Behavioral Similarity Analysis of Validators in Byzantine Distributed Control Networks

Rachid Guedjali, Jean-Philippe Georges, Sylvain Kubler

Behavioral Similarity Analysis of Validators in Byzantine Distributed Control Networks

Rachid Guedjali * **Jean-Philippe Georges** * **Sylvain Kubler** **

* *Université de Lorraine, CNRS, CRAN, F-54000 Nancy, France (e-mail: {rachid.guedjali, jean-philippe.georges}@univ-lorraine.fr)*
** *University of Luxembourg, SnT, L-1359 Luxembourg, Luxembourg (e-mail: sylvain.kubler@uni.lu)*

Abstract: Byzantine Fault Tolerant (BFT) consensus protocols, such as PBFT, face scalability and security challenges due to static validator sets and high communication overhead. While recent work models validator behavior in binary terms (honest/faulty), this oversimplifies real-world dynamics. Inspired by decentralized systems that grant users autonomous control, such as in healthcare data sharing and smart city remote sensing, we enhance PBFT by introducing a nuanced validator behavior model, aligning consensus dynamics with adaptive trust management in dynamic environments. We propose a multi-state framework, classifying nodes as majority validators, minority honest validators, faulty validators, or non-validators. Using empirical state distributions and Jensen Shannon Divergence (JSD), we quantify behavioral similarity among nodes, enabling dynamic clustering and anomaly detection. Our approach enhances PBFT's adaptability by tracking nuanced validator roles and transitions, improving resilience in dynamic networks. Simulations demonstrate the method's effectiveness in identifying stable and anomalous behaviors, supporting proactive validator selection and trust management.

Keywords: blockchain, mathematics of networked systems, Jensen-Shannon divergence, cluster

1. INTRODUCTION

Blockchain and distributed systems provide decentralized solutions across various domains, including healthcare, smart cities, and vehicular networks, the food chain and other emerging areas (Khan et al., 2024b). Generally speaking, blockchain technology research trends in secured device to device communication and secures Internet of Things communication. Hence (Khan et al., 2024a) leveraged blockchain as a distributed ledger to secure remote sensing data in smart cities, ensuring data integrity and traceability without centralized control while, (Dibaei et al., 2021) investigated the integration of blockchain and machine learning to enhance security in vehicular communications. While these studies demonstrate blockchain's growing role as a trust infrastructure in decentralized environments, they do not explicitly address the challenges of validator dynamics in Byzantine consensus protocols. Our work bridges this gap by introducing a probabilistic approach to model and analyze validator behavior, thereby enhancing the robustness of decentralized systems.

Byzantine Fault Tolerant (BFT) protocols, such as Practical Byzantine Fault Tolerance (PBFT) (Castro et al., 1999), are designed to preserve system consistency even in the presence of arbitrary node failures or malicious actors. Operating in partially synchronous environments, PBFT guarantees correctness as long as no more than f out of $N \geq 3f + 1$ replicas behave Byzantine. Despite its robustness, PBFT incurs a communication overhead of $\mathcal{O}(N^2)$, making scalability and dynamic validator management increasingly important.

A key limitation of PBFT and similar protocols is the static or pseudo-static selection of validators. The compromise or degradation of a fixed subset of validators can severely impact system security and performance. To address this, recent research has explored alternative validator selection strategies. These include randomness-based models, as in Algorand (Gilad et al., 2017), token-weighted selection (Li et al., 2020), reputation-based methods (Qiu et al., 2022), and even spatial optimization techniques (Leduc et al., 2023). However, these approaches often lack real-time behavioral tracking and rarely adapt to the dynamic nature of validator trustworthiness over time.

Recent efforts have proposed incorporating probabilistic and anomaly detection models to enhance validator assessment. For instance, (Nischwitz et al., 2021) introduce probabilistic failure modeling, while the Blockchain Ensemble Machine Learning (BEML) framework (Musa Baig et al., 2023) applies ensemble learning to detect anomalies in validator behavior. Most notably, (Guedjali et al., 2025) proposed integrating opinion dynamics with BFT protocols to model validator reliability as a stochastic process. Their work models validators using binary states (honest vs. faulty), predicting transitions based on shared characteristics and behavioral patterns.

Traditional binary models of validator behavior (Guedjali et al., 2025) often fail to capture the complexity and variability inherent in decentralized consensus systems.

⋆ The authors acknowledge the support of the Agence National de la Recherche (ANR), under grant ANR-20-CE25-001 (project TIC-TAC-SDN).

To address this, we introduce a probabilistic multi-state framework where each node is represented by a distribution over four discrete behavioral categories: majority validator, honest minority validator, faulty minority validator, and non-validator. To quantify behavioral similarity between nodes, we compute the Jensen-Shannon Divergence (JSD) (Menéndez et al., 1997), which effectively captures differences between probability distributions. The resulting divergence matrix is then used as input for the Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) algorithm (Campello et al., 2013), which identifies clusters of nodes exhibiting similar behavioral dynamics and detects anomalous behavior, all without requiring prior specification of the number of clusters. In contrast to traditional clustering methods that rely on geometric metrics such as Euclidean distance, the integration of JSD enables a more meaningful analysis of behavioral patterns in Byzantine consensus settings by directly leveraging the probabilistic nature of validator states.

Our approach enables a macroscopic view of validator dynamics while preserving individual node histories. These similarities serve to guide dynamic validator selection or weighting, addressing our central problem of capturing nuanced validator behavior beyond binary classifications. This framework aims to improve both the decentralization and resilience of PBFT-like systems by providing adaptive tools to detect validator drift, enhance trust assessments, and support secure consensus in dynamic environments.

The remainder of this article is organized as follows: Section 2 outlines the challenges associated with validator behavior modeling, particularly regarding the states of nodes over time. Subsection 2.5 introduces our similarity-based framework using JSD to analyze validator behaviors. Section 3 presents our clustering methodology with HDBSCAN, which is subsequently evaluated in Section 4. Finally, Section 5 discusses the implications of our findings and suggests directions for future research.

2. PROBLEM STATEMENT

The key objective of studying the states of nodes is to measure the behavioral similarity between nodes at different time points. By comparing the dynamics of multiple nodes over time, it is possible to identify the degree of similarity in their behaviors, which can help detect recurring patterns or significant changes in their role within the network. This similarity measure is crucial for tracking the evolution of individual nodes and for observing the emergence of groups of nodes with similar behavioral dynamics. Such clustering processes are essential for a better understanding of the network's structure and for the proactive management of its evolution.

2.1 PBFT Protocol

The Practical Byzantine Fault Tolerance (PBFT) protocol (Castro et al., 1999) is a foundational consensus algorithm for distributed systems designed to tolerate Byzantine faults. Operating in a partially synchronous network, PBFT ensures consistency across a set of replicas, tolerating up to f faulty nodes in a network of at least $N \geq 3f+1$ nodes, where N represents the total number of nodes in the system. In PBFT-based blockchain systems, all nodes act as validators, collectively participating in the consensus process. However, PBFT incurs a message complexity of $O(N^2)$, due to the need for multiple rounds of communication between all nodes in the system.

2.2 Validator Selection Protocols

In the standard PBFT model, the set of validators is typically fixed, and all nodes are assumed to participate in every consensus round. However, static validator sets may become suboptimal over time due to changing network conditions, misbehaviors, or faults. To address these limitations, recent works have proposed dynamic validator selection protocols to enhance security, decentralization, and scalability. Hybrid BFT protocols such as those discussed in (Belotti et al., 2019) introduce an initial proof-based selection phase followed by a classical BFT round, while systems like Algorand (Gilad et al., 2017) and Gosig (Li et al., 2020) use randomized or token-based methods to assign validator roles. Other mechanisms leverage trust or reputation metrics (Qiu et al., 2022), and proximity-aware models (Leduc et al., 2023) aim to optimize validator communication latency.

While (Guedjali et al., 2025) focuses on binary validator behavior, this work extends the modeling by introducing a four-state representation, capturing finer-grained roles and behaviors within the PBFT consensus process. By analyzing the similarity between validators based on their observed transitions across these states—majority, minority (non-faulty), faulty, and non-validator—we aim to identify behavioral patterns and structural vulnerabilities in the validator set. This similarity-based approach enables more informed decisions for dynamic validator selection, improving both the robustness and adaptability of the consensus protocol.

2.3 Node Behavioral States

In a PBFT-based blockchain system, each node at any given time step can occupy one of four discrete states, defined by its role in the consensus process and its behavioral alignment. Let S_i denote the state of a node, where $i \in \{0, 1, 2, 3\}$, and each state is described as follows (see color coding in Fig. 1):

- **State S_0: Majority Validator** (blue in Fig. 1)
$$S_0 = \{\text{Validator} \wedge \text{Majority}\}$$
The node participates in the consensus protocol and aligns with the majority decision.
- **State S_1: Minority Validator (Non-Faulty)** (green in Fig. 1)
$$S_1 = \{\text{Validator} \wedge \text{Minority} \wedge \neg\text{Faulty}\}$$
The node is an honest validator whose view deviates from the majority—often due to network delays or partial information—but without exhibiting faulty behavior.
- **State S_2: Faulty Minority Validator** (red in Fig. 1)
$$S_2 = \{\text{Validator} \wedge \text{Minority} \wedge \text{Faulty}\}$$
The node validates from a minority position while violating protocol assumptions. This category includes

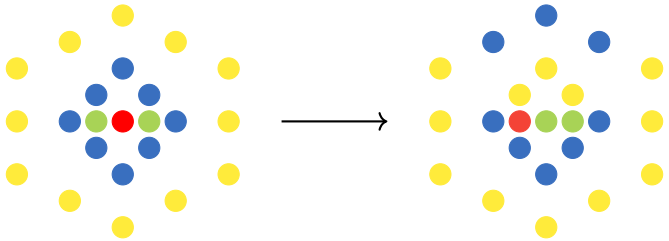

Fig. 1. Illustration of the 4 node states across two iterations

both malfunctioning nodes and adversarial participants.

- **State S_3: Non-Validator** (yellow in Fig. 1)

$$S_3 = \{\neg\text{Validator}\}$$

The node does not engage in the validation process and remains passive with respect to consensus activities.

These four states form the basis for modeling individual node behavior over time and enable the characterization of system-wide dynamics in PBFT networks. An illustrative representation of this classification, as used in our selection process, is shown in Figure 1, where color-coded nodes indicate their respective consensus states. This visualization serves as a practical reference for understanding how nodes transition and distribute across roles in real consensus rounds.

2.4 Dynamic Node Behaviours

From an empirical standpoint, the evolution of node states is studied by estimating the individual probabilities that each node occupies one of the four protocol-defined states. For a given node v, and over a computation horizon H, its state trajectory is observed as:

$$\{S_t(v), S_{t+1}(v), \ldots, S_{t+H}(v)\},$$

allowing us to compute an empirical distribution of its state occurrences. Specifically, the empirical probability that node v is found in state S_i, with $i \in \{0, 1, 2, 3\}$, is defined as:

$$P_i(v) = \frac{1}{H+1} \sum_{k=0}^{H} \mathbb{1}_{\{S_{t+k}(v)=S_i\}},$$

where $\mathbb{1}$ is the indicator function. This distribution characterizes the behavioral profile of a node over the given horizon.

The size of each sector corresponds directly to the estimated probability $P_i(v)$ over the horizon H. This pie chart thus offers an intuitive representation of a node's behavioral tendencies and participation profile in the consensus process. Such local distributions, when aggregated over the entire network, support the identification of systemic patterns (e.g., clusters of persistent minority validators or high failure rates), providing insights into the robustness and dynamics of the consensus mechanism under real conditions.

2.5 Similarity

To systematically compare the behavioral dynamics of nodes in the network, we represent each node v by the empirical distribution of its state occurrences over a given time horizon H. Let $P(v) = (P_0(v), P_1(v), P_2(v), P_3(v))$ denote the probability vector associated with node v, where each component $P_i(v)$ corresponds to the fraction of time node v spends in state S_i, as previously defined.

To quantify the similarity between two nodes u and v, we compute the JSD (Menéndez et al., 1997) between their empirical distributions:

$$\text{JSD}(P(u) \parallel P(v)) = \frac{1}{2}\text{KL}(P(u) \parallel M) + \frac{1}{2}\text{KL}(P(v) \parallel M),$$

where the mean distribution M is defined as:

$$M = \frac{1}{2}(P(u) + P(v)).$$

The quantity $\text{KL}(P \parallel Q)$ denotes the Kullback-Leibler divergence (Kullback, 1951), a measure of relative entropy that quantifies the information loss when distribution Q is used to approximate P. Formally, it is defined as:

$$\text{KL}(P \parallel Q) = \sum_{i=0}^{3} P_i \log_2 \left(\frac{P_i}{Q_i}\right),$$

By convention, any term with $P_i = 0$ is taken to be zero (i.e., $0 \log 0 = 0$), and the KL divergence is only defined when $Q_i > 0$ for all i such that $P_i > 0$.

While the KL divergence is asymmetric and can diverge to infinity if the supports of P and Q do not match, the JSD mitigates these issues by averaging and symmetrizing the divergences with respect to the mean distribution M. As a result, the JSD is always well-defined, symmetric, and bounded:

$$0 \leq \text{JSD}(P(u) \parallel P(v)) \leq \log_2(2) = 1.$$

A smaller JSD value indicates a greater similarity between the two behavioral profiles. This divergence-based similarity metric thus provides a robust foundation for clustering and tracking nodes with comparable behavioral tendencies across the network. Such comparisons are especially valuable for detecting changes, persistent behavioral roles, or emerging anomalies, and can be used as a basis for higher-level structural analyses of the network.

3. NODE CLUSTERING BASED ON JENSEN-SHANNON DIVERGENCE METRIC

The analysis of validator behaviors in Byzantine fault-tolerant networks requires robust methods to quantify and cluster nodes according to their consensus participation patterns. Building on our multi-state framework, this section presents a novel clustering approach using the JSD as a statistical similarity metric between nodes' behavioral distributions.

3.1 State Distributions

Table 1 presents the probability distribution of each node across four predefined behavioral states (S_0 to S_3) prior to the execution of a consensus round. Each column corresponds to a node in the network, and each row shows the likelihood of a node being in a particular state. This probabilistic modeling facilitates a more detailed and realistic depiction of node behavior, acknowledging

State	Node 1	Node 2	Node 3	...	Node N
S_0	$P_1^{(0)}$	$P_2^{(0)}$	$P_3^{(0)}$	...	$P_N^{(0)}$
S_1	$P_1^{(1)}$	$P_2^{(1)}$	$P_3^{(1)}$	...	$P_N^{(1)}$
S_2	$P_1^{(2)}$	$P_2^{(2)}$	$P_3^{(2)}$	...	$P_N^{(2)}$
S_3	$P_1^{(3)}$	$P_2^{(3)}$	$P_3^{(3)}$	...	$P_N^{(3)}$

Table 1. Distribution of each node over the four predefined behavioral

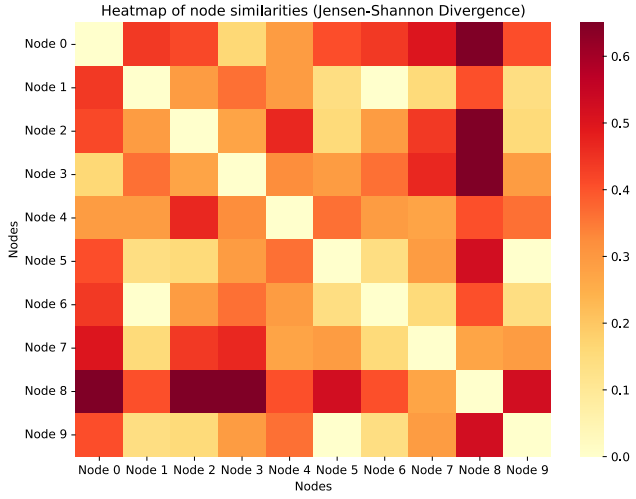

Fig. 2. Jensen-Shannon similarity of nodes' state distributions during one consensus round.

inherent uncertainties and facilitating soft classification rather than rigid state categorizations.

These distributions are foundational to our similarity analysis. As introduced in Section 2.5, we use the JSD to measure how similar or dissimilar the behavior profiles of two nodes are. A low JSD value between two nodes indicates behavioral alignment, while a high value may highlight divergence or abnormality. This similarity framework supports clustering and behavioral grouping, offering insights for validator management, anomaly detection, and consensus robustness.

3.2 Data Analysis

To assess behavioral similarity among nodes in the PBFT network, we utilize the JSD metric, as introduced in Section 2.5. This measure captures the divergence between each node's empirical state distribution $P_i(v)$, detailed in Table 1, allowing for a principled comparison of node behaviors within a given consensus round.

Figure 2 shows the similarity matrix derived from pairwise JSD values. The color gradient encodes behavioral proximity: light cells indicate low divergence and similar node behaviors, while dark cells reflect higher divergence and thus behavioral inconsistency. Although a deeper interpretation is deferred to the simulation results section, this visualization already highlights meaningful structural patterns, such as clustering of aligned nodes, behavioral anomalies, and deviations from consensus norms.

3.3 Behavioral Clustering and Analysis Process

The behavioral clustering process based on JSD is applied iteratively across each consensus round in the PBFT network, following the steps outlined below:

(1) **State Distribution Computation:** For each node i, we calculate the empirical probability distribution $P_i(v)$ over the predefined behavioral states (e.g., honest, deviating, faulty, inactive) observed during a consensus round. These distributions are organized into a matrix, as illustrated in Table 1, where each row corresponds to a state, and each column represents a node.

(2) **Pairwise Similarity Estimation:** The JSD is computed for all pairs of nodes based on their respective state distributions. The resulting JSD values are symmetrically arranged in a similarity (or divergence) matrix $\mathbf{D}$, where $\mathbf{D}_{ij}$ quantifies the behavioral dissimilarity between nodes i and j.

(3) **Behavioral Clustering:** The JSD-based similarity matrix is then input into a clustering algorithm, revealing latent groupings of nodes exhibiting similar behaviors during the consensus round. These groupings emerge independently of the nodes' nominal roles or statuses within the protocol.

(4) **Temporal Aggregation:** For extended analyses, the above steps are repeated across successive consensus rounds to assess the temporal stability of node behaviors. Persistent clustering patterns suggest stable behavioral roles, whereas temporal changes may indicate evolving strategies or disruptions in the protocol.

This analytical pipeline facilitates a comprehensive, data-driven understanding of the interrelationships between individual node behaviors during the consensus process, both at a given point in time and across multiple rounds. These insights are crucial for evaluating the security and reliability of the network.

3.4 Temporal Evolution of Node Behavior

To capture how node behaviors evolve throughout the consensus process, we compute the JSD between the empirical state distributions of each node across consecutive rounds. This divergence quantifies behavioral shifts at the node level and serves as an indicator of temporal stability or volatility. The resulting sequence of divergence values allows us to:

- Track each node's behavioral consistency over time;
- Detect sudden deviations possibly related to faults or adversarial behavior;
- Observe broader behavioral trends, such as synchronization or instability phases within the network.

This temporal perspective is essential for dynamic analysis and contributes directly to the design of adaptive clustering and validator selection mechanisms presented in later sections.

3.5 Clustering Node Behaviors

In this study, we apply HDBSCAN a density-based clustering algorithm designed to handle complex data distributions. We use it to cluster node behaviors based

on their probabilistic state distributions, measured using JSD. HDBSCAN constructs a mutual reachability graph, builds a minimum spanning tree, and generates a hierarchical cluster tree, which is pruned using the Excess-of-Mass (EOM) criterion to extract the most stable clusters. Nodes that do not fit well into any cluster are labeled as noise, allowing for the identification of anomalous behaviors. This method is particularly effective for detecting patterns in decentralized systems where node behaviors can vary significantly.

4. SIMULATIONS

This simulation aims to study the temporal evolution of node behavior in a PBFT network and to evaluate clustering effectiveness using JSD with the HDBSCAN algorithm. Specifically, we pursue two main objectives: (1) to analyze how individual node behaviors stabilize or diverge across consensus iterations, and (2) to determine whether behavioral similarities can be effectively captured via unsupervised clustering. We simulate a PBFT network composed of $N = 50$ nodes. In each round, $n = 10$ validators are randomly selected from the network, ensuring the PBFT majority rule is preserved. Each node is modeled with a probability distribution over four behavioral states—e.g., honest, faulty, inactive, or neutral—generated through synthetic behavioral patterns that reflect different levels of protocol adherence.

This state distribution is observed across 5 consecutive consensus rounds to capture both stable and transient behavior. Node transitions are governed by probabilistic rules that simulate fluctuations in behavior, making it possible to observe how trustworthiness evolves dynamically. For each round, we compute and store the state distribution of all nodes, while also checking that consensus is consistently reached according to protocol constraints. We now detail how the state distributions and divergence metrics are computed and used for clustering analysis.

4.1 Analysis of Node Behavior

To analyze the dynamics of node behavior during the consensus process, we examine the evolution of each node's state distribution across successive consensus rounds. At each round, the empirical probability distribution over the four behavioral states is computed for every node, resulting in a matrix of node–state probabilities. This matrix forms the basis for comparing node behaviors over time. We employ the JSD to quantify the dissimilarity between the state distributions of nodes. Specifically, for each node, we compute the JSD between its state distribution at round t and round $t + 1$. This approach allows us to measure how each node's behavior shifts across consecutive consensus iterations, capturing both stability and divergence.

Figure 3 presents a heatmap illustrating the JSD for each node across consecutive rounds. This visualization reveals the behavioral consistency of individual nodes over time. For instance, between rounds 4 and 5, the distributions for 32 nodes remained relatively unchanged (indicated by lighter shades), whereas between rounds 5 and 6, this number increased to 40 nodes, suggesting a growing behavioral stability across the network.

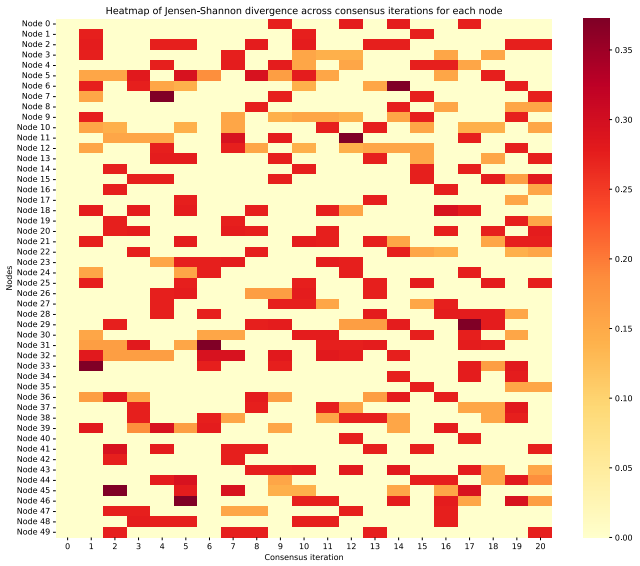

Fig. 3. Heatmap of JSD between successive consensus iterations for each node.

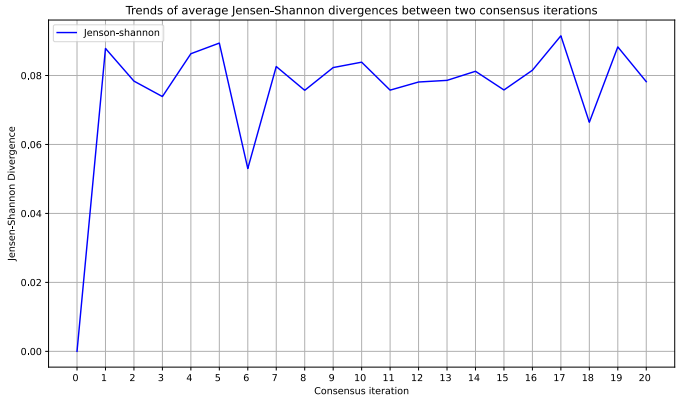

Fig. 4. Average JSD between successive consensus rounds.

To further quantify this trend, we compute the average JSD across all nodes for each pair of successive rounds. This average provides a global measure of behavioral drift in the system. Figure 4 confirms this observation, showing a decreasing trend in average divergence at round 6. The decrease in divergence indicates the stabilization of the network, with most nodes exhibiting persistent behaviors and states.

4.2 Clustering Results

Once the pairwise JSD values are computed, they are used as input to the HDBSCAN algorithm, as detailed in Section 3.5. This algorithm clusters nodes based on their behavioral similarity while automatically identifying outliers or nodes that exhibit significantly different behaviors. To investigate the stability of node behavior throughout the consensus iterations, we applied the HDBSCAN clustering algorithm to the JSD values calculated between successive rounds. Between iterations 4 and 5, HDBSCAN identified 5 clusters among which 18 nodes remained in a stable state: 17 in cluster 0 and 1 in cluster 1. This results in a total of 18 nodes exhibiting stable distributions. This outcome is consistent with the analysis of the heatmap

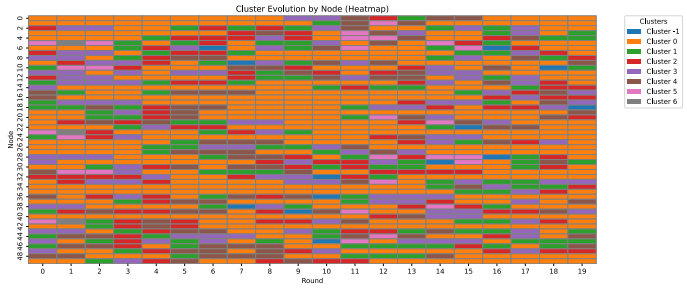

Fig. 5. Heatmap representing the evolution of cluster stability throughout the consensus iterations.

(3), where a significant number of nodes indicates that 32 out of 50 nodes underwent a state change but did not experience a major transition, or even no transition, during these two iterations. In contrast, clustering between iterations 5 and 6 (Figure 5) revealed greater stability, with 31 nodes remaining unchanged: 15 in cluster 0, 1 in cluster 1, 4 in cluster 2, 3 in cluster 3, and 8 in cluster 4. In total, 31 nodes retained similar distributions, suggesting a marked improvement in the overall stability of the consensus process.

These results confirm the trend observed in Figure 4, where the average JSD between successive iterations decreases, highlighting a growing consistency in node behavior. Additionally, the emergence of several clusters in the subsequent transition may indicate the formation of subgroups of nodes converging towards different but stable states. The stability of certain nodes in a network, particularly those exhibiting consistent and predictable behavior across iterations, may introduce security risks if adversaries are able to identify and exploit these patterns. Such predictable nodes could become targets for malicious actors aiming to disrupt the consensus process and undermine network integrity.

To mitigate this risk and bolster network security, nodes or clusters displaying excessive stability in their state distributions can be excluded from both the network and the validator selection process in subsequent iterations. Nodes that show minimal variation over multiple iterations become more predictable, making them susceptible to targeted attacks. By removing these stable nodes from the network and preventing their future selection as validators, the system ensures a more dynamic, diverse, and secure set of validators. This approach enhances the resilience of the consensus process by reducing the potential for exploitation based on predictable node behaviors.

5. CONCLUSION

In this study, we introduced a multi-state probabilistic framework for analyzing validator behavior within PBFT consensus systems, grounded in the mathematics of network systems. Validators were categorized into four distinct behavioral states, and inter-node similarity was quantified using Jensen-Shannon Divergence. These similarity measures were then clustered using HDBSCAN to identify persistent behavioral patterns and detect anomalous nodes, supporting more robust and adaptive validator selection. Our simulations demonstrated the framework's effectiveness in capturing both behavioral stability and divergence across consensus rounds. Future research may integrate these similarity metrics with reinforcement learning to design self-optimizing validator selection mechanisms tackling dynamically the adversarial behavior.

REFERENCES

Belotti, M., Božić, N., Pujolle, G., and Secci, S. (2019). A vademecum on blockchain technologies: When, which, and how. *IEEE Communications Surveys & Tutorials*, 21(4), 3796–3838.

Campello, R.J., Moulavi, D., and Sander, J. (2013). Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, 160–172. Springer.

Castro, M., Liskov, B., et al. (1999). Practical byzantine fault tolerance. In *OsDI*, volume 99, 173–186.

Dibaei, M., Zheng, X., Xia, Y., Xu, X., Jolfaei, A., Bashir, A.K., Tariq, U., Yu, D., and Vasilakos, A.V. (2021). Investigating the prospect of leveraging blockchain and machine learning to secure vehicular networks: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(2), 683–700.

Gilad, Y., Hemo, R., Micali, S., Vlachos, G., and Zeldovich, N. (2017). Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th symposium on operating systems principles*, 51–68.

Guedjali, R., Georges, J.P., and Kubler, S. (2025). Validator's opinion dynamic in a practical byzantine fault tolerant network. *IFAC-PapersOnLine*, 59(1), 223–228.

Khan, A.A., Laghari, A.A., Alroobaea, R., Baqasah, A.M., Alsafyani, M., Bacarra, R., and Alsayaydeh, J.A.J. (2024a). Secure remote sensing data with blockchain distributed ledger technology: a solution for smart cities. *Ieee Access*.

Khan, I., Majib, Y., Ullah, R., and Rana, O. (2024b). Blockchain applications for internet of things — a survey. *Internet of Things*, 27, 101254. doi:https://doi.org/10.1016/j.iot.2024.101254.

Kullback, S. (1951). Kullback-leibler divergence.

Leduc, G., Kubler, S., and Georges, J.P. (2023). A centre-based validator selection approach for a scalable bft blockchain. *IFAC-PapersOnLine*, 56(2), 10192–10197.

Li, P., Wang, G., Chen, X., Long, F., and Xu, W. (2020). Gosig: A scalable and high-performance byzantine consensus for consortium blockchains. In *Proceedings of the 11th ACM Symposium on Cloud Computing*, 223–237.

Menéndez, M.L., Pardo, J.A., Pardo, L., and Pardo, M.d.C. (1997). The jensen-shannon divergence. *Journal of the Franklin Institute*, 334(2), 307–318.

Musa Baig, S., Javed, M.U., Almogren, A., Javaid, N., and Jamil, M. (2023). A blockchain and stacked machine learning approach for malicious nodes' detection in internet of things. *Peer-to-Peer Networking and Applications*, 16(6), 2811–2832.

Nischwitz, M., Esche, M., and Tschorsch, F. (2021). Bernoulli meets pbft: Modeling bft protocols in the presence of dynamic failures. In *2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*, 291–300. IEEE.

Qiu, X., Qin, Z., Wan, W., Zhang, J., Guo, J., Zhang, S., and Xia, J. (2022). A dynamic reputation-based consensus mechanism for blockchain. *Computers, Materials & Continua*, 73(2).