Smart Contract Execution – the (+−)-Biased Ballot Problem*

Lin Chen[1], Lei Xu[2], Zhimin Gao[3], Nolan Shah[4], Yang Lu[5], and Weidong Shi[6]

1 **Department of Computer Science, University of Houston, Houston, USA**
chenlin198662@gmail.com
2 **Department of Computer Science, University of Houston, Houston, USA**
xuleimath@gmail.com
3 **Department of Computer Science, University of Houston, Houston, USA**
mtion@hotmail.com
4 **Department of Computer Science, University of Houston, Houston, USA**
nolanshah212@gmail.com
5 **Department of Computer Science, University of Houston, Houston, USA**
ylu17@uh.edu
6 **Department of Computer Science, University of Houston, Houston, USA**
wshi3@uh.edu

Abstract

Transaction system build on top of blockchain, especially smart contract, is becoming an important part of world economy. However, there is a lack of formal study on the behavior of users in these systems, which leaves the correctness and security of such system without a solid foundation. Unlike mining, in which the reward for mining a block is fixed, different execution results of a smart contract may lead to significantly different payoffs of users, which gives more incentives for some user to follow a branch that contains a wrong result, even if the branch is shorter. It is thus important to understand the exact probability that a branch is being selected by the system. We formulate this problem as the (+−)-Biased Ballot Problem as follows: there are n voters one by one voting for either of the two candidates A and B. The probability of a user voting for A or B depends on whether the difference between the current votes of A and B is positive or negative. Our model takes into account the behavior of three different kinds of users when a branch occurs in the system – users having preference over a certain branch based on the history of their transactions, and users being indifferent and simply follow the longest chain. We study two important probabilities that are closely related with a blockchain based system – the probability that A wins at last, and the probability that A receives d votes first. We show how to recursively calculate the two probabilities for any fixed n and d, and also discuss their asymptotic values when n and d are sufficiently large.

1998 ACM Subject Classification G.3 Statistical computing, C.2.1 Distributed networks, F.1.2 Probabilistic computation

Keywords and phrases Blockchain, Probability, Random Walk, Smart Contract

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2017.21

1 Introduction

Blockchain technology provides a decentralized way for bookkeeping and has been proved to be a powerful tool in various areas, especially fintech applications [3]. Bitcoin [16], as the first blockchain based cryptocurrency system, is now accepted by more than 100,000

* A full version of the paper is available at [6], http://i2c.cs.uh.edu/tiki-download_wiki_attachment.php?attId=73&download=y.

merchants worldwide [10], and the value of each bitcoin is more than \$1,000. Besides purely cryptocurrency transactions, blockchain is also used to build decentralized smart contract platform, where the execution of contracts is enforced by all nodes participating the blockchain system [5]. This greatly extends the range of application of blockchain technology and could revolutionize world business [19].

Briefly speaking, a blockchain is a chain of blocks with each block containing some information (e.g., the transactions, the execution result of a program and so on) and the hash of the previous block. Users of the system keep appending blocks to the blockchain. Smart contracts can be built upon a blockchain system. A smart contract is an event-driven program which can be viewed as a contract in a decentralized system. In smart contract supported blockchain systems (e.g., Ethereum), a user runs the contract locally, puts the execution result of the smart contract in a block and then tries to append the block to the blockchain. Further details of a blockchain system can be found in Section 1.1.

Ideally, a blockchain always remains as a chain. However, intentionally or not, a user may append a new block after some other block instead of the one at the end of the current blockchain, yielding a branch (or "fork"). There are several different criteria for eliminating branches. One common criterion is the *longest-chain rule* [16], which ensures that the branch that first receives d blocks afterwards will be selected as the "legal" branch, that is, blocks in all the other branches will be neglected by the system.

Users of the system that are financially driven may behave strategically to maximize their own profit. In a blockchain system where a user is paid by a fixed amount of coins whenever he/she successfully appends a block (e.g., Bitcoin), it is natural that he/she will append a block to a branch that is most likely to be selected by the system eventually, which is essentially the current longest branch [16]. Things become much more complicated when smart contracts are involved, since the execution result of the contract may involve a huge gain or loss among users. Suppose there two branches A and B where different execution results of some smart contract are stored. A user who is involved in the smart contract can have strong pereference over these two branches. Indeed, let r_A (r_B) be the reward that the user can get if eventually A (B) is selected by the system. If $r_A > 100r_B$, then even if A is currently the shorter branch and will be selected by the system with the probability of 0.01, the user may still append blocks to A, trying to maximize his/her expected reward. Such a phenominon has been observed by Chen et al. [7]. To better understand and predict the behavior of users in a blockchain system supporting smart contracts like Ethereum, it is thus important to characterize the exact probability that certain branch gets selected, which is the goal of this paper.

1.1 The Blockchain Based Transaction System

A blockchain is a public ledger that records transactions between users [16]. Typically, it is a chain of blocks with each block containing some information (e.g., the transactions, the execution result of a program and so on) and the hash of the previous block. Consequently, the precedence order between blocks are fixed and it is possible to trace back from a recent block to the very first block in the system. Users of the system append blocks to the blockchain through a process called mining. They are called miners[1].

As blockchain is a decentralized system, one of the critical requirements is that all users have to reach consensus on the sequence and content of blocks. There are mainly two types of techniques involved in this process: block construction and chain selection. For block construction, the most common approach is proof-of-work [24], which is widely used in many blockchain based transaction systems, e.g., Bitcoin [16] and Ethereum [25]. Proof-of-work

[1] Throughout this paper, we are only concerned with such users, thus users and miners are used interchangeably.

requires the node to solve a hard problem to build a valid block, and the success probability is determined by physical resources (e.g., CPU and storage) owned by the node [15]. Because the amount of physical resources is relatively fixed, the probability for each node to produce a valid block is stable. It is worth mentioning that besides proof-of-work, there are also other protocols for block construction, e.g., proof-of-stake [4] and proof-of-elapsed-time [8]. However, as proof-of-work much more widely used than other protocols, we focus on proof-of-work throughout this paper and consequently, we treat the probability of producing a valid block as a fixed value. If more than one branches are generated, participating nodes have to decide which branch to follow to add new blocks. Common chain selection criteria include longest-chain rule [16] and GHOST (greedy heaviest-observed sub-tree) rule [14]. Byzantine fault tolerant protocol is also proposed to eliminate disagreements on the chain [9, 11]. Throughout this paper, we focus on the longest-chain rule.

A fixed reward[2] is paid to a user who successfully adds a block, thus giving incentive to mining. However, when a miner adds a block, it is not guaranteed that he/she always appends this block at the bottom of the chain. He/she may choose any previous block to append the new block or sometimes two or more users simultaneously add blocks after the same block yielding a branch. Eventually only one branch will be chosen as the "legal" branch and all others will be discarded. The longest chain rule is a common branch selection method used in Bitcoin and other blockchains. This rule chooses the branch that first accumulates d blocks for some constant d. Note that when a branch is discarded, miners in that branch do not receive any award, thus incentivizing users to follow the longest chain rule.

The strategic behavior of users may cause serious problems when a blockchain system includes smart contracts. A smart contract is an event-driven program which can be viewed as a contract in a decentralized system. In smart contract supported blockchain systems (e.g., Ethereum [25]), a user runs the contract locally, then publishes the execution result of the smart contract in a block. The user is rewarded for mining and contract execution. However, since a smart contract defines the payoff among users involved in the contract, different execution results may lead to significantly different payoffs of users involved in the contract. If a miner is directly or indirectly involved, he/she may want to branch the chain by adding a block containing a (right or wrong) result favorable to them. In this case, miners benefiting from the result will work on different branches in their favor, while indifferent users will simply work on the (temporarily) longer branch. Eventually, one branch will be selected by the system.

This poses the question, what is the probability that a certain branch is selected? This could be cast as a ballot problem: we view both branches as two candidates A and B. Every block added to the system is viewed as a vote and voting for A or B represents which branch the block is added to. As indifferent users choose the longer branch, the probability that a voter votes for A or B is subject to variation on the current number of votes A and B get. We give the formal definition of the ballot problem in the following subsection.

1.2 Problem Statement

We propose the following model to study the behavior of users in a blockchain based smart contract system.

(+−)-Biased Ballot Problem. Suppose there are two candidates A and B. There are n voters one by one making their votes. Each voter either votes for A or B. The probability that the voter votes for A or B depends on the total number of votes received by A and B at the time he votes. Specifically, let n_A^i and n_B^i be the number of votes received by A and B when voter i votes, respectively. Voter i votes for A and B with probability p_+ and

[2] The reward may be adjusted after a significantly long time. In the short term, it is treated as fixed.

$q_+ = 1 - p_+$, if $n_A^i - n_B^i > 0$; with probability p_- and $q_- = 1 - p_-$, if $n_A^i - n_B^i < 0$; with probability p_0 and $q_0 = 1 - p_0$, if $n_A^i - n_B^i = 0$. We are interested in the following two questions in this paper. First, what is the probability that A receives more votes than B at last? Second, if there are infinite voters (indeed, $n \geq 2d$ suffices), what is the probability that A first receives d votes.

An equivalent formulation of the problem under the framework of random walk is the following:

1-Dimensional (+−)-Biased Random Walk. Suppose there is a ball located at the x-axis. Let L_i be the location of the ball after i-steps. Originally, $L_0 = 0$.

$$\mathrm{P}(L_{i+1} - L_i = 1) = \begin{cases} p_+, & \text{if } L_i > 0 \\ p_-, & \text{if } L_i < 0 \\ p_0, & \text{if } L_i = 0 \end{cases}$$

and $\mathrm{P}(L_{i+1} - L_i = -1) = 1 - \mathrm{P}(L_{i+1} - L_i = 1)$.

Note that $L_{i+1} - L_i = 1$ implies that voter $i+1$ votes for A, and $L_{i+1} - L_i = -1$ implies that he votes for B. Therefore, the probability that A receives more votes than B at last is exactly $\mathrm{P}(L_n > 0)$. We call it the ending probability. The event that A receives d votes first is denoted as H_d. We call $\mathrm{P}(H_d)$ as the hitting probability.

We discuss the random walk that starts at the origin in Section 3. In general, the random walk need not start at the origin, i.e., L_0 could be an arbitrary integer. We provide the result as Theorem 12. The reader may refer to the full version [6] of this paper for details.

▶ Remark. In our model, the two candidates A and B represent the two chains a blockchain branches into. Users' preferences over the two chains, based on the history of their transactions, are indicated by the parameters $p_+, q_+, p_0, q_0, p_-, q_-$. Suppose the probability of users favoring chain A is p, users favoring chain B is q, and users being indifferent is λ. Further, we assume that users being indifferent will always add a block to the longer chain, and when A and B have the same length, they add a block randomly. In this case, parameters will take the following values.

$$\begin{aligned} p_+ &= p + \lambda, & p_0 &= p + \lambda/2, & p_- &= p \\ q_+ &= q, & q_0 &= q + \lambda/2, & q_- &= q + \lambda \end{aligned}$$

The ending and hitting probabilities will indicate the chance that eventually A or B becomes the chain that is accepted by the system. Weighing such probabilities against their potential gain or loss (due to the history of their transactions recorded on A and B), users may decide on whether to follow the chain they prefer, or to follow the other chain if the probability that the chain they prefer has too low probability of being accepted by the system.

1.3 Related Work on Random Walk and the Ballot Problem

We give a brief overview of random walk and the ballot problem.

Bertrand's Ballot Problem. In an election where candidate A receives p votes and candidate B receives q votes with $p > q$, what is the probability that A will be strictly ahead of B throughout the count?

The ballot problem is a classical problem in combinatorics whose study dates back to 1878 by Whitworth [13]. The answer of the problem is $\frac{p-q}{p+q}$, which is known as the Bertrand's ballot theorem and could be proved via various approaches, e.g., by a recursion relation [13] or by Andre's reflection method [17].

An equivalent problem is also studied in the context of random walk [1]. Consider the *1-dimensional symmetric simple random walk* on $\mathbb{Z}$, that is, let the random walk start at the origin $L_0 = 0$ and $L_{i+1} = L_i \pm 1$, each with probability 1/2 and independent of other steps. What is the probability that $L_i > 0$ for all $1 \le i \le p+q$ conditioned on $L_{p+q} = p-q$? This probability is given precisely by the Bertrand's ballot theorem. The number of all such random walks is studied in [2]. Various generalizations of the ballot problem in the context of random walk has been studied by Takacs [20, 21, 22].

Random walk, as a general subject, has been studied extensively in the literature. We refer the readers to [23] as a nice survey on geometric random walk and [1] as a nice survey on various ballot problem-related results when the ballot problem is viewed as a random walk.

In recent years, research on the ballot problem and its extensions has found applications in the study of the blockchain based transaction systems. In his seminal paper, Nakamoto [16] introduces the Bitcoin system and uses the result from the ballot problem to study the security of the system. Indeed, the security problem in Bitcoin could be cast as the following modified ballot problem: suppose each voter votes independently with probability p to A and probability $q = 1-p$ to B, what is the probability that A is always strictly ahead of B? Such a model is also adopted in a series of subsequent studies [18, 12].

1.4 Our Contribution

We study the strategic behavior of users in a blockchain based transaction system. Instead of adopting the classical model that assumes attackers and honest users, in our model we assume there are three groups of users when the blockchain branches into chain A and chain B, with two groups favoring chain A and chain B respectively, and the third group being indifferent and favoring the longer chain. We formulate our model as the $(+-)$-Biased Ballot Problem where the two candidates A and B represent the two chains. In our model, we do not necessarily require that chain A is always ahead of chain B. Indeed, we care about the probability that A exceeds B after n blocks are generated, which we call the ending probability, and the probability that A is extended by d blocks at first, which we call the hitting probability. That means, we also take into account of the possibility that chain A is temporarily behind chain B but takes over later on. However, once A is behind B, the probability that the next block is added to A will be adjusted. In the extreme case when $p_- = 0$, i.e., when A is behind no blocks will be added to A, our model reduces to the classical model in which we only consider the probability that A is always ahead of B.

We show how to calculate the ending and hitting probability for the $(+-)$-Biased Ballot Problem and study their asymptotic values when n and d are sufficiently large. Applying our results to blockchain, we show that, if the third group (i.e., users that are indifferent) has much larger probability of generating the next block, then the probability that chain A wins eventually is roughly $0.5 + \theta\Delta$, where Δ is the difference in the probability of generating a block between users favoring A and users favoring B, and $\theta \in [3/4, 3/2]$. We further consider the ballot problem starting at the situation that A is already ahead of B by s votes. Let μ_s be the probability that A wins eventually, then for sufficiently large n the probability chain B can win is no more than $(1-\mu_0)(q_+/p_+)^k$ when $p_+ > q_+$. In the classical ballot problem, the probability that B can catch up by k blocks is $(q_+/p_+)^k$, as is shown by Nakamoto [16] in the Bitcoin system. In our model, however, this probability is further reduced by a factor of $1-\mu_0$.

2 Preliminaries

Bailey's number [2]. Consider a sequence of numbers $x_1, x_2, \cdots, x_{n+r}$ where $x_i \in \{-1, 1\}$. In total, the number of 1's and -1's in the sequence is n and r, respectively. Furthermore, $\sum_{i=1}^{j} x_i \ge 0$ holds for any $1 \le j \le n+r$. The number of all such sequences is denoted as

$\sigma(n,r)$ in this paper. It is shown by Bailey that

$$\sigma(n,r) = \frac{n+1-r}{n+1} \cdot \binom{n+r}{r}.$$

Similar as the binomial coefficient, the followings are true for the Bailey's number [2]:

$$\sigma(n,r) = \sigma(n-1,r) + \sigma(n,r-1), \quad 2 \le r \le n-1 \tag{1}$$

$$\sigma(n,n-1) = \sigma(n,n). \tag{2}$$

Specifically, if $r = n$, $\sigma(n,r)$ becomes the same as the n-th Catalan number. We will make use of the following generating function for the Catalan number:

$$\sum_{n=0}^{\infty} \sigma(n,n)x^n = \frac{2}{1+\sqrt{1-4x}}. \tag{3}$$

3 Solution for the (+−)-Biased Ballot Problem

In this section, we show how to calculate the ending and hitting probability for the (+−)-Biased Ballot Problem for any fixed n and d. We also discuss their asymptotic values when n and d are sufficiently large. We focus on the random walk version of the problem. We consider the case that the random walk starts at the origin, i.e., $L_0 = 0$. The reader may refer to to the full version [6] oft this paper for the random walk that starts at an arbitrary location.

Let Λ_k be the event that the random walk starts at $L_0 = 0$, and returns to 0 for the first time after $2k$ steps. Furthermore, we define

$$\Lambda_k^+ = \Lambda_k \cap \{L_i > 0, \forall\, 1 \le i \le 2k\}, \quad \Lambda_k^- = \Lambda_k \cap \{L_i < 0, \forall\, 1 \le i \le 2k\}.$$

Consequently, $\Lambda_k = \Lambda_k^+ \cup \Lambda_k^-$.

Consider the event $L_n > 0$. There are two possibilities, either the random walk starts at 0, goes right and never returns to 0 within n steps, or it returns to 0 for the first time after $n' < n$ steps, implying that after n' steps it re-starts at 0, and ends at some positive location after $n - n'$ steps. Notice that n' must be even. Let B_n^+ be the event that it goes right and never returns to 0 within n steps, we have the following recursive calculation:

$$\mathrm{P}(L_n > 0) = \mathrm{P}(B_n^+) + \sum_{k=1}^{\lfloor n/2 \rfloor} \mathrm{P}(\Lambda_k) \cdot \mathrm{P}(L_{n-2k} > 0) \tag{4}$$

Here $\mathrm{P}(\Lambda_k) \cdot \mathrm{P}(L_{n-2k} > 0)$ is the probability of the event that the random walk starts at 0, goes right, returns to 0 for the first time after $2k$ steps, and then re-start at 0, ends at some positive location after $n - 2k$ additional steps.

We can apply similar arguments for the hitting probability. Let $H_d = \cup_{j=1}^{d}\{L_{2d-j} = j\}$ denote the event that the random walk goes right for d steps and goes left for less than d steps. There are two possibilities, either the random walk starts at 0, goes right and never returns to 0, or it returns to 0 for the first time after $2k$ steps, and the event H_{d-k} happens afterwards. Let D_d^+ denote the event that the random walk never goes back to 0, then we have the following.

$$\mathrm{P}(H_d) = \mathrm{P}(D_d^+) + \sum_{k=1}^{d-1} \mathrm{P}(\Lambda_k) \cdot \mathrm{P}(H_{d-k}) \tag{5}$$

The recursive formulas (4) and (5) has a very similar structure. In the following we show how to calculate $\mathrm{P}(B_n^+)$, $\mathrm{P}(D_n^+)$ and $\mathrm{P}(\Lambda_k)$, whereas for every fixed n and d we can always calculate $\mathrm{P}(L_n > 0)$ and $\mathrm{P}(H_d)$ recursively.

▶ **Lemma 1.**

$$\mathrm{P}(B_n^+) = p_0 \cdot \sum_{0 \le r \le (n-1)/2} \sigma(n-1-r,r) p_+^{n-1-r} q_+^r,$$

where $\sigma(n-1-r,r) = \binom{n-1}{r} \cdot \frac{n-2r}{n-r}$.

Proof. The first step of the random walk should go right, which happens with probability p_0. We denote by $x_i = L_{i+1} - L_i \in \{-1, 1\}$. Consider the case that from the second step to the n-th step, there are in total r steps that go left and $n-1-r$ steps go right. To make sure that the random walk never goes back to 0, we have

$$\sum_{j=1}^{h} x_j \ge 0, \quad \forall 1 \le h \le n-1. \tag{6}$$

Note that when $h = n-1$, we have $n-1-r-r \ge 0$, whereas $r \le (n-1)/2$.

According to the definition of the Bailey's number, the number of integral solutions for Inequality (6) is $\sigma(n-1-r,r)$, and the probability that each solution happens is $p_+^{n-1-r} q_+^r$. Hence, the lemma is proved. ◀

For D_d^+ the argument is exactly the same.

▶ **Lemma 2.**

$$\mathrm{P}(D_d^+) = p_0 \cdot \sum_{0 \le r \le d-1} \sigma(d-1,r) p_+^{d-1} q_+^r,$$

where $\phi(d,r) = \binom{d+r}{r} \cdot \frac{d+1-r}{d+1}$.

Proof. The first step of the random walk should go right, which happens with probability p_0. Again let $x_i = L_{i+1} - L_i \in \{-1, 1\}$. Suppose starting from the second step, there are in total r steps that go left. Notice that in total there are $d-1$ steps go right (excluding the first step) where $0 \le r \le d-1$, then we have

$$\sum_{j=1}^{h} x_j \ge 0, \quad \forall 1 \le h \le d+r-1.$$

The number of integral solutions for the above is $\sigma(d-1,r)$, and the probability that each solution happens is $p_+^{d-1} q_+^r$. Hence, the lemma is proved. ◀

Now we consider Λ_k.

▶ **Lemma 3.**

$$\mathrm{P}(\Lambda_k^+) = p_0 \cdot \sigma(k-1,k-1) p_+^{k-1} q_+^k,$$

where $\sigma(k-1,k-1) = \binom{2k-2}{k-1}/k$.

Proof. Consider Λ_k^+. Starting at 0, the first step of the random walk must go right, and the last step must go left (from 1 to 0). From step 2 to step $2k-1$, we consider $x_i = L_{i+1} - L_i \in \{-1, 1\}$, then it is easy to see that

$$\sum_{j=1}^{h} x_j \ge 0, \quad \forall 1 \le h \le 2k-2.$$

Therefore, the number of all possible solutions is given by the Bailey's number $\sigma(k-1,k-1)$. Each solution occurs with probability $p_+^{k-1} q_+^{k-1}$. Further multiplying the probability of the first and last step $p_0 q_+$, the lemma is proved. ◀

Using the same argument, we have

▶ **Lemma 4.**

$$\mathrm{P}(\Lambda_k^-) = q_0 \cdot \sigma(k-1, k-1) p_-^k q_-^{k-1},$$

where $\sigma(k-1,k-1) = \binom{2k-2}{k-1}/k$.

With the above lemmas, for any fixed n and d we can always recursively calculate $\mathrm{P}(L_n > 0)$ and $\mathrm{P}(H_d)$. In the following we discuss their values when n and d go to infinity, which provide an estimation of the two probabilities when n and d are sufficiently large.

▶ **Lemma 5.** *If* $p_+ > q_+$,

$$\lim_{n\to\infty} \mathrm{P}(B_n^+) = p_0 \cdot \frac{p_+ - q_+}{p_+}.$$

Otherwise, $\lim_{n\to\infty} \mathrm{P}(B_n^+) = 0$.

To show the above lemma, we need the following.

▶ **Lemma 6** ([13], pp.272)**.** *Consider a random walk starting at* $L_0 = 0$, $\mathrm{P}(L_{i+1} - L_i = 1) = p$, $\mathrm{P}(L_{i+1} - L_i = -1) = q$ *where* $p + q = 1$*. If* $p > q$*, then*

$$\lim_{n\to\infty} \mathrm{P}(L_i \geq 0, \forall 1 \leq i \leq n) = \frac{p-q}{p}.$$

If $p < q$*, the above limit is* 0.

Proof of Lemma 5. This infinite summation could be calculated directly by plugging in Lemma 1 and using generating functions. An easier way, however, is to apply Lemma 6. Note that

$$B_n^+ = \{L_0 = 0, L_1 = 1, L_2 \geq 1, L_3 \geq 1, \cdots, L_n \geq 1\}.$$

From the second step to the n-th step, the event B_n^+ could be viewed as starting at 1 and always remaining at the rightside of 1 (including 1 itself). Note that in this case the probability of going left is always q_+ and going right is always p_+, hence, if $p_+ > q_+$

$$\lim_{n\to\infty} \mathrm{P}(B_n^+) = p_0 \cdot \frac{p_+ - q_+}{p_+}.$$

Otherwise, $\lim_{n\to\infty} \mathrm{P}(B_n^+) = 0$. ◀

▶ **Lemma 7.** *If* $p_+ > q_+$,

$$\lim_{d\to\infty} \mathrm{P}(D_d^+) = p_0 \cdot \frac{p_+ - q_+}{p_+^2}.$$

Otherwise, $\lim_{d\to\infty} \mathrm{P}(D_d^+) = 0$.

The proof is a bit involved, the reader is referred to the full version [6] of this paper for details.

Next we consider Λ_k.

▶ **Lemma 8.**

$$\sum_{k=1}^{\infty} \mathrm{P}(\Lambda_k^+) = \frac{2p_0q_+}{1 + \sqrt{1 - 4p_+q_+}}.$$

Proof. Using the generating function (3), we know

$$\sum_{k=1}^{\infty} \mathrm{P}(\Lambda_k^+) = \sum_{k=1}^{\infty} p_0 \sigma(k-1,k-1) p_+^{k-1} q_+^k = p_0 q_+ \sum_{k=0}^{\infty} \sigma(k,k) p_+^k q_+^k = \frac{2p_0 q_+}{1+\sqrt{1-4p_+q_+}}.$$

◀

Similarly, we have

▶ **Lemma 9.**

$$\sum_{k=1}^{\infty} \mathrm{P}(\Lambda_k^-) = \frac{2q_0 p_-}{1+\sqrt{1-4p_-q_-}}.$$

▶ **Theorem 10.** *If* $p_+ \le q_+$, $\lim_{n\to\infty} \mathrm{P}(L_n > 0) = 0$. *Otherwise,*

$$\lim_{n\to\infty} \mathrm{P}(L_n > 0) = \frac{p_0 \cdot \frac{p_+ - q_+}{p_+}}{1 - \frac{2p_0 q_+}{1+\sqrt{1-4p_+q_+}} - \frac{2q_0 p_-}{1+\sqrt{1-4p_-q_-}}}$$

Proof. The limit of $\mathrm{P}(L_n > 0)$ could be calculated in the following way. Given that

$$\sum_{k=1}^{\infty} \mathrm{P}(\Lambda_k) = \sum_{k=1}^{\infty} (\mathrm{P}(\Lambda_k^+) + \mathrm{P}(\Lambda_k^-)) = \frac{2p_0 q_+}{1+\sqrt{1-4p_+q_+}} + \frac{2q_0 p_-}{1+\sqrt{1-4p_-q_-}},$$

For N being sufficiently large, we know

$$0 \le \sum_{k=N}^{\infty} \mathrm{P}(\Lambda_k) \le \epsilon_N,$$

whereas for $n > 2N$ we have

$$\mathrm{P}(B_n^+) + \sum_{k=1}^{N} \mathrm{P}(\Lambda_k)\mathrm{P}(L_{n-2k} > 0) \le \mathrm{P}(L_n > 0) \le \mathrm{P}(B_n^+) + \sum_{k=1}^{N} \mathrm{P}(\Lambda_k)\mathrm{P}(L_{n-2k} > 0) + \epsilon_N.$$

Let n goes to infinity and let $\mu_0 = \lim_{n\to\infty} \mathrm{P}(L_n > 0)$, $\nu_0 = \lim_{n\to\infty} \mathrm{P}(B_n^+)$, the following holds for any fixed integer N,

$$\nu_0 + \sum_{k=1}^{N} \mathrm{P}(\Lambda_k)\mu_0 \le \mu_0 \le \nu_0 + \sum_{k=1}^{N} \mathrm{P}(\Lambda_k)\mu_0 + \epsilon_N.$$

Let N goes to infinity, we know ϵ_N goes to 0, hence

$$\mu_0 = \nu_0 + \sum_{k=1}^{\infty} \mathrm{P}(\Lambda_k)\mu_0.$$

Plug in ν_0, if $p_+ \le q_+$, $\lim_{n\to\infty} \mathrm{P}(L_n > 0) = 0$, otherwise,

$$\mu_0 = \lim_{n\to\infty} \mathrm{P}(L_n > 0) = \frac{p_0 \cdot \frac{p_+ - q_+}{p_+}}{1 - \frac{2p_0 q_+}{1+\sqrt{1-4p_+q_+}} - \frac{2q_0 p_-}{1+\sqrt{1-4p_-q_-}}}$$

◀

Using the same argument, we have the following theorem.

▶ **Theorem 11.**

$$\lim_{d\to\infty} \mathrm{P}(H_d) = \frac{p_0 \cdot \frac{p_+ - q_+}{p_+^2}}{1 - \frac{2p_0 q_+}{1+\sqrt{1-4p_+q_+}} - \frac{2q_0 p_-}{1+\sqrt{1-4p_-q_-}}}.$$

So far we have discussed the random walk starting at the origin. Indeed, if the random walk starts at some point $s > 0$ (the case $s < 0$ could be handled in the same way), then its ending probability μ_s satisfies the following (See the full version [6] of this paper for details).

▶ **Theorem 12.** *For $s \geq 1$, if $p_+ > q_+$*

$$\mu_s = 1 - (1-\mu_0)(\frac{q_+}{p_+})^s.$$

Otherwise, $\mu_s = 0$.

3.1 Application to Blockchain

In this subsection we apply our results to the blockchain system. Specifically, we estimate the value of ending probability when the parameters $p_+, q_+, p_0, q_0, p_-, q_-$ are taking certain fixed values. Note that these parameters are taken as fixed constants.

We first consider Theorem 12. Suppose A represents the main chain and B represents some private chain of attackers. Users favoring the main chain A are the honest users in the traditional studies of blockchain, users favoring chain B are the attackers, and users being indifferent (the third group of users) are actually the majority of users in the system – they do not really have a preference but will just follow the longer chain. Theorem 12 implies that if the main chain on which the honest users keep working on is already k blocks ahead, then the attackers can win with a probability no more than $(1-\mu_0)(\frac{q_+}{p_+})^s$ as long as $p_+ > q_+$, that is, as long as the honest users together with the third group of users can generate the next block with the probability of more than 50%.

We now give a more detailed analysis. Suppose the probability of users favoring chain A is p, users favoring chain B is q, and users being indifferent is λ. We assume that the third group of users, i.e., users being indifferent will always add a block to the longer chain, and when A and B have the same length, they add a block randomly since it makes no difference to them. In this case, we have the following.

$$\begin{aligned} p_+ &= p + \lambda, \qquad & p_0 &= p + \lambda/2, \qquad & p_- &= p \\ q_+ &= q, & q_0 &= q + \lambda/2, & q_- &= q + \lambda \end{aligned}$$

If users favoring A are so powerful that $p \geq q + \lambda$, then μ_0 could be simplified as

$$\mu_0 = \lim_{n\to\infty} \mathrm{P}(L_n > 0) = \frac{(p+\frac{1}{2}\lambda)\cdot \frac{p-q+\lambda}{p+\lambda}}{1 - \frac{2(p+\frac{1}{2}\lambda)q}{1+p+\lambda-q} - \frac{2(q+\frac{1}{2}\lambda)p}{1+p-q-\lambda}}$$

Simple calculations show that it becomes 1, that means A can always win. If $q \geq p + \lambda$, however, then the probability is 0.

In the following we assume that $\lambda > |p-q|$, whereas $p_+ > q_+$ and $q_- > p_-$. Under this condition, we can simplify the formula in Theorem 10 as follows,

$$\mu_0 = \frac{p_0 \cdot \frac{p_+ - q_+}{p_+}}{1 - \frac{p_0 q_+}{p_+} - \frac{p_- q_0}{q_-}} = \frac{\frac{(p+\lambda/2)(p-q+\lambda)}{p+\lambda}}{1 - \frac{q(p+\lambda/2)}{p+\lambda} - \frac{p(q+\lambda/2)}{q+\lambda}}$$

We can rewrite μ_0 as

$$\mu_0 = \frac{p + \frac{1}{2}\lambda - \frac{1}{2}q - \frac{\frac{1}{2}pq}{p+\lambda}}{1 - \frac{1}{2}p - \frac{1}{2}q - \frac{\frac{1}{2}pq}{p+\lambda} - \frac{\frac{1}{2}pq}{q+\lambda}}$$

If p and q are tiny compared with λ, say, $p, q \le 0.1$, then we can ignore $pq/(p+\lambda)$ and $pq/(q+\lambda)$ and derive the following

$$\mu_0 \approx \frac{1}{2} + \frac{\frac{3}{2}p - \frac{3}{2}q}{1+\lambda}.$$

That means, if A is more powerful in generating blocks than B but is still not as powerful as the third party, then his probability of winning the game is larger than half by approximately $\frac{3}{2(1+\lambda)}(p-q) \in [\frac{3}{4}(p-q), \frac{3}{2}(p-q)]$.

4 Conclusion

We study the behavior of users in a blockchain system supporting smart contracts. As different execution results of a smart contract can lead to significantly different payoffs among users, it becomes critical to know the exact probability that a certain branch is selected by the system. We propose a generalized model called the (+−)-Biased Ballot Problem. In our model, we classify users into three groups when the blockchain branches into two chains A and B: the group of user favoring chain A, the group of users favoring chain B and the third group of users that are indifferent and will simply follow the longer chain. Instead of requiring one chain, say, chain A, to be always ahead of the other chain, we allow chain A to be temporarily behind chain B by considering the probability that chain A exceeds B after n blocks are generated (which we call the ending probability), and the probability that chain A exceeds B and when it is extended by d blocks (which we call the hitting probability). We provide recursive equations which allow us to compute the ending and hitting probabilities for any fixed n and d, and discuss the asymptotic values of the ending and hitting probabilities when n and d are sufficiently large.

References

1 L Addario-Berry and BA Reed. Ballot theorems, old and new. In *Horizons of combinatorics*, pages 9–35. Springer, 2008.

2 DF Bailey. Counting arrangements of 1's and-1's. *Mathematics Magazine*, 69(2):128–131, 1996.

3 Shagun Bali and Terry Roche. Blockchain technology: Pushing the envelope in fintech. In *Industry report TABB Forum*, 2015.

4 Vitalik Buterin. What proof of stake is and why it matters. *Bitcoin Magazine, August*, 26, 2013.

5 Vitalik Buterin. A next-generation smart contract and decentralized application platform. *white paper*, 2014.

6 Lin Chen, Lei Xu, Zhimin Gao, Nolan Shah, Yang Lu, and Weidong Shi. Smart contract execution – the (+−)-biased ballot problem. `http://i2c.cs.uh.edu/tiki-download_wiki_attachment.php?attId=71&download=y`.

7 Lin Chen, Lei Xu, Zhimin Gao, Nolan Shah, Yang Lu, and Weidong Shi. Decentralized execution of smart contracts: Agent model perspective and its implications. In *1st Workshop on Trusted Smart Contracts*, 2017.

8 Lin Chen, Lei Xu, Nolan Shah, Zhimin Gao, Yang Lu, and Weidong Shi. On security analysis of proof-of-elapsed-time (poet). In *19th Annual International Symposium on Stabilization, Safety, and Security of Distributed Systems*, 2017.

9 Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, et al. On scaling decentralized blockchains. In *International Conference on Financial Cryptography and Data Security*, pages 106–125. Springer, 2016.

10 Anthony Cuthbertson. Bitcoin now accepted by 100,000 merchants worldwide, 2015.

11 Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. Bitcoin-ng: A scalable blockchain protocol. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pages 45–59, 2016.

12 Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *International Conference on Financial Cryptography and Data Security*, pages 436–454. Springer, 2014.

13 William Feller. *An introduction to probability theory and its applications: volume I*, volume 3. John Wiley & Sons New York, 1968.

14 Yoad Lewenberg, Yonatan Sompolinsky, and Aviv Zohar. Inclusive block chain protocols. In *International Conference on Financial Cryptography and Data Security*, pages 528–547. Springer, 2015.

15 Debin Liu and L Jean Camp. Proof of work can work. In *WEIS*, 2006.

16 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

17 Marc Renault. Lost (and found) in translation: Andre's actual method and its application to the generalized ballot problem. *The American Mathematical Monthly*, 115(4):358–363, 2008.

18 Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal selfish mining strategies in bitcoin. *arXiv preprint arXiv:1507.06183*, 2015.

19 Melanie Swan. *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.", 2015.

20 Lajos Takács. A generalization of the ballot problem and its application in the theory of queues. *Journal of the American Statistical Association*, 57(298):327–337, 1962.

21 Lajos Takács. The distribution of the majority in a ballot. *journal for probability theory and related fields*, 2(2):118–121, 1963.

22 Lajos Takács. Fluctuations in the ratio of scores in counting a ballot. *Journal of Applied Probability*, 1(02):393–396, 1964.

23 Santosh Vempala. Geometric random walks: a survey. *Combinatorial and computational geometry*, 52(573-612):2, 2005.

24 Marko Vukolić. The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In *International Workshop on Open Problems in Network Security*, pages 112–125. Springer, 2015.

25 Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 2014.