Delay analysis of the BFT blockchain data dissemination : case of narwhal protocol

Khouloud Hwerbi, Ichrak Amdouni, Cédric Adjih, Philippe Jacquet, Leila Azouz Saidane, Anis Laouiti

Delay Analysis of the BFT Blockchain Data Dissemination: Case of Narwhal Protocol

Khouloud Hwerbi * ‡, Ichrak Amdouni *, Cédric Adjih §, Philippe Jacquet §, Leila Azouz Saidane *, Anis Laouiti ‡
*ENSI, Tunisia, University of Manouba, name.surname@ensi-uma.tn
‡Telecom SudParis, France, khouloud_hwerbi@telecom-sudparis.eu, anis.laouiti@telecom-sudparis.eu
§Inria, France, name.surname@inria.fr

***Abstract*—This article investigates data dissemination delays in a Directed Acyclic Graph (DAG)-based Byzantine Fault Tolerant (BFT) blockchain. We focus particularly on the Narwhal protocol, a mempool-based approach for efficiently disseminating transactions and constructing a DAG. Narwhal is designed to work alongside a BFT consensus protocol like Tusk. Tusk then orders the transaction metadata based on the DAG information. Through an in-depth analysis of the protocol messages, we establish a mathematical model for message propagation delays. We start by considering a specific probability distribution for data network propagation delays: Gaussian Distribution. Then, we consider a general propagation delay distribution. Also, we assume large networks and apply some approximations, i.e., the Central Limit Theorem (CLT). Finally, we develop the Narwhal protocol and demonstrate that the simulated delays are compatible with the theoretical ones.**

***Index Terms*—BFT Blockchain, Network Delay Analysis, Narwhal, Order Statistics, large-scale, CLT.**

I. Introduction

The BFT principle, introduced in 1982 as the Byzantine generals problem [1], ensures that a system can function correctly and reach consensus despite Byzantine failures, that is arbitrary or malicious behavior by up to f nodes out of n. These systems attempt to achieve a consistent state, regardless of the actions of faulty nodes. Besides the failure mode, the communication network model is also crucial in designing consensus protocols, ranging from synchronous models with time constraints to asynchronous models with unbounded message delays. In asynchronous settings, achieving deterministic consensus is impossible with even a single failure, as shown in [2]. One solution, proposed by [3], uses probabilistic methods like random coin flips to reach consensus. Another approach [4] assumes partial synchronization and ensures safety and liveness during asynchronous periods and guarantees termination once the system becomes synchronous. The continuous interest in enhancing the scalability and performance of blockchain networks has sparked renewed interest in DAG-based protocols [5]. These protocols rely on a DAG structure, where each vertex represents a message and edges connect messages to their predecessors. Furthermore, unlike traditional blockchains where transactions are sequentially organized into blocks, DAG-based protocols enable concurrent transaction processing. This capability enables blockchains to support a larger number of transactions per second, potentially enhancing overall throughput and scalability. This paper investigates the Narwhal protocol [6] which focuses on improving a key part of a blockchain called the mempool. The mempool is a temporary storage area for transactions waiting to be included in a block. By improving the mempool, Narwhal aims to make blockchains faster and more efficient. In Narwhal, all nodes disseminate transactions around the network simultaneously. Then, a separate part of the blockchain system, that is the consensus layer, takes care of ordering these transactions. This separation allows Narwhal to handle transactions efficiently while keeping the overall system secure. The Narwhal protocol potential for high performance has garnered significant attention. As a result, new blockchain platforms like Aptos [7] and Celo [8] are actively exploring its integration with their protocols. Narwhal employs a round-based DAG structure to organize its messages. Each message (vertex) is associated with a specific round. To start a new round, a node must validate a quorum of messages from the previous round. To achieve high performance, recent studies utilize Narwhal as a mempool protocol and add, on top of it, a zero communication overhead consensus algorithm like in [9], [4], [10], [11], [12]. In these references each node interprets its local view of the DAG to establish a total ordering of the accumulated transactions. While consensus algorithms used in conjunction with the Narwhal mempool protocol are constantly evolving and being optimized, a comprehensive analysis of Narwhal standalone performance remains lacking. In this paper, we fill this gap and analyze the propagation delays of messages exchanged in Narwhal. The performance analysis of the blockchain consensus algorithms has been addressed by several studies using different methodologies. For instance, [13] modeled the PBFT protocol with an M/G/1 queue. Similarly, [14] introduced a new theoretical model to calculate the transaction latency for the Hyperledger Fabric blockchain under different network settings, such as varying block sizes and block intervals. [15] focused on Hotstuff consensus and proposed a mathematical model of the protocol and provided a performance evaluation under different network conditions. For our case, we establish a mathematical model to determine the propagation delays of the different delays of Narwhal messages. Specifically, the main contributions of this work are: (1) the detailed analysis of the Narwhal protocol, (2) an asymptotic mathematical delay analysis assuming Gaussian propagation delay, (3) an asymptotic mathematical delay anal-

ysis for general propagation delay, and (4) the development of the Narwhal protocol and comparing simulation results with theoretical model results.

II. Related Work

A. DAG-based protocols

The benefits of DAG-based consensus protocols for blockchains have been well documented [16].One of the main advantages is that all nodes can generate and process transactions independently and simultaneously. This enables parallel processing, spreading the workload of validating new transactions across the network, speeding up transaction processing, and reducing waiting times for clients. In DAG-based protocols, nodes broadcast messages with transaction data and metadata to determine the final order of committed transactions. Nodes build the DAG by adding new transactions that reference and validate previous ones, creating links between them. Each new transaction points to one or more preceding transactions, forming a directed acyclic graph that maintains transaction dependencies. For instance, IOTA [17] introduced Tangle as a specific type of DAG data structure for its cryptocurrency platform. In order to send an IOTA transaction, we need to validate two previously approved transactions. A central coordinator node is used to manage the Tangle. However, the IOTA community is working towards entirely removing this coordinator, making the network fully decentralized. Hashgraph consensus [18], based on the concepts of "gossip about gossip" and "virtual voting", is an asynchronous DAG-based consensus mechanism with a degree of two as in the IOTA tangel. Through the gossip process, nodes collaboratively construct a hash diagram that reflects the chronology of such gossip events. Each node retains a copy of the hash diagram, enabling it to achieve BFT on any decision without ever explicitly casting a vote.

B. BFT-consensus based on Narwhal

The leader-based BFT consensus Hotstuff [19], which assumes a partially synchronous environment, has been combined with Narwhal in the Narwhal-Hotstuff [6]. In Hotstuff, a node proposes blocks of transactions that other nodes have validated. To improve the throughput, a leader in Narwhal-Hotstuff proposes only a meta-data created by the Narwhal protocol. The consensus process works in a succession of views, each has a unique leader and a committed proposal is achieved in three phases. DAG-Rider [9] and Tusk [6], operating in an asynchronous environment, utilize the randomization technique. They employ the distributed global perfect coin for leader election. The main theoretical difference between them lies in their termination. DAG-Rider waves consist of 4 rounds, where the leader from the first round is committed if it has enough strong edges from a vertex in round four. In contrast, Tusk waves consist of only 3 rounds, where each odd round uses a shared random coin to elect a leader in round $r-2$ and ensures enough edges to round $r-1$. Similar to previous protocols such as DAG-Rider and Tusk, BullShark [10] operates without the need of additional communication to achieve consensus following DAG construction. It is a BFT consensus working on the partially synchronous environment. It maintains all the theoretical properties of DAG-Rider while incorporating additional techniques to leverage periods of synchrony. These techniques involve defining two types of votes for either a predefined or randomly chosen leader.

III. Study of the Narwhal Protocol

A. Overview

Narwhal is a mempool-based protocol that reliably broadcasts transactions from the mempool. This mempool serves as a buffer for transactions waiting to be validated and included in a block. Narwhal proceeds in rounds and build its DAG during these rounds. Each vertex is linked to a specific round. The DAG structure efficiently organizes and orders transactions, enhancing the robustness and scalability of the blockchain. By maintaining a structured round-based DAG, Narwhal ensures efficient data availability and transaction integrity in a fully asynchronous network. Nodes which are responsible for processing and validating transactions, operate in rounds and exchange messages.

B. Narwhal Protocol Messages

Narwhal considers three types of messages: the block, the signature, and the certificate of availability:

1) **Block**: The DAG vertices in Narwhal are data blocks. Each block includes the signature of the block generator node, a list of transactions, and a list of certificates of availability of blocks from the previous round.
2) **Signature**:A signature is a message that is relative to a specific block. It is sent by a node when this latter validates (or approves) this block. A specific quorum of such signatures constitutes a certificate of availability relative to the block in question.
3) **Certificate of availability**: A certificate message is associated to a block. It contains the hash of this block, along with a quorum of signatures from other validators and the round number r.

C. Protocol Workflow

The Narwhal protocol operates as the following (Fig. 1):

1) Each validator continually receives client transactions and stores them in a local transaction list. It also receives certificates, which are stored in a certificate list.
2) Upon receiving a block, a validator first validates the block signature, ensuring it contains $q = 2f + 1$ certificates from the previous round and is the first block it receives from this validator in the current round. If these conditions are satisfied, the validator signs it.
3) Once a validator receives q signatures for the block it has broadcast, it generates a certificate of availability for the block and disseminates this certificate to the network.
4) Upon receiving certificates from a quorum q of validators for the current round, the validator advances to the next round and broadcasts a new block containing its current transactions.

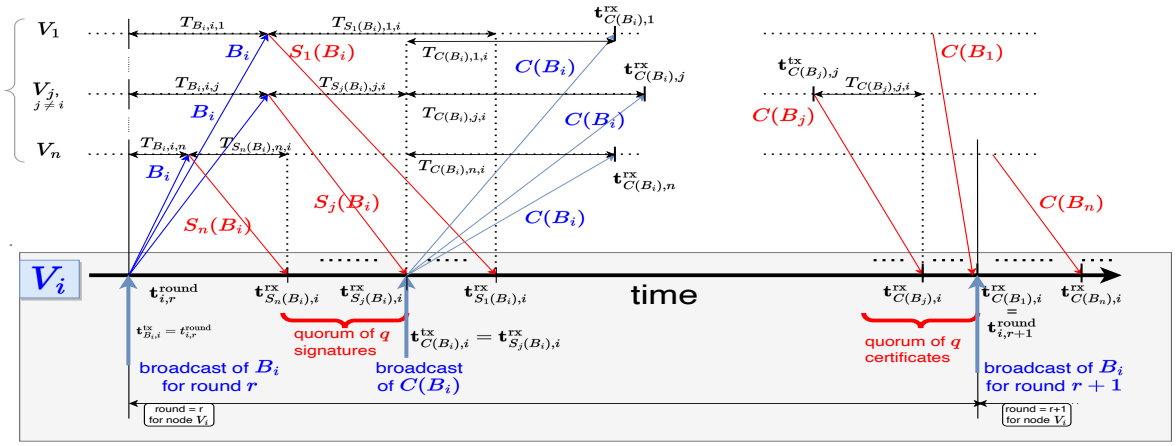

Fig. 1: Exchanged Messages from the point of view of validator V_i

After the transactions are disseminated through the DAG, a separate consensus protocol takes over to determine the final order in which these transactions are included in the blockchain. This protocol achieves zero additional message overhead because nodes do not need to send extra messages to each other. Instead, each node independently examines its local copy of the DAG and uses predefined rules to order transactions.

IV. Analysis of Data Dissemination Delays

The objective now is to mathematically model the transmission and reception times for each message before validators can proceed to the next round.

A. Notations and Assumptions

Table I summarizes the notations used in our analysis. We will consider only a single round r. Hence we are dropping the index r from all notations, except the starting time of the round denoted $\mathbf{t}^{\text{round}}_{i,r}$.

TABLE I: Notation Used in the Model Establishment

Notation	Description
$\mathbf{t}^{\text{round}}_{i,r}$	Time at which node i starts the round r.
$\mathbf{t}^{\text{tx}}_{m,i}$	Time at which node i transmits a message m.
$\mathbf{t}^{\text{rx}}_{m,j}$	Time at which node j receives a message m.
$T_{m,i,j}$	The propagation delay of message m sent from node i to node j
B_i	A block message sent by node i.
$S_j(B_i)$	A signature message of node j for the block B_i sent by node j.
$C(B_i)$	A certificate message of the block B_i sent by node i.

We consider the following assumptions:

A1. We consider a network including $n \geq 3f+1$ nodes acting as Narwhal validators with the same processing power operating over a fully connected peer-to-peer network in which nodes are directly connected pairwise.

A2. We assume that the network propagation delays for both unicast and broadcast follow the same probability distribution $\mathcal{D}$. Specifically, for any pair of nodes i and j, the transmission delay $T_{m,i,j}$ of a message m follows the distribution $\mathcal{D}_{i,j} = \mathcal{D}$ across all P2P network links.

A3. We assume that there is no message loss for either unicast or broadcast transmissions. The reception time of a message m sent by node i and received by node j is given by the following equation:

$$\mathbf{t}^{\text{rx}}_{m,j} = \mathbf{t}^{\text{tx}}_{m,i} + T_{m,i,j} \tag{1}$$

A4. For the purpose of simplification we assume that the delay of messages generation needed for a node i to process any message m and perform the necessary computations is zero.

B. Mathematical Preliminaries

To facilitate the understanding of the model, we start by presenting some essential mathematical preliminaries.

1) Order Statistics: In many steps of the protocol, a node has to wait for a quorum of q messages. This means it must wait for the first q messages among n expected messages before proceeding to the next step of the protocol. This waiting time corresponds to the q-th order statistic of the sample; that is, the q-th smallest value among them [20].

Formally, let $X_1, X_2, \ldots, X_n$ be n independent and identically distributed (i.i.d.) continuous random variables (r.v.) with probability density function (PDF) f and cumulative distribution function (CDF) F.

$$X_{(1)} = \text{smallest value among } X_1, X_2, \ldots, X_n$$

$$X_{(q)} = \text{q-th smallest value among } X_1, X_2, \ldots, X_n$$

$$X_{(n)} = \text{largest value among } X_1, X_2, \ldots, X_n$$

The ordered values $X_{(1)}, \ldots, X_{(n)}$ are known as the order statistics of the random variables $X_1, \ldots, X_n$.

2) Central Limit Theorem (CLT) for Order Statistics: Let $X_1, X_2, \ldots, X_n$ be a sequence of n i.i.d random variables representing measurements sampled from a population having CDF F and PDF f. The CLT theorem states that, as the

sample size increases, the sample mean tends to be normally distributed around the population mean regardless of the underlying distribution of the population [20]. Several versions of the classical CLT theorem have been proposed for order statistics; see, for instance, [21]. One result states that the distribution of $X_{(pn)}$ for a fixed $p \in (0,1)$ and as $n \to \infty$ asymptotically converges to:

$$X_{(np)} \sim AN\left(F^{-1}(p), \frac{p(1-p)}{n(f(F^{-1}(p)))^2}\right) \tag{2}$$

where "AN" stands for "Asymptotically Normal". Here, $F^{-1}(p)$ is the inverse of the CDF function F, also known as the quantile function. For a random variable X with a Gaussian distribution $X \sim \mathcal{N}(\mu, \sigma^2)$, the quantile function is given by:

$$F^{-1}(p; \mu, \sigma) = \mu + \sigma\Phi^{-1}(p) \tag{3}$$

where $\Phi^{-1}(p)$ is the quantile function for a standard normal distribution. By defining the parameters $\mu_{n,p}$ and $\sigma^2_{n,p}$ for Eq.(2), we have:

$$\mu_{n,p} = \mu + \alpha\sigma, \quad \sigma^2_{n,p} = \beta\sigma^2 \tag{4}$$

where:

$$\alpha_p = \Phi^{-1}(p), \quad \text{and} \quad \beta_{n,p} = \frac{2\pi p(1-p)}{n}\exp\left(\alpha^2\right) \tag{5}$$

C. Narwhal Protocol Modeling

Our goal now is to compute the necessary time duration for any validator node i to switch to a new round. The round duration determines how quickly transactions can be processed and included in a block, which directly affects the complexity of the DAG, the speed of the transaction confirmation, and the stability of the blockchain network. We model the delays of the exchanged messages from the point of view of i which starts at time t_i^r. We assume that validator i has received the required quorum of certificate messages from the previous round. We apply the Narwhal algorithm and analyze the message exchanges as shown in Fig. 1, while considering the aforementioned assumptions.

Every validator node i broadcasts a block B_i at time:

$$\mathbf{t}^{\text{tx}}_{B_i,i} = \mathbf{t}^{\text{round}}_{i,r} \tag{6}$$

As a result, every other validator node j receives such broadcast at time $\mathbf{t}^{\text{rx}}_{B_i,j}$ which is represented by the following equation as in our general model (see Eq. (1)):

$$\mathbf{t}^{\text{rx}}_{B_i,j} = \mathbf{t}^{\text{tx}}_{B_i,i} + T_{B_i,i,j} \tag{7}$$

Upon receiving the broadcast block B_i, any other validator node j signs this block and sends back the signature message (the signed block) $S_j(B_i)$ to the sender node i. The transmission time is given by the following equation (assuming no processing delays: Assumption **A4**):

$$\mathbf{t}^{\text{tx}}_{S_j(B_i),j} = \mathbf{t}^{\text{rx}}_{B_i,j} \tag{8}$$

The signature $S_j(B_i)$ is received by node i at time:

$$\mathbf{t}^{\text{rx}}_{S_j(B_i),i} = \mathbf{t}^{\text{tx}}_{S_j(B_i),j} + T_{S_j(B_i),j,i} \tag{9}$$

Every node i has to wait for a quorum of the first $q = 2f + 1$ signatures. However, as node i itself signs its block B_i, it actually waits for $2f$ signatures from any node $j \neq i$ to generate the certificate of its block $C(B_i)$. Waiting for the first q signatures corresponds to the q-th order statistic of $\mathbf{t}^{\text{rx}}_{S_j(B_i),i}$. The sorted sequence of these r.v. is denoted:

$$\mathbf{t}^{\text{rx}}_{S_{j_1}(B_i),i} \leq \ldots \leq \mathbf{t}^{\text{rx}}_{S_{j_q}(B_i),i} \leq \ldots \leq \mathbf{t}^{\text{rx}}_{S_{j_n}(B_i),i}$$

where j_1 is the index of the validator whose signature $S_{j_1}(B_i)$ is received first by V_i, j_2 for the second, etc. Once the quorum of signatures is reached, the certificate C of B_i is broadcast by node i to all other nodes at time:

$$\mathbf{t}^{\text{tx}}_{C(B_i),i} = \mathbf{t}^{\text{rx}}_{S_{j_q}(B_i),i} \tag{10}$$

Such certificate is received by node j at time $\mathbf{t}^{\text{rx}}_{C(B_i),j}$:

$$\mathbf{t}^{\text{rx}}_{C(B_i),j} = \mathbf{t}^{\text{tx}}_{C(B_i),i} + T_{C(B_i),i,j} \tag{11}$$

When the validator node i has $q = 2f + 1$ certificates ($2f$ received from other nodes with its own certificate), it moves to the round $r+1$ and broadcasts a new block. Node i receives the certificates relative to the blocks of other nodes j at time $\mathbf{t}^{\text{rx}}_{C(B_j),i}$:

$$\mathbf{t}^{\text{rx}}_{C(B_j),i} = \mathbf{t}^{\text{tx}}_{C(B_j),j} + T_{C(B_j),j,i} \tag{12}$$

Following the same reasoning, waiting for the first q certificates corresponds to the q-th order statistic of the certificates reception times from all nodes at node i. The sorted sequence of these r.v. can be denoted as:

$$\mathbf{t}^{\text{rx}}_{C(B_{j'_1}),i} \leq \ldots \leq \mathbf{t}^{\text{rx}}_{C(B_{j'_q}),i} \leq \ldots \leq \mathbf{t}^{\text{rx}}_{C(B_{j'_n}),i}$$

where j'_1 is the index of the validator whose certificate $C(B_{j'_1})$ is received first by validator i, j'_2 for the second, etc. The transmission time of the new block $\mathbf{t}^{\text{tx}}_{B_i,i,r+1}$ in the new round $r+1$ is equal to $\mathbf{t}^{\text{round}}_{i,r+1}$. The transition to the new round is represented by:

$$\mathbf{t}^{\text{round}}_{i,r+1} = \mathbf{t}^{\text{rx}}_{C(B_{j'_q}),i} \tag{13}$$

V. Gaussian Propagation Delays

In this section we assume that the propagation delay $T_{m,i,j}$ of any Narwhal protocol message m follows a Gaussian distribution. We denote such delay as

$$T_{m,i,j} \sim \mathcal{N}(\mu, \sigma^2)$$

.

We also make the assumption:

$$\mathbf{t}^{\text{round}}_{i,r} \sim \mathcal{N}(\mu_r, \sigma_r^2)$$

.

We will now apply the protocol steps introduced in Section IV to find the distribution of

$$\mathbf{t}^{\text{round}}_{i,r+1}$$

as a Gaussian

$$\mathcal{N}(\mu_{r+1}, \sigma^2_{r+1})$$

,

by establishing the equations from

$$(\mu_r, \sigma^2_r)$$

to

$$(\mu_{r+1}, \sigma^2_{r+1})$$

. The reception time of a signature message is equal to the sum of the round starting time and the propagation delays of a block and a signature :

$$\mathbf{t}^{\text{rx}}_{S_j(B_i),i} - \mathbf{t}^{\text{round}}_{i,r} = T_{B_i,i,j} + T_{S_j(B_i),j,i} \tag{14}$$

Since $T_{B_i,i,j}$ and $T_{S_j(B_i),j,i}$ follow Gaussian distribution with parameters $\mathcal{N}(\mu, \sigma^2)$ respectively, $\mathbf{t}^{\text{rx}}_{S_j(B_i),i} - \mathbf{t}^{\text{round}}_{i,r} \sim \mathcal{N}(2\mu, 2\sigma^2)$.
$\mathbf{t}^{\text{tx}}_{C(B_i),i}$ is the transmission time of the certificate C of the block B_i as represented by Eq. (10). It is equal to the waiting time of the reception of $q = 2f$ signatures from nodes $j \neq i$ for the block B_i. This waiting time corresponds to the q-th order statistic of the signature reception times random variables. We apply, as an approximation, the variant of the CLT theorem that has been extended to order statistics, as discussed in [20] and Section IV. We then obtain from Eqs. (2), (4) and (5) the following approximation:

$$\mathbf{t}^{\text{tx}}_{C(B_i),i} - \mathbf{t}^{\text{round}}_{i,r} \approx \mathcal{N}(2\mu + \alpha\sigma\sqrt{2}, 2\beta\sigma^2) \tag{15}$$

Based on equations (10), (12) and (15), $\mathbf{t}^{\text{rx}}_{C(B_j),i}$, is the sum of three random variables:

$$\mathbf{t}^{\text{rx}}_{C(B_j),i} = (\mathbf{t}^{\text{tx}}_{C(B_j),j} - \mathbf{t}^{\text{round}}_{j,r}) + \mathbf{t}^{\text{round}}_{j,r} + T_{C(B_j),j,i} \tag{16}$$

approximated by Gaussian distributions respectively $\mathcal{N}(2\mu + \alpha\sigma\sqrt{2}, 2\beta\sigma^2)$, $\mathcal{N}(\mu_r, \sigma^2_r)$, and $\mathcal{N}(\mu, \sigma^2)$. Thus,

$$\mathbf{t}^{\text{rx}}_{C(B_j),i} \approx \mathcal{N}(\mu_r + 3\mu + \alpha\sigma\sqrt{2}, \sigma^2_r + (2\beta+1)\sigma^2). \tag{17}$$

In order for node i to transmit a block in the new round, it must wait for a quorum of $C(B_j)$. We use the same approximation of the CLT theorem extended to order statistics to deduce that $\mathbf{t}^{\text{rx}}_{C(B_{j'_q}),i} = \mathcal{N}(\mu', \sigma'^2)$ and we obtain the parameters:

$$\mu' = \mu_r + 3\mu + \alpha\sigma\sqrt{2} + \alpha\sqrt{\sigma^2_r + (2\beta+1)\sigma^2} \tag{18}$$

$$\sigma'^2 = \beta\sigma^2_r + (2\beta+1)\beta\sigma^2. \tag{19}$$

Notice that the certificate quorum might be reached before the signature quorum, in which case the nodes will have to wait the later before changing rounds, but as approximation we ignore this, since, as show in next section, this happens rarely.

Thus:

$$\mathbf{t}^{\text{round}}_{i,r+1} \approx \mathcal{N}(\mu_{r+1}, \sigma^2_{r+1}) \quad \text{with} \quad \mu_{r+1} = \mu', \quad \sigma^2_{r+1} = \sigma'.$$

When $\beta < 1$, σ_r will converge to a fixed point obtained by setting $\sigma_r = \sigma_{r+1}$:

$$\sigma^2_\infty = \frac{(2\beta+1)\beta}{1-\beta}\sigma^2. \tag{20}$$

As a result, when $r \to \infty$, the asymptotic round duration is

$$(\mu_{r+1} - \mu_r) \approx 3\mu + \left(\sqrt{2} + \sqrt{\frac{2\beta+1}{1-\beta}}\right)\alpha\sigma. \tag{21}$$

VI. General propagation delay

In this section, we assume a general propagation delay distribution. We denote $F_1(x)$ (resp. $F_2(x)$) as the CDF of the one-way (resp. round trip) delay. A node i starts its round at time $\mathbf{t}^{\text{round}}_{i,r}$, we know that the relative delay $\mathbf{t}^{\text{tx}}_{C(B_i),i}$ it will have received its signature quorum and transmit its certificate is:

$$\mathbf{t}^{\text{tx}}_{C(B_i),i} - \mathbf{t}^{\text{round}}_{i,r} \approx \mathcal{N}_{F_2,2/3}. \tag{22}$$

where $\mathcal{N}_{F,p} = \left(F^{-1}(p), \frac{(1-p)p}{(nf(F^{-1}(p)))^2}\right)$. And we denote $\mathbf{t}^{\text{rx}}_{C(B_i),i}$ the absolute time at which node i receives its signature quorum. A random node i will receive the certificates from a random node j at time:

$$\mathbf{t}^{\text{rx}}_{C(B_j),i} = \mathbf{t}^{\text{round}}_{j,r} + (\mathbf{t}^{\text{tx}}_{C(B_j),j} - \mathbf{t}^{\text{round}}_{j,r}) + T_{C(B_j),j,i} \tag{23}$$

and therefore receives its quorum of certificates at a time $\mathbf{t}^{\text{rx}}_{C(B_{j_q}),i}$ close to $\mathcal{N}_{F_3,2/3}$ where F_3 is the cumulative function distribution of $\mathbf{t}^{\text{round}}_{j,r} + T_{C(B_j),j,i} + F_2^{-1}(2/3)$.

The event $\mathbf{t}^{\text{tx}}_{C(B_i),i} > \mathbf{t}^{\text{rx}}_{C(B_{j_q}),i}$ is called an *inversion*.

Theorem 1: Assume that $\mathbf{t}^{\text{round}}_{i,r}$ satisfies the Central Limit Theorem with a mean μ_r and large deviation exponentially small in n (*i.e.* $\forall \epsilon > 0\ Pr(|\mathbf{t}^{\text{round}}_{i,r} - \mu_r| > \epsilon < \exp(-O(n)))$, thus the variance $\sigma_r = O(1/n)$. The probability of an inversion is exponentially small in n.

Proof: We know that $\mathbf{t}^{\text{rx}}_{C(B_{j_q}),i}$ has a mean of $F_3^{-1}(p)$. Since $\mathbf{t}^{\text{tx}}_{C(B_j),j}$ is already of mean $F_2^{-1}(2/3)$ with an exponential large deviation, assuming that $\mathbf{t}^{\text{round}}_{i,r}$ has also an exponential large deviation implies that $F_3^{-1} = \mu_r + F_2^{-1}(2/3) + F_1^{-1}(2/3)$. Thus the inversion probability is within an exponentially small error term equal to $Pr(\mathbf{t}^{\text{round}}_{i,r} > \mu_r + F_1^{-1}(2/3))$. The latter is also exponentially small. ■

Remark: the average time to wait before an inversion is exponentially large in n.

Theorem 2: $\mathbf{t}^{\text{round}}_{i,r}$ satisfies the Central Limit Theorem with distribution $\mathcal{N}(\mu_r, \sigma_r)$ with exponential large deviation.

Proof: Since we already know that the inversion probability is exponentially small, then we have with probability 1 minus the inversion probability the fact that:

$$\mathbf{t}^{\text{round}}_{i,r+1} = \mathbf{t}^{\text{rx}}_{C(B_{j_q}),i} \tag{24}$$

Since $\mathbf{t}^{\text{round}}_{j,r} + \mathbf{t}^{\text{tx}}_{C(B_j),i} = \mu_r + F_2^{-1}(2/3)$ a gaussian distribution with a mean $\mu_r + F_2^{-1}(2/3)$ and a $O(1/n)$ variance it impacts the variance of $\mathbf{t}^{\text{round}}_{j_q,r} + \mathbf{t}^{\text{tx}}_{C(B_{j_q}),i}$ by a $O(1/n^2)$ term.

Consequently $\mathbf{t}_{i,r+1}^{\text{round}}$ is gaussian with a mean $\mu_r + F_2^{-1}(2/3) + F_1^{-1}(2/3)$ and the variance is $\frac{2/9}{(f_1(F_1^{-1}(2/3)))^2 n} + O(1/n^2)$. ∎

The corollary is that the round duration is asymptotically normal with mean $F_2^{-1}(2/3) + F_1^{-1}(2/3)$ and variance $\frac{2/9}{(f_1(F_1^{-1}(2/3)))^2 n} + O(1/n^2)$.

For the Gaussian propagation delay, we have $F_1(x) = \Phi((x-\mu)/\sigma)$ and $F_1^{-1}(p) = \mu + \sigma\Phi^{-1}(p/\sigma)$, $F_2^{-1}(p) = 2\mu + \sqrt{2}\sigma\Phi(p))$. For the exponential delay we have $F_1(x) = 1 - e^{-x/\mu}$, thus $F_1^{-1}(p) = \mu \log(1-p)$, $F_2(x) = 1 - (1 + x/\mu)e^{-x/\mu}$ and therefore $F_2^{-1}(2/3) = \mu W(\frac{e^{-1}}{3}) \approx \mu 2.289281414\ldots$, where W is the Lambert function.

VII. Simulation and Results

In this section, we present the simulation results of the mathematical model and compare them with those of the Narwhal protocol. We utilized a batch size of 10000, with a total of 100 nodes, initially fully synchronized, and ran 30 rounds. We represent in Fig. 2 the distribution of the different phases of the protocol (round start, signatures received, certificates received, round end) of the last round (shifted so that start time is $t = 0$ on average). Each event is represented with a distinct color in the histogram, clearly depicting their relative timing across different rounds. Additionally, we overlay empirical Gaussian distributions on these histograms to compare the observed data with theoretical models. It includes an empirical Gaussian distribution for round start times, and an estimated Gaussian for the next round.

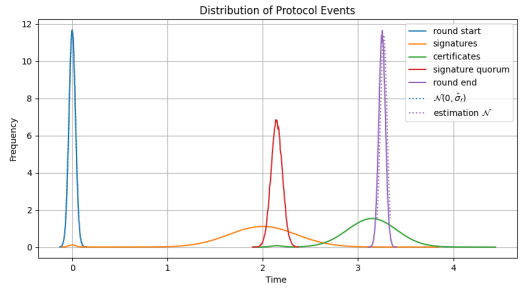

Fig. 2: Distribution of Protocol Events Over Time

Fig. 3 illustrates both empirical observations from the simulation and theoretical expectations derived from our model. The blue line represents the mean round durations observed throughout the simulation in a very different scenario where nodes are heavily desynchronized when they begin the protocol, i.e. their start time follow a Gaussian $\mathcal{N}(0, 10)$. The red dashed line signifies the asymptotic duration mean, as calculated in Eq. (21). As the protocol progresses, the empirical round durations quickly converge towards the asymptotic duration mean predicted by our model. The close alignment between the empirical data and the model predictions confirms the accuracy of our calculations. This also evidences a self-synchronization property.

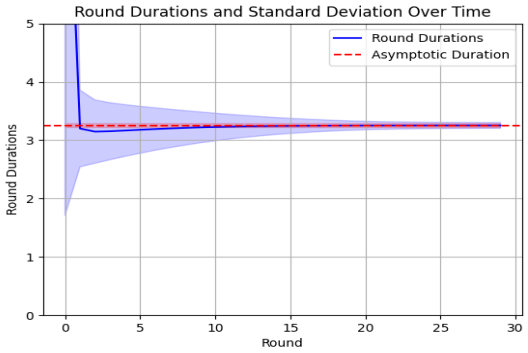

Fig. 3: Mean and Standard Deviation of Round Duration $\mathbf{t}_{i,r+1}^{\text{round}} - \mathbf{t}_{i,r}^{\text{round}}$ Over Multiple Rounds

VIII. Conclusion

This article examined how long it takes for data to travel within the Narwhal protocol, using a mathematical approach. We developed both a general model and a model that assumes data propagation delays follow a Gaussian distribution. Simulations of the Narwhal protocol closely matched the delays predicted by our model, which confirms its accuracy and reliability. One remarkable property is that the protocol tends to automatically (re-)synchronize the nodes despite each of them needs only a quorum q to progress.

Acknowledgment

This research was conducted under the project PHC-Maghreb ANGEL 24MAG18.

References

[1] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," ACM Trans. Program. Lang. Syst., 1982.

[2] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," J. ACM, 1985.

[3] M. Ben-Or, "Another advantage of free choice (extended abstract): Completely asynchronous agreement protocols," in Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing, 1983.

[4] A. Spiegelman, N. Giridharan, A. Sonnino, and L. Kokoris-Kogias, "Bullshark: Dag bft protocols made practical," in Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2022.

[5] J. Xu, C. Wang, and X. Jia, "A survey of blockchain consensus protocols," ACM Computing Surveys, 2023.

[6] G. Danezis, L. Kokoris-Kogias, A. Sonnino, and A. Spiegelman, "Narwhal and tusk: a dag-based mempool and efficient bft consensus," in Proceedings of the Seventeenth European Conference on Computer Systems, 2022.

[7] T. A. Team. (2022) Aptos networks.

[8] T. C. Team. (2020) Celo networks.

[9] I. Keidar, E. Kokoris-Kogias, O. Naor, and A. Spiegelman, "All you need is dag," in Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing, 2021, pp. 165–175.

[10] A. Spiegelman, N. Giridharan, A. Sonnino, and L. Kokoris-Kogias, "Bullshark: The partially synchronous version," arXiv preprint arXiv:2209.05633, 2022.

[11] A. Spiegelman, B. Aurn, R. Gelashvili, and Z. Li, "Shoal: Improving dag-bft latency and robustness," arXiv preprint arXiv:2306.03058, 2023.

[12] S. Blackshear, A. Chursin, G. Danezis, A. Kichidis, L. Kokoris-Kogias, X. Li, M. Logan, A. Menon, T. Nowacki, A. Sonnino et al., "Sui lutris: A blockchain combining broadcast and consensus," arXiv preprint arXiv:2310.18042, 2023.

[13] M. Alaslani, F. Nawab, and B. Shihada, "Blockchain in iot systems: End-to-end delay evaluation," IEEE Internet of Things Journal, 2019.
[14] X. Xu, G. Sun, L. Luo, H. Cao, H. Yu, and A. V. Vasilakos, "Latency performance modeling and analysis for hyperledger fabric blockchain network," Information Processing & Management, 2021.
[15] Y. Shahsavari, K. Zhang, and C. Talhi, "Performance modeling and analysis of hotstuff for blockchain consensus," in 2022 Fourth International Conference on Blockchain Computing and Applications (BCCA), 2022.
[16] Q. Wang, J. Yu, S. Chen, and Y. Xiang, "Sok: Dag-based blockchain systems," ACM Computing Surveys, 2023.
[17] S. Popov, "The tangle," White paper, 2018.
[18] L. Baird, "The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance," Swirlds Tech Reports SWIRLDS-TR-2016-01, Tech. Rep, 2016.
[19] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, "Hotstuff: Bft consensus with linearity and responsiveness," in Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, 2019.
[20] B. C. Arnold, N. Balakrishnan, and H. N. Nagaraja, A first course in order statistics. SIAM, 2008.
[21] M. Cardone, A. Dytso, and C. Rush, "Entropic central limit theorem for order statistics," IEEE Transactions on Information Theory, 2022.