Scaling Multichain Systems using DAG-based Architecture

Maksym Kotov[1,†], Serhii Toliupa[1,*,†], Serhii Buchyk[1,†] and Ivan Parkhomenko[1,†]

[1]*Taras Shevchenko National University of Kyiv, 60 Volodymyrska St., Kyiv, 01033, Ukraine*

Abstract

The demand for decentralized value exchange solutions has been steadily growing for the last few decades. In that light, blockchain technology has gained momentum as a trustless platform for arbitrary computations. The myriads of consensus protocols, networks, and custom functionalities enable the migration of a wide range of traditional centralized finance into a distributed environment, reducing the impact of a single custodian entity. Nonetheless, the technology has met significant limitations related to scaling. Modern blockchain networks are faced with high transaction latencies, low throughput, and ever-growing storage requirements. Gradual iterative solutions have been proposed to offload computational complexity tied to consensus onto subnetworks or an off-chain extension. Such solutions often come in the form of second-layer structures, such as rollups, or an entirely new approach towards consensus similar to the Ethereum 2.0 multichain system. A tree-based state sharding and load balancing has been proposed in recent studies but is associated with limitations of its own. In the following article we will provide an overview of existing scaling solutions and propose an improvement of tree-based architecture by generalizing the approach to a directed acyclic graph (DAG)-based system. Its structure, security considerations, and communication management between interconnected chains will be laid out, providing a basis for further research in scalability and efficiency in multichain networks.

Keywords

Blockchain; Multichain Systems; Blockchain State Sharding; Security Incidents; Multichain Communication Protocols; DAG-based Multichain Architecture.

1. Introduction

1.1. Scaling Blockchain Networks

Traditional financial institutions require both interacting clients to put trust into a third party and its accounting capabilities. The outsourcing of building a compliant asset exchange system simplified security management and integration efforts, while in exchange, a centralized entity keeps a private ledger holding records about each transaction ever processed within the institution and holds responsibility for its consistency and validity.

Unfortunately, there were plenty of historical occasions when such a trusted entity became a reason for financial data leakage or fraudulent activities either because of a coordinated attack against the institution or unintentional changes made by employees. Centralized solutions highly depend on political and social landscape which could endanger trusted assets in critical situations. Hence, the demand for decentralized solutions has been steadily rising for the past few decades.

The most widely adopted technology for decentralized asset exchange is a blockchain network. Comprised of thousands of participating peers, each responsible for validation and verification of

Information Technology and Implementation (IT&I-2025), November 20-21, 2025, Kyiv, Ukraine

*Corresponding author.

† These authors contributed equally.

maksym_kotov@ukr.net (M. Kotov); tolupa@i.ua (S. Toliupa); buchyk@knu.ua (S. Buchyk); ivan.parkhomenko@knu.ua (I. Parkhomenko)

0000-0003-1153-3198 (M. Kotov); 0000-0002-1919-9174 (S. Toliupa); 0000-0003-0892-3494 (S. Buchyk); 0000-0001-6889-9284 (I. Parkhomenko)

data consistency, blockchain technology lays foundation for trustless and secure accounting of occurring transactions [1-2].

Nonetheless, the basic Consistency, Availability, and Partition Tolerance (CAP) theorem has its implication for building such networks [3]. It is of paramount importance for a finance system to provide consistency guarantees as a top priority. Hence, the decision comes down to a balance between availability and partition tolerance. Since the nature of a decentralized solution implies distribution, the tradeoffs were mainly centered around the availability, namely the throughput, transaction finality, and latencies associated with the addition of new records to the ledger.

Most modern consensus protocols in blockchain networks require an agreement by a majority of participants to either accept or deny a new block [1-2]. The majority could be represented by the held assets of validating nodes, customized trust rules, or a simple popular count. That creates a significant bottleneck in the throughput of transactions and their respective latencies.

Significant efforts have been put into research to improve the finalization speed and throughput, which are mostly centered around the second layer of blockchain systems [1-2]. This layer is built on top of the basic consensus protocol and usually provides an off-chain computational power for executing smart contracts and providing proof of their validity [4].

A prominent example of such an approach used in Ethereum is called rollups and helps to significantly reduce latencies and increase the throughput of the system [5]. In its essence, the side network provides the computation, while the main is responsible solely for keeping the records. While improving latencies and throughput, it only optimizes the network but does not solve the horizontal scaling problem.

1.2. Expansion to Multichain Systems

The previously discussed extensions improve these conditions, but to allow horizontal scaling while keeping strong consistency guarantees, another approach is required using sharding as a core principle. Such a system is comprised of multiple subnetworks or autonomous networks interacting with each other through custom consensus or communication protocols [6-8].

As of now, Ethereum is one of the most popular networks utilizing such a principle. By leveraging a Beacon chain and a Proof of Validity mechanism, it allows the distribution of processing of transactions between multiple subnetworks [9-11]. Ethereum's multichain system establishes a shared security mechanism with validators pooling and rotation, where each validator is assigned to a particular subnetwork for a limited timeframe through a consensus process established on the Beacon chain.

Cosmos Network is another example of an attempt to build a large-scale multichain system and serves as a communication layer between external networks. In contrast to the Ethereum 2.0 solution, its main goal is to conjoin multiple independent networks rather than build an isolated multichain system [12], [13].

Lastly, Avalanche leverages subnetworking and multiple ledgers for division of responsibilities. C-Chain executes smart contracts, P-Chain manages the lifecycle of subnets, while X-Chain manages the native assets available on Avalanche. [14].

1.3. Tree-Based State Sharding

Sharding the state of a blockchain in a tree-like structure is an attempt to combine the scalability properties of the Cosmos network while locally improving security on par with the Ethereum network. The main purpose of such architecture is to divide traffic both by cohesion and responsibility zones. The rationale and comparison between existing solutions are provided in their own dedicated article [15].

The network is comprised of three main structures: the chain trunk, bough chains, and spring chains. Here, the chain trunk serves as a root of the network and is a gene-sis point of a multichain system, its governance, and its coordination. The trunk connects distant leaves and might become the primary bottleneck in distant communication [15].

Bough chains are the managing chains responsible for implementing the communication protocol among its subjected spring chains. A bough chain could potentially manage and validate another bough chain, enabling growth of the structure. Spring chains are the main computational units to be used by clients. While it is possible to execute a transaction on a bough chain, it would cost more and is incentivized against due to scalability considerations [15].

Figure 1 shows a diagram of the tree-based multichain system architecture:

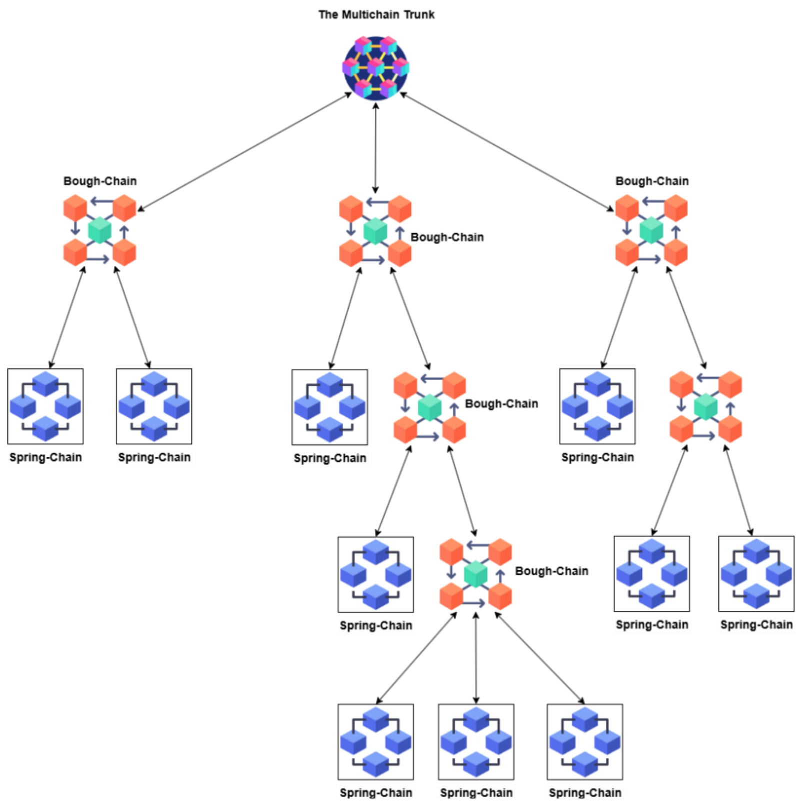

Figure 1: Tree-based architecture of a multichain system.

Here we can clearly see a combination of properties from previously discussed multichain solutions. The tree-based invariant incorporates Proof of Validity similar to one used in Ethereum 2.0 for the bough chain and its child networks. It also promotes distant communication through parents or custom channels similar to Cosmos. The division between governance ledger and client-oriented load is separated akin to the Avalanche approach, resulting in an efficient sharding method.

The tree-based architecture also describes three primary communication modes between its subnetworks. The primary and most efficient is through the parent bough chain and a Proof of Validity mechanism. Secondary is a multi-hop transaction passing to non-distant relatives. Lastly, the architecture allows establishing custom communication bridges between distant chains to avoid congestions [15].

1.4. Purpose of the Article

The purpose of the article is to improve the limitation tied to the trunk chain in the tree-based architecture. The overall goal is to develop an architecture based on Directed Acyclic Graphs (DAG) that aims to improve the scalability and security properties of the proposed model [16-17].

When constructing a multichain system model, evaluating its efficiency against existing solutions is crucial. Therefore, this paper discusses the comparison model as one of its key points. The model includes parameters representing security and scalability properties and evaluates them based on the previous iteration, a recently suggested enhancement, and a collection of pre-existing networks.

2. Architecture Based on Directed Acyclic Graphs

2.1. Coordination Hierarchy

The architecture is a finite, simple, directed graph $G = (V, E)$ [16-17], where:

- V is a finite set of nodes (chains). The cardinality $|V|$ is the number of nodes.
- $E \subseteq V \times V$ is a set of directed edges (ordered pairs), $(u, v) \in E$.
- Simplicity: no self-loops $(v, v) \notin E$ and no multi-edges (because edges are elements of a set of ordered pairs).
- Acyclicity: there is no directed cycle (no sequence $v_0 \to v_1 \to \cdots \to v_k = v_0$ with $k \geq 1$).

A directed graph is acyclic iff it admits a topological ranking [16-17].

A level function (topological ranking):

$$\ell : V \to \mathbb{N} = \{0,1,2, \dots\}. \tag{1}$$

This function assigns an integer level to each node such that $(u, v) \in E \;\Rightarrow\; \ell(u) < \ell(v)$. Then, for each $k \in \mathbb{N}$, define $L_k := \{v \in V : \ell(v) = k\}$ as a set of assigned levels for each node. ℓ can be chosen to be strict (no equal levels along any edge) since we topologically require $\ell(u) < \ell(v)$.

Any two nodes drawn "horizontally" in a diagram can be totally ordered by refining ℓ (tie-breaking) without changing reachability.

We partition edges into backbone vs cross edges.

There exist disjoint sets $E_h, E_x \subseteq E$ such that:

$$E = E_h \dot\cup E_x, \qquad E_h \cap E_x = \emptyset, \tag{2}$$

where $\dot\cup$ disjoint union, $\emptyset$ is the empty set, E_h is a set of backbone edges, and E_x is a set of cross edges.

For every $v \in V$:

$$deg^-_{E_h}(v) \leq 1, \tag{3}$$

where $deg^-_{E_h}(v) := |\{u \in V : (u, v) \in E_h\}|$.

Consequences:

- $H = (V, E_h)$ is a forest of arborescences (disjoint union of rooted out-trees), with (possibly multiple) roots R.
- No node has two different backbone parents.

For every $(u, v) \in E_x$:

$$\ell(u) < \ell(v) \land (u, v) \notin E_h. \tag{4}$$

Cross edges preserve acyclicity and cannot duplicate a backbone edge. They may skip levels arbitrarily (as long as $\ell(u) < \ell(v)$). The level function organizes edges by levels:

$$E_h \subseteq \bigcup_{k \in \mathbb{N}} L_k \times L_{k+1} \tag{5}$$

Backbone edges move exactly one level down:

$$E_x \subseteq \bigcup_{i<j} L_i \times L_j \tag{6}$$

Cross edges move one or more levels down.

Any DAG with a strict ℓ satisfies equations 5 and 6 is a modelling choice that makes the backbone "one-step" layered and is compatible with refining ℓ.

Out-degree in the backbone:

$$deg^{+}_{E_h}(v) := |\{w \in V : (v, w) \in E_h\}|. \tag{7}$$

Roles:

- Root node: $v \in R \Leftrightarrow deg^{-}_{E_h}(v) = 0$;
- Bough node: $deg^{+}_{E_h}(v) \geq 1$;
- Spring node (leaf): $deg^{+}_{E_h}(v) = 0$.

Roles are derived, not labelled: they depend only on degrees in E_h. Multiple roots are allowed (no global "trunk").

Figure 2 shows a diagram of the DAG-based multichain system architecture:

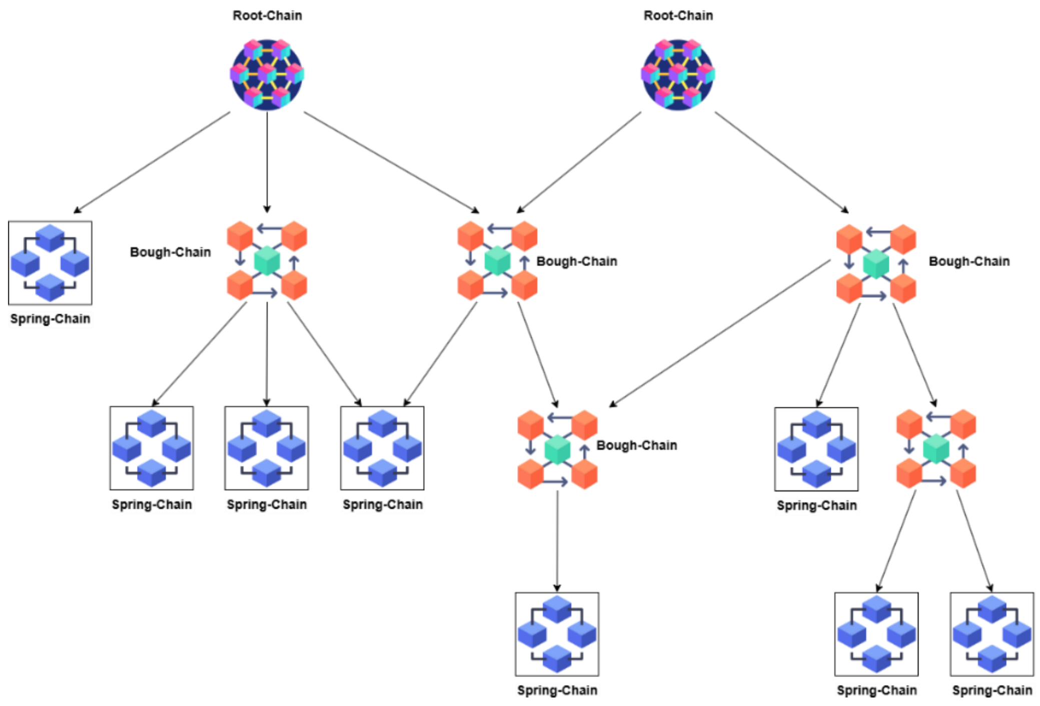

Figure 2: DAG-based architecture of a multichain system.

A quadruple (V, E_h, E_x, ℓ) defines a valid DAG-multichain architecture iff all of the following hold:

- DAG: $G = (V, E)$ with $E = E_h \dot\cup E_x$ is acyclic and admits ℓ with $(u, v) \in E \Longrightarrow \ell(u) < \ell(v)$.
- Partition: $E_h \cap E_x = \emptyset$.
- Backbone forest: $deg^{-}_{E_h}(v) \leq 1$ for all $v \in V$ (thus $H = (V, E_h)$ is a disjoint union of rooted out-trees with roots R.
- Cross-edge forwardness: $(u, v) \in E_x \Longrightarrow \ell(u) < \ell(v)$.

- Layering (optional normalization): $E_h \subseteq \bigcup_{k\in\mathbb{N}} L_k \times L_{k+1} \wedge E_x \subseteq \bigcup_{i<j} L_i \times L_j$.

Here, (1)–(4) are the essential structural rules and (5) is a convenient normalization matching layered diagrams; it's always achievable by strict topological numbering.

2.2. Governance Structures and Security

2.2.1. Defining a Governance Structure

A Governance Structure is a shared security mechanism initially introduced for the tree-based architecture. Within DAG-based architecture it could be represented as a subgraph of the multichain backbone [15]:

$$G_d = \left(V_g, E_g\right), \tag{8}$$

where:

- $V_g \subseteq V$ is the subset of bough nodes (blockchain networks) participating in this governance structure.
- $E_g \subseteq E_h$ is the set of parent-child edges between consecutive boughs inside V_g representing default direct control.

By default, G_d consists of all immediate parent–child pairs:

$$G_d = \{\left(v, p(v)\right) \mid v \in V_g, \quad p(v) \neq \emptyset\}, \tag{9}$$

where:

- $p(v)$ = parent function, defined as $p(v) \in V \cup \emptyset$.
- $p(v) = \emptyset$ iff v is a root.

A Governance Structure is essentially the restriction of the backbone forest to a consecutive chain (or small hierarchy) of boughs. It captures their hierarchical organization. There could be many Governance Structures within a multichain system.

2.2.2. Defining a United Governance Structure

A United Governance Structure is a higher-order governance subgraph [15]:

$$G_u = (V_u, E_u), \tag{10}$$

where:

- $V_u \subseteq V$ is a subset of participating bough chains that agreed to extend governance through the consensus process.
- $E_u \subseteq V_u \times V_u$ is the set of agreed governance relations among them with an established shared security.

Formally:

$$G_u = \{\left(v_i, v_j\right) \mid v_i, v_j \in V_u\}. \tag{11}$$

If $\left(v_i, v_j\right) \in G_u$ and $\left(v_j, v_k\right) \in G_u$, then $(v_i, v_k) \in G_u$. Unlike G_d, which is local and hierarchical, G_u allows extended coordination: validator rotation, policy alignment, and inter-chain communication. For simplicity we did not index G_u in the definition, but it should be clear that there could be multiple G_u as well as G_d in a multichain system.

The purpose of a United Governance Structure is to expand shared security principle farther than a direct parent to child relation. Withing such a structure there is a pool of shared and rotated validators to reduce the likelihood of 51% attacks. Such a structure does not scale horizontally since the assignment of validators requires general consensus, hence there could be many disjoined structures throughout the multichain system.

2.3. Communication and Routing

A large-scale multichain system could have numerous blockchain networks interconnected within a global network. Addressing a single chain in such a scenario could pose a problem when relying on unstructured unique names. A directory service would be required either on-chain or off-chain to manage requests routing.

Maintaining such a directory could have its limitations due to update propagation and bottlenecks. Hence, within a discussed architecture, we propose to utilize a structured and routable naming system for individual chains, similar to DNS.

Figure 3 shows a diagram of the DNS-like routing system within the multichain architecture:

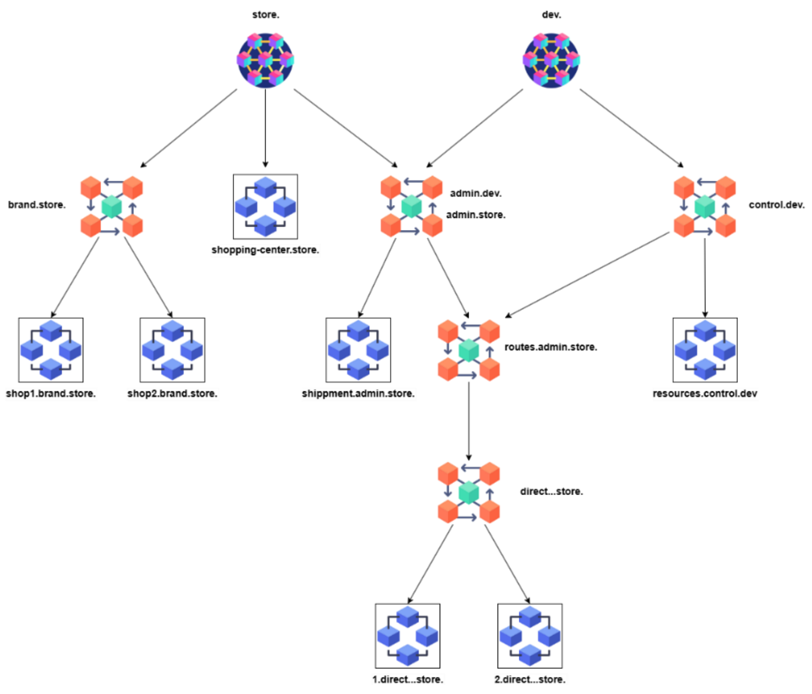

Figure 3: DNS-like routing system within the multichain architecture.

In a classical DNS system, there are multiple levels of servers, including root domain servers, top-level servers, and authoritative servers [18], [19]. Within the proposed DAG-based multichain architecture, the root chains mirror responsibilities of top-level domains, while their direct children would function akin to authoritative nodes.

An advantage in contrast to some centralized directory service is apparent from the perspective of distant networks. When a client from spring chain A tries to create a transaction involving assets on spring chain B, all that chain A has to know is the root chain address, which then points to its authoritative record, allowing for recursive traversal of the graph.

Now a similar problem happening with unstructured names could happen if there are many root chains dynamically created or updated. In such cases the role of a root DNS server could be allocated to a single off-chain directory service or a separate blockchain network. Since the cardinality of the root-level chain should not be significant, the capabilities of a single blockchain network would be enough for such a purpose.

3. Comparison With Existing Solutions

3.1. Scoring Model

3.1.1. Security Scoring Model

Security assessment includes multiple categories discretely scored following ISO/IEC 27005 risk-assessment practice [20]. Each dimension will be scored using a discrete 0-5 ordinal (Likert-style) scale to transform qualitative security properties into quantitative judgments. Such an approach aligns with ISO's guidance to use qualitative/ordinal scales when precise measurements are infeasible due to complexities related to assessed systems and has grounds as a common practice in NIST SP 800-30 [21].

Shared Security (SS), "How much validator power is pooled across domains": no shared security, every domain fully secluded (1); shared security in specific clusters (2); shared security for most, not all, cross-chain traffic (3); shared security covers >90% of inter-chain transactions (4); all interchain operations are validated through a shared pool (5).

Pooled validators set raise economic security and reduce the probability of 51% attacks since it would require an infeasible amount of computational and voting power for an attacker to influence the consensus process. Validators rotation is essential to ensure that there could not be organized pooling of consensus votes.

Coordinator Centralization (CC), "How concentrated cross-chain verification is": single coordination network (1); major coordination network with multiple subordinate chains (3); multiple dominant chains with a balanced workload (3); the multichain is distributed with local coordinators but without global (4); distributed architecture without reliance on coordination hubs (5).

Message Verification (MV), "Trust assumptions in cross-chain message validation": requires trusted external actors for correctness (1); relays required, with partial on-chain checks (2); external agents required for liveness with on-chain verification (3); on-chain proofs exist with partial reliance on coordinator (4); self-sufficient cryptographic proofs for all inter-chain transactions (5).

Trust introduces potential attack vectors, the less trust is put into cross-ledger transaction the better (e.g. verifiable proof are preferred). External actors could get compromised which diminishes decentralized security principles. Global coordination chains are potential single points of failure.

Data Retention & Redundancy (DR), "Ability to re-verify/recover state commitments over time": state could be pruned (1); periodic roots for short history (2); moderate historical checkpoints (3); strong redundancy and proofs (4); explicit retention with configurable depth (5).

Blast Radius / Containment (BR) — "Extent of damage from a compromised domain": full compromise propagates globally (1); spillover limited to hub cluster (2); spillover limited to a governance group (3); spillover limited to parent-child only (4); compromise fully contained (5).

To combine the previously discussed scores, we use a weighted geometric mean [22]. Such a method allows us to expose weaknesses within individual metrics by penalizing disproportionately low scores. We calculate Composite Security Score (CSS) as follows:

$$CSS = (SS^{w_1} \cdot CC^{w_2} \cdot MV^{w_3} \cdot DR^{w_4} \cdot BR^{w_5})^{\frac{1}{w_1+w_2+w_3+w_4+w_5}}, \tag{12}$$

where we will use the following weights during final calculations: $w_1 = 3$ (SS), $w_2 = 3$ (CC), $w_3 = 2$ (MV), $w_4 = 1$ (DR), $w_5 = 2$ (BR).

So the final formula looks as follows:

$$CSS = (SS^3 \cdot CC^3 \cdot MV^2 \cdot DR^1 \cdot BR^2)^{\frac{1}{11}}. \tag{13}$$

Within this scoring model, higher means more secure. Poor dimensions drag down the score (no averaging-away weaknesses). To normalize the score and use it later for cumulative assessment model, we use: $S^* = CSS/5$.

3.1.2. Performance Assessment Model

Within this section we'll introduce 4 performance characteristics that will be later used in cumulative assessment of the architecture. The performance metrics are calculative in relative perspective to compared networks.

Latency utility L^* shows relative performance of an architecture. It compares latencies with respect to the fastest available $L^* = L_{min}/L$, where L_{min} is the lowest latency observed within assessed multichain systems. $\bar{L}$ is an average latency associated with an inter-chain transaction within target architecture. Latencies here include routing, consensus/finality delay, periodic proof interval, and queueing.

Throughput utility T^* measures relative ability of cross-chain message processing per second and defined as $T^* = \mu/\mu_{max}$, where μ is the cross-chain throughput capacity of the architecture and μ_{max} is the highest capacity among the compared set. If the throughput is unlimited within a multichain system, its total score is $T^* = 1.5$.

Const utility C^* shows the cost associated with an inter-chain transaction. The following formula is used: $C^* = C_{min}/\bar{C}$, where C_{min} is the lowest cost observed within assessed multichain system and $\bar{C}$ is the average cost of a target architecture. Since cost could fluctuate due to economic reasons, we assume the cost as an amount of records required on multiple ledgers to perform an asset transfer. If there is no additional cost, then $C^* = 1.5$.

Each cross-chain operation requires proof so that participants from different ledgers could verify the validity of the state. Data availability DA^* shows extra bytes related to the construction of such proofs $DA^* = O_{min}/O$, where O is the overhead within a target system and O_{min} is the minimal overhead among tested systems. If there is no overhead, then $DA^* = 1.5$.

3.1.3. Cumulative assessment model

Within a cumulative assessment model, we include 5 dimensions discussed previously:

$$OverallScore = \ w_L L^* + w_T T^* + w_C C^* + w_{DA} DA^* + w_S S^* \ , \tag{14}$$

where, the weights are summing to 1 with $w_L = 0.2$, $w_T = 0.3$, $w_C = 0.1$, $w_{DA} = 0.1$, $w_S = 0.3$. Therefore, the formula is as follows:

$$OverallScore = \ 0.2 \cdot L^* + 0.3 \cdot T^* + 0.1 \cdot C^* + 0.1 \cdot DA^* + 0.3 \cdot S^*. \tag{15}$$

Here, the higher the value the better.

It is possible to get score larger than 1 due to metrics such as $T^* = 1.5$.

3.2. Ethereum 2.0

Latency in Ethereum 2.0 could be represented $\bar{L}_{Ethereum2.0} \approx 984$ seconds, which describes a finalized assets transfer between shards [23], [24].

Moving on to the throughput μ, each shard should post a commitment to a beacon chain per epoch, where an epoch is roughly 384 seconds [23], [24]. One commitment can contain up to 40000 transactions (~105 transactions per second). Since the architecture limits shards to 64, we can assume a throughput of $\mu_{Ethereum2.0} \approx 105 \cdot 64 = 6720$ cross-chain transactions per second [9].

Cost $\bar{C}$ as a count of records per cross-chain transaction could be expressed as follows: a record on chain A for the transaction, a record on chain A for the proof object, a record for commitment on the beacon chain shared in a batch, a record on chain B for the transaction and one for proof verification [9]. Meaning, $\bar{C}_{Ethereum2.0} \approx 4$.

Data overhead $O_{Ethereum2.0} \approx 544$ bytes for a Merkle proof with a depth of 16 [9], [23], [24]. The beacon chain is also responsible for storing commitments ~256 bytes per shard for an epoch but this is for all transactions inside an epoch.

Security Score:

- Shared Security (SS = 5) – shared validation pool coordinated through a beacon chain [9].
- Coordinator Centralization (CC = 1) – centralized beacon chain is responsible for security management and coordination [9].
- Message Verification (MV=4) – on-chain verification of roots received from the beacon chain [9].
- Data Retention & Redundancy (DR = 2) – the beacon stores periodic roots for data verification but does not support data restoration [9].
- Blast Radius / Containment (BR = 4) – security breaches are isolated to a child chain but potential major issues could happen if the beacon itself gets compromised [9].

Composite Security Score:

$$CSS_{Ethereum2.0} = (5^3 \cdot 1^3 \cdot 4^2 \cdot 2^1 \cdot 4^2)^{\frac{1}{11}} \approx 2.73. \tag{16}$$

Normalized Security Score:

$$S^*{}_{Ethereum2.0} = \frac{2.73}{5} \approx 0.55. \tag{17}$$

3.3. Polkadot

Latency $\bar{L}_{Polkadot} \approx 12$ second to finalize a transaction which includes sending it for validation to the relay chain and inclusion into the destination ledger [10], [11], [25].

Throughput $\mu_{Polkadot} \approx 2000$ transaction per second and is directly tied to the relay chain's ability to validate transactions depending on allocated cores and including limitations on messages to parachains [10], [11], [25].

Cost $\bar{C}_{Polkadot} \approx 5$ includes the original transaction and XCMP message on chain A, entry on the relay chain, and respective transaction and XCMP receipt on chain B [10], [11], [25].

Data overhead $O_{Polkadot} \approx 1536$ bytes for three XCMP entries on chain A, relay, and B respectively [10], [11], [25]. Each XCMP message on both chain A and B contain header information and payload, while relay chain record hold Merkel proof.

Security Score:

- Shared Security (SS = 5) – shared validation pool coordinated through a relay chain [10], [11].
- Coordinator Centralization (CC = 1) – relay chain involved in all inter-chain operations [10], [11].
- Message Verification (MV=4) – XCMP/XCM with verification of proof from the relay chain [10], [11].
- Data Retention & Redundancy (DR = 3) – relay finality and checkpoints for data verification [10], [11].
- Blast Radius / Containment (BR = 3) – parachain faults are isolated but relay faults could propagate [10], [11].

Composite Security Score:

$$CSS_{Polkadot} = (5^3 \cdot 1^3 \cdot 4^2 \cdot 3^1 \cdot 3^2)^{\frac{1}{11}} \approx 2.69. \tag{18}$$

Normalized Security Score:

$$S^*{}_{Polkadot} = \frac{2.69}{5} \approx 0.54. \tag{19}$$

3.4. Cosmos

Latency $\bar{L}_{Cosmos} \approx 30$ seconds to finalize a transaction which includes request commitment on chain A, its transfer to the hub, and later to the chain B [12], [13], [25].

Since Cosmos is a distributed system without an apparent limitation in scale, its throughput final estimation $T^*{}_{Cosmos} = 1.5$ [12], [13].

Cost $\bar{C}_{Cosmos} \approx 8$ in case if we consider a single hub hop. Each hop would include commitment and deletion records on the source and receipt and acknowledgement records on the destination [12], [13].

Data overhead $O_{Cosmos} \approx 192$ bytes the commitment and acknowledgement records [12], [13]. Each additional hop would increase the value but for estimation purposes we will consider only the most frequent case.

Security Score:

- Shared Security (SS = 1) – connects uncoordinated ledgers, does not ensure shared security [12], [13].
- Coordinator Centralization (CC = 4) – does not enforce centralized topology, multiple hubs could be used for connecting ledgers [12], [13].
- Message Verification (MV=4) – on-chain light clients verify headers and proofs with minimized trust [12], [13].
- Data Retention & Redundancy (DR = 2) – light clients windowing and pruning are typical, proof is isolated to a short context [12], [13].
- Blast Radius / Containment (BR = 5) – security breach on one chain does not propagate to another since Cosmos serves as a communication media rather than a shared security coordinator [12], [13].

Composite Security Score:

$$CSS_{Cosmos} = (1^3 \cdot 4^3 \cdot 4^2 \cdot 2^1 \cdot 5^2)^{\frac{1}{11}} \approx 2.68. \tag{20}$$

Normalized Security Score:

$$S^*{}_{Cosmos} = \frac{2.68}{5} \approx 0.54. \tag{21}$$

3.5. Avalanche

Latency $\bar{L}_{Avalanche} \approx 4$ seconds to finalize a transaction which includes commitment to chain A, its transfer through the teleporter, and later to the chain B [14], [25].

Avalanche is a highly distributed system without an apparent limitation in scale of transaction processing. Though it could have unlimited number of subnetworks, the primary chain is responsible for recording the assignments of validators which could become a bottleneck. Its throughput final estimation $T^*{}_{Avalanche} = 1$ [14].

Cost $\bar{C}_{Avalanche} \approx 2$ in case if we consider AWM/Teleporter. One record for the request is written on chain A and another for the import on the chain B [14]. There are no intermediary writes to internal chains.

Data overhead $O_{Avalanche} \approx 512$ bytes where each side writes a normal transaction that carries an AWM proof including the message, BLS aggregate signature, and the metadata [14].

The apparent performance of the Avalanche network stems from its internal organization and reliance on a few coordinating networks. In contrast to completely disjointed systems such as Cosmos, Avalanche requires each subnetwork to register and manage its validators through the main chain. This cohesion allows boosting performance but at the cost of a potential blast in case of the main chain compromise.

Security Score:

- Shared Security (SS = 4) – centralized pooling in place with subnets including validators from the primary chain [14].
- Coordinator Centralization (CC = 2) – shared security coordination network [14].
- Message Verification (MV=4) – cryptographic proof through the warp mechanism [14].
- Data Retention & Redundancy (DR = 2) –history is local to the subchain [14].
- Blast Radius / Containment (BR = 4) – faults are typically isolated to a subnetwork [14].

Composite Security Score:

$$CSS_{Avalanche} = (4^3 \cdot 2^3 \cdot 4^2 \cdot 2^1 \cdot 4^2)^{\frac{1}{11}} \approx 3.1. \tag{22}$$

Normalized Security Score:

$$S^*{}_{Avalanche} = \frac{3.1}{5} \approx 0.62. \tag{23}$$

3.6. Tree-based Architecture

Latency could vary greatly depending on the communication path. It could be either with the siblings or to the distant nodes. Since the design of architecture assumes high cohesion, the most frequent method of communication should be done through the bough chain where latency should be similar to Ethereum 2.0. $\bar{L}_{Tree-based} \approx 984$ seconds to finalize a transaction which includes sending it for validation to the relay chain and inclusion into the destination ledger [15], [23], [24].

The tree-based architecture in its essence as a data structure does not strictly impose the communication mode between siblings, but the original article assumed coordination like Ethereum 2.0 between siblings. Since the Trunk chain underpins the entire system $\mu_{Tree-based} \approx 6720$ transactions per second [15].

Cost $\bar{C}_{Tree-based} = 4$ similar to estimation for Ethereum 2.0 but could grow significantly larger for distant-node communication [15].

Data overhead $O_{Tree-based} = 544$ bytes for coordination through the bough chain between siblings [15]. Again, this could grow with N factor depending on the number of hops needed for communication with distant nodes.

Security Score:

- Shared Security (SS = 2) – shared security within boughs and its direct child network.
- Coordinator Centralization (CC = 2) – has a major coordination network (The Tree Trunk).
- Message Verification (MV=4) – trust minimized cryptographic proofs for most operations.
- Data Retention & Redundancy (DR = 5) – explicit and configurable N-ancestor data retention.
- Blast Radius / Containment (BR = 3) – faults are typically isolated to a subnetwork but could propagate if happen on local coordination networks.

Composite Security Score:

$$CSS_{Tree-based} = (2^3 \cdot 2^3 \cdot 4^2 \cdot 5^1 \cdot 3^2)^{\frac{1}{11}} \approx 2.65. \tag{24}$$

Normalized Security Score:

$$S^*{}_{Tree-based} = \frac{2.65}{5} \approx 0.53. \tag{25}$$

The purpose of the Tree-based architecture was to improve the horizontal scalability capabilities of multichain systems.

3.7. DAG-based Architecture

In contrast to the tree-based, the newly proposed DAG-based architecture assumes communication between siblings similar to the Polkadot's solution. Hence, since communication between siblings is expected to be the dominant way due to cohesion principles, the estimated latency $\bar{L}_{DAG-based} = 12$ second to finalize a transaction which includes sending it for validation to the relay chain and inclusion into the destination.

Throughput in such an architecture is not limited to the trunk chain or any other general coordination network, making its final estimation $T^*_{DAG-based} = 1.5$. There could be many root chains connecting different branches. Moreover, cross-connection facilitates efficient transactions.

Cost $\bar{C}_{DAG-based} = 5$ similar to Polkadot and includes the original transaction and XCMP message on chain A, entry on the relay chain, and respective transaction and XCMP receipt on chain B [10], [11].

Data overhead between sibling chains is similar to the Polkadot, $O_{DAG-based} = 1536$ bytes for three XCMP entries on chain A, relay, and B respectively [10], [11]. Each XCMP message on both chain A and B contain header information and payload, while relay chain record hold Merkel proof. The estimation would differ for a multi-hop transfer but due to cohesion principles, such transactions should be rare.

Security Score:

- Shared Security (SS = 3) – shared security within boughs and its direct child network with additional governance structures that could be shared.
- Coordinator Centralization (CC = 4) – no global coordination network.
- Message Verification (MV=4) – trust minimized cryptographic proofs for most operations.
- Data Retention & Redundancy (DR = 5) – explicit and configurable N-ancestor data retention.
- Blast Radius / Containment (BR = 3) – faults are typically isolated to a subnetwork but could propagate if happen on local coordination networks.

Composite Security Score:

$$CSS_{DAG-based} = (3^3 \cdot 4^3 \cdot 4^2 \cdot 5^1 \cdot 3^2)^{\frac{1}{11}} \approx 3.58. \tag{26}$$

Normalized Security Score:

$$S^*_{DAG-based} = \frac{3.58}{5} \approx 0.72. \tag{27}$$

3.8. Cumulative Assessment Results

The results obtained in the previous subsections could be organized in a table for a more convenient comparison.

Table 1
Comparison of the multichain architecture assessment results.

Title	L^*	T^*	C^*	DA^*	S^*	Total
Ethereum 2.0	0.01	1	0.5	0.35	0.55	0.55
Polkadot	0.33	0.3	0.4	0.13	0.54	0.37
Cosmos	0.13	1.5	0.25	1	0.54	0.76
Avalanche	1	1	1	0.38	0.62	0.82
Tree-based	0.01	1	0.5	0.35	0.53	0.55
DAG-based	0.33	1.5	0.4	0.13	0.72	0.79

From the results we can see the improvement within DAG-based architecture over the tree-based.

4. Conclusions

The increasing growth in traffic related to decentralized operations has led to a surge in research on scalable multichain systems. Novel consensus protocols and inter-chain protocols are being developed to allow the sustainable growth of distributed and decentralized computations. In this article we've provided an overview of existing solutions and proposed an improvement for a tree-based multichain system by generalizing it to a DAG-based system.

A new coordination hierarchy has been discussed, showcasing previous consensus constraints and their mitigation. The DAG-based approach allowed us to avoid a single bottleneck tied to the trunk chain of the system. In contrast to the tree-based architecture, multiple authoritative connection points could be established, allowing for parallelized request flow.

Additionally, the DAG-based approach allows for improving the security properties of the system by enabling multiple higher-level validations of transactions. The intersection of security boundaries allows spreading the validation responsibility to participants in multiple networks, which improves the security guarantees.

Overall, the purpose of this article is to improve a previously developed tree-based architecture by leveraging the properties of a DAG. A comparison between the proposed architecture and existing systems has been provided, showing its relevance. The research aims to spark further interest in building scalable and secure systems.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

Acknowledgements

The authors would like to thank colleagues and reviewers for their valuable comments and suggestions, which helped improve the quality of this work. No external funding was received for this research.

References

[1] Gad, A.G., et al.: Emerging Trends in Blockchain Technology and Applications: A Review and Outlook. Journal of King Saud University - Computer and Information Sciences (2022). https://doi.org/10.1016/j.jksuci.2022.03.007

[2] Habib, G., et al.: Blockchain Technology: Benefits, Challenges, Applications, and Integration of Blockchain Technology with Cloud Computing. Future Internet 14(11), 341 (2022). https://doi.org/10.3390/fi14110341

[3] Gilbert, S., Lynch, N.: The CAP theorem. Computer **45**(2), 30–36 (2012). https://doi.org/10.1109/MC.2011.389

[4] Taherdoost, H.: Smart Contracts in Blockchain Technology: A Critical Review. Information 14(2), 117 (2023). https://doi.org/10.3390/info14020117

[5] Thibault, L.T., Sarry, T., Hafid, A.S.: Blockchain Scaling using Rollups: A Comprehensive Survey. IEEE Access, 1 (2022). https://doi.org/10.1109/access.2022.3200051

[6] Liu, W., et al.: Distributed and Parallel Blockchain: Towards A Multi-Chain System with Enhanced Security. IEEE Transactions on Dependable and Secure Computing, 1–16 (2024). https://doi.org/10.1109/tdsc.2024.3417531

[7] Hashim, F., Shuaib, K., Zaki, N.: Sharding for Scalable Blockchain Networks. SN Computer Science 4(2) (2023). https://doi.org/10.1007/s42979-022-01435-z

[8] Stone, D.: Trustless, Privacy-Preserving Blockchain Bridges. arXiv 2102.04660 (2021). https://doi.org/10.48550/arXiv.2102.04660

[9] Kudzin, A., et al.: Scaling Ethereum 2.0s Cross-Shard Transactions with Refined Data Structures. Cryptography 6(4), 57 (2022). https://doi.org/10.3390/cryptography6040057

[10] Petrowski, J.: The Path of a Parachain Block. Polkadot. https://polkadot.com/blog/the-path-of-a-parachain-block

[11] Abbas, H., Caprolu, M., Di Pietro, R.: Analysis of Polkadot: Architecture, Internals, and Contradictions. In: 2022 IEEE International Conference on Blockchain (Blockchain), pp. 61–70. IEEE, Espoo (2022). https://doi.org/10.1109/Blockchain55522.2022.00019

[12] Essaid, M., Kim, J., Ju, H.: Inter-Blockchain Communication Message Relay Time Measurement and Analysis in Cosmos. Applied Sciences 13(20), 11135 (2023). https://doi.org/10.3390/app132011135

[13] Peelam, M.S., Chaurasia, B.K., Sharma, A.K., Chamola, V., Sikdar, B.: Unlocking the Potential of Interconnected Blockchains: A Comprehensive Study of Cosmos Blockchain Interoperability. IEEE Access 12, 171753–171776 (2024). https://doi.org/10.1109/ACCESS.2024.3497298

[14] Avalanche Docs: Primary Network. Avalanche Docs. https://docs.avax.network/protocol/primary-network

[15] Kotov, M.S.: Tree-based state sharding for scalability and load balancing in multichain systems. Electronic Professional Scientific Journal «Cybersecurity: Education, Science, Technique» 2(26), 392–408 (2024). https://doi.org/10.28925/2663-4023.2024.26.702

[16] Fan, S., Zhang, S., Wang, X., Shi, C.: Directed acyclic graph structure learning from dynamic graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. **37**(6), 7512–7521 (2023)

[17] Directed Acyclic Graph (DAG). Hazelcast Foundations: Distributed Computing. https://hazelcast.com/foundations/distributed-computing/directed-acyclic-graph/ (last accessed 2025/07/22)

[18] Callahan, T., Allman, M., Rabinovich, M.: On modern DNS behavior and properties. SIGCOMM Comput. Commun. Rev. **43**(3), 7–15 (2013). https://doi.org/10.1145/2500098.2500100

[19] van der Toorn, O., Müller, M., Dickinson, S., Hesselman, C., Sperotto, A., van Rijswijk-Deij, R.: Addressing the challenges of modern DNS: a comprehensive tutorial. Comput. Sci. Rev. **45**, 100469 (2022). https://doi.org/10.1016/j.cosrev.2022.100469

[20] ISO/IEC 27005:2018 Information technology – Security techniques – Information security risk management. International Organization for Standardization (2022). https://www.iso.org/standard/80585.html (last accessed 2025/07/28)

[21] Joint Task Force Transformation Initiative: Guide for conducting risk assessments. NIST Special Publication 800-30 Rev. 1. National Institute of Standards and Technology, Gaithersburg (2012). https://doi.org/10.6028/NIST.SP.800-30r1

[22] Siegel, I.H.: Index-number differences: geometric means. J. Am. Stat. Assoc. 37(218), 271–274 (1942). https://doi.org/10.1080/01621459.1942.10500636

[23] The Beacon Chain: Ethereum 2.0 explainer. https://ethos.dev/beacon-chain (last accessed 2025/08/23)

[24] Consensys: The Ethereum 2.0 Beacon Chain Explained. https://consensys.io/blog/the-ethereum-2-0-beacon-chain-explained (last accessed 2025/08/06)

[25] LCX: Blockchain Platform Comparison: A Deep Dive into the Top Networks. https://www.lcx.com/blockchain-platform-comparison-a-deep-dive-into-the-top-networks/ (last accessed 2025/08/06)