Enhancing Availability for IPFS Using Diffused Repository

Shu Wang
Graduate School of Information Science and Engineering
Ritsumeikan University
Osaka, Japan
gr0695vx@ed.ritsumei.ac.jp

Noriaki Kamiyama
College of Information Science and Engineering
Ritsumeikan University
Osaka, Japan
kamiaki@fc.ritsumei.ac.jp

***Abstract*—The InterPlanetary File System (IPFS), a decentralized storage system, is based on content addressing and peer-to-peer networking. However, the data stored in each node is determined by the free will of owners, which often results in the geographical concentration of identical data. Consequently, when large-scale disasters or power outages occur, there is a significant risk of losing access to specific data. Existing approaches such as Filecoin and Kademlia DHT attempt to achieve data distribution through economic incentives or random allocation, but their ability to enhance fault tolerance is limited. To address this problem, we propose a novel method in which nodes permanently store data if the upper bits of their Peer ID match the upper bits of the Data ID. In addition, we introduce a Peer ID assignment algorithm that allocates identical bit strings to repositories distributed across different geographic regions. This design enables the system to maintain data availability even in the event of large-scale disasters. Through simulation-based evaluations, we demonstrate that the proposed method outperforms conventional approaches by reducing retrieval latency under normal conditions and significantly improving recovery success rates during disaster scenarios.**

***Index Terms*—IPFS, Network Topology, Disaster Resilience.**

I. Introduction

The InterPlanetary File System (IPFS) has recently attracted significant interest as a next-generation P2P technology alongside the advancement of decentralized technologies. A fundamental characteristic of IPFS is its adoption of a content-based addressing method, in contrast to the location-based addressing method used by the widely utilized HTTP protocol. Specifically, instead of relying on fixed URLs for data retrieval, IPFS manages data using unique identifiers generated by applying a hash function to the content itself, and possesses a mechanism for efficiently exploring the optimal retrieval location among multiple nodes on the network. This approach eliminates dependence on specific servers, potentially providing strong resistance against failures and censorship.

Moreover, IPFS divides and stores data in small units called blocks, distributing them across multiple nodes [2]. This mechanism enhances redundancy and availability as the number of nodes holding the data increases, and is expected to enable efficient retrieval from the nearest node, reducing communication latency and effectively utilizing bandwidth. Additionally, compared to traditional Content Delivery Network (CDN) distribution mechanisms, a major advantage is the realization of large-scale, autonomous, and low-cost distribution.

On the other hand, a challenge exists in that the replication strategy adopted by the current IPFS is static and does not sufficiently reflect temporal demand fluctuations, such as data popularity (access frequency). Consequently, delayed replication in response to sudden concentration of demand may lead to increased latency, while conversely, redundant storage of unpopular data can result in wasteful resource consumption. Against this backdrop, recent research has explored the introduction of strategies for dynamically adjusting the number and placement of replicas based on demand prediction and analysis. Particularly, adaptive management techniques utilizing machine learning to forecast future popularity are gathering attention as effective means for improving the quality of service in IPFS.

Furthermore, IPFS is not merely a decentralized storage system; it is an important foundational technology supporting the development of Web3.0 and the new Internet. Its characteristics, such as high censorship resistance for information sharing, communication during disasters, and efficient distribution of academic content, hold potential for diverse applications, and broader advancements are anticipated in both research and practical applications in the future.

In actual operation, the act of Pinning, where a node's owner voluntarily decides to persistently store specific data on their own IPFS node, is crucial. However, heavy reliance on the autonomous decisions of individual nodes for data retention leads to the following significant challenges:

- Geographical concentration problem: Identical data may be concentrated and stored within a group of nodes located in close physical proximity.
- Lack of persistence guarantee: Data may become inaccessible during power outages or disasters.
- Lack of redundancy control: The owner cannot explicitly control the number and placement of nodes storing the data [3].

This paper proposes an efficient availability enhancement method for IPFS that is resilient to large-scale disasters such as earthquakes, utilizing an autonomous redundancy control and spatially optimized ID assignment technology for repositories. The following sections are structured as follows: Section II discusses related work, Section III describes the proposed method, Section IV presents numerical evaluation results, and finally, Section V concludes the paper.

II. Related Work

A. IPFS

IPFS (InterPlanetary File System) is a decentralized P2P storage and sharing system, primarily realized through the following two technologies:

- DHT-based Content Placement: IPFS search and content placement utilize a Distributed Hash Table (DHT) based on Kademlia [4]. Each node is assigned a generated Peer ID, and each content item has a unique Content Identifier (CID) determined by a hash function. The DHT defines an XOR distance between the CID and Peer ID, selecting the node with the smallest distance to this content as the candidate holder. When a search request is issued, the query is relayed among nodes with progressively closer XOR distances until the target CID is reached. This enables a decentralized system without a central server.
- Block-based Management: Data stored on IPFS is typically divided and managed in 256KB units called blocks. Each block has its own unique CID and is replicated across multiple nodes. When a specific file is requested, parallel searches for its constituent blocks are conducted, and the retrieved blocks are sequentially combined to fully reconstruct the original file. While this mechanism provides a certain degree of redundancy, if nodes holding specific blocks are lost due to disasters, it remains difficult to restore the entire file if it relies on those blocks. Thus, a challenge exists in that conventional methods cannot sufficiently guarantee availability.

B. Filecoin

Existing approaches like Filecoin [7] and Kademlia DHT [5], which attempt to achieve distribution through economic incentives or random allocation for the practical application and performance improvement of IPFS, have limitations in enhancing fault tolerance against large-scale disasters.

Filecoin manages storage contracts and cryptocurrency rewards between providers and users on the IPFS network, incentivizing nodes to reliably store data over the long term. It thus offers a solution to the sustainability problem inherent in IPFS alone, namely "who will persistently store data for extended periods?". However, issues exist, such as the high computational requirements and associated processing delays for contract proofs, leading to increased overhead, which is a noted drawback for small-scale or short-term usage.

C. IPFS Cluster

IPFS Cluster is a framework for integrally managing multiple IPFS nodes, controlling replica placement, and performing load balancing [6]. Compared to relying on the autonomous storage decisions of individual IPFS nodes, using Cluster allows administrators to flexibly specify the number and placement of replicas, offering advantages in stable and reliable quality assurance. It is particularly effective in enhancing availability for large-scale data and improving resource efficiency. However, this control fundamentally relies on a centralized approach, compromising complete decentralization and autonomy. Furthermore, in environments with dynamic node participation or large-scale failures, manual adjustment is necessary, indicating a limitation due to the substantial burden on administrators.

III. Availability Enhancement Technology for IPFS Using Diffused Repository

A. ID-based Storage Scheme

This paper defines "Repositories" as nodes provided by multiple entities for persistent storage. A key feature is that each repository operates as a standard IPFS node but performs permanent storage for data deemed its responsibility. The method for determining target data involves introducing a rule: a node permanently stores data if the upper y bits of its own Peer ID match the upper y bits of the data's Content ID (CID). By controlling y, the total number of nodes N results in each data item being stored by approximately $N/2^y$ nodes. This mechanism allows for a trade-off: a smaller y increases the number of storing nodes, enhancing availability against node failures, while a larger y reduces the number of storing nodes, decreasing overhead. This controllability is a major advantage of the proposed method.

Furthermore, two operational methods are considered: one where a common y value is set for all repositories, and another where each owner arbitrarily selects their y value. In the latter case, applying Filecoin's verification mechanism or having repositories periodically mutually confirm compliance with storage obligations could ensure adherence. This allows for flexible enhancement of IPFS availability with low control overhead, enabling arbitrary persistent storage of any data by any repository.

B. ID Assignment

The design method for assigning IDs is described. The core idea of this method is to assign identical upper-bit IDs to repositories that are geographically dispersed [1]. This increases the probability that identical data will be stored in different geographic locations, enhancing fault tolerance against large-scale disasters and reducing retrieval latency for each access.

If a central management authority exists, this authority manages all repositories and deterministically assigns the upper bits of the ID. Specifically, starting with the set S_k , a recursive bipartitioning operation is performed. For each new partition at step $k+1$, the sum of the shortest path distances T_n between nodes within each new subset $S_(k+1)$ is calculated, and the 0/1 assignment is made to minimize $\sum T_n$. A greedy algorithm is applied for this recursive assignment. In addition to T_n defined based on network connectivity, a weighted term incorporating the geographic distance $d_(geo)$ can be used. Here, $d_(geo)$ is defined as the straight-line distance between any two nodes, based on a planar coordinate system. This enables efficient hierarchical ID assignment that reflects the network topology.

On the other hand, in a fully decentralized environment without a central authority, when a new node joins, it collects the ID and coordinate information of neighboring nodes. Based on this, it determines its own upper bits by referencing the prefix of the nearest existing group of nodes and maximizing the distance (e.g., XOR distance) from them. This ensures distinguishability between geographically close nodes and maintains a balanced distributed structure overall.

By applying the proposed method, it resolves the challenge of "inability to control the number and location of storing nodes" inherent in conventional IPFS. It addresses the issue of "geographical redundancy" that economic incentives like Filecoin or replica management like IPFS Cluster do not fully address, enabling controllable availability. A major characteristic of this method is the flexible adjustment of availability through the parameter y. Furthermore, by adopting a design applicable to both centrally managed and fully decentralized operational environments, it possesses versatility tailored for diverse practical deployment scenarios.

In summary, the proposed framework of "Repository-based scheme + ID assignment" provides a new framework for efficiently and flexibly enhancing IPFS availability through the combination of introducing repositories and ID assignment. Nodes provided by multiple entities persistently store data based on specific conditions, adhering to the rule that they must store data whose upper y bits of the Content ID match the upper y bits of their own Peer ID. Any data is, on average, guaranteed to be persistently stored by $N/2^y$ nodes.

IV. Performance Evaluation

A. Evaluation Scenarios

Evaluations were conducted under the following two primary scenarios:

- Normal State: Assuming an environment where all nodes are operational, retrieval latency was measured. This evaluation assesses how efficiently the proposed method performs searches compared to the conventional method under normal operating conditions.
- Disaster Occurrence Scenario: Simulating large-scale failures such as earthquakes and power outages, all nodes within a specific defined area were simultaneously disabled. Specifically, a disaster area was defined as a circular region with radius r, containing nodes. Subsequently, retrieval requests were issued for each data item, success rates were calculated, and the fault tolerance of both methods was compared. By progressively expanding the disaster area, changes in overall resilience were analyzed.

B. Comparison Method

In conventional IPFS, the storage of data by nodes is left to the autonomous decisions of each node, with no mechanism forcing long-term storage of specific data. This was simulated for comparison in the simulations. The configuration for the conventional method determined storing nodes based on this principle, simulating the autonomous and potentially uneven storage of actual IPFS, including the possibility of data being stored with regional bias. This inherently contains significant limitations from the perspectives of availability and fault tolerance.

C. Evaluation Metrics

The primary metrics used for evaluation were as follows: Under normal conditions, the average retrieval time per request (Average Retrieval Delay), the average number of hops required for a requesting node to reach a storing node (Average Hop Count), and the probability of successful retrieval within 4 hops (4-hop Recovery Rate) were measured. For disaster scenarios, the ratio of successful complete file recovery (Recovery Success Rate) was used as the evaluation metric.

D. Ensuring Fairness

To ensure a fair comparison between the two methods, the number of nodes storing each data item was adjusted to be identical for both the conventional and proposed methods (by selecting the value of y corresponding to data popularity). The number of nodes, network topology, communication bandwidth, and data characteristics were set identically. Each experiment was repeated 120 times, and the average values were used for evaluation. For disasters, the center of the disaster area was fixed, and the radius was varied. This eliminates randomness and facilitates analysis of the impact of increasing damage scale.

E. Dataset Used

For evaluation, a network topology was generated based on existing research to reflect realistic distribution characteristics. As shown in Table 1, file size distributions followed a long-tailed distribution, known to be characteristic of actual Web and FTP distributions [8], and a Lognormal distribution was used.

TABLE I: Properties of file size distribution (M bits)

	MEAN	*MID*	*STD*	*MAX*	*MIN*
File Size	4.52	2.73	7.06	139.82	0.1

Evaluations were conducted for three scales: 10, 100, and 1000 files, to compare the performance of both methods across different network sizes. Request generation was based on a Zipf distribution for file popularity, with the popularity parameter θ varying between 0.6 and 0.9[9], replicating the phenomenon of concentration on a few highly popular files.

F. Simulation Environment

The proposed method and the conventional method were applied and their performance compared under identical conditions using the US commercial ISP "at home" network topology shown in Figure 1.

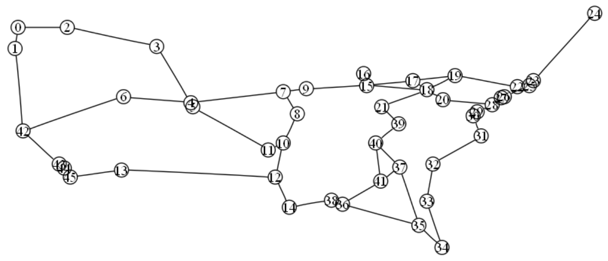

Fig. 1: Network Topology of at home network

G. Numerical Results

Figure 2 shows the results for the normal state scenario with 1000 files following a Zipf distribution, as the parameter θ varies. It displays (a) average retrieval delay per file, (b) average hop count, and (c) success rate within 4 hops. Regarding average retrieval delay, the proposed method consistently achieved about 10% to 20% lower delay than the conventional method, indicating stable low-latency performance. Next, regarding the average hop count, the proposed method consistently required about 10% to 15%fewer hops than the conventional method, confirming more efficient retrieval. The success rate within 4 hops was over 20% higher for the proposed method compared to the conventional method, demonstrating maintained high availability even in environments including low-popularity content. These results suggest that the proposed method delivers stable performance regardless of fluctuations in θ and the bias of the popularity distribution. Particularly, in disaster scenarios where numerous low-popularity items exist, the proposed method offers superior recovery efficiency and availability compared to the conventional method, which is a major strength of this approach.

Figure 3 shows the recovery success rate against the failure radius for the three different file counts (10, 100, 1000) under disaster conditions. The proposed method consistently outperformed the conventional method across all scales, with the difference becoming particularly pronounced as the disaster scale increased. The proposed method maintained a recovery success rate approximately 20% higher than the conventional method, showing a gradual decline as the failure radius increased. For 100 and 1000 files, where the conventional method exhibited a sharp drop in success rate, the proposed method ensured a success rate nearly 1.5 times higher, confirming its stability. These results support the claim that the proposed method maintains high availability even when facing widespread node failures due to large-scale disasters. Notably, while the conventional method's performance degraded rapidly with increasing scale, the proposed method demonstrated consistent and robust resilience independent of scale.

In addition to evaluations based on the overall file set, an evaluation focusing on low-popularity data was conducted. Specifically, under conditions where the number of storing nodes for low-popularity data was less than 6.25% of the total nodes, the recovery success rates of both methods against the failure radius are shown in Figure 4(b). For comparison, the results for high-popularity data (stored by at least 12.5% of the nodes) are shown in Figure 4(a). The proposed method maintained performance comparable to the conventional method for high-popularity data, while for low-popularity data, its recovery success rate was confirmed to be higher than even the high-popularity case. The recovery success rate of the proposed method was superior to the conventional method, especially for data with lower overall popularity.

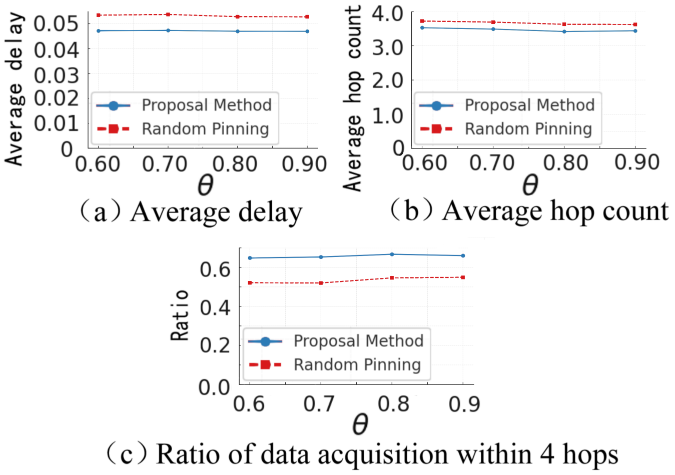

Fig. 2: Three properties against Zipf parameter θ in normal state

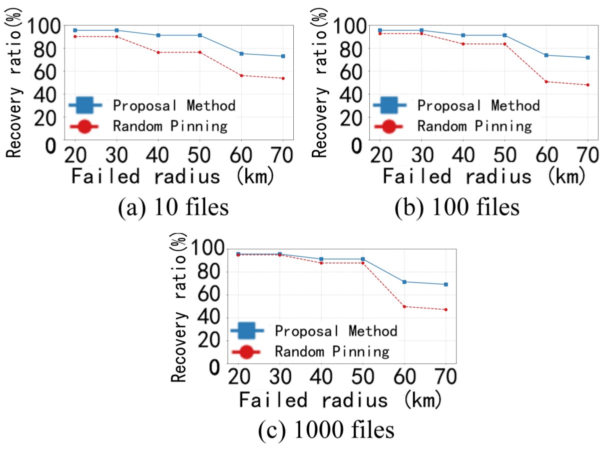

Fig. 3: Recover ratio against failed radius with different file count

V. Conclusion

This research focused on the challenge of degraded availability in decentralized storage systems, stemming from the autonomy of data storage decisions. We proposed a novel availability enhancement method utilizing a repository-based scheme and optimized ID assignment. In conventional IPFS, reliance on the free will of individual nodes leads to geographical concentration of identical data and high risk of inaccessibility during large-scale disasters or power outages.

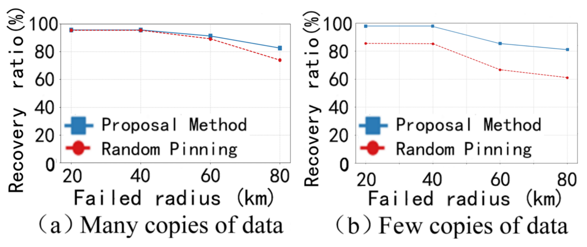

Fig. 4: Recover ratio against failed radius when (a)many copies of data exist at all nodes and (b)few copies of data exist at all nodes

While existing solutions like Filecoin and IPFS Cluster offer certain improvements, their limitations, such as computational intensity and management dependence, prevent fundamental resolution of the issue.

The proposed method in this paper introduces repositories provided by multiple entities, establishing a rule where each repository permanently stores data if the upper y bits of its Peer ID match the upper y bits of the Data ID. Adjusting y allows flexible control over the availability level. Furthermore, the ID assignment method allocates identical prefix sequences to geographically dispersed repositories, confirming reduced risk of simultaneous failure during disasters and ensured redundancy. Evaluations demonstrated that the proposed method outperformed the conventional method under normal conditions in terms of average delay and hop count, and significantly improved disaster recovery success rates. Notably, the proposed method showed markedly effective performance for large networks and for scenarios involving numerous low-popularity items. In conclusion, the proposed method offers a promising approach for efficiently and flexibly enhancing data availability in IPFS. Future work will involve performance verification incorporating dynamic node participation and departure in larger-scale network environments, and implementation towards practical deployment to enhance the feasibility of this method.

Acknowledgment

This research was supported in part by JSPS KAKENHI Grant Numbers 25K03113 and 23K28078. The authors would like to express their gratitude for this support.

References

[1] N. and M. Murata, “Dispersing Content Over Networks in InformationCentric Networking”, IEEE Transactions on Network and Service Management 2019.

[2] J. Batiz-Benet, “IPFS - Content Addressed, Versioned, P2P File System”, arXiv 2014.

[3] R. Shi, R. Cheng, Y. Fu, B. Han, Y. Cheng, and S. Chen, “Centralization in the Decentralized Web: Challenges and Opportunities in IPFS Data Management”, Proceedings of the ACM 2025.

[4] Stoica, I. Morris, R. Liben-Nowell, D. Karger, D.R. Kaashoek, M.F. Dabek, F. Balakrishnan, H., “Chord: a scalable peer-to-peer lookup protocol for Internet applications”, IEEE/ACM Transactions on Networking 2003

[5] M. Petar, M. David, “Kademlia: A Peer-to-peer Information System Based on the XOR Metric”. IPTPS 2002.

[6] Protocol Labs, IPFS Cluster Documentation. [Online]. Available: https://cluster.ipfs.io/

[7] L. Zhihao, L. Dave, S. Neil, and Bobby Bhattacharjee, “Internet anycast: performance, problems, & potential”, ACM SIGCOMM 2018.

[8] A. B. Downey, “Evidence for Long-Tailed Distributions in the Internet”, SIGCOMM Internet Measurement Workshop 2001.

[9] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web Caching and Zipf-like Distributions: Evidence and Implications”, IEEE INFOCOM 1999.