

Artificial Intelligence: Adaptive Certainty Propagation in Distributed Cognitive Systems

Shiva Kumar Bommakati
sbommakanti2@gmail.com
Independent Researcher

Abstract

Distributed artificial intelligence (AI) systems require protocols with scalability and fault tolerance for making decisions asynchronously over networks. Through the adaptive propagation of a certain strength and the recursive structure of consensus summits, a new consensus protocol is proposed in this paper, which is robust and flexible. The proposed framework allows agents to reach agreement with various confidence levels rather than using binary commitments like in traditional approaches. Moreover, these confidence levels are based on validator reliability and consistency. By utilizing a Directed Acyclic Graph (DAG), the flow of information can be organized effectively. This allows for increased scaling and a reduction in the costs of communication. Furthermore, aggregate summit proof validation can be done efficiently as the networks get enhanced. With iteration of the recursive summit mechanism, more trust gets established in the system's decision, and it becomes robust to noisy observations, network latencies, and malicious behavior among parties. The protocol performs its task asynchronously and does not require the existence of a global clock. Moreover, the protocol achieves Byzantine fault tolerance in the presence of up to one-third malicious validators. The adaptive confidence model enhances collective decision-making across heterogeneous AI ecosystems, making the framework suitable for decentralized machine learning platforms and large language model deployments. The method achieves the scalable verification of finality for strong consensus. These attributes make the protocol a potential solution for next-generation distributed artificial intelligence systems.

Keywords

• Adaptive Uncertainty Propagation • Distributed Consensus • Directed Acyclic Graph (DAG) • Recursive Summits • Byzantine Fault Tolerance (BFT) • GHOST Rule • Distributed Cognitive Systems

1. Introduction to Distributed Cognitive Systems and Consensus Challenges

In recent years, the evolution of artificial intelligence has seen a growing convergence with decentralized computing architectures. Distributed cognitive systems comprising numerous agents that perceive, process, and adapt collectively have emerged as powerful frameworks for autonomous decision-making at scale. These systems are especially relevant for environments characterized by uncertainty, dynamism, and a need for robust consensus without centralized control [1, 2].

Consensus protocols in distributed AI are tasked with achieving agreement among heterogeneous agents that may have differing data, latency, or trust assumptions. Classical Byzantine Fault Tolerant (BFT) protocols, such as PBFT and Tendermint, provide deterministic safety guarantees but often incur high communication overheads and limited scalability. Alternatively, probabilistic mechanisms like Proof-

of- Work (PoW) and Proof- of- Stake (PoS) offer broader decentralization at the expense of liveness guarantees and energy efficiency [3, 4].

As distributed AI agents evolve toward more adaptive and resource- efficient models, consensus must become more flexible. A rigid consensus mechanism imposes bottlenecks in settings where agent availability fluctuates or real- time coordination is required. This motivates the need for protocols that propagate not only binary agreement but also evolving levels of certainty about the system state, enabling more informed and agile decision- making.

Adaptive certainty propagation refers to the mechanism by which agents in a distributed system maintain and communicate graded confidence levels about proposed values, rather than waiting for binary finality before acting.

The concept of adaptive certainty propagation is central to this paradigm. Rather than waiting for full consensus before action, agents may act on a graded understanding of what is likely to be agreed upon. This is particularly relevant in distributed learning, swarm robotics, sensor fusion, and trust management systems, where confidence- based behavior is more natural and efficient [5].

The Highway protocol introduces a novel solution to these challenges. Built atop a Directed Acyclic Graph (DAG) structure, it allows agents (validators) to asynchronously build consensus by creating messages that cite and justify earlier messages. Finality is not enforced by a central round or hard threshold, but rather through accumulated confidence, derived from recursive endorsements and support from other validators.

Unlike traditional chain- based blockchains, the Highway protocol enables flexible timing, asynchronous validator activity, and finality based on “summits” sets of validators that vouch for a proposed value above a predefined confidence threshold. This ensures that agents with partial views can still reason about the system’s likely trajectory without re- running the entire protocol state.

Summits are recursively structured collections of validator votes that provide evidence of agreement over time; a summit at level k consists of validators that support votes from level $k - 1$, creating a chain of confidence that leads to finality.

Moreover, Highway facilitates a clean separation between safety and liveness. Finality and commitment to a decision emerge through local observations and cryptographic attestations, whereas liveness is achieved by timely construction of messages and propagation of votes. This separation allows system designers to optimize for either latency or robustness depending on application context [6].

This paper explores the architectural design, theoretical guarantees, and practical implementations of Highway- style adaptive consensus. We analyze its DAG construction, message finality, fork resolution, and resistance to spam or equivocation. Through this, we argue that Highway provides a principled and scalable foundation for consensus in intelligent distributed cognitive systems.

In the following sections, we delve into the system architecture and DAG model, define the logic of flexible confidence propagation, and present mechanisms for fork resolution and adversarial robustness, concluding with practical implications and future directions.

2. System Architecture and DAG-Based Information Flow

At the heart of the Highway protocol lies a Directed Acyclic Graph (DAG) structure that models message propagation among validators in a distributed system. Each validator periodically produces messages known as units that record its observations of the current state of the system. These units cite other prior messages, forming a DAG that reflects causality and validator support [7, 8].

Unlike linear blockchains where each block has a single predecessor, the DAG in Highway enables

multiple concurrent units to be valid, so long as they do not conflict. This allows for asynchronous behavior and better utilization of validator bandwidth, especially in heterogeneous or high-latency environments. Validators construct units that reference multiple earlier units, including their own and others', thereby recursively building trust.

Each unit comprises the following fields, presented in Table 1:

Table 1: Unit message structure in Highway DAG

Field	Description
Creator ID	Cryptographic identifier (public key hash) of the validator producing the unit
Round number	Logical timestamp indicating the validator's round when producing this unit
Parent references	Set of hashes pointing to previous units cited as justification (typically 2–10 parents)
Justification payload	Cryptographic evidence supporting decisions, including votes, endorsements, and equivocation proofs
Signature	Validator's digital signature over all preceding fields

These fields collectively ensure that each unit is uniquely attributable, causally linked to prior knowledge, and verifiable by any recipient without requiring full DAG history.

Equivocation detection is a key architectural component. If a validator produces two conflicting units at the same round, other validators can detect this by comparing justifications and mark the sender as faulty. This prevents malicious actors from manipulating the DAG or injecting forks without being noticed.

Validators maintain a local view of the DAG, which grows as they receive new messages. Each node in the DAG (i.e., each unit) implicitly contains a chain of prior knowledge, and validators use this to reason about the confidence in any given proposal or vote. This recursive referencing enables adaptive finality, where the system does not rely on a global clock or hard rounds.

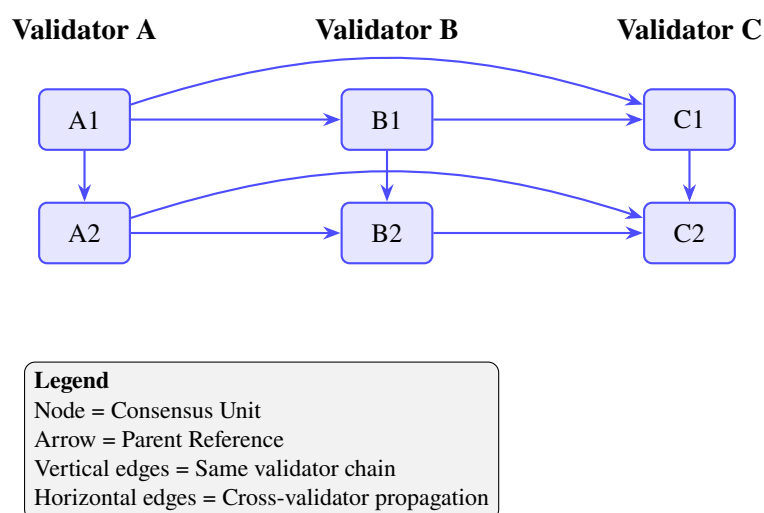


Figure 1: Directed acyclic graph illustrating unit propagation across three validators. Each validator generates sequential units that reference earlier units locally and across validators, enabling efficient dissemination and causal ordering of consensus information.

The DAG also supports pruning and optimization. Units that are fully justified and confirmed can be archived to prevent indefinite DAG growth. Moreover, the acyclic nature guarantees a well- founded order, ensuring that units can be topologically sorted for further analysis or snapshot generation [9, 10].

In summary, the DAG- based architecture supports asynchronous, scalable, and fault- tolerant information propagation among distributed cognitive agents. It lays the groundwork for confidence accumulation, flexible voting, and finality detection described in subsequent sections.

3. Flexible Confidence Propagation and Finality Logic

Traditional consensus protocols in distributed systems often depend on rigid finality thresholds, such as two-thirds of nodes agreeing before a decision is considered irreversible. While this ensures strong safety, it limits flexibility in heterogeneous environments where some agents may be slow or temporarily offline. The Highway protocol introduces a novel mechanism called summits to support flexible and adaptive finality [11, 12].

To illustrate the summit concept concretely, consider a system with seven validators $V_1 \dots V_7$, where validator weights are equal for simplicity. When validator V_1 proposes a value x , it gathers votes from other validators. A Level- 1 summit for x forms when five validators ($> 2f + 1$ where $f = 2$) explicitly support x in their units. Subsequently, a Level- 2 summit emerges when five of these supporting validators produce new units that cite and endorse the Level- 1 supporting units, demonstrating continued commitment. This recursive endorsement continues; a Level- 3 summit finalizes x when five validators endorse the Level- 2 structure. Each level adds a layer of confirmation, making reversal increasingly improbable. In practice, validators need not wait for Level- 3 if application requirements tolerate lower confidence—this tunability is the essence of adaptive certainty.

This approach enables confidence propagation: validators may act on information not yet finalized but supported by a growing number of trusted peers. It reflects a shift from binary finality to graded certainty, enabling faster responsiveness and fault- tolerant inference under partial information.

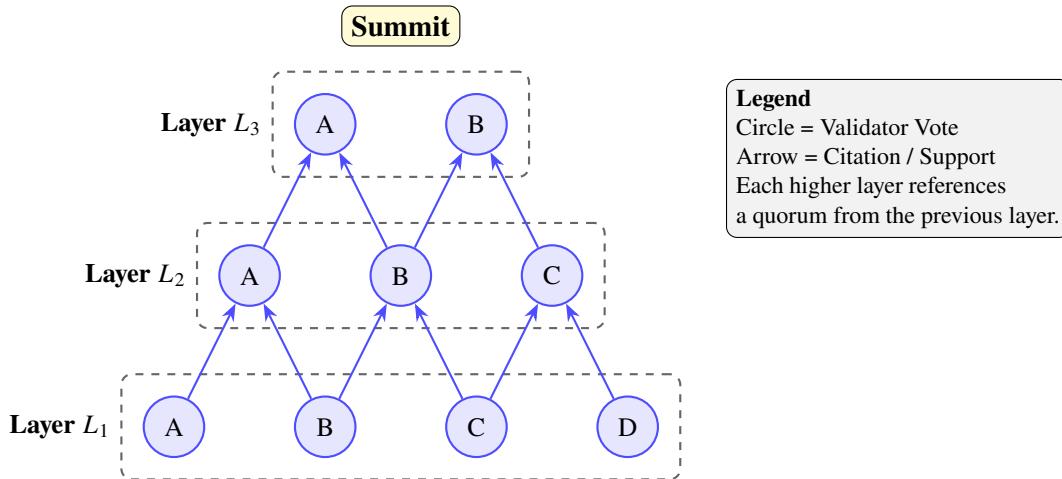


Figure 2: Summit structure in DAG-based Byzantine fault tolerant consensus. Each successive summit layer is formed by validator units that collectively cite a quorum of supporting votes from the preceding layer, thereby strengthening the evidence required for finality.

Figure 2 shows how summit layers are constructed. Layer 1 contains initial votes (V_1 - V_3); Layer 2 contains validators that endorse Layer 1 votes, and Layer 3 further endorses Layer 2. Finality increases with each additional supported layer.

Summits are defined in terms of their level (i.e., the depth of support) and the total validator weight included in each level. For example, a Level- 1 summit requires that more than $2f + 1$ validators support a vote, where f is the maximum number of faulty validators. A Level- 2 summit recursively requires $2f + 1$ support of the Level- 1 summit, and so on [13, 14].

Table 2 outlines how summit levels correspond to growing confidence. The system designer can select the required level based on the application's latency and safety needs. For low- latency use cases, Level- 1 may be sufficient, while critical systems may demand Level- 3 before commitment.

Table 2: Finality Thresholds for Summit Levels

Summit Level	Min. Validator Weight	Confidence Implication
1	$> 2f + 1$	Tentative Agreement
2	$> 2f + 1$ (supporting Level 1)	Strong Consensus
3	$> 2f + 1$ (supporting Level 2)	Finalized

This flexibility allows asynchronous and heterogeneous agents to contribute to consensus without halting the protocol for stragglers. Validators that miss a round can rejoin by citing recent summits and continue contributing to finality in future rounds.

Importantly, the recursive nature of summits enables efficient proofs. A validator need not evaluate the entire DAG- only a compact justification structure. This greatly reduces computational overhead for finality verification, making the system scalable and light- client friendly.

Overall, flexible confidence propagation via summits supports fast finality in typical cases while maintaining robustness under adversarial conditions. The next section explores how votes and forks are resolved using the GHOST rule and how validators navigate ambiguity in the DAG.

4. Voting, Fork Resolution, and the GHOST Rule

In decentralized systems built upon Directed Acyclic Graphs (DAGs), conflicting messages or forks may emerge when validators asynchronously produce units that reference diverging histories. Efficient and safe fork resolution mechanisms are therefore essential to ensure that all honest validators converge on the same final value. The Highway protocol employs a variation of the GHOST (Greedy Heaviest Observed Sub- Tree) rule, adapted for a DAG structure, to resolve such ambiguities [10, 15].

The GHOST rule operates by assigning weights to validator votes and selecting the heaviest branch as the canonical chain. In the context of Highway, GHOST is applied virtually- validators compute the most supported unit (or chain of units) by recursively traversing child nodes with the most cumulative support. This does not require active on- chain votes but is derived from the structure of the DAG.

To initiate voting, each validator assigns its weight to a specific unit often one that builds on the latest finalized summit. Virtual voting then aggregates these weights through the DAG's causality links. The heaviest path emerging from the DAG is interpreted as the most probable consensus trajectory, and validators update their estimates accordingly [16, 17].

Figure 3 shows how GHOST resolves forks by selecting the path with the most downstream votes. Although U0 has three children, U1 has more cumulative support from subsequent validators, making it the preferred chain for finalization.

To further formalize voting dynamics, validators implement threshold-based vote propagation. For a value to be considered "confirmed," it must be supported by a set of validators whose cumulative weight

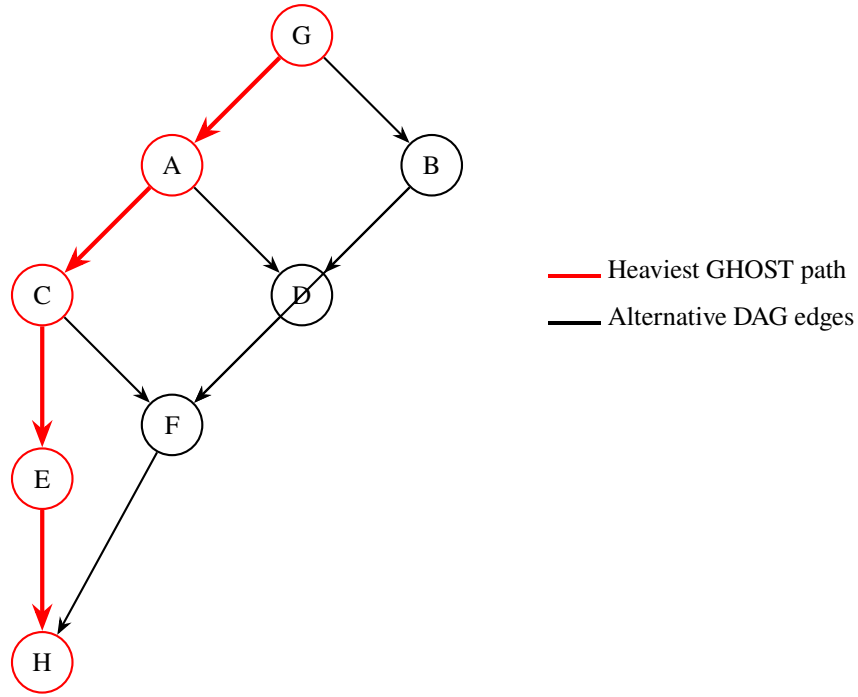


Figure 3: Illustration of GHOST path selection in a Directed Acyclic Graph (DAG). The red path represents the heaviest chain selected by the GHOST rule, while the black edges denote alternative branches that contribute less cumulative weight.

exceeds a predefined threshold (e.g., $> 2f + 1$). These votes may be implicit- embedded within message justifications- or explicit via endorsements.

Table 3: Comparison of Voting Strategies

Strategy	Vote Type	Latency	Fork Safety
On-chain Voting	Explicit	High	Strong
Virtual Voting (GHOST)	Implicit	Low	High
Summit Voting	Recursive	Moderate	Very High

Table 3 compares three key voting strategies. Highway leverages virtual voting for responsiveness and summit- based finality for safety. Together, they form a hybrid consensus that adapts to validator activity and network conditions.

Additionally, validators must handle equivocations. If a validator issues conflicting units, its vote weight is reduced or eliminated. Equivocation is detected by analyzing DAG references- if two units from the same validator exist without a causal relationship, the system flags them as faulty.

Importantly, the GHOST rule does not enforce finality by itself. Instead, it guides fork resolution and vote direction. Finality is achieved once summit conditions are met along the GHOST path. This decouples value estimation from commitment, improving performance and reducing deadlocks.

In summary, Highway’s voting protocol combines the efficiency of GHOST- based fork resolution with the robustness of recursive summits. Validators can make rapid progress even in asynchronous settings while preserving safety.

5. Spam Prevention and Security via Endorsements

As DAG- based consensus protocols become more widely adopted in distributed cognitive systems, controlling network spam becomes essential. Unbounded message creation or malicious equivocation can inflate the DAG, delay finality, and exhaust validator resources. The Highway protocol mitigates these challenges through a mechanism called endorsements, which acts as a rate limiter and spam filter [4, 18].

Validators may produce multiple units per round, referencing previous messages and justifying their votes. However, without constraints, a malicious actor could initiate a fork bomb- - producing divergent messages across multiple rounds to increase DAG complexity. Additionally, vertical spam occurs when a validator produces a long sequence of messages without meaningful justifications, artificially inflating their local view.

To address this, the protocol introduces endorsements: validators explicitly reference a subset of units they support for finality. These endorsements serve as cryptographic filters, reducing ambiguity and ensuring that only well- supported units participate in summit formation and vote propagation.

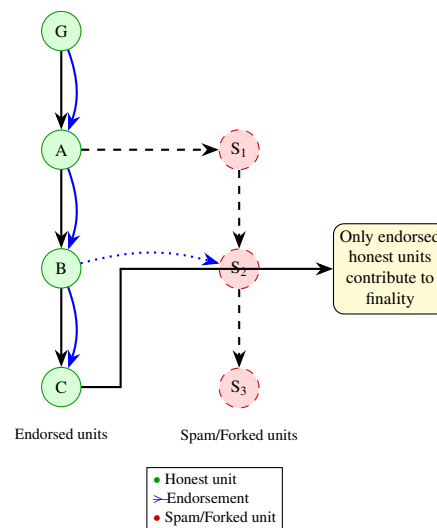


Figure 4: Spam control through endorsements in a DAG-based consensus protocol. Honest units receive endorsements from validators and participate in finality, whereas spam or forked units lacking sufficient endorsements are excluded from the consensus decision.

Each validator is limited to a configurable message rate (e.g., one unit per δ seconds). If a validator exceeds this rate without meaningful endorsements or votes, their messages are deprioritized. This prevents vertical spam and reduces DAG bloating. Validators are incentivized to produce high- quality, well- justified units, reinforcing honest participation.

Table 4 summarizes common spam types and their corresponding countermeasures. Endorsements offer a lightweight but effective method for curating the DAG without requiring centralized validators or external auditors. Additionally, finality logic leverages only endorsed units. This prevents malicious validators from forming illusory summits using spammy branches. If a message lacks sufficient endorsements from non- equivocating validators, it is ignored in summit construction and vote tallying.

In a broader security context, endorsements can also be interpreted as “soft slashing signals.” If a validator repeatedly produces unendorsed or conflicting messages, their trust weight can be reduced over time. This creates a reputation- aware consensus model, aligning validator incentives with protocol health.

Thus, spam prevention in Highway is not enforced through rigid exclusion but rather through adaptive filtering and economic disincentives. Endorsements provide a principled way to maintain DAG integrity,

Table 4: Spam Types and Defenses

Type	Behavior	Defense
Fork Bomb	Conflicting branches	Equivocation check
Vertical Spam	Fast units, no votes	Rate limit
Flooding	Repetitive messages	Endorsement filter
Cross-Spam	Unendorsed mixing	Finality rule

reduce overhead, and preserve finality under adversarial stress.

The next section explores how Highway maintains liveness, balances flexibility with safety, and adapts to dynamic validator participation, concluding the system’s practical readiness for real- world distributed intelligence [19].

5.1 Adversarial Model and Fault Assumptions

The Highway protocol operates under the partially synchronous network model and assumes up to f Byzantine faulty validators among a total of $n = 3f + 1$ or more participants. Byzantine validators may exhibit arbitrary behavior including equivocation (producing conflicting messages), censorship (selectively dropping messages), or spam attacks. The protocol guarantees [3, 5]:

- **Safety:** No two honest validators finalize conflicting values, provided the fraction of faulty validators remains below $\frac{1}{3}$.
- **Liveness:** Transactions will eventually be finalized if network asynchrony resolves and at least $2f + 1$ honest validators are actively participating.
- **Adaptive security:** Validator weights can be dynamically adjusted based on observed behavior (e.g., equivocation reduces effective voting power).

These assumptions align with standard Byzantine fault tolerance literature and ensure compatibility with permissioned or proof- of- stake permissionless deployments.

6. Liveness, Practical Implementation, and Theoretical Guarantees

In distributed cognitive systems, liveness is the assurance that the system will eventually make progress, even under asynchrony, latency, or partial validator failure. The Highway protocol achieves liveness through adaptive round scheduling, cryptographic attestations, and dynamic validator participation, ensuring that decisions are reached even in imperfect conditions [17, 18].

Unlike synchronous protocols that rely on global round boundaries, Highway allows validators to operate in loosely timed epochs. Each validator independently determines when to create new units based on a round length function, which increases over time if consensus is delayed. This adaptive scheduling minimizes wasted effort while allowing recovery from network delays.

Validators also produce “witness units” at scheduled intervals, serving as heartbeat signals and points of coordination. These units help build the summit structure by anchoring votes and endorsements, and allow validators to gauge progress even if some peers are temporarily unresponsive [20].

To ensure that forks do not stall progress, the protocol includes a mechanism for fallback summits. If a preferred proposal fails to reach consensus, validators can converge on the next- best supported

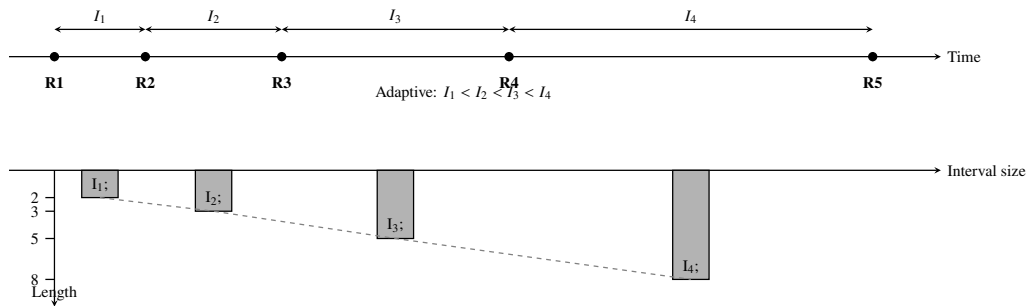


Figure 5: Adaptive round scheduling with increasing intervals under delay

value using inherited justifications. This prevents deadlock and guarantees that some value is eventually finalized.

From an implementation standpoint, Highway is efficient. Validators maintain a local DAG view, prune obsolete units, and need only verify a compact justification (i.e., summit proof) to confirm finality. This supports scalability, mobile- friendly clients, and efficient state synchronization.

Real- world deployments have demonstrated that Highway can finalize thousands of units per second across dozens of globally distributed validators. Its modular design enables it to be integrated with various runtime environments, including smart contract chains, IoT systems, and distributed knowledge networks[21].

On a theoretical level, Highway guarantees:

- **Safety:** No two honest validators finalize conflicting values.
- **Liveness:** With at least $2f + 1$ honest validators active, new values will eventually be finalized.
- **Adaptive Finality:** Time to finality varies based on observed validator activity and message support.

These properties are expected under the assumptions of the partially synchronous model and are consistent with prior analyses of DAG-based consensus protocols.

7. Conclusion

The Highway protocol is a consensus mechanism designed specifically for distributed cognitive systems that require adaptive and scalable Byzantine-fault-tolerant decision-making. Distinct from traditional protocols that impose inflexible binary finality, Highway incorporates the concept of flexible certainty propagation, enabling agents to reason using graded confidence levels derived from recursive summit structures. The transition from deterministic to probabilistic finality facilitates faster responses in dynamic environments characterized by heterogeneous validator participation, network latency, and short-term faults, while preserving safety guarantees.

The architecture is organized as a Directed Acyclic Graph (DAG) that reflects the causal relations between validator messages, allowing information to be transmitted asynchronously and enabling efficient fork resolution using the GHOST rule. The layers of the chain of evidence provided by the summit mechanism strengthen progressively confidence in a value while allowing for tuning latency against assurance. Endorsements are a simple tool for spam filtering and equivocation detection, ensuring only well-justified units contribute to the consensus, and preserving the integrity of the DAG under attack. The

above features yield a protocol that is safe and live in the standard partially synchronous model with up to one-third faulty validators. The protocol implements adaptive round scheduling and adjusts committee weights based on recent behaviour observed.

Highway is practically suitable for decentralized artificial intelligence applications such as federated learning, swarm robotics, sensor fusion, large-scale language model coordination and more. The modular structure and compact justification proofs allow for lighter implementations and resource-constrained edge devices, which extend beyond traditional blockchain applications. Prior studies of DAG-based consensus protocols suggest that Highway-style approaches may achieve high throughput and low finality overhead in a globally distributed validator set.

Nonetheless, several questions are still open for future work. First, though the protocol can handle Byzantine faults, further empirical investigations as well as tuning of parameters is required for extreme network partitions or large-scale validator churns. The proper design of the economic incentives, particularly stake-based slashing and reward mechanisms for honest participation in permissionless settings, is crucial to align the long-term behaviour of a validator with the health of the protocol. Integration of Highway with machine learning pipelines for scenario which can expect consensus requirements (e.g. model update or gradient aggregation) can offer a scope for cross-layer optimisation, e.g. when optimising confidence propagation, uncertainty estimates of neural networks could be incorporated. Moreover, adaptation of adversarial mechanisms could also help reinforce protocol safety and liveness properties through mechanised proof assistant formal verification that may provide even better guarantees for safety-critical systems.

The principles of Highway adaptive finality, recursive evidence, and DAG-based information flow are a promising basis for the next generation of distributed cognitive infrastructures. The evolution of consensus protocols must go from rigid synchrony and binary agreement, as AI systems become decentralised and collaborative. The highway offers a flexible, robust, and scalable design for such evolution and provides a bridge from the theoretical distribution of computing to the actual coordination of intelligent agents. Future research could lead to fresh perspectives and analyses of autonomous, trustworthy, and resilient collective intelligence of systems.

References

- [1] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition, 2021.
- [2] Reza Olfati-Saber, J. Alex Fax, and Richard M. Murray. Consensus and Cooperation in Networked Multi-Agent Systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.
- [3] Miguel Castro and Barbara Liskov. Practical Byzantine Fault Tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, pages 173–186, 1999.
- [4] W. Zhong, C. Yang, W. Liang, J. Cai, L. Chen, J. Liao, and N. Xiong. Byzantine fault-tolerant consensus algorithms: A survey. *Electronics*, 12(18):3801, 2023.
- [5] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the Presence of Partial Synchrony. *Journal of the ACM (JACM)*, 35(2):288–323, 1988.
- [6] M. Levin. Technological approach to mind everywhere: an experimentally-grounded framework for understanding diverse bodies and minds. *Frontiers in Systems Neuroscience*, 16:768201, 2022.

- [7] S. Byeon and W. Lee. Directed acyclic graphs for clinical research: a tutorial. *Journal of Minimally Invasive Surgery*, 26(3):97, 2023.
- [8] J. C. Digitale, J. N. Martin, and M. M. Glymour. Tutorial on directed acyclic graphs. *Journal of Clinical Epidemiology*, 142:264–267, 2022.
- [9] Christian Decker and Roger Wattenhofer. Information Propagation in the Bitcoin Network. In *IEEE P2P 2013 Proceedings*, pages 1–10. IEEE, 2013.
- [10] Yonatan Sompolinsky and Aviv Zohar. Secure High-Rate Transaction Processing in Bitcoin. In *Financial Cryptography and Data Security*, pages 507–527. Springer, 2015.
- [11] Vlad Zamfir. The Correct-by-Construction Casper Protocol, 2018. Accessed: 2026-06-26.
- [12] Lewis Gudgeon, Daniel Perez, Dominic Harz, and Benjamin Livshits. Sovereign Consensus: Towards a Global Standard for Blockchain Governance, 2020. arXiv preprint.
- [13] C. Wang, X. Qiang, M. Xu, and T. Wu. Recent advances in surrogate modeling methods for uncertainty quantification and propagation. *Symmetry*, 14(6):1219, 2022.
- [14] A. Sasikumar, L. Ravi, K. Kotecha, A. Abraham, M. Devarajan, and S. Vairavasundaram. A secure big data storage framework based on blockchain consensus mechanism with flexible finality. *IEEE Access*, 11:56712–56725, 2023.
- [15] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin Backbone Protocol: Analysis and Applications. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 281–310. Springer, 2015.
- [16] Rafael Pass and Elaine Shi. Hybrid Consensus: Efficient Consensus in the Permissionless Model. In *Proceedings of the 31st International Symposium on Distributed Computing (DISC)*, 2017.
- [17] Ling Ren, Tianyu Yu, and Zonghang Li. Analysis of the Liveness and Safety of DAG-Based Consensus Protocols. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 115–123, 2019.
- [18] Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, and Dawn Song. The Honey Badger of BFT Protocols. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 31–42, 2016.
- [19] C. Schwarz-Schilling, J. Neu, B. Monnot, A. Asgaonkar, E. N. Tas, and D. Tse. Three attacks on proof-of-stake ethereum. In *International Conference on Financial Cryptography and Data Security*, pages 560–576, Cham, May 2022. Springer International Publishing.
- [20] Gavin Wood. Polkadot: Vision for a Heterogeneous Multi-Chain Framework, 2016.
- [21] D. L. Barack and J. W. Krakauer. Two views on the cognitive brain. *Nature Reviews Neuroscience*, 22(6):359–371, 2021.