *IACR Cryptology ePrint Archive*, 2015:796, 2015.

17 Rafael Pass, Lior Seeman, and Abhi Shelat. Analysis of the blockchain protocol in asynchronous networks. Technical report, Cryptology ePrint Archive, Report 2016/454, 2016.

18 Rafael Pass and Elaine Shi. Fruitchains: A fair blockchain. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 315–324, 2017.

19 Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal selfish mining strategies in Bitcoin. In *Financial Cryptography and Data Security*, 2016.

20 R. Sedgewick and K. Wayne. *Algorithms*. Addison-Wesley, fourth edition, 2011.

21 Jakub Sliwinski and Roger Wattenhofer. Blockchains cannot rely on honesty. `https://disco.ethz.ch/courses/distsys/lnotes/rationalblockchainpaper.pdf`, 2019.

22 Yonatan Sompolinsky, Shai Wyborski, and Aviv Zohar. Phantom ghostdag: a scalable generalization of nakamoto consensus. In *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies*, pages 57–70, 2021.

23 Gavin Wood. Ethereum yellow paper. `https://web.archive.org/web/20160820211734/http://gavwood.com/Paper.pdf`, 2015.

24 H. Yu, Nikolić I., R. Hou, and P. Saxena. Ohie: Blockchain scaling made simple. In *2020 IEEE Symposium on Security and Privacy (SOSP)*, 2020.