Bitcoin
To fork or not to fork: the blockchain's propensity to converge

Bassam El Khoury Seguias

August 28, 2018

1 Introduction

The revolution that has been brought about by **Bitcoin's blockchain** is a direct result of its open nature. Indeed, anyone can be part of it, suggest changes to it, mine new blocks in it, or simply conduct routine validations on it. It is in many respects, the epitome of **decentralization** and **censorship-resistance**. Its appealing nature is in large part rooted in its rich interdisciplinary foundation that spans across philosophy, mathematics and economics.

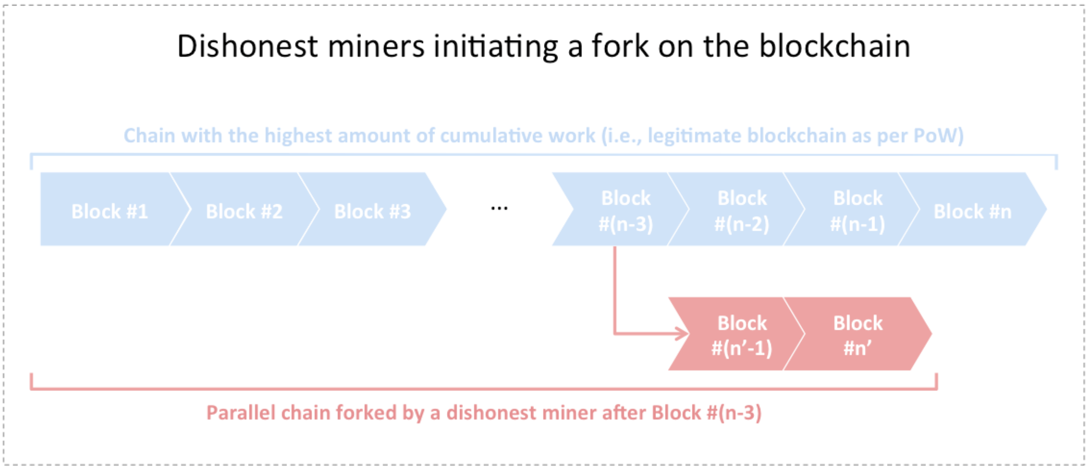
But beyond the elegance of its theoretical underpinning, the blockchain's seamless implementation rests on an inherent agreement between its different participants. Without agreement, this harmonious apparatus would likely decay into chaos. The rather flawless operation of the system is the result of a particular **consensus** protocol known as **Proof of Work** (or PoW for short).

The consensus is meant to be amongst all of the miners on the network. It stipulates that any miner always extend the chain of blocks with the **highest amount of cumulative work**. In this context, work is a measure of the expected computational effort that a miner exerts in order to solve a given cryptographic challenge. In essence, the challenge consists in finding a value that makes the computation emit an output with a mandatory minimum number l of leading 0's. The work associated with mining a given block corresponds to the value 2^l, where l is dynamically adjusted to ensure that the network's average block rate remains constant at ~ 0.00167 blocks / second (i.e., 1 block per 10 minutes). We discuss PoW in more detail in another post.

In an ideal setting where all miners are **honest** (i.e., abide by the PoW consensus protocol) and where blocks are propagated instantaneously on the network, all the nodes will always have a unified view of the blockchain – barring the extremely unlikely scenario of two distinct miners generating two valid blocks at the exact same time. However, imperfections do exist:

- **Imperfection #1**: The network incurs an information propagation delay. As a result, every new block takes a positive amount of time to reach all the other nodes on the network.

- **Imperfection #2**: There exists a subset of dishonest miners that decide to disregard the PoW consensus protocol. Such miners could **fork** the blockchain and start mining on top of a parallel chain different from the one with the highest amount of cumulative work.


Dishonest miners initiating a fork on the blockchain
Chain with the highest amount of cumulative work (i.e., legitimate blockchain as per PoW)
Block #1
Block #2
Block #3
...
Block #(n-3)
Block #(n-2)
Block #(n-1)
Block #n
Block #(n'-1)
Block #n'
Parallel chain forked by a dishonest miner after Block #(n-3)

In the first case, miners are bound to momentarily experience diverging views of the blockchain. Even if all miners were honest, a network propagation delay would still cause **natural forks** to form on the blockchain. This is not desirable because one of the most important tenets of a well-functioning ledger consists in ensuring a unified view of the state of the system at any point in time. We will show in section 2 that the probability of a natural fork occurring at some point in time on a system incurring an information propagation delay is equal to 1. However, we prove that the probability of sustaining a natural fork over a certain time interval is upper-bounded by a quantity that decays exponentially with the length of the interval. Consequently, any natural fork will collapse within finite time with high likelihood. A blockchain subject to the PoW consensus protocol is hence inclined to rapidly settle any natural fork that emerges.

In the second case, dishonest miners could stage an attack and maliciously attempt to redirect the blockchain to another chain of their liking. The likelihood of success of such an attack (also known as a **51% attack**) depends on the hashing power of the dishonest miner or pool of miners. We will discuss this case in section 3.

2 Probability of sustaining a natural fork when all miners are honest

We start by defining the following network parameters:

- Let $q \geq 2$ denote the total number of miners on the network. We let m_i denote the i^{th} miner, $i \in \{1, ..., q\}$.
- Let h_i denote m_i's hashing power, $i \in \{1, ..., q\}$. And let $H \equiv \Sigma_{i=1}^{q} h_i$ denote the network's total hashing power.
- Let λ denote the network's block rate in units of blocks per second. Moreover, let $\lambda_i \equiv (\frac{h_i}{H})\lambda$ denote m_i's specific block rate, $i \in \{1, ..., q\}$. Hence $\lambda = \sum_{i=1}^{q} \lambda_i$.
- Let t_{ij} denote the average time in seconds it takes to propagate a block from m_i to m_j and vice versa (assuming symmetry). We define the network's propagation delay to be $t_p \equiv max_{i,j \in \{1,...,q\}} t_{ij}$. We also define the network's propagation unit to be $t_u \equiv min_{i,j \in \{1,...,q\}} t_{ij}$.
- The probability that m_i generates k blocks ($k \in \mathbb{N}^+$) within an interval of $t > 0$ seconds follows a Poisson distribution with mean equal to $\lambda_i t$ blocks. As a result, $\forall i \in \{1, ..., q\}$ we have:

$$P[m_i \text{ generates } k \text{ blocks within } t \text{ seconds }] = \frac{(\lambda_i t)^k}{k!} e^{-\lambda_i t}$$

We also define the following four events:

- $\mathcal{F}_T \equiv$ *"At least one fork of the blockchain is formed at some point in time $t_f \in (0, T]$"*. The time origin $t = 0$ can be chosen arbitrarily.
- $\mathcal{D}^B \equiv$ *"A fork is formed at block #B of the blockchain. All miners shared a unified view of the blockchain right before #B's addition by a given miner."*
- $\mathcal{S}_T \equiv$ *"A fork that was created at time $t_f > 0$ is sustained (i.e., does not collapse) for a minimum duration of T seconds right after its formation."*
- $\mathcal{DS}_T \equiv$ *"A double-spend transaction coexists with the legitimate transaction, T seconds after the latter got added to a certain block"*.

Note that in what follows, we make the following two assumptions:

1. The PoW consensus protocol consists in extending the chain of blocks that has the highest amount of accumulated work. Theoretically, this is not equivalent to the longest chain (i.e., the chain with the highest number of blocks). However, for all practical matters, we assume that they are equivalent going forward.
2. The block propagation delay t_p as defined earlier could be a large number since there might be nodes on the network that take a large amount of time on average to receive a given block. In general, block propagation statistics are described in terms of percentiles tracking how long it takes to propagate a block to a certain fraction of the network [2]. For all practical matters, we assume that t_p corresponds to the average time taken to propagate a block to 99% of the network.

Our objective is four-fold:

1. We first calculate the probability $P[\mathcal{F}_\infty]$ of a natural fork occurring at some point in time (assuming all miners are honest) and show that it is equal to 1.
2. Second, we derive a non-trivial upper bound on $P[\mathcal{D}^B]$, for any given block $\#B$.
3. Third, we derive a non-trivial upper bound on $P[\mathcal{S}_T]$, decaying exponentially in T.
4. Finally, we derive a non-trivial upper bound on $P[\mathcal{DS}_T]$.

Objective #1: Natural forks on the blockchain happen with probability 1:

- Choose t_m such that $0 < t_m < t_u$. We can write:

$$P[F_\infty] = lim_{n\to\infty} P[F_{nt_m}] = 1 - lim_{n\to\infty} P[\overline{F_{nt_m}}]$$

  where $\overline{F_{nt_m}} \equiv$ *”No fork is formed on the blockchain at any time in the interval* $(0, nt_m]$*”*

- $\forall k \in \{1, ..., n\}$, let $E_{(k,t_m)} \equiv$ *”No fork is formed on the blockchain at any time in the interval* $((k-1)t_m,\ kt_m]$*”*. We can then write:

$$\overline{F_{nt_m}} = \cap_{k=1}^{n} E_{(k,t_m)}.$$

- Let $(E_0)_{(k,t_m)} \equiv$ *”None of the q miners generate any block in the interval* $((k-1)t_m,\ kt_m]$*”*, and let $(E_1)_{(k,t_m)} \equiv$ *”One and only one miner out of q generates at least 1 block in the interval* $((k-1)t_m,\ kt_m]$*”*. We claim that:

$$E_{(k,t_m)} \mid \overline{F_{(k-1)t_m}} \Rightarrow (E_0)_{(k,t_m)} \cup (E_1)_{(k,t_m)}, \quad \text{for all } 2 \leq k \leq n$$

  To see why, note that if $(E_0)_{(k,t_m)}$ and $(E_1)_{(k,t_m)}$ were not true, there would exist at least two miners that generate at least 1 block each in interval $((k-1)t_m,\ kt_m]$. Since $t_m < t_u$, it must be that at least two parallel blocks (one from each miner) coexist, hence forming a fork. Indeed, the choice of t_m guarantees that none of the two blocks will have sufficient time to propagate to the other node. We conclude that $(E_0)_{(k,t_m)} \cup (E_1)_{(k,t_m)}$ is a necessary condition for $E_{(k,t_m)} \mid \overline{F_{(k-1)t_m}}$ to hold. Observing that $(E_0)_{(k,t_m)}$ and $(E_1)_{(k,t_m)}$ are disjoint, we get:

$$P[E_{(k,t_m)} \mid \overline{F_{(k-1)t_m}}] \leq P[(E_0)_{(k,t_m)} \cup (E_1)_{(k,t_m)}] = P[(E_0)_{(k,t_m)}] + [(E_1)_{(k,t_m)}].$$

- We can calculate $P[(E_0)_{(k,t_m)}] = \Pi_{i=1}^{q} e^{-\lambda_i t_m} = e^{-\lambda t_m}$. This quantity is independent of k and so $P[(E_0)_{(k,t_m)}] \equiv P[(E_0)_{t_m}]$

- Similarly, $P[(E_1)_{(k,t_m)}] = \Sigma_{i=1}^{q}[(1 - e^{-\lambda_i t_m})\Pi_{j=1,j\neq i}^{q} e^{-\lambda_j t_m}] = e^{-\lambda t_m}(\Sigma_{i=1}^{q} e^{\lambda_i t_m} - q)$. This quantity is also independent of k and so $P[(E_1)_{(k,t_m)}] \equiv P[(E_1)_{t_m}]$

- Putting it altogether, we get:

$$P[\overline{F_{nt_m}}] = P[\cap_{k=1}^{n} E_{(k,t_m)}] \ = \ \Pi_{k=2}^{n} \left(P[E_{(k,t_m)} \mid \overline{F_{(k-1)t_m}}]\right) \times \ P[E_{(1,t_m)}]$$

$$\leq \Pi_{k=2}^{n} \; (P[E_{(k,t_m)} \mid \overline{F_{(k-1)t_m}}]) \;\leq\; \Pi_{k=2}^{n} \; (P[(E_0)_{t_m}] + P[(E_1)_{t_m}])$$

$$= (P[(E_0)_{t_m}] + P[(E_1)_{t_m}])^{n-1}$$

- $\forall j \in \{0, ..., q\}$, let $(E_j)_{t_m}$ denote the event *"j and only j miners out of q generate at least 1 block each in an interval of t_m seconds"*. One can easily see that for a finite value of t_m, $(E_j)_{t_m} > 0, \; \forall j \in \{0, ..., q\}$. As a result, we have:

$$P[(E_0)_{t_m}] + P[(E_1)_{t_m}] < \Sigma_{j=0}^{q} P[(E_j)_{t_m}] = 1$$

- Consequently, we get:

$$0 \leq P[\overline{F_\infty}] = lim_{n\to\infty} P[\overline{F_{nt_m}}] \leq lim_{n\to\infty}(P[(E_0)_{t_m}] + P[(E_1)_{t_m}])^{n-1} = 0$$

  And since $P[\overline{F_\infty}] = 0$, we conclude that $P[F_\infty] = 1$

Objective #2: A non-trivial upper bound on the probability of a natural fork occurring at an arbitrary block:

Suppose that all miners share a unified view of the blockchain. At time t_B, one of the miners adds block $\#B$ to the blockchain. Miners that still haven't received block $\#B$, (which for all practical matters could take up to $t_B + t_p$) could generate their own block and start a fork at $\#B$. We are interested in computing the probability that such an event occurs, i.e. $P[\mathcal{D}^B]$. We define the following:

- $\mathcal{U}^B \equiv$ *"No fork gets formed at block $\#B$, knowing that all miners shared a unified view of the blockchain right before $\#B$'s addition by some miner."*
- $\mathcal{U}^B_{k,\Delta t} \equiv$ *"None of the miners that still haven't received block $\#B$ by time $t_B + (k+1)\Delta t$, generate any block in time interval $(t_B + k\Delta t, \; t_B + (k+1)\Delta t]$."*
- Let $q_{k,\Delta t}$ be the total number of miners out of a total of q miners, that still haven't seen block $\#B$ by time $t_B + (k+1)\Delta t$. Let these miners be indexed as $\{m_{k_1}, ..., m_{k_{q_{k,\Delta t}}}\}$, and let λ_{k_j} denote the block rate of miner $m_{k_j}, \; j \in \{1, ..., q_{k,\Delta t}\}$.

By letting $\Delta t \to 0$, we can write:

$$\mathcal{U}^B = lim_{\Delta t\to 0}\{\cap_{k=0}^{\infty} \; \mathcal{U}^B_{k,\Delta t}\}$$

And so letting $\lambda_{max} = max_{i=1}^{q} \lambda_i$, we can write:

$$P[\mathcal{U}^B] = lim_{\Delta t\to 0}\{\Pi_{k=0}^{\infty} \; e^{-\Sigma_{j=1}^{q_{k,\Delta t}}(\lambda_{k_j}\Delta t)}\} \;\geq\; lim_{\Delta t\to 0}\{\Pi_{k=0}^{\infty} \; e^{-q_{k,\Delta t}(\lambda_{max}\Delta t)}\}$$

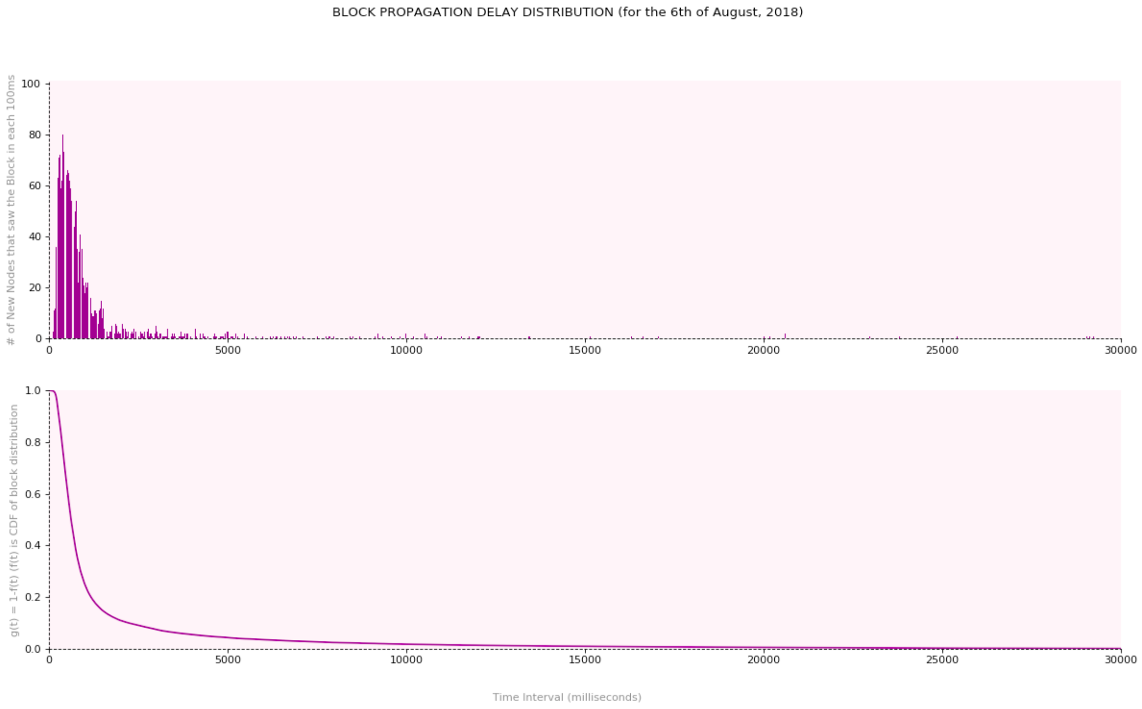
$$= lim_{\Delta t\to 0}\{e^{-\Sigma_{k=0}^{\infty}[q_{k,\Delta t}(\lambda_{max}\Delta t)]}\} \;=\; lim_{\Delta t\to 0}\{e^{-\Sigma_{k=0}^{\infty}[(\frac{q_{k,\Delta t}}{q})(q\lambda_{max}\Delta t)]}\}$$

Letting $g(t)$ be the ratio of miners that still haven't received block $\#B$ by time t, we get:

$$P[\mathcal{U}^B] \geq e^{-q\lambda_{max}(\int_0^\infty g(t)dt)}$$

$$P[\mathcal{D}^B] = 1 - P[\mathcal{U}^B] \leq 1 - e^{-q\lambda_{max}(\int_0^\infty g(t)dt)}$$

Note that if all miners share the same block rate $\frac{\lambda}{q}$, then $q\lambda_{max} = \lambda$. As a result, we would get $P[\mathcal{D}^B] = 1 - e^{-\lambda(\int_0^\infty g(t)dt)}$ (note the equality in this case). For small λ, the upper bound can be approximated by $1 - (1-\lambda)^{\int_0^\infty g(t)dt}$, as found in [1]. Below is a representation of a block propagation delay as observed on the 6^{th} of August 2018 [2]. In order to compute $g(t)$, we make the assumption that the share of miners that still haven't received the block by a certain time is the same as the corresponding share of generic nodes.

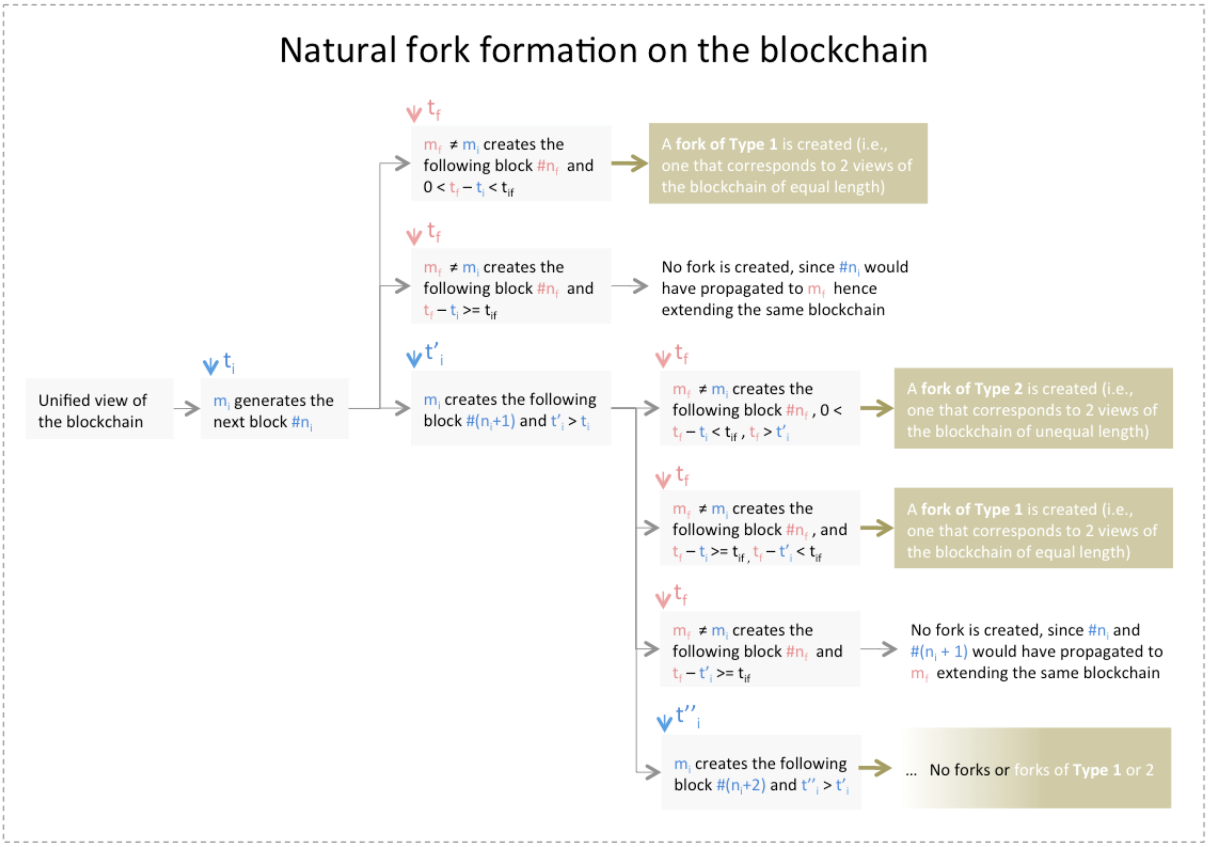


We find that $\int_0^\infty g(t)dt = 1.25$. Using $\lambda = \frac{1}{600}$ blocks/sec, we get $P[\mathcal{D}^B] \leq 0.208\%$.

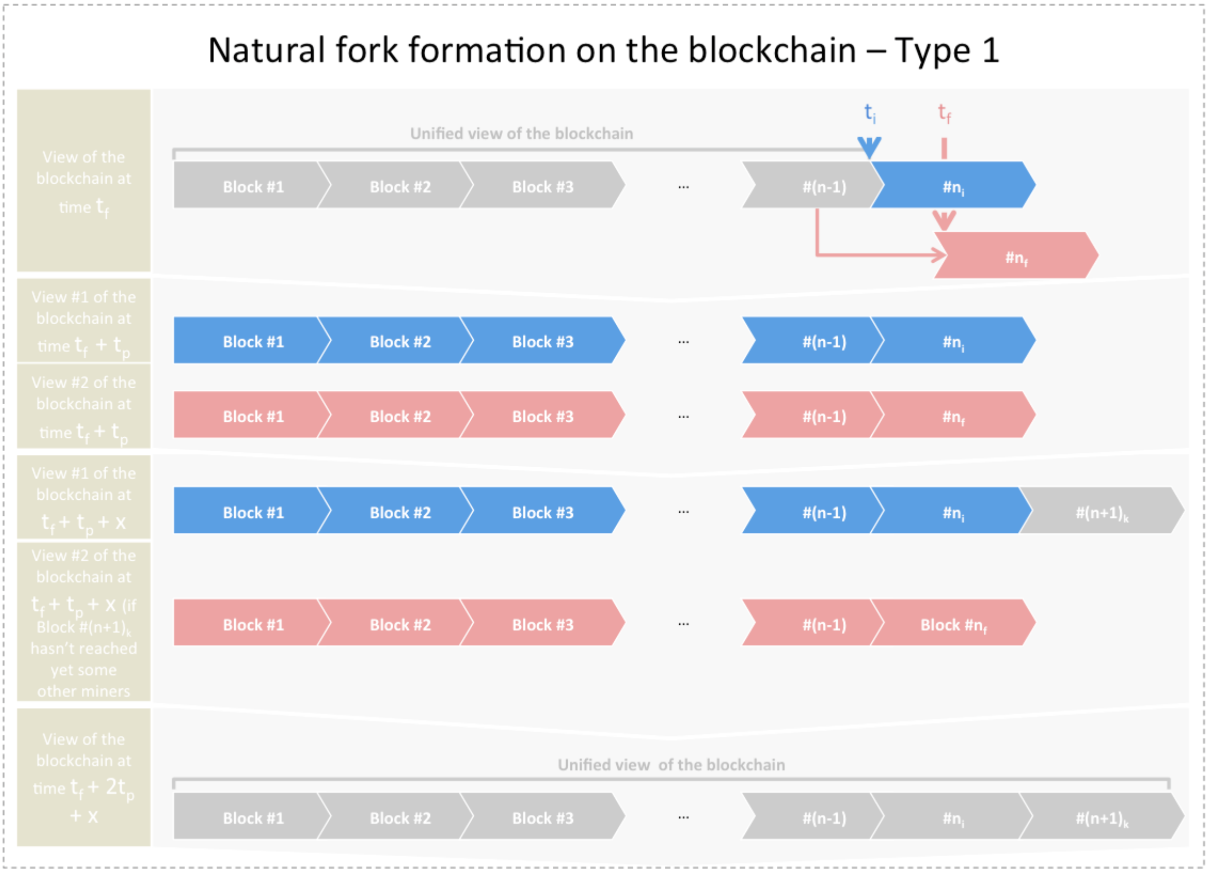
Objective #3: Natural forks collapse with high likelihood in a finite amount of time:

In what follows, we derive a non-trivial upper bound on $P[\mathcal{S}_T]$. To do so, we will first find a lower bound on the probability of occurrence of its complement $\overline{\mathcal{S}_T} \equiv$ "*A fork that was created at time* $t_f > 0$, *collapses at some point in time in the interval* $(t_f,\ t_{f+T}]$". A lower bound on $P[\overline{\mathcal{S}_T}]$ would be given by $P[E]$, where E is an event whose occurrence is a sufficient condition for $\overline{\mathcal{S}_T}$ to hold true. Our objective becomes one of finding an appropriate E and calculating a lower bound on $P[E]$.

In the tree below, we depict the different scenarios that lead to a natural fork formation, starting with a shared view of the blockchain. t_i, t'_i, t''_i and t_f are time instances corresponding to block formations by miners m_i and m_f respectively. Recall that t_{if} denotes the average block propagation time between m_i and m_j. Scenarios that lead to a fork formation are one of two types: those that lead to parallel chains of equal length (i.e., Type 1) and those that lead to chains of different lengths (i.e., Type 2):

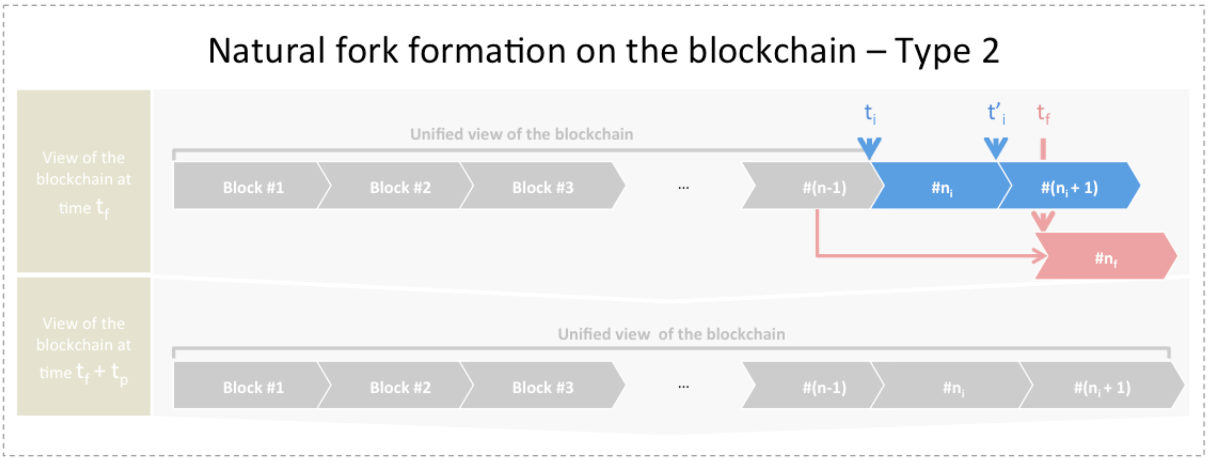

Natural fork formation on the blockchain
Unified view of the blockchain
t_i
m_i generates the next block $\#n_i$
t_f
$m_f \neq m_i$ creates the following block $\#n_f$ and $0 < t_f - t_i < t_{if}$
A **fork of Type 1** is created (i.e., one that corresponds to 2 views of the blockchain of equal length)
t_f
$m_f \neq m_i$ creates the following block $\#n_f$ and $t_f - t_i >= t_{if}$
No fork is created, since $\#n_i$ would have propagated to m_f hence extending the same blockchain
t'_i
m_i creates the following block $\#(n_i+1)$ and $t'_i > t_i$
t_f
$m_f \neq m_i$ creates the following block $\#n_f$, $0 < t_f - t_i < t_{if}$, $t_f > t'_i$
A **fork of Type 2** is created (i.e., one that corresponds to 2 views of the blockchain of unequal length)
t_f
$m_f \neq m_i$ creates the following block $\#n_f$, and $t_f - t_i >= t_{if}$, $t_f - t'_i < t_{if}$
A **fork of Type 1** is created (i.e., one that corresponds to 2 views of the blockchain of equal length)
t_f
$m_f \neq m_i$ creates the following block $\#n_f$ and $t_f - t'_i >= t_{if}$
No fork is created, since $\#n_i$ and $\#(n_i + 1)$ would have propagated to m_f extending the same blockchain
t''_i
m_i creates the following block $\#(n_i+2)$ and $t''_i > t'_i$
… No forks or forks of **Type 1** or 2

Forks of type 1: Before time t_i, all miners share the same view of the blockchain. At t_i, m_i (for some $i \in \{1, .., q\}$) generates block $\#n_i$. At t_f, m_f (for some $f \in \{1, .., q\}, f \neq i$) generates the following block $\#n_f$ such that $t_f - t_i < t_{if}$. If we wait for t_p seconds and no miner generates any block in interval $(t_f, t_f + t_p]$, then at time $(t_f + t_p)$ there will be more than one view of the blockchain, all of which have the same length. If for the following x seconds $(x > 0)$, one and only one $m_k, k \in \{1, ..., q\}$ generates at least one block $\#(n + 1)_k$, and then for the following t_p seconds no miner generates any block, then by time $t_f + x + 2t_p$ all forks would collapse. This is depicted in the figure below:


Natural fork formation on the blockchain – Type 1

View of the blockchain at time t_f
Unified view of the blockchain
t_i
t_f
Block #1
Block #2
Block #3
...
#(n-1)
$\#n_i$
$\#n_f$
View #1 of the blockchain at time $t_f + t_p$
View #2 of the blockchain at time $t_f + t_p$
View #1 of the blockchain at $t_f + t_p + x$
$\#(n+1)_k$
View #2 of the blockchain at $t_f + t_p + x$ (if Block $\#(n+1)_k$ hasn't reached yet some other miners
Block $\#n_f$
View of the blockchain at time $t_f + 2t_p + x$
Unified view of the blockchain

Forks of type 2: Before time t_i all miners share the same view of the blockchain. At t_i, m_i (for some $i \in \{1, .., q\}$) generates block $\#n_i$. At t'_i, m_i mines the next block $\#(n_i + 1)$. At t_f, m_j (for some $j \in \{1, .., q\}, j \neq i$) generates the following block $\#n_j$ $(0 < t_f - t_i < t_{if})$ Two views of the blockchain will coexist at t_f, with one being a longer chain than the other. If we wait for t_p seconds, and no miner generates any block in interval $(t_f,\ t_f + t_p]$, then by $(t_f + t_p)$ all forks would collapse. This is depicted in the figure below:


Natural fork formation on the blockchain – Type 2

View of the blockchain at time t_f
Unified view of the blockchain
t_i
t'_i
t_f
Block #1
Block #2
Block #3
...
#(n-1)
$\#n_i$
$\#(n_i + 1)$
$\#n_f$
View of the blockchain at time $t_f + t_p$
Unified view of the blockchain

One consequence is that given a fork that was formed at time t_f, if we ensure that for the next t_p seconds no miner generates any blocks, and that for the subsequent x

seconds one and only one miner generates at least 1 block, and finally for the subsequent t_p seconds no miner generates any block, then the fork would collapse in the interval $(t_f,\ t_f + x + 2t_p]$. This construction ensures a sufficient condition for a fork to collapse within a specific time interval. In what follows, we formalize our approach:

- Let $k \in \mathbb{N}^+$ and let $x \in \mathbb{R}_+^*$. We refer to x as the **interval design parameter** expressed in units of seconds. We define the following events:

  1. $(E_0)_{(k,x)} \equiv$ *"No miner out of q generates any block in time interval $(t_f + k(x+2t_p),\ t_f + kx + (2k+1)t_p]$"*
  2. $(E_1)_{(k,x)} \equiv$ *"One and only one miner out of q generates at least 1 block in time interval $(t_f + kx + (2k+1)t_p,\ t_f + (k+1)x + (2k+1)t_p]$"*
  3. $(E_2)_{(k,x)} \equiv$ *"No miner out of q generates any block in time interval $(t_f + (k+1)x + (2k+1)t_p,\ t_f + (k+1)(x+2t_p)]$"*
  4. $E_{(k,x)} \equiv$ *"Any fork that could have existed at time $t_f + k(x+2t_p)$ collapses at some time in the interval $(t_f + k(x+2t_p),\ t_f + (k+1)(x+2t_p)]$."*
  5. $\overline{E_{k,x}} \equiv$ *"At least one fork that existed at time $t_f + k(x+2t_p)$ is sustained throughout time interval $(t_f + k(x+2t_p),\ t_f + (k+1)(x+2t_p)]$."*

- The previous logic allows us to conclude that if $(E_0)_{(k,x)}, (E_1)_{(k,x)}$ and $(E_2)_{(k,x)}$ are all satisifed, then $E_{(k,x)}$ will hold true $(k \geq 0)$, and so will $E_{(k,x)} \mid \overline{E_{(k-1,x)}}, ..., \overline{E_{(0,x)}}$ $(k \geq 1)$. We conclude that the event $(E_0)_{(k,x)} \cap (E_1)_{(k,x)} \cap (E_2)_{(k,x)}$ is sufficient for the occurrence of $E_{(k,x)}$ $(k \geq 0)$ and $E_{(k,x)} \mid \overline{E_{(k-1,x)}}, ..., \overline{E_{(0,x)}}$ $(k \geq 1)$. We get:

$$p[E_{(k,x)}],\ p[E_{(k,x)} \mid \overline{E_{(k-1,x)}}, ..., \overline{E_{(0,x)}}] \ \geq \ p[(E_0)_{(k,x)} \cap (E_1)_{(k,x)} \cap (E_2)_{(k,x)}] \equiv$$

$$P\ [\cap_{l=1}^{q}(m_l \text{ does not create any block in time interval } (t_f + k(x+2t_p),\ t_f + kx + (2k+1)t_p]) \cap$$

$$\cup_{i=1}^{q}\ \{m_i \text{ creates at least one block in time interval } (t_f + kx + (2k+1)t_p,\ t_f + (k+1)x + (2k+1)t_p\ ] \cap_{j=1, j\neq i}^{q} (m_j \text{ does not create any block in time interval } (t_f + kx + (2k+1)t_p,\ t_f + (k+1)x + (2k+1)t_p])\}$$

$$\cap_{l=1}^{q}(m_l \text{ does not create any block in } (t_f + (k+1)x + (2k+1)t_p,\ t_f + (k+1)(x+2t_p)])\ ]$$

Recognizing that all the events appearing under the union symbol above are disjoint, and that all the intersections are taken over independent events, we get:

$$p[E_{(k,x)}],\ p[E_{(k,x)} \mid \overline{E_{(k-1,x)}}, ..., \overline{E_{(0,x)}}] \geq$$

$$[\Sigma_{i=1}^{q}(1 - e^{-\lambda_i x})e^{-(\lambda - \lambda_i)x}]e^{-2\lambda t_p} \ = \ e^{-\lambda(x+2t_p)}[\Sigma_{i=1}^{q} e^{\lambda_i x} \ - \ q]$$

- Now note that $\Sigma_{i=1}^{q} e^{\lambda_i x} = q \times \Sigma_{i=1}^{q} \frac{1}{q} e^{\lambda_i x}$. And since the exponential function is convex, we can invoke Jensen's inequality to conclude that $\forall x \in \mathbb{R}_+^*$ we have:

$$\Sigma_{i=1}^{q} e^{\lambda_i x} \geq q \times e^{(\Sigma_{i=1}^{q} \lambda_i x)/q} = q e^{(\lambda x)/q}$$

- As a result, $\forall x \in \mathbb{R}_{+}^{*}$,

$$P[(E)_{(k,x)}],\ p[E_{(k,x)} \mid \overline{E_{(k-1,x)}}, ..., \overline{E_{(0,x)}}] \geq\ qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1]$$

  And so

$$P[\overline{E_{(k,x)}}] = 1 - P[E_{(k,x)}] \leq 1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1], \text{ and}$$

$$P[\overline{E_{(k,x)}} \mid \overline{E_{(k-1,x)}}, .., \overline{E_{(0,x)}}] = 1 - P[E_{(k,x)} \mid \overline{E_{(k-1,x)}}, .., \overline{E_{(0,x)}}] \leq$$

$$1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1]$$

- Let $n \in \mathbb{N}^{+}$. In order to sustain a fork over a minimum duration of $n(x + 2t_p)$ seconds after its formation at time t_f, we must sustain it over each interval of time of the form $(t_f + k(x + 2t_p),\ t_f + (k + 1)(x + 2t_p)],\ \forall k \in \{0, ..., n - 1\}$. This implies that:

$$\mathcal{S}_{n(x+2t_p)}\ \Rightarrow\ \cap_{k=0}^{n-1}\overline{E_{(k,x)}}$$

  As a result, $\forall x \in \mathbb{R}_{+}^{*}$, we can write:

$$P[\mathcal{S}_{n(x+2t_p)}]\ \leq\ P[\cap_{k=0}^{n-1}\overline{E_{(k,x)}}]\ =\ \Pi_{k=1}^{n-1}P[\overline{E_{(k,x)}} \mid \overline{E_{(k-1,x)}}, .., \overline{E_{(0,x)}}]\ \times\ P[\overline{E_{(0,x)}}] \leq$$

$$(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{n}$$

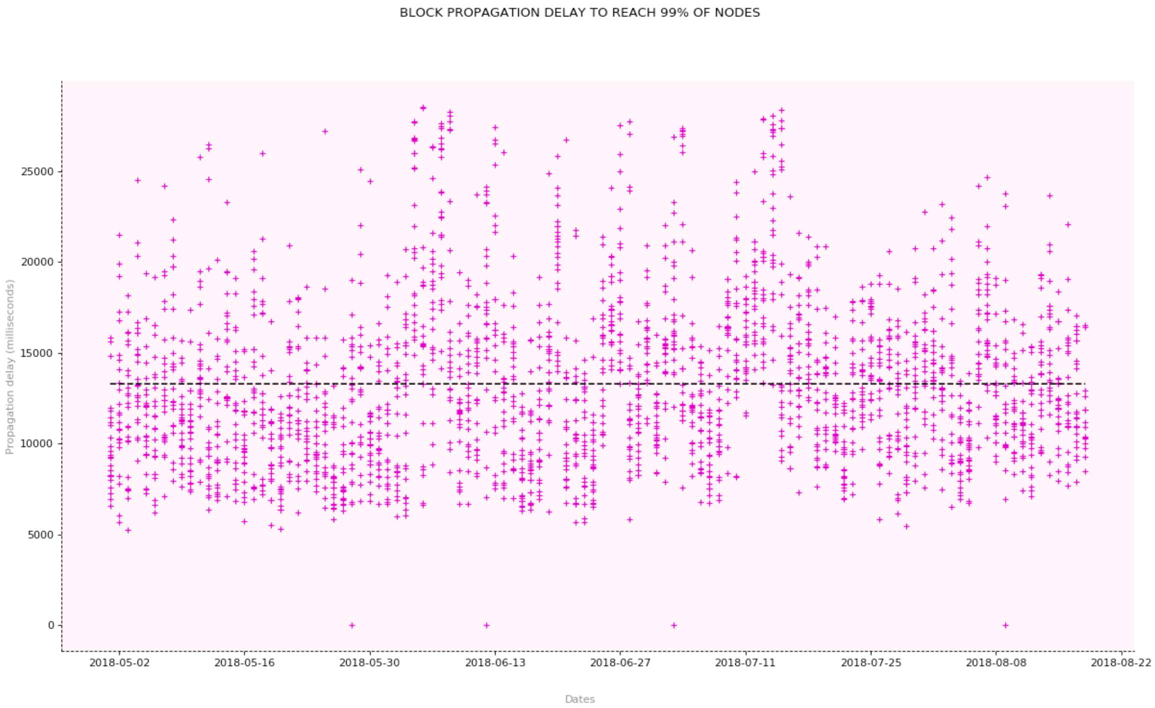
  And in particular, $P[\mathcal{S}_{n(x+2t_p)}]\ \leq\ min_{x>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{n}$

- Finally, for a given time interval of T seconds, we can write:

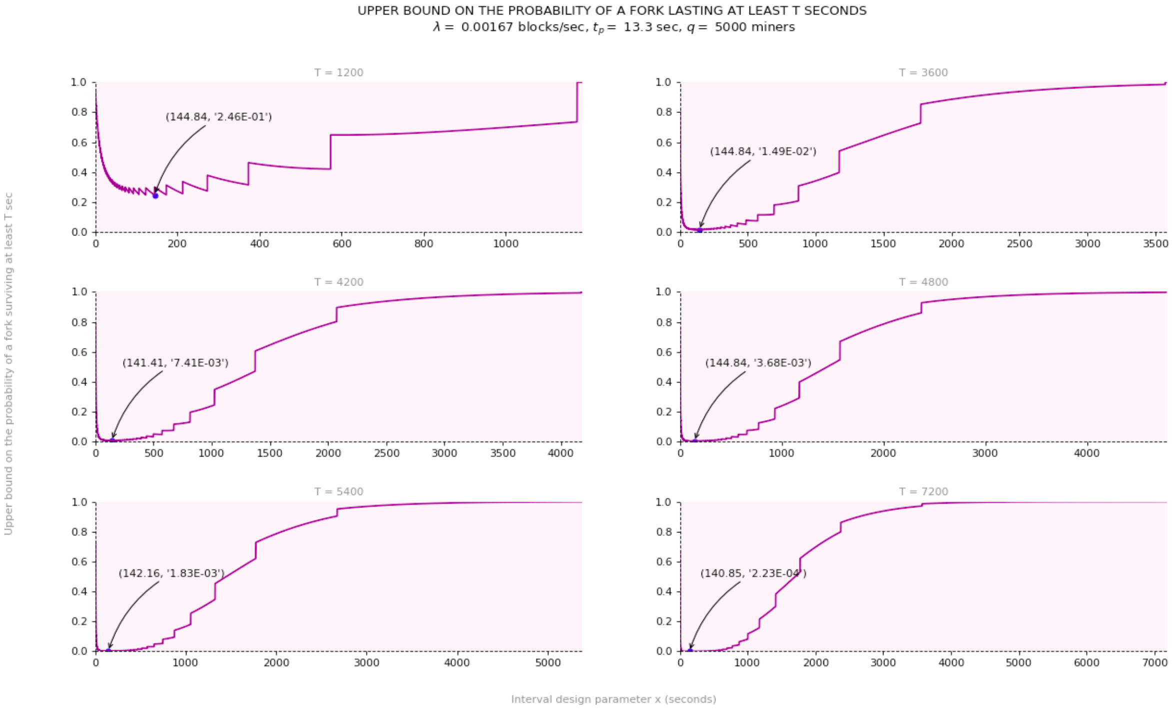
$$P[\mathcal{S}_T]\ \leq\ min_{x>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\lfloor \frac{T}{x+2t_p} \rfloor}$$

  Where $\lfloor . \rfloor$ denotes the floor function.

The objective function on the right-hand side of this inequality is not smooth due to the presence of the floor function. As a result, finding a closed form analytical solution might prove difficult. However, numerical methods could be employed to find the optimal upper bound. Observe that this upper bound decays exponentially with T and eventually converges to 0 when $T \to \infty$. We use the value of $t_p = 13.3s$ that corresponds to the average block propagation delay to reach 99% of the network observed over a period of time extending from May 1st 2018 to August 18th 2018 [2].



Below, we include various graphs of this upper bound for different values of T and for fixed $\lambda = \frac{1}{600}$ $blocks/s,$ $t_p = 13.3s$, and $q = 5000$ $miners$.



These graphs show that for the pre-defined values of λ, t_p and q, the probability that a natural fork survives $T = 20$ minutes (which on average corresponds to the addition of 2 new blocks on the blockchain) is upper-bounded by 0.25 (or almost 1 in 4 cases). When $T = 60$ minutes (i.e., the duration required to add an average of six new blocks on the blockchain), the upper bound goes down to 0.015 (or almost 1 in 67 cases). And when $T = 120$ minutes, it goes down to 0.00022 (or almost 1 in 4500 cases).

In the graphs above, we assumed a fixed block rate λ of 1 block per 10 minutes. This is the value used in the Bitcoin protocol. For what it is worth, one could further optimize the upper bound over positive values of λ. For fixed t_p, q, and T, the tightest upper bound becomes:

$$min_{x,\lambda>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\lfloor \frac{T}{x+2t_p} \rfloor}$$

In order to solve it, we first find the optimal value of $\lambda = \lambda^*(x)$ that solves the following optimization problem:

$$min_{\lambda>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\lfloor \frac{T}{x+2t_p} \rfloor}$$

Note that since the exponent appearing in the objective function does not depend on λ, and since the base is a positive quantity, we can solve the following equivalent optimization problem whose objective function is now smooth:

$$min_{\lambda>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])$$

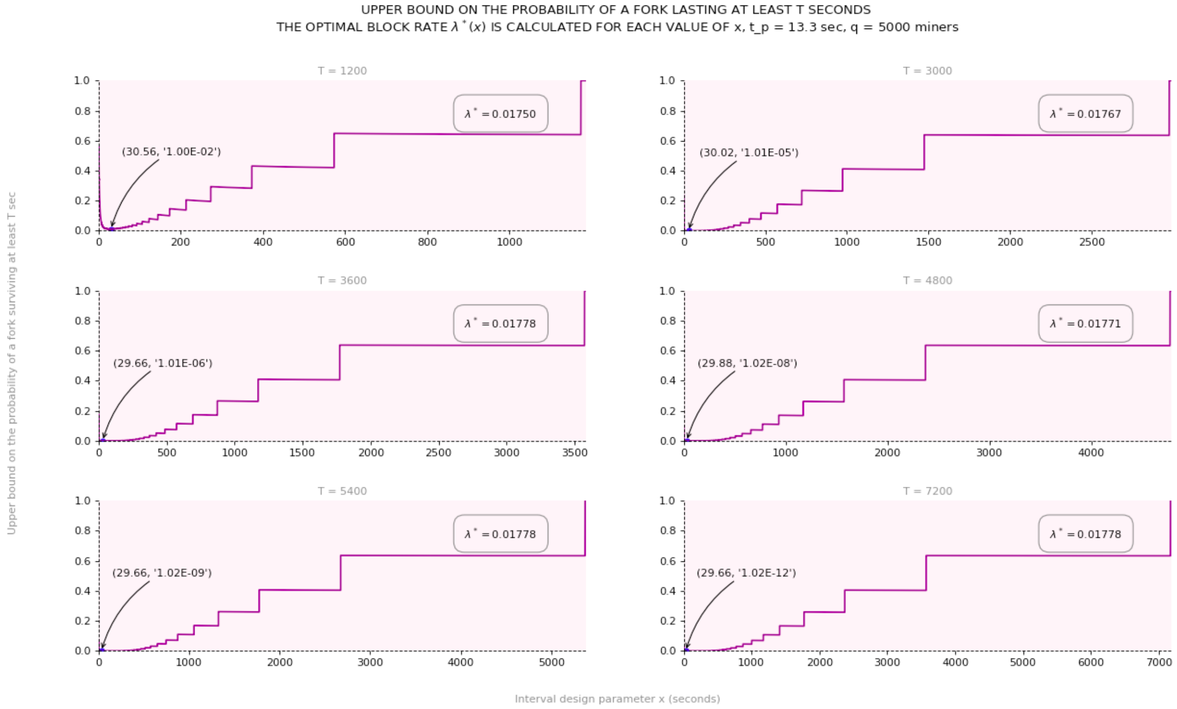
Note that when $\lambda = 0$ or when $\lambda \to \infty$, the objective function tends to 1. The objective function turns out to be convex in λ and we can solve for $\lambda^*(x)$ by setting its first derivative with respect to λ equal to 0. Doing so, yields:

$$\lambda^*(x) = (\tfrac{q}{x})\ ln(\tfrac{q(x+2t_p)}{(q-1)x+2qt_p}).$$

The tightest upper bound over all positive values of x and λ can then be written as:

$$min_{x>0}(1 - qe^{-\lambda^*(x+2t_p)}[e^{(\lambda^* x)/q} - 1])^{\lfloor \frac{T}{x+2t_p} \rfloor}$$

For each of the following graph, we let $t_p = 13.3s,\ q = 5000\ miners.$ and specify a particular value for T. For each value of x, we then calculate the corresponding λ^* as outlined above, and plot the graph of $(1 - qe^{-\lambda^*(x+2t_p)}[e^{(\lambda^* x)/q} - 1])^{\lfloor \frac{T}{x+2t_p} \rfloor}$:



Objective #4: A non-trivial upper bound on the probability of a double-spend transaction coexisting with the legitimate transaction T seconds after the legitimate transaction is added to the blockchain:

The derivations above demonstrate that even if all miners were honest, forks are bound to happen, although they collapse with high likelihood after a finite time of their formation. The existence of natural forks could encourage dishonest customers to engage in double-spending behavior.

To see how, consider a scenario in which a customer spends some satoshis to purchase a physical product. Let the transaction be denoted $\mathcal{T}_{legit}$. Suppose that at $t = 0$, the vendor sees his transaction $\mathcal{T}_{legit}$ included in block $\#B$ for the first time on the blockchain. Suppose that the customer issues another transaction $\mathcal{T}_{malicious}$ (before or after $t = 0$) destined to himself and that uses the same UTXOs as $\mathcal{T}_{legit}$. There is a chance that both $\mathcal{T}_{legit}$ and $\mathcal{T}_{malicious}$ be selected by two different miners and included in two separate coexisting blocks. We define the following double-spending event:

$\mathcal{DS}_T \equiv$ *"$\mathcal{T}_{malicious}$ and $\mathcal{T}_{legit}$ coexist on the blockchain T seconds after $\mathcal{T}_{legit}$'s addition to the blockchain in block $\#B$"*

Note that if no fork is formed at or before block $\#B$, then $\mathcal{T}_{malicious}$ will not constitute a double-spending risk since $\mathcal{T}_{legit}$ would have propagated to all nodes. Consequently, a necessary condition for a double-spending risk to exist consists of the union of the following events:

- $\mathcal{D}^B \equiv$ *"A fork is formed at block $\#B$ and $\mathcal{T}_{malicious}$ is added to this fork at some point in time. All miners shared a unified view of the blockchain right before $\#B$'s addition"*

- $\mathcal{D}^{B_p} \equiv$ *"A fork was formed at some block* $\#B_p$ *preceding* $\#B$*and* $\mathcal{T}_{malicious}$ *is added to this fork at some point in time. All miners shared a unified view of the blockchain right before* $\#B_p$*'s addition"*

We would like to calculate a non-trivial upper bound on $P[\mathcal{DS}_T]$. We can write:

$$P[\mathcal{DS}_T] = P[\mathcal{DS}_T \cap \mathcal{D}^B] + P[\mathcal{DS}_T \cap \mathcal{D}^{B_p}]$$

1. $P[\mathcal{DS}_T \cap \mathcal{D}^B] = P[\mathcal{DS}_T \mid \mathcal{D}^B] \times P[\mathcal{D}^B]$

   We saw earlier that $P[\mathcal{D}^B] \leq 1 - e^{-q\lambda_{max}(\int_0^\infty g(t)dt)}$. Moreover, $\mathcal{DS}_T \mid \mathcal{D}^B \;\Rightarrow \mathcal{S}_T$.

   And so $P[\mathcal{DS}_T \mid \mathcal{D}^B] \leq \; P[\mathcal{S}_T] \;\leq\; min_{x>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\lfloor \frac{T}{x+2t_p} \rfloor}$

   Assuming that $T > x + 2t_p$, we get

   $$P[\mathcal{DS}_T \mid \mathcal{D}^B] \leq \; min_{x>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\frac{T}{x+2t_p}-1}$$

2. Let k be a non-negative integer and let Δt be an arbitrary interval of time. Without loss of generality, assume block $\#B$ was added at $t = 0$. Define $\mathcal{D}^{B_{p,k}} \equiv$ *"Block* $\#B_p$ *preceding* $\#B$ *was created in interval* $[-(k+1)\Delta t, -k\Delta t)$*, a fork was formed at* $\#B_p$ *and* $\mathcal{T}_{malicious}$ *is added to this fork at some point in time. All miners shared a unified view of the blockchain right before* $\#B_p$*'s addition".*

   We can write:

   $$P[\mathcal{DS}_T \cap \mathcal{D}^{B_p}] = lim_{\Delta t \to 0}\, \Sigma_{k=0}^{\infty}\, P[\mathcal{DS}_T \cap \mathcal{D}^{B_{p,k}}]$$

   $$= lim_{\Delta t \to 0}\, \Sigma_{k=0}^{\infty}\, P[\mathcal{D}^{B_{p,k}}] \times P[\mathcal{DS}_T \mid \mathcal{D}^{B_{p,k}}]$$

   Note that

   - $P[\mathcal{D}^{B_{p,k}}] \;=\; P[\mathcal{D}^{B_{p,k}} \;\cap\; \mathcal{D}^{B_p}] \;=\; P[\mathcal{D}^{B_p}] \;\times\; P[\mathcal{D}^{B_{p,k}} \mid \mathcal{D}^{B_p}]$.
   - $\mathcal{D}^{B_{p,k}} \mid \mathcal{D}^{B_p} \;\Rightarrow$ At least one block is added in interval $[-(k+1)\Delta t, \; -k\Delta t)]$.
   - $\mathcal{DS}_T \mid \mathcal{D}^{B_{p,k}} \;\Rightarrow\; \mathcal{S}_{T+k\Delta t}$

   We then have:

   $$P[\mathcal{D}^{B_{p,k}}] \leq P[\mathcal{D}^{B_p}] \times P[\text{ At least one block is added in } [-(k+1)\Delta t, \; -k\Delta t)]$$

   As a result,

   $$P[\mathcal{DS}_T \cap \mathcal{D}^{B_p}] \leq$$

   $$lim_{\Delta t \to 0}\, \{P[\mathcal{D}^{B_p}] \times (1 - e^{-\lambda \Delta t})\, \Sigma_{k=0}^{\infty}\, min_{x>0}(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\lfloor \frac{T+k\Delta t}{x+2t_p} \rfloor}\}$$

Assuming $T > x + 2t_p$, we get $P[\mathcal{DS}_T \cap \mathcal{D}^{B_p}] \leq$

$$lim_{\Delta t \to 0} \; \{P[\mathcal{D}^{B_p}] \times (1 - e^{-\lambda \Delta t}) \; \Sigma_{k=0}^{\infty} \; min_{x>0} (1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\frac{T+k\Delta t}{x+2t_p} - 1}\}$$

In the limit when Δt becomes infinitesimally small and tends to dt, the quantity $(1 - e^{-\lambda \Delta t})$ tends to λdt. We can then write

$$P[\mathcal{DS}_T \cap \mathcal{D}^{B_p}] \leq P[\mathcal{D}^{B_p}] \; min_{x>0} \int_0^{\infty} \lambda (1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\frac{T+t}{x+2t_p} - 1} dt$$

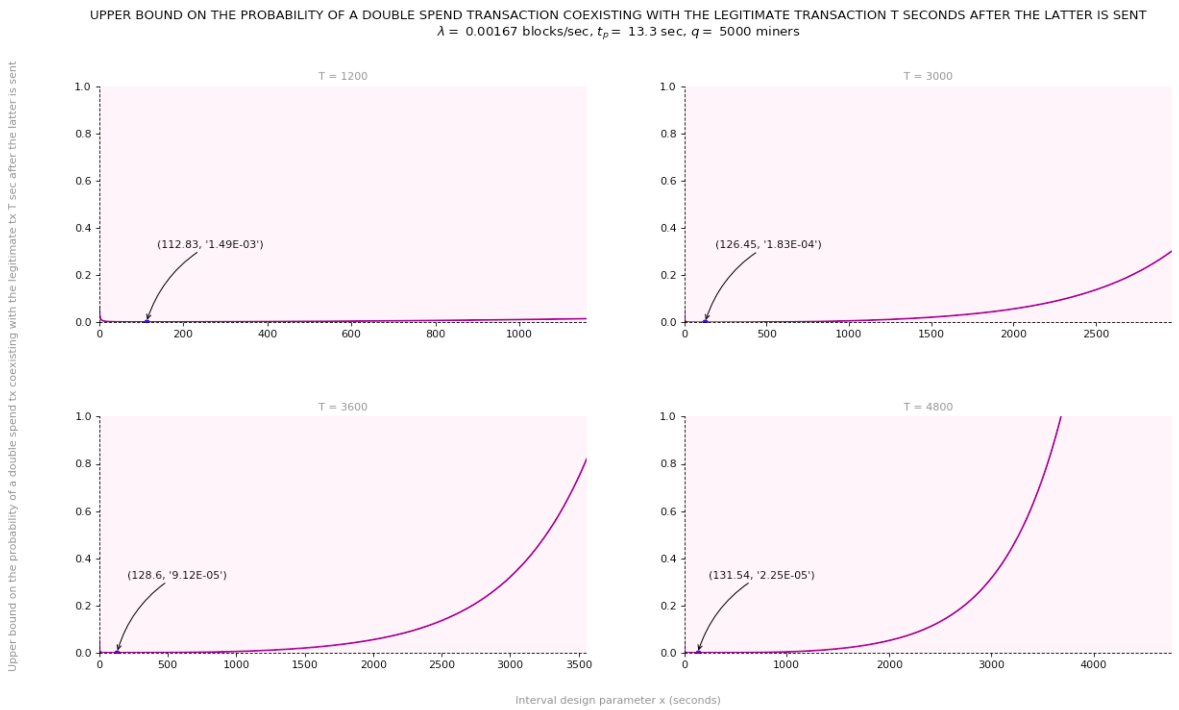
$$= P[\mathcal{D}^{B_p}] \; min_{x>0} \; \frac{-\lambda(x+2t_p)}{ln(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])} (1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\frac{T}{x+2t_p} - 1}$$

Similar to $P[\mathcal{D}^{B}]$, we have $P[\mathcal{D}^{B_p}] \leq 1 - e^{-q\lambda_{max}(\int_0^{\infty} g(t)dt)}$.

We then conclude that $P[\mathcal{DS}_T] \leq$

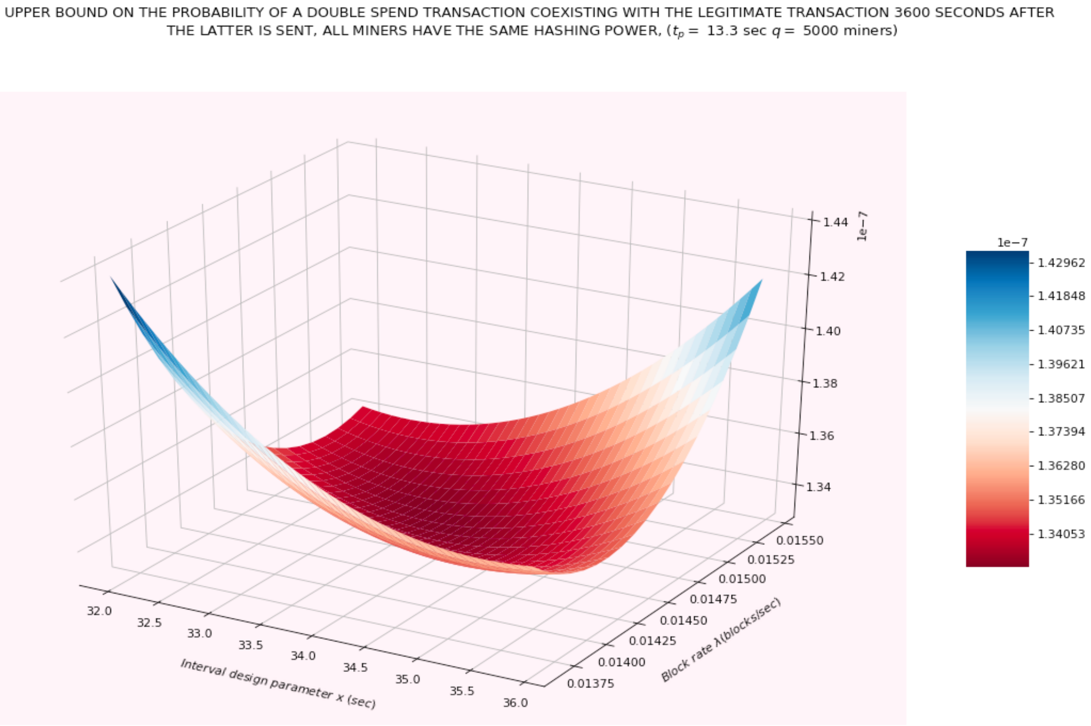
$$min_{x>0} \; (1 - e^{-\lambda(\int_0^{\infty} g(t)dt)}) \times (1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])^{\frac{T}{x+2t_p} - 1} (1 - \frac{\lambda(x+2t_p)}{ln(1 - qe^{-\lambda(x+2t_p)}[e^{(\lambda x)/q} - 1])})$$

The graphs below depict the upper bound on the probability that a double spend transaction $\mathcal{T}_{malicious}$ coexists with $\mathcal{T}_{legit}$ on the blockchain T seconds after $\mathcal{T}_{legit}$ is added to block $\#B$, for various values of x and T.



The above graphs show that if before handing over the goods, the vendor waits for an additional $T = 20$ minutes after he sees $\mathcal{T}_{legit}$ added to block $\#B$ on the blockchain (which at a block rate of $\lambda = \frac{1}{600}$ $blocks/s$ would roughly correspond to two additional blocks on top of $\#B$), then for $\lambda = \frac{1}{600}$ $blocks/s$, $t_p = 13.3s$, $q = 5000$, and all miners being honest and sharing the same hash rate, the probability of a double-spend attempt coexisting with $\mathcal{T}_{legit}$ at time T is upper-bounded by 0.00149. That means that there is at most a probability of 0.00149 that $\mathcal{T}_{malicious}$ is still part of a parallel fork. By virtue of being sustained, it could still become part of the longest chain. If this happens, $\mathcal{T}_{legit}$ gets thrown back into the mempool and $\mathcal{T}_{malicious}$ gets validated instead. On the other hand, waiting for an an additional $T = 50$ minutes (i.e., roughly 5 additional blocks on top of $\#B$), would bring down this probability to 0.000183.

One could also optimize the upper bound not only over x but also over λ. Below is a plot showing the optimal value for the case $T = 60$ minutes. In this case, the tightest upper bound is 1.33 $\times 10^{-7}$, which is achieved for $\lambda \approx 0.01476$ blocks / sec and $x \approx 33.5$ sec.



UPPER BOUND ON THE PROBABILITY OF A DOUBLE SPEND TRANSACTION COEXISTING WITH THE LEGITIMATE TRANSACTION 3600 SECONDS AFTER THE LATTER IS SENT, ALL MINERS HAVE THE SAME HASHING POWER, ($t_p =$ 13.3 sec $q =$ 5000 miners)

A small note on the value of λ : To the extent of our knowledge, Satoshi's choice of $\lambda = \frac{1}{600}$ $blocks/s$ is not the result of a pure mathematical optimization exercise. The larger the value of λ, the higher the probability of a natural fork occurring. Natural forks are not desirable as they possibly pave the way to double-spending attempts. Another important consideration has to do with the storage capacity requirement and the rate of growth of such capacity that needs to be maintained at the level of each full node on the network. A higher λ means faster transaction processing but also faster growth of ever-increasing storage requirement. It is most likely that only a handful of nodes would be able to afford such storage requirements, subsequently leading to a centralization scenario. This stands in sharp contrast with Bitcoin's fundamental philosophy. On the other hand, a very small λ would cause the network to be painfully slow. As a result of this tradeoff, Satoshi's choice of $\lambda = \frac{1}{600}$ $blocks/s$ is probably a good compromise.

3 Probability that dishonest miner(s) succeed in creating a dominant fork: 51% attack

In this section we look at the second type of imperfections. More specifically, we turn to the possibility that a subset of dishonest miner(s) decide to disregard the PoW consensus protocol and mine on top of a parallel chain different than the one with the highest amount of cumulative work.

This type of behavior has been introduced and analyzed in section 11 of Nakamoto's seminal paper [3]. The analysis demonstrates that dishonnest miner(s) could possibly generate a parallel chain that overtakes the original honest chain. As a consequence, malicious miner(s) could potentially engage in double spending behavior.

Such a scenario is commonly referred to as a **51% attack**, although malicious miner(s) do not necessarily need 51% of the total hashing power of the network to launch a double spending attack. A control of **51% or more** of the total hashing power will however **guarantee** that the attack will be successful. On the other hand, control of **less than 51%** is not associated with a deterministic state of success, but rather a **probabilistic** one. The analysis in [3] quantifies the probability of success as a function of said control. In this section, we simply clarify the mathematical foundation of this analysis.

Building on the notation used in section 2, we define the following quantities:

- Let $\{m_{i_1}, ..., m_{i_s}\} \subset \{m_1, ..., m_q\}$ be the subset of malicious miner(s) staging a 51% attack.
- Let $r = \frac{\Sigma_{j=1}^{s} \lambda_{i_j}}{\lambda}$ denote the fraction of the total hashing power controlled by the malicious miner(s). Note that in [3], this quantity is denoted by q (in our notation, q refers to the total number of miners).
- Let $p = 1 - r$ denote the fraction of the total hashing power controlled by the honest miners.
- Let $\mathcal{T}_{legit}$ correspond to the legitimate transaction against which $\{m_{i_1}, ..., m_{i_s}\}$ will stage a 51% attack, and let $\#B$ be the block containing $\mathcal{T}_{legit}$.
- Let $(\Delta t)_z$ denote the interval of time extending from the moment $\mathcal{T}_{legit}$ was added to $\#B$ to the time the z^{th} subsequent block following $\#B$ got added to the honest chain.

We also define the following two events:

1. $\mathcal{A}_{r,z}^{51} \equiv$ *"The subset of malicious miner(s) with a fraction r of the network's total hashing power conduct a successful 51% attack on $\mathcal{T}_{legit}$, and $\mathcal{T}_{legit}$ has been previously validated by the addition of z blocks on top of $\#B$ on the honest chain."*
2. $\mathcal{M}_{k,r,z} \equiv$ *"The subset of malicious miner(s) with a fraction r of the network's total hashing power created and added k blocks to the parallel chain in interval $(\Delta t)_z$."*

We can write:

$$P[\mathcal{A}^{51}_{r,z}] = \Sigma_{k=0}^{\infty}\ P[\mathcal{A}^{51}_{r,z}\ \cap\ \mathcal{M}_{k,r,z}] = \Sigma_{k=0}^{\infty}\ P[\mathcal{A}^{51}_{r,z}\ |\mathcal{M}_{k,r,z}] \times P[\mathcal{M}_{k,r,z}]$$

Calculating $P[\mathcal{M}_{k,r,z}]$: Given a network block rate of λ blocks/sec, the rate associated with honest miners is $p\lambda$ blocks/sec, and that associated with malicious miner(s) is $r\lambda$ blocks/sec. As a result, $(\Delta t)_z = \frac{z}{p\lambda}$ sec. In this interval, the subset of malicious miner(s) generate an average of $(r\lambda).(\frac{z}{p\lambda}) = \frac{r}{p}z$ blocks. As a result, we can model malicious miner(s) block generation over this inteval as a Poisson process with mean $\frac{r}{p}z$ blocks. We get:

$$P[\mathcal{M}_{k,r,z}] = \frac{(\frac{r}{p}z)^k\ e^{-\frac{r}{p}z}}{k!}$$

Calculating $P[\mathcal{A}^{51}_{r,z}\ |\mathcal{M}_{k,r,z}]$: Knowing that in interval $(\Delta t)_z$ the malicious miner(s) generated k blocks on the parallel chain, we need to calculate the probability that the malicious miner(s) catch-up to the honest chain and generate a parallel chain that is at least as long (note that technically speaking, the parallel chain should be one block longer than the honest chain for the attack to be successful, but Nakamoto's analysis considers the case of equal length instead). Clearly, if $k \geq z$, the probability is 1. When $k < z$, we can model the process as a **binomial random walk** whereby given that the malicious miner(s)'s parallel chain is $(z - k)$ blocks shorter than the honest chain:

a Every time the malicious miner(s) generate a new block, the gap gets reduced by 1.

b Every time the honest miners generate a new block, the gap gets increased by 1.

c Otherwise, the gap remains the same.

This problem turns out ot be a slight variant of the **Gambler's Ruin Problem** that we introduce next.

- **The Gambler's Ruin Problem**:
  - The setting consists of a gambler who initially has $x > 0$ units of currency (UOC for short).
  - The gambler engages in a series of bets whereby every time she wins she gets UOC 1, and every time she loses, she gives UOC 1.
  - Suppose that the gambler has a probability $0 \leq w < 1$ of winning each bet and a probability $l = 1 - w$ of losing. All bets are mutually independent.
  - The objective is for the gambler to reach a fortune of UOC $f > x$ before losing all her capital. As a result, the process will stop if either the gambler accumulates UOC f or if she sees her capital reduced to 0.

  In what follows, we calculate the probability that the gambler wins knowing that she started the game with UOC x. This is the probability that she reaches a fortune of UOC f at some point in time knowing that she started off with UOC x.

We denote it by $w_{(f,x)}$. A similar derivation can be found in [5] and [4]. Note that in this version of the game, the gambler cannot play if she does not have a positive amount of capital to start the betting process with. This stands in contrast to the aforementioned situation where malicious miner(s) start off with a block deficit. Later on, we will account for this variation.

We start by defining the following events:

{ $\mathcal{W} \equiv$ *"The gambler wins"*

{ $\mathcal{W}_1 \equiv$ *"The gambler wins the first bet in the series"*

{ $\mathcal{L}_1 \equiv$ *"The gambler looses the first bet in the series"*

And let X be a random variable denoting the amount of capital held by the gambler at the beginning of the game.We can write:

$$P[\mathcal{W} \mid X = x] = P[\mathcal{W} \cap \mathcal{W}_1 \mid X = x] + P[\mathcal{W} \cap \mathcal{L}_1 \mid X = x]$$

$$= P[\mathcal{W} \mid X = x,\ \mathcal{W}_1] \times P[\mathcal{W}_1 \mid X = x] + P[\mathcal{W} \mid X = x,\ \mathcal{L}_1] \times P[\mathcal{L}_1 \mid X = x]$$

$$= P[\mathcal{W} \mid X = x+1] \times P[\mathcal{W}_1 \mid X = x] + P[\mathcal{W} \mid X = x-1] \times P[\mathcal{L}_1 \mid X = x]$$

In other terms, we have:

$$w_{(f,x)} = w.w_{(f,x+1)}\ +\ l.w_{(f,x-1)}$$

$$\iff (w+l).w_{(f,x)} = w.w_{(f,x+1)}\ +\ l.w_{(f,x-1)}$$

$$\iff (w_{(f,x+1)} - w_{(f,x)}) = (\tfrac{l}{w}).(w_{(f,x)} - w_{(f,x-1)})$$

$$\Rightarrow w_{(f,x+1)} - w_{(f,x)} = (\tfrac{l}{w})^x.\ w_{(f,1)},\ \text{ since } w_{(f,0)} = 0$$

We can write:

$$w_{(f,x+1)} - w_{(f,1)} = \Sigma_{j=1}^{x}(w_{(f,j+1)} - w_{(f,j)}) = [\Sigma_{j=1}^{x}(\tfrac{l}{w})^j].\ w_{(f,1)}$$

And so, $w_{(f,x)} = [\Sigma_{j=0}^{x-1}(\frac{l}{w})^j].\ w_{(f,1)} =$

$$\{\ [\tfrac{1-(\frac{l}{w})^x}{1-\frac{l}{w}}].\ w_{(f,1)},\ \text{ if } w \neq l$$

$$\{\ x.\ w_{(f,1)},\ \text{ if } w = l = \tfrac{1}{2}$$

Noting that $1 = w_{(f,f)}$ and applying the above when $x = f$, we get

$$\{\ 1 = [\frac{1-(\frac{l}{w})^f}{1-\frac{l}{w}}].\ w_{(f,1)}\ ,\ \text{if}\ w \neq l$$

$$\{\ 1 = f.\ w_{(f,1)},\ \text{if}\ w = l = \tfrac{1}{2}$$

Which then allows us to conclude that $w_{(f,1)} =$

$$\{\ [\frac{1-\frac{l}{w}}{1-(\frac{l}{w})^f}],\ \text{if}\ w \neq l$$

$$\{\ \frac{1}{f},\ \text{if}\ w = l = \tfrac{1}{2}$$

Putting it altogether, we get $w_{(f,x)} =$

$$\{\ [\frac{1-(\frac{l}{w})^x}{1-(\frac{l}{w})^f}],\ \text{if}\ w \neq l$$

$$\{\ \frac{x}{f},\ \text{if}\ w = l = \tfrac{1}{2}$$

- **Adapting the Gambler's Ruin Problem to the case of a 51% attack**: In the context of a 51% attack, and for $k < z$, $P[\mathcal{A}^{51}_{r,z}\ |\mathcal{M}_{k,r,z}]$ is the probability that the subset of malicious miner(s) catch up to the honest chain when the malicious miner(s)' parallel chain is $(z-k)$ blocks behind. In other terms, the malicious miner(s) have a deficit of $(z-k)$ blocks at the beginning of the process and we are interested in calculating the probability that they catch-up at some point in time. Note that under this setting, the block deficit may widen without having any lower-bound constraint.

  In the setting of the Gambler's Ruin Problem, the gambler cannot play when she is in a deficit and will have to stop as soon as her betting capital reaches 0. As such, the problem must be modified to account for a scenario where the gambler could borrow unlimited credit if need be, to continue playing the game.

  An equivalent formulation consists of a gambler having an infinite amount of capital to start with. Formally, we assume the same setting as the original problem. The gambler however, has a deficit of UOC d. She then borrows UOC $x > d$ and starts the game with the objective of reaching UOC f where $f = x + d$. If she wins, she returns UOC x to her creditor and keeps UOC d so as to break-even. In case she loses, she does not return anything to her creditor.

  We then let $x \to \infty$, and compute the corresponding probability of success. In our block deficit case, $d = z - k$ blocks.

We get $lim_{x\to\infty}\ [w_{(x+(z-k),\ x)}] =$

$$\{\ lim_{x\to\infty}\ [\frac{1-(\frac{l}{w})^x}{1-(\frac{l}{w})^{x+(z-k)}}],\ \text{if } w \neq l$$

$$\{\ lim_{x\to\infty}\ [\frac{x}{x+(z-k)}],\ \text{if } w = l = \frac{1}{2}$$

Next, note that if $w < l$, then $lim_{x\to\infty}(\frac{l}{w})^{-x} = 0$. And so

$$lim_{x\to\infty}\ [\frac{1-(\frac{l}{w})^x}{1-(\frac{l}{w})^{x+(z-k)}}] = lim_{x\to\infty}\ [\frac{(\frac{l}{w})^{-x}-1}{(\frac{l}{w})^{-x}-(\frac{l}{w})^{(z-k)}}] = (\frac{w}{l})^{(z-k)}$$

And if $w > l$, then $lim_{x\to\infty}(\frac{l}{w})^x = 0$. And so

$$lim_{x\to\infty}\ [\frac{1-(\frac{l}{w})^x}{1-(\frac{l}{w})^{x+(z-k)}}] = 1$$

Finally, if $w = l$, then $lim_{x\to\infty}\ [\frac{x}{x+(z-k)}] = 1$.

As a result, $lim_{x\to\infty}\ [w_{(x+(z-k),\ x)}] = (\frac{w}{l})^{(z-k)}$ if $w < l$, and 1 if $w \geq l$

If malicious miner(s) are not incurring a block deficit, i.e., $k \geq z$, then

$$P[\mathcal{A}^{51}_{r,z}\ |\mathcal{M}_{k,r,z}] = 1$$

Otherwise, if $k < z$, then noting that the probability of malicious miner(s) finding the next block is equal to r and the one corresponding to the honest miners is equal to p, we get:

$$P[\mathcal{A}^{51}_{r,z}\ |\mathcal{M}_{k,r,z}] = (\frac{r}{p})^{(z-k)} \text{ if } r < p, \text{ and 1 if } r \geq p$$

The calculations above allow us to conclude that:

$$P[\mathcal{A}^{51}_{r,z}] = \Sigma_{k=0}^{z-1}\ [\frac{(\frac{r}{p}z)^k\ e^{-\frac{r}{p}z}}{k!}(\frac{r}{p})^{(z-k)}]\ +\ \Sigma_{k=z}^{\infty}\ [\frac{(\frac{r}{p}z)^k\ e^{-\frac{r}{p}z}}{k!}]$$

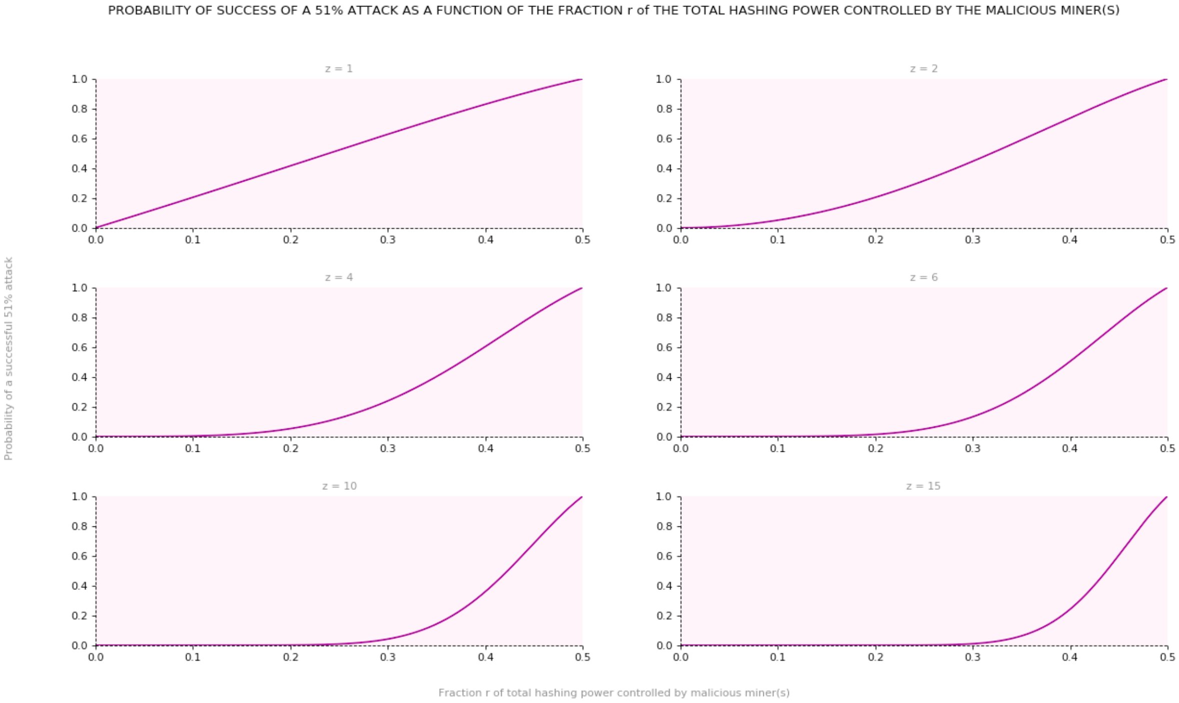
$$= \Sigma_{k=0}^{z-1}\ [\frac{(\frac{r}{p}z)^k\ e^{-\frac{r}{p}z}}{k!}(\frac{r}{p})^{(z-k)}]\ +\ 1 - \Sigma_{k=0}^{z-1}\ [\frac{(\frac{r}{p}z)^k\ e^{-\frac{r}{p}z}}{k!}]$$

$$= 1 - \Sigma_{k=0}^{z-1}\ [\frac{(\frac{r}{p}z)^k\ e^{-\frac{r}{p}z}}{k!}][1-(\frac{r}{p})^{(z-k)}],\ \text{ if } r < p$$

$$\text{and } P[\mathcal{A}^{51}_{r,z}] = \Sigma_{k=0}^{\infty}\ [\frac{(\frac{r}{p}z)^k\ e^{-\frac{r}{p}z}}{k!}] = 1,\ \text{ if } r \geq p$$

In the graphs below we plot the probability of a successful 51% attack as a function of the fraction r of the total hashing power controlled by the subset of malicious miner(s).

We do so for different values of the block validation height z :



For example, assuming that malicious miner(s) controlled 15% of the total hashing power of the network (i.e., $r = 0.15$), then a block validation height of 6 blocks (i.e., $z = 6$) is associated with a probability of a successful attack of 0.268%.

References

[1] Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network, 2013.

[2] DSN. Block propagation delay history. https://dsn.tm.kit.edu/bitcoin/.

[3] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *White Paper*, 2008.

[4] A. Pinar Ozisik and Brian Neil Levine. An explanation of nakamoto's analysis of double-spend attacks. https://arxiv.org/pdf/1701.03977.pdf, 2017.

[5] Karl Sigman. Gambler's ruin problem.
http://www.columbia.edu/ ks20/FE-Notes/4700-07-Notes-GR.pdf, 2016.