Keyword-based multimedia data lookup in decentralized systems

Keyword-based multimedia data lookup in decentralized systems

Emanuele Fazzini*, Mirko Zichichi†, Stefano Ferretti‡, Gabriele D'Angelo*,
*Department of Computer Science and Engineering, University of Bologna, Italy
†IOTA Foundation, Italy
‡Department of Pure and Applied Sciences, University of Urbino Carlo Bo, Italy
emanuele.fazzini@studio.unibo.it, mirko.zichichi@iota.org,
stefano.ferretti@uniurb.it, g.dangelo@unibo.it

***Abstract*—The use of decentralized systems has the potential to offer secure and efficient collaboration without intermediaries, and in the absence or fault of a central server. However, searching for content in a decentralized system is still an issue, especially when real-time information needs to be retrieved (e.g. emergency situations). We propose a keyword-based decentralized lookup scheme, based on the International Standard Content Code (ISCC), which enables the unique identification of digital content without the need for a centralized registry or authority. Our approach exploits a hypercube Distributed Hash Table (DHT) for storing and retrieving shared resources and compares different approaches to index contents within it. Results show that the use of ISCC, as the basis for creating identifiers, enables more efficient content placement and retrieval, which could pave the way for novel decentralized solutions that surmount the presence of centralized servers for content lookup.**

***Index Terms*—Distributed Hash Table, Keyword Search, Decentralized File Storage, Distributed Ledger Technology**

I. Introduction

Decentralized systems have the potential to transform modern (yet, still centralized) traditional services. The combined use of modern Distributed Ledger technologies (DLTs) and classic peer-to-peers approaches allow the design of systems that can surmount the limits related to the presence of a single point of failure (i.e. the server), that can easily shut down a service in presence of sudden ICT failures, and thus ensure higher resilience in disaster contexts [1]. In fact, the offered redundancy minimizes the impact of individual failures and ensures that essential services can continue to operate even in challenging circumstances.

Moreover, decentralized systems empower individuals by ensuring freedom of expression and information exchange, offering a resilient and censorship-resistant platform that safeguards against oppressive control [2]. Traditional centralized systems can sometimes be susceptible to misinformation or manipulation, particularly during times of crisis. Decentralized systems, built on transparent and immutable blockchain technology, enable secure and efficient interactions without the need for intermediaries [3], [4], [5]. This helps to mitigate the spread of false information, enabling better-informed decision-making and collaboration.

This work has received funding from the University of Urbino through the "Bit4Food" research project.

A relevant problem in decentralized systems arises when looking for content [6], [7]. In a previous work, we proposed a solution for content lookup, based on a Distributed Hash Table (DHT) [8], [9]. A DHT is a distributed system that provides a lookup service similar to a hash table: any participating node can efficiently retrieve the value associated with a given key [10]. In particular, we resort to a Hypercube DHT. P2P nodes form an overlay topology shaped as a hypercube, with each node having a unique identifier represented as a binary string. The length of the binary string determines the number of dimensions of the hypercube. The storing and lookup of a resource in the DHT is based on the navigation inside the different dimensions of the hypercube, in order to reach the appropriate node that maintains information about that resource. The main limitation of that solution was that it assumes a naive keyword mapping lookup scheme that can create scalability issues in large decentralized systems.

We propose a novel keyword-based decentralized lookup scheme thought for multimedia content. The goal is to define a similarity content lookup scheme. able to identify similar documents and present them as potential search results. With this in view, we employ the International Standard Content Code (ISCC). It is a standardized content identification system, that enables the unique identification of digital content without the need for a centralized registry or authority. The ISCC system uses a combination of cryptographic hashing and content analysis to generate a unique identifier for any given piece of digital content. This identifier allows verifying the authenticity and integrity of the content. Moreover, we believe it enables content discovery. In fact, through this approach, similar contents have similar identifiers (ids), and this might ease the lookup process. Another key benefit of the ISCC system is that it is decentralized and open-source, meaning that anyone can use it to identify and verify digital contents without the need for a centralized authority or proprietary software. This makes it ideal for use in decentralized systems such as blockchain networks or P2P content sharing platforms, where there is a need for a reliable and standardized content identification system, not in charge to a single entity.

An array of diverse use case applications could benefit from this solution. Indeed, the combination of content discoverability, system reliability, scalability, together with the

cryptographic features of decentralized architectures bolster security by guarding against unauthorized access and data tampering. These qualities are highly relevant in contexts such as data sharing, data censorship resilience, information retrieval, collaborative networks as well as traceability and supply chain management.

To assess the our proposal, we set up a set of experiments using classes of images. We compare different approaches to index contents within the Hypercube DHT, with respect to the classic hashing scheme commonly used in DHTs. Results show that ISCC-based identifiers enable more efficient content placement and retrieval. This eases the discovery of multimedia content, which is distributed over Decentralized File Storage (DFS) systems.

The remainder of this paper is organized as follows. Section II provides some background. Section III presents the proposed system. Section IV discusses the experimental evaluation and results. Finally, conclusions are provided in Section V.

II. Background and Related Work

A. Content Lookup in Decentralized Systems

Content discovery in distributed systems has a relevant state of the art [9], [11]. There are two main categories of approaches, i.e. client/server and P2P based solutions. Searching content in a client/server system is quite simple. The server has complete knowledge about where contents are located. Thus, clients can ask the server in order to retrieve data. Things get more complicated when a decentralized system (i.e., P2P) is employed [12], [13]. Two main approaches to content lookup in P2P systems exist, i.e. unstructured and structured.

An unstructured content lookup is a simple approach where peers forward requests to their neighbors without any specific organization or structure [14]. Nodes are usually connected randomly and do not follow any specific topology, or if they do, this topology is not related to the contents nodes store [15]. In unstructured content lookup, a node disseminates a query message to its neighbors, and each of these neighbors further send the message to their neighbors, until the message reaches the desired content [6]. Unstructured content lookup can be inefficient in large-scale P2P networks, where the number of nodes and the amount of data to be stored are large.

Structured content lookup is a more organized approach that usually involves Distributed Hash Tables (DHTs) to locate contents [16]. DHTs are a type of data structure that distributes data evenly among nodes. Each node is responsible for storing a small subset of the data, and the distribution is based on a predefined key-space. In structured content lookup, when a node wants to retrieve content, it first hashes the content's unique identifier to obtain a key. The node then uses the key to locate the node responsible for storing the content in the DHT. The node can then retrieve the content from that node directly. Structured content lookup is more efficient than unstructured content lookup because it allows for content retrieval without extensive message flooding [17]. In our system, we resort to a specific DHT shaped as a hypercube.

B. Hypercube

The Hypercube DHT is a structured P2P network that organizes nodes in a hypercube topology [8], [18]. Nodes are assigned unique identifiers, encoded as binary strings of a fixed length. These nodes are responsible for maintaining information about specific contents, which are indexed through ids mapped in the same key-space of node ids. The main concept is defining how identifiers are associated with contents. The typical solution is to compute the hash of the desired data object to obtain a binary string that corresponds to the id in the DHT. In this case, the content lookup turns into identifying the node whose id is closer to the hashed identifier. This solution works if the content to look for is known, i.e. the user does have the actual data, but he knows precisely which data is needed.

When there is the need to make more complex queries to the system, some alternative solution is needed. For example, a user might want to locate images of a specific location (e.g. the Colosseum in Rome, Italy). In this case, he does not know which is the file to retrieve, but he knows a specific keyword associated with the type of the image he needs (e.g. `location:Rome`, `subject:Colosseum`). In this case, content indexing can rely to keywords associated with the metadata of the content. This is the goal we pursue in our system. In the rest of this section, we review the approach we already employed in [8].

To sum up, we are looking for a method that, based on a set K of keywords exploited to perform a query, computes an identifier that describes a typology of contents (e.g. images of the Colosseum in Rome). This leads to the recognition of rID as the id for that query, which is used to locate the node v in the DHT that has knowledge about all the contents resolving the query. Thus, the node will answer with all the references to these contents available in the decentralized storage. Clearly, these references are, in turn, other identifiers xID to locate contents (that have nothing to do with the DHT ids).

Table I provides a description of identifiers involved in our system. Consider the xID as a generic identifier for some particular content in the decentralized storage. For instance, xID might be the CID for an IPFS Object or a specific DLT transaction identifier. xID is associated to a keyword set K describing the content resolved by xID. We assume these keywords belong to a domain $K \subseteq W$.

Now, a uniform hash function $h : W \rightarrow \{0, 1, \ldots, r-1\}$ is employed to map the keyword set K to a keyword set id rID. In particular, for each $k \in W$, $h(k)$ sets to 1 one specific bit of the r-bit string given by $mod_r(h(k))$. (Put in other words, each $k \in W$ has an assigned position in the r-bit string). Thus, the rID related to the keyword set K is generated as a r-bit string where the positions are "activated" (i.e., set to 1), by all the $k \in K$, i.e. $one(rID) = \{mod_r(h(k)) \mid k \in K\}$.

These r-bit strings represent not only a keyword set K, but they are used to identify logical nodes in the Hypercube DHT network. For example, if we fix the size of the r-bit string to $r = 4$, then node ids can take binary values from 0000 to 1111.

ID type	Description
Content id, xID	It is the identifier for the content to be retrieved in the decentralized storage. It can be the CID in the case of IPFS, a DLT transaction ID, etc. In the DHT, it is the *value* of the <*key*,*value*> mapping.
Node id, v	It represents a vertex of the hypercube DHT. It is encoded as a r-bit string. This logical node is associated with a physical node in the distributed system that maintains pointers to where data with certain characteristics, i.e., set of keywords, are physically stored and retrievable.
Keyword set id, rID	It represents an identifier for a certain set of keywords K describing given contents. Thus, rID aggregates in one single information different keywords. It is encoded as a r-bit string in the same space of node ids.

TABLE I: Summary of identifiers employed in this work.

We can formally define a r-dimensional hypercube $H_r(V, E)$ as a set of vertices V and a set of edges E connecting them. Each of the 2^r vertices represents a logical DHT node, while edges are formed when two vertices ids differ by only one bit, e.g., 1011 and 1010 share an edge. The distance between two vertices u and v can be measured using the Hamming distance, i.e., $Hamming(u, v) = \sum_{i=0}^{r-1}(u_i \oplus v_i)$, where $\oplus$ is the XOR operation and u_i is the bit at the i-th position of the u string, e.g., for $u = 1011$ and $v = 1010$, we have $Hamming(u, v) = 1$.

Contents are discovered through queries based on a keyword set id rID, which corresponds to a point in the hypercube. In fact, content references are stored on the node u with the identifier closest to rID (recall that node ids and resource ids are mapped into the same key-space). The query takes as input a keyword set id rID and, starting from a random node v, the request is propagated in the net until reaching a node u, which is responsible for that keyword set rID. This basic search will return all and only the xIDs exactly associated with a keyword set K, i.e., $\{xID \in D \mid K_{xID} = K\}$, maintained by the responsible node u (pin search). What might happen, however, is that one could be interested also in contents that are described by keyword sets that include K, i.e., $\{xID \in D \mid K_{xID} \supseteq K\}$ (superset search). Thus, it is also possible to ask for contents not only at the u node, but at its neighbours (in the hypercube) as well. Clearly enough, in this case a limit l is set to the number of returned results obtained by all the nodes with id associated to the keyword sets $K_{xID} \supseteq K$.

C. International Standard Content Code

The International Standard Content Code (ISCC) [19] is an ISO-approved standard that, given an input file, creates a corresponding code to identify the file itself. Generating the code consists of employing a set of content-driven, locality-sensitive, and similarity-preserving hash functions. Unlike cryptographic hash functions, these hash functions aim to preserve similarity between data, so that two similar contents do not have different codes. This process consists of four main components.

- Meta-Code: encodes metadata similarity, e.g., *AAA5C73C3GZDHDHD*;
- Content-Code: encodes perceptual or syntactic similarity of contents, e.g., *EEA2T2CQVF6ZR7BU*;
- Data-Code: encodes raw bitstream data similarity, e.g., *GAACDVmDpTfvWZfP*;
- Instance-Code: encodes checksum for data integrity, e.g., *IAACRtoh1WeiDvEi*.

These components can be considered separately, all together, or used to generate a code represented by a digest derived from the four components. This digest is composed of 52 characters, totaling 36 bytes (288 bits), and it is the result of a base32 function applied to the four components. (In this work, we will use Meta-Code and Content-Code, only.)

Each component consists of 72 bits, 8 bits for the header, and 64 bits for the body, and has as its return value a string encoded in base58-iscc [20]. The header is intended to recognize the type of code component and, in the case of Content Similarity Code, to indicate the file type from four main choices: text, audio, video, and image.

An example of a research work that uses the ISCC standard to build a similarity-based lookup scheme for multimedia content in decentralized systems is presented in [21]. That paper compares an ISCC based indexing and a classic hashing scheme commonly used in DHTs. Results show that the use of ISCC enables more efficient content placement and retrieval, if compared to standard hash functions employed in other DHTs. This is basically what we are going to do, but coupling the indexing approach with the use of a Hypercube DHT for the retrieval of contents in decentralized storage.

III. Multimedia Multiple Keywords Search

The deployment of DFS, containing multimedia contents to be indexed, leads to the need for a similarity content-based indexing service that is more sophisticated than a simple keyword-value based index [8]. We thus envision a system that allows for multimedia content retrieval, not only thanks to one (or a set of) keywords but based on metadata and data contents [22]. With this goal in mind, we evaluate some alternative decentralized indexing approaches based on the hashing of the content itself, or on the use of the ISCC standard and the hashing of its metadata [19].

A. System Model

The system architecture has two layers: the hypercube DHT network works on top of the DFS network, i.e., IPFS (see Figure 1). Contents are stored in the underlying layer, while they are indexed in the upper layer. The system maintains three different kinds of information, i.e. the (multimedia) contents, their identifiers, and the metadata (e.g., keywords) associated with them. Thus, the Hypercube DHT maintains an association between keywords K and the related content identifiers xIDs,

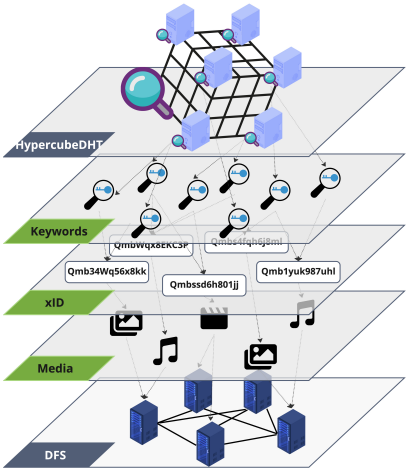


Fig. 1: Multi-level architecture of the system model.

i.e., a CID in IPFS. These identifiers are used to retrieve the actual data, i.e., multimedia content stored in the DFS.

B. Indexing scheme

The goal of the indexing scheme is to minimize the number of hops required to search contents through the Hypercube DHT, while automatizing the creation of keywords for multimedia content. In other words, we are considering the opportunity of having a decentralized lookup scheme that clusters similar contents into near peers in the logical overlay. We aim to identify the best scheme that maps similar contents to similar keyword set ids, so that nearby nodes in the Hypercube DHT maintain pointers to similar contents located in a decentralized storage or ledger. In this work, we compare two approaches, i.e., one based on the use of a traditional hash-based function and some variants of a scheme based on the use of the ISCC standard.

1) The SHA-256 method: The first approach is a traditional cryptographic hash function based scheme that, given content, computes the hash of the considered information and takes it to generate a keyword set id. Thus, given the content, this method produces a r-bit string that is, in turn, exploited by the approaches described in Section III-B3 to generate a final keyword set id. Hereinafter, we refer to this indexing scheme as *SHA-256*, which is the typical employed hashing function.

2) The ISCC methods: The alternative approach (referred to as *ISCC* method) exploits the ISCC standard. In particular, we consider different combinations of the ISCC Meta-Code and Content-Code associated with multimedia content:

- *ISCC-M* - only the Meta-Code associated with the metadata is employed to obtain the keyword set id used in the hypercube DHT.
- *ISCC-C* - only the Content-Code is used to obtain the keyword set id.
- *ISCC-CM* - both Meta and Content codes are computed and then their keyword set ids are combined through an OR operation (Meta-Code OR Content-Code).

Starting code	**Key set id generation scheme**	
	OR	concat
SHA-256	*SHA-OR*	*SHA-concat*
ISCC Meta	*ISCC-M-OR*	*ISCC-M-concat*
ISCC Content	*ISCC-C-OR*	*ISCC-C-concat*
ISCC Content OR Meta	*ISCC-CM-OR*	*ISCC-CM-concat*

TABLE II: Indexing schemes considered in the evaluation.

3) Keyword Set Id generation method: Given a (hexadecimal base representation of a) r-bit string s, generated using one among the *SHA-256* or *ISCC* methods described above, we further manipulate s to generate the final keyword set id. Two (slightly) different strategies are adopted for generating the id.

- **OR-based indexing**. s is subdivided into a sequence of chunks of size g. For each chunk, c_i, we compute its modulo c_i mod r, which identifies the position in the r-bit string that is set equal to 1. By repeating this procedure for all chunks, we obtain a final bit string by OR-ing all the modulo operations for all the chunks.
- **Concatenation-based indexing** Given s expressed in hexadecimal format, every single character c_i is considered (i.e., with respect to the approach above, we set $g = 1$). The final keyword set id is built through the concatenation of the result of c_i mod r, for all the c_i.

Table II shows the names we assigned to all the variants of the employed methods, based on the use of *SHA-256* or *ISCC*, and on the specific keyword set id generation method.

IV. Experimental Evaluation

This section discusses how we evaluated the different indexing schemes and the obtained results.

A. Metrics of Interest

Our goal was to assess which set of parameters and keyword set id generation technique are able to maximize the distance between classes of contents, while minimizing the internal distance between contents belonging to the same class. (As reported before, since we are dealing with bit-string ids, the distance we consider in this case is a classic Hamming distance.) Thus, for each class c, we compute a type of clustering index (*CI*), measured as the average distance between the c-centroid and the centroids of all other classes, over the average distance of items in c with respect to the c-centroid. Finally, we compute the average of these clustering values, to obtain a single outcome for the considered setup scheme. The metric *CI* is a variation of the Dunn index, which is the ratio between the minimum inter-cluster distance and the maximum intra-cluster distance. Clearly enough, the higher the value of *CI* the better it is.

B. Experimental Results

The test dataset was composed of 30 classes, each containing 15 photos of the same subject, like a monument or a painting. The system was analyzed through simulation [23].

Figure 2 shows the average results when using the *OR-based* indexing method. It is possible to appreciate how the

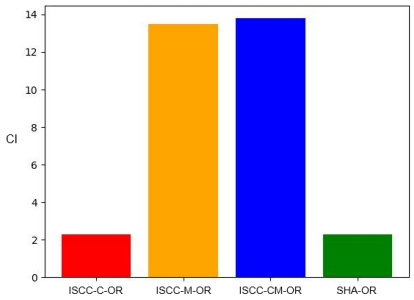


Fig. 2: Clustering Index (*CI*) for *OR-based* methods. It measures the method ability to cluster similar contents on the same nodes of the Hypercube DHT. Thus, the higher the better.

ISCC-M-OR and *ISCC-CM-OR* indexing schemes, based on the use of ISCC meta-codes, perform better than others. In particular, *ISCC-CM-OR* slightly outperforms *ISCC-M-OR*. This result is not surprising, since *ISCC-CM-OR* uses more information than *ISCC-M-OR* to index contents. Indeed, adding more information should improve the ability to differentiate between different classes of multimedia content. However, this improvement seems only marginal in these experiments.

Results for the *concat-based* methods are shown in Figure 3. In particular, Figure 3a shows the results obtained from different values of the r parameter, with g kept fixed and equal to 2, while Figure 3b provides results when $g = 4$. In both plots, it is possible to notice that the use of Meta-Code, i.e. *ISCC-M-concat*, is the best choice for allocating images on the DHT. In fact, this approach guarantees better performance in terms of maximizing the distance between different classes and, at the same time, minimizing the distance between contents of the same class. In this experiment, *ISCC-CM-concat* that employs both Content-Code and Meta-Code has a similar, but slightly worse, behaviour with respect to the *ISCC-M-concat*. This result suggests that the most important information that allows separating classes is the Meta-Code. From the results shown in the figure, it seems that the configuration with the parameter $g = 2$ provides slightly better performance with respect to $g = 4$, but the improvement is actually limited. An interesting thing to notice is that the best result obtained with the *OR-based* method, i.e. *ISCC-CM-OR*, is almost twice as much as the highest value obtained with the *concat-based* method, i.e. *ISCC-CM-concat*. This allows us to conclude that *ISCC-CM-OR* performs better in terms of the distribution of multimedia content on the DHT.

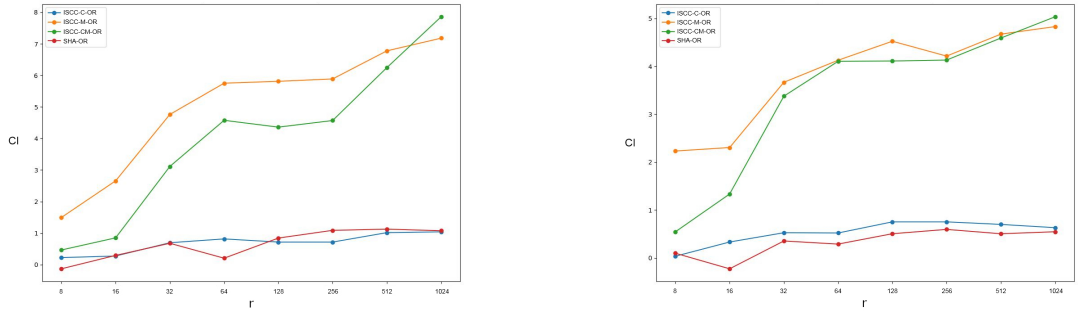
Figure 4 shows how the images were allocated over the hypercube nodes, in a configuration with a number of bits $r = 8$, respectively for *ISCC-CM-OR* (Figure 4a) and the typical *SHA-256* (Figure 4b). What each chart shows is a part of the Hypercube DHT. Each node of the graph corresponds to a node in the DHT. Each node is coloured based on the images it maintains. (To be more precise, it maintains pointers to where images are located.) Thus, for instance, a node coloured in orange means that it maintains images related to the "Altare della Patria" (a popular monument in Rome, Italy). Different colours pertain to different classes of images, with the exception of the grey colour, which represents the fact that the node contains (pointers to) images of different classes. Moreover, each node has a weight (label) associated with it, which represents the number of contents it stores. Simply put, what we would like to see here is a graph of nodes, with a high weight, which are highly connected when they have the same colour and with no nodes coloured in grey. It is possible to notice that *ISCC* creates more effective clustering for images of the same class. Nodes containing images of the same class stay closer, and those containing images of different classes are far apart. Instead, *SHA-256* causes an allocation of multimedia contents that is spread over the hypercube, causing a higher number of hops in content lookups over the DHT, and consequently performing worse in terms of returned multimedia contents and lookup delay. It is worth mentioning that both schemes were applied to DHTs of the same size. The fact that *ISCC* subgraph has fewer nodes shown in the figure, w.r.t. *SHA-256*, means that contents were actually allocated in fewer nodes in the DHT, thus leaving other nodes not utilized. In our view, this is a benefit, since, as mentioned, the allocation is more clustered. In a situation with a higher amount of multimedia content, our results suggest that a *ISCC Meta-Code* method guarantees that different classes of contents would be managed by nearby DHT nodes, with different classes going on different portions of the DHT. Conversely, *SHA-256* allocation does the job it is supposed to do, i.e. it distributes contents (even those of the same class) throughout the DHT. Few images are allocated in each DHT node. Indeed, several of these nodes have only one image associated, as reported in the node weights. This leads to a significant number of hops to lookup for a content type and fewer returned results correlated with the query.

V. Conclusions

We proposed a keyword-based decentralized lookup scheme, based on the ISCC standard to retrieve contents stored in a decentralized system such as DFS or DLTs. Our experiments show that the use of the ISCC enables a more efficient content placement and retrieval, compared to the classic hashing schemes. Overall, our proposal can pave the way for novel decentralized solutions that overcome the limitations of centralized servers for content lookup, making decentralized systems more scalable, efficient, and resilient to node failures when subject, for instance, to sudden and unexpected emergency situations or censorship. Such a system represents an instrumental tool that effectively addresses the challenges of data tracing and efficient retrieval across various application domains.

References

[1] L. Chen, W. Yu, J. Li, S. Yang, and H. V. Poor, "A distributed deep learning approach for intelligent multimedia content analysis in the internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5234–5245, 2021.



(a) g=2 (b) g=4

Fig. 3: Clustering Index (*CI*) for *concat-based* methods, with different g and r settings. The metrics refer to the ability of the method to cluster similar multimedia contents on the same nodes of the Hypercube DHT. Thus, the higher the better.

(a) ISCC allocation. (b) SHA allocation.

Fig. 4: Subgraph of DHT nodes containing references to images of different classes. Links represent direct connections among nodes of the DHT. The more nodes with same the color are linked together, the better it is.

[2] M. Zichichi, S. Ferretti, and G. D'Angelo, "On the efficiency of decentralized file storage for personal information management systems," in *Proc. of the 25th IEEE Symposium on Computers and Communications 2020 (ISCC2020)*. IEEE, 2020, pp. 1–6.

[3] J. Zhou, F. Tang, H. Zhu, N. Nan, and Z. Zhou, "Distributed data vending on blockchain," 2018, p. 1100 – 1107.

[4] M. Zichichi, S. Ferretti, and G. D'Angelo, "A distributed ledger based infrastructure for smart transportation system and social good," in *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC)*, 2020, pp. 1–6.

[5] S. Peng, W. Bao, H. Liu, X. Xiao, J. Shang, L. Han, S. Wang, X. Xie, and Y. Xu, "A peer-to-peer file storage and sharing system based on consortium blockchain," *Future Generation Computer Systems*, vol. 141, p. 197 – 204, 2023.

[6] G. D'Angelo and S. Ferretti, "Highly intensive data dissemination in complex networks," *Journal of Parallel and Distributed Computing*, vol. 99, pp. 28–50, 2017.

[7] M. A. Belarbi, S. A. Mahmoudi, S. Mahmoudi, and G. Belalem, "A new parallel and distributed approach for large scale images retrieval," *Lecture Notes in Networks and Systems*, vol. 49, p. 185 – 201, 2019.

[8] M. Zichichi, L. Serena, S. Ferretti, and G. D'Angelo, "Complex queries over decentralised systems for geodata retrieval," *IET Networks*, 2022.

[9] S. Wasiq, S. H. A. Bukhari, and M. A. Halepoto, "Distributed image retrieval in a decentralized network using content-addressable networking," *Multimedia Systems*, vol. 26, no. 4, pp. 445–461, 2020.

[10] I.-H. Gu and T. Tjahjadi, "Multiresolution feature detection using a family of isotropic bandpass filters," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 32, no. 4, pp. 443–454, 2002.

[11] S. S. Roy and S. K. Das, "Content lookup in peer-to-peer networks: A survey," *Journal of Network and Computer Applications*, vol. 105, pp. 35–56, 2018.

[12] T. Bocek, E. Hunt, D. Hausheer, and B. Stiller, "Fast similarity search in peer-to-peer networks," in *NOMS 2008 - 2008 IEEE Network Operations and Management Symposium*, 2008, pp. 240–247.

[13] A. Crespo and H. Garcia-Molina, "Semantic overlay networks for p2p systems," in *Agents and Peer-to-Peer Computing*, G. Moro, S. Bergamaschi, and K. Aberer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 1–13.

[14] S. Ferretti, "Gossiping for resource discovering: An analysis based on complex network theory," *Future Generation Computer Systems*, vol. 29, no. 6, p. 1631 – 1644, 2013.

[15] ——, "Shaping opportunistic networks," *Computer Communications*, vol. 36, no. 5, pp. 481–503, 2013.

[16] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. on Networking*, vol. 11(1), pp. 17–32, 2001.

[17] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, 2001, pp. 161–172.

[18] R. M. Karp and S. Shenker, "A randomized algorithm for finding frequent elements in streams and bags," in *Proc. of the ACM symposium on Theory of computing*. ACM, 2000, pp. 163–174.

[19] "Iscc," https://iscc.codes.

[20] "Base-iscc," https://iscc.codes/specification/#base58-iscc.

[21] M. Couceiro, L. Nunes, and J. Silva, "A similarity-based lookup scheme for multimedia content in decentralized systems," in *2020 International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2020, pp. 468–474.

[22] Z. Wu, C. R. Yang, S. Vargas, and A. Balasubramanian, "Is ipfs ready for decentralized video streaming?" 2023, p. 3002 – 3010.

[23] G. D'Angelo and S. Ferretti, "Adaptive parallel and distributed simulation of complex networks," *Journal of Parallel and Distributed Computing*, vol. 163, p. 30 – 44, 2022.