Content Based Video Identification in Peer-to-Peer Networks: Requirements and a Novel Solution

Alper Koz and R.(Inald) L. Lagendijk, *Fellow IEEE*

Abstract— **Content identification in peer-to-peer (P2P) networks has until now been achieved by using metadata or cryptographic hashes. However, with increasing number of duplicates in different names and formats especially in (unmanaged) P2P networks, these tools have become insufficient for proper content finding and access right management. A complementary possible approach is to identify the content in P2P networks by using perceptual hashes (or fingerprints) extracted from the perceptual features of the content robust to typical processing. In this paper, we first discuss the essential differences in fingerprint size, fingerprint extraction complexity, and fingerprint search methodology for a distributed video identification system in P2P networks compared with traditional central database implementations. Then, we propose a novel method optimized for P2P networks that uses only differences of video frame means. The proposed method reduces the fingerprint sizes into kilobytes, extraction time to seconds, and search duration into milliseconds, and achieves more than 90 % detection rates with 1-4 minutes granularities. Furthermore, the uniform distribution of the extracted fingerprints enables the usage of existing DHT-based keyword search mechanisms for fingerprint queries.**

Index Terms—**Peer to Peer, Perceptual Hash, Fingerprint, Content Search, Access Right Management, Video Identification, DHT, Gaussian Scales**

I. Introduction

After the introduction of the Bit-torrent protocol [1] for transferring large-size files in 2001, P2P networks have become one of the most popular medium for downloading and sharing video files over the internet [2]. By the year 2009, P2P file sharing has been estimated to form about 40-70 % of all internet traffic depending on the geographical location [3]. The reasons of such popularity of P2P networks are argued in [4] as the tendency to free and easy access to digitally perfectly copied content and the diversity of the choice for music and video in comparison to regular retail shops. However, such popularity has also brought problems of proper content identification and access right managements in these mediums that have also turned into a disarray of duplicates with different names and formats [4]-[6].

Although metadata and cryptographic hash based identifications have been used up to now in P2P networks [1], [7],[8] they have the distinctive disadvantage of user and format dependency. In an environment of thousands of peers that can tag, edit and format a file in different manners, solutions are necessary that are dependent on the content itself, not on subjective user inputs.

Solutions to this problem can be achieved by using perceptual hashes, also called fingerprints, based on the robust perceptual features of the content [9]-[19]. Previous applications of fingerprints include identification of radio and TV broadcasts, digital music library organization, and copy detections on web [9]-[19]. In these systems, the fingerprint extracted from unlabeled content is searched in a centralized database of precomputed fingerprints and the metadata associated to the matched fingerprint is returned from the database.

A naive approach for a fingerprint based video identification system in a P2P network would be to use also a central server for fingerprint queries. However, the problems regarding the maintenance of the server, bottleneck in the server traffic, and the possibility of server failure have been already experienced in similar central indexing systems for keyword search in P2P [20]-[22].

Our proposal is a fingerprinting approach that is consistent with the P2P philosophy: a system where the fingerprints are extracted by the peers, stored at the peers, and searched by the peers in a collaborative manner, similar to the state-of the art indexing systems based on distributed hash tables (DHT) in P2P networks [20]-[22]. Such a system involves the advantages of P2P networks such as decentralization of the network traffic, availability of the resources, and collaboration of the peers [20]-[22]. However, such a system also imposes more challenging restrictions on the fingerprint algorithms compared to the centralized database approaches.

First of all, the peers forming a P2P network are just ordinary internet users with simple personal computers. Therefore, the complexity of the fingerprint extraction should be very low. The methods requiring full decoding and high level analysis of video [10]-[14] can obviously not be used. Second, fingerprint storage at the peers should be in tolerable sizes. The fingerprint size should be comparable with the size

Alper Koz was with the Multimedia Signal Processing Group, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, The Netherlands. (Corresponding author tel: 00 33 (0) 665541253; fax: 00 33 (0) 169851765; e-mail: alperkoz@yahoo.com).

R.(Inald) L. Lagendijk is with the Multimedia Signal Processing Group, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 Delft, The Netherlands (e-mail: R.L.Lagendijk@tudelft.nl)

This research is funded by…

of similar data in current P2P clients, such as crypto-hashes [7]-[8], in order to avoid a significant increase in cache sizes. Therefore, the methods with small granularities in *seconds* [15]-[19] are not suitable in P2P due to their high fingerprint sizes. Third, fingerprint search should be compliant with the network traffic. Although searching the nearest neighbours (erroneous versions) of a given N bit fingerprint query is possible in a central database with new queries [15], such an opportunity is not realizable in a P2P network due to the low bandwidth of internet users. This requires robust fingerprinting algorithms that survive typical processing in P2P such as re-encoding and format changes, without distortions on the fingerprints. Finally, the indexing load for fingerprints should be uniformly distributed among the peers for a fair and decentralized design as in DHTs [20]-[22]. In its simplest form, this imposes a uniform distribution on fingerprints in their represented space to uniformly map the video clips to the peers.

The previous work on content based identification in P2P networks consists of only a few review studies. Kundur et al. [5] gave an overview of the content based retrieval methods in P2P and briefly discussed the security and privacy aspects. Kalker et al. [4] described a distributed P2P protocol in system level that allows access to music of guaranteed quality. In [23] and [24], some distributed fingerprinting approaches are proposed for audio. To the best knowledge of the authors, none of the former studies explicitly figured out the essential requirements for a distributed fingerprint based video copy detection system in P2P networks and proposed a solution.

In this paper we first discuss these requirements for a distributed fingerprint system and then propose a novel video fingerprinting method that is simple, robust and well suited to P2P networks.

The next section gives an overview of the identification methods and search mechanisms in P2P networks. Our focus will be more on one of the well-known DHT structures, *Content-Addressable Networks* (CAN) [20], which forms a base for our proposed system. Section 3 presents the proposed distributed system and its requirements, and discusses the suitability of existing video fingerprinting methods to these requirements in P2P. Section 4 describes the proposed video fingerprinting method based on the mean intensity of video frames. Section 5 gives a novel solution to the synchronization problem between the query video and the shared video at the peers. In Section 6, an efficient method is presented to decrease the fingerprint extraction complexity at the peers by only decoding a subset of video frames. Section 7 explains and justifies the uniform distribution of fingerprints for a DHT based indexing. In Section 8, the selections of design parameters are discussed for the adaptation of the method to P2P environment. Finally, conclusions are given in Section 9.

II. Identification Methods and Search Mechanisms in P2P Networks

A file in current P2P networks is identified by its name or by a cryptographic hash of the entire file.

A. Identification with File Names

A file in the network is represented by its name. The equivalence or difference of the files is concluded by comparing their names. For a given keyword as a query, the search mechanism finds and returns the files whose names match the keyword, and the peers possessing those files. Depending on the structure of underlying overlay, search with file names in P2P networks is achieved by using a centralized index [25]-[26], by flooding the network [27]-[35], or by employing distributed hash tables [20]-[22].

Search by using a Centralized Index

In this pioneer solution, a centralized index in the network holds the information of the peers where each file is located. A query by a user is searched in the centralized index and the peers possessing the desired file are returned. The well-known example is Napster [25]-[26]. Simplicity and easiness to employ sophisticated search tools are basic advantages of centralized indexes. However, the problems about scalability, maintenance of the centralized server, and the high traffic and storage load at the server make this approach inappropriate for large scale systems [20], [33].

Search by flooding the Network

Flooding the request for a desired file in the network forms the second generation of searching mechanisms. *Gnutella* [27] was the first representative of this category. A user looking for a file in *Gnutella* network sent a request to its neighbors and the neighbors recursively multicast the request for a predefined number of hops, which is called TTL value. The default TTL value of 7 covers about ten thousands nodes [5]. When a request is successful or failed, file locations or failure report is routed back to the user.

The later approaches have made the flooding mechanism more structured by exploiting the heterogeneities in peer capacities [28]-[31] and the similarities in peer interests [32]-[35]. *Kazaa* [28]-[29] and its variants [30]-[31] have combined the peers with higher bandwidth and computational capacities, namely super-peers, in an upper level network. Each super-peer holds the index of a number of its responsible peers. A query from a peer is first sent to the super-peer and the super-peer floods the query in his network if the query is not available after a local search.

The proposed system in [32] discovers the peers with similar interests by tracking the query history to further improve the flooding. The peers resolving the highest numbers of queries are hold locally at each peer as *neighbors* and are given priority in the future searches. This passive way of exploring the neighbors have been returned into an active search in [33]. The peers register their major interests to the index and also query the index to select neighbors. Finally, *Tribler* [34]-[35] has developed a peer similarity evaluator to compare the preference lists and to determine the interest similarity of two peers. The users first indicate their preferences for certain files and then this information is exchanged among the peers via an epidemic protocol to find the interest neighbors.

Compared to the centralized solutions, flooding based approaches avoids the network congestion at the server by distributing the search traffic to all peers, and also provides

robustness to failures and node transience. It is also relatively easy to implement multiple-keyword search or partial match [20], [33]. On the down side, finding a file that is actually in the network is not always guaranteed as the flooding should end at some point. Second, scalability is poor as the search performance in a fixed TTL duration is directly proportional with the number of peers in the network [20], [33]. Finally, flooding causes high volumes of traffic overhead as several peers are unnecessarily visited until the desired file is found.

Search by Distributed Hash Tables

Distributed hash tables (DHTs) [20]-[22] are proposed as the latest approach to overcome the mentioned disadvantages of previous solutions in scalability, guaranteed search, and traffic and storage balance. A hash table for an indexing system in a P2P network maps the file names *(keys)* to the IP addresses of file owners *(values)*. DHTs are distributed implementation of hash tables to uniformly spread the storage and mapping load of a hash table among the peers.

Such a structure is achieved by constructing a virtual *key* space that is partitioned by all the peers in the network. In the typical representatives of DHT systems, namely CAN [20] and Chord [21], this key space is chosen as a d-dimensional torus in Cartesian space and a uni-dimensional ring, respectively. Each peer stores a zone of the *key* space and holds the information of small number of neighbor zones. Requests (lookup, delete etc.) for a key are routed by intermediate peers towards the peer whose zone contains that key [20]-[21].

Figure 1 gives a simple illustration for the partition and routing in CAN structure [20], which also form a base for our proposed distributed fingerprint system (Section III). The 2-D Cartesian space is entirely partitioned among all the peers. In order to store a pair (K_1,V_1), key K_1 is deterministically mapped onto a point P in the coordinate space by using a *uniform hash function*. The key-value pair is then stored at the peer that owns the zone where P lies. During a query for key K_1, any peer apply the same hash function to map K_1 onto point P and then retrieve the corresponding value from the peer whose zone covers P. If the point P is not in the zones of the requesting peer, then the request is routed through the CAN infrastructure until it reaches the peer that holds P. For such a routing, each CAN peer maintains a coordinate routing table that holds the IP address and virtual coordinate zone of its neighbors. Using its coordinate set, a node routes a message by forwarding to the neighbor with coordinates closest to the destination coordinates [20] (Fig. 1).

Such a structuring and neighborhood description in DHT systems enables a proper routing path to the targeted peer without visiting the unnecessary peers during a query and hence, avoids the traffic overheads in flooding. As all the keys in the system are uniformly and deterministically mapped to a point in the virtual key space, such a structuring also guarantees to find the existing content in the network and balances the traffic and storage load between the peers. Furthermore, the linear relation of the search performance with the total number of peers in *flooding* turns into a logarithmic relation with such structures [20]. However, as a critical limitation, DHT systems require the exact knowledge of the key during a query to properly find the virtual zone of the peer storing that item.

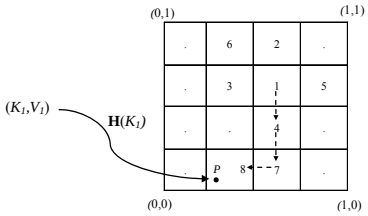


Fig. 1: Partition of 2D coordinate space by the peers in CAN, mapping of a (key, value) pair and routing of a query (K_1) (from peer 1 to peer 8).

Table 1: Total crypto-hash sizes for a 350 MB video. (A file is divided into chunks. For each chunk, a crypto hash is generated.)

P2P Network	Alg.	Output Size	Chunk Size	Total Hash Size (~ KB)
Emule [8] (main block)	MD4	128 bits	9.28 MB	0.59
Emule [8] (subblocks)	MD4	128 bits	180 KB	31.11
Bittorrent [1] (new)	SHA-1	160 bits	256 KB	27.34
Bittorrent [1] (old)	SHA-1	160 bits	1 MB	6.84
FastTrack [7]	MD4	128 bits	64 KB	87.49
Gnutella [7]	SHA-1	160 bits	500 KB (min)	14.00 (max)

B. *Identification with Cryptographic Hashes*

The second identification method in P2P networks is to represent a file with the cryptographic hash of the entire content. As the cryptographic hashes are very hard to be spoofed, such a representation enables a robust identification after the changes in the name and location of a file. When a compatible user gets the hash value of a file, he can find the locations of the content in the network by performing a search with the provided hash [36]-[37].

Magnet links [36]-[37] is the standard description for crypto hash based identification in P2P networks. Although more famous *torrent links* also include crypto hashes [1], the main usage is verification after downloads, but not identification for file search. *Magnet links* are used for both purposes [36]-[37]. They also provide easier dissemination with their smaller sizes, plain-text format and platform independent characteristics.

The description includes a series of parameters specifying the name, size, keyword topics and more importantly, file hash and hash set for the parts of the file [8], [36]. Hash set is formed by dividing the entire file into chunks and calculating the hash of each chunk. Then, the resulting hash set is used to calculate the *file hash*. While the *file hash* is mainly used for identification, hash set is employed to check the integrity of each file chunk after the download.

Most of the current P2P networks including *eMule*, *FastTrack*, *Overnet* and *Gnutella* employ magnet links [7]. The minor difference is mostly the utilized algorithm for hashing and the size of the chunks [7]. Table 1 shows the chunk size, hashing algorithm and the total resulting hash size

for a file of 350 MB (as an approximate size for a one hour episode) in some of the current P2P clients, which explicitly mention the hash sizes in their descriptions. The total hash sizes are in KB levels. The table is particularly important for our proposed fingerprint system to decide on the compatible sizes of the fingerprints that can be stored and employed in existing P2P networks.

Although crypto hashes offer a robust identification after the name and location changes in P2P networks, their distinctive disadvantage is their failure even in one bit of changes in the content. Therefore, they fail to identify the video in different formats in P2P networks.

III. Proposed Distributed Fingerprint System and Its Requirements

The previous discussion has revealed the following important points: (i) file names and cryptographic hashes are not sufficient to identify the video files in different names and formats in current P2P systems; (ii) DHT systems are the latest state-of-the-art in P2P search mechanisms with the advantages of distributed traffic and storage, scalability, and guaranteed search.

Based on these conclusions, our target in this paper is to design a DHT based distributed fingerprint system for video identification in P2P networks. Our strategy for this aim would be to construct an analogy to the well-known DHT system, CAN structure [20]. The basic principles in this analogy also apply for Chord [21].

The proposed distributed system mainly includes the following stages:

- A virtual fingerprint space representing the fingerprint vectors is constructed and this space is partitioned by all the peers in the network as in CAN.
- Fingerprints of a shared video at a peer are automatically extracted by the client program at that peer.
- Extracted fingerprints are mapped to the fingerprint space and indexing information about the fingerprint (the time interval in the whole video corresponding to the extracted fingerprint and the peer possessing that video) are stored at the peer containing the mapped position, similar to CAN.
- Fingerprint queries are collaboratively routed by the peers as in CAN.

The mentioned stages of the proposed distributed system impose more restrictive requirements on the fingerprint methodology including fingerprint extraction, searching, and matching compared to the previous central server applications of fingerprints. The other aspects of the design such as bootstrapping, joining to the network, and node departures are independent from the type of the keys and can be directly barrowed from CAN.

The following sections describe these requirements in parallel with a discussion on the suitability of the existing fingerprinting methods to such requirements. Although our main focus is a DHT based solution, the applicability of the *flooding* approach with the existing fingerprinting methods is also discussed for the sake of completeness.

Table 2: Main operations during fingerprint extraction

Method	Main Operations on Video Frames
Coskun et al [10]	Temporal and Spatial Normalization + 3D DCT
Joly et al [11]	Key frames + *Harris* point detector + Local Descriptors
Law-To et al. [12]	*Harris* point detector + Trajectory building by KLT
S. Chen et. al [13]	*SURF* point detector + Tracking the points + Categorization of trajectories
Chaisorn et. al [14]	Key Frames + Color and Face feature extractions
Oostveen et al [15]	Averaging frame blocks + Taking Spatio-temporal Difference
Mohan et al. [16]	Averaging frame blocks + Spatial Ordering
L. Chen et al. [17]	Averaging frame blocks + Temporal Ordering
Kim et al. [18]	Averaging frame blocks + Spatial Ordering + Taking Temporal Difference
Uchida et al. [19]	Computing the luminance centroid of frame blocks + Categorization of the centroid positions

Table 3: Fingerprint sizes (MB) for a one hour video (30 frames/sec)

Method	[11]	[12]	[15]	[16]	[17]	[18]	[19]
Size(~MB)	1.17	1.70	0.41	0.46	0.29	0.41	0.41

A. *Fingerprint Extraction Complexity*

Table 2 shows the main characteristics of the fingerprint extraction for different methods [10]-[19]. Although the list does not include all the methods in the literature, it is sufficiently comprehensive to draw conclusions. A fingerprint extraction algorithm should not turn the simple P2P client into a loaded background process for a personal computer of an ordinary internet user. Therefore, the methods using 3D transformations [10], *Harris* point detectors [11]-[12], *SURF* point detectors [13] and face features [14] are not acceptable for P2P clients due to their computational load. The methods using low level features such as average intensity [14]-[18] or centroid positions [19] of the frame blocks are computationally more suitable. However, full decoding of the video should still be performed in these methods, which is still a time consuming operation during fingerprint extraction [16]. Furthermore, the storage and search requirements should also be acceptable for P2P.

B. *Fingerprint Storage Size*

Table 3 shows approximate fingerprint sizes for the methods in Table 2 that can be explicitly computed. Fingerprint sizes are given for a video of one hour and 350 MB compressed size. Compared to the sizes of cryptographic hashes which are stored and employed in current P2P networks (Table 1) and 1Mbits/sec of typical ADSL upload rate for internet users, fingerprint sizes in megabytes are very large. Therefore, it is necessary to decrease the sizes to kilobytes for possible application of fingerprints compatible with the existing storage sizes in P2P networks.

C. Fingerprint Search and Match Complexity

Fingerprint search in existing fingerprint methods is mainly achieved by a sequential search [14], [16]-[18] or a preprocessed indexing structure [11]-[13],[15]. These two approaches can be applied in P2P in two ways similar to the flooding and DHT based searches.

The first approach is flooding the fingerprint queries in the network [27]-[35] and performing a sequential search in each of the visited peer. In the current fingerprint methods [16]-[18], search of a few seconds video (50-150 frames) in a one hour video with fingerprints takes about 2 to 10 seconds. For a peer sharing 40 hour video, this takes minutes, just for one peer, excluding the network latencies. Therefore, the number of the matching operations and the complexity of the dissimilarity computations should be decreased to adapt the sequential search algorithms [16]-[18] to P2P environment.

The second approach is our preferred DHT based approach [20]-[22]. A fingerprint query is routed to the peer storing the related part of the fingerprint index and the query is searched in that peer. Although the search and the matching of fingerprint are performed only in the targeted peer, this approach requires the exact knowledge of the queried key. In other words, the extracted fingerprints should not include any bit error to correctly route the fingerprint to the targeted peer. As the upload rate of typical ADSL users is mostly lower than 1Mbits/sec, the possibility of performing extra searches for the likely erroneous version of the queried fingerprint is limited to an extent in contrary to central server based fingerprint implementations as in [15].

D. Fingerprint Distribution in its Represented Space

In addition to the mentioned requirements, a DHT based fingerprint system requires also the structuring of the fingerprint space to define deterministic locations for the peers and to describe the neighborhood relations between the peers. If the fingerprint distribution is not uniform in its represented space, such a structuring should include a proper segmentation of the fingerprint space to achieve a uniform storage and routing load between the peers. Such a segmentation and neighborhood description between the segments would be a complicated task for the current fingerprinting methods with their high level feature vectors [11]-[14].

Our proposal is instead to guarantee the uniform distribution of fingerprints in their represented space in order not to require such segmentation. If a uniform distribution is guaranteed, then the *uniform hashing operation* mapping the keys to key space in CAN design simply replaces with the fingerprint extraction operation mapping the video files into the fingerprint space, and then the same structuring and neighborhood description in CAN can be used for the proposed fingerprint system.

IV. Proposed Video Fingerprinting Method

Based on the previous discussions, a fingerprint method enabling a DHT structure in P2P networks should have the following properties: (1) the fingerprint extraction should be simple, avoiding the decoding of every frame; (2) the hash sizes should be comparable to existing sizes in P2P; (3) fingerprints should be binary vectors with a uniform distribution; (4) simple distances should be used for (inexact) matching; (5) fingerprints should be very robust to typical distortions in P2P because the DHT structure requires a new query to find each possible distorted version of a fingerprint.

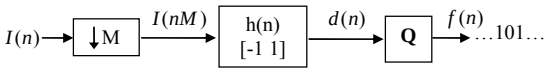


Fig. 2: General scheme of the proposed method

Our strategy in this paper is to decrease the storage size of fingerprints at the expense of increasing granularity (i.e. the smallest duration of a video segment required for identification) from second levels to minutes. This also decreases the number of matching operations. In order to achieve simplicity and robustness, we use one of the simplest and most robust features of video frames against compression, line shifts and scaling, namely the mean of the pixel intensities in a frame. In order to achieve the discriminability, we use the variations in one dimensional frame mean sequence for fingerprint extraction. Finally, uniform distribution is realized by decorrelating the consecutive bits of the fingerprints by extracting these bits from sufficiently large intervals on the mean sequence.

A. Proposed Method based on Frame Means

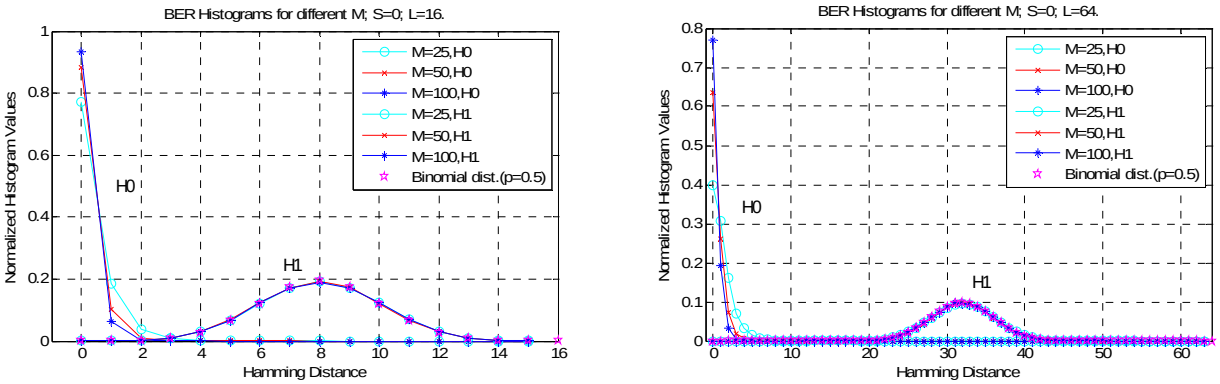
The proposed method involves the following steps (Figure 2):

- For a shared video at a peer, compute the mean of every frame, denoted as $I(n)$ in Figure 2.
- Subsample the frame mean sequence $I(n)$ by M and take the difference of the consecutive samples. The difference sequence is denoted by $d(n) = I(nM) - I((n-1)M)$.
- Quantize the difference sequence, $d(n)$, with respect to its sign. Resulting binary sequence is denoted by $f(n)$.
- Store $f(n)$ into binary vectors of size L: $v_n = [\, f(n) \ldots f(n-L) \,]$, $v_{n-1} = [\, f(n-1) \ldots f(n-L-1) \,]$,…
- Use the vectors v_n as the fingerprints to identify the corresponding MxL frames of the video. Note the overlapping of $L-1$ bits between the vectors v_n and v_{n-1}.

The granularity of such a method is *MxL* frames. Given a video of this duration as a query, the fingerprint vector of *L bits* is extracted and mapped to the fingerprint space to find the zone of the targeted peer storing the index information. The fingerprint query is routed to the targeted peer. Then the targeted peer returns the peers possessing the video files including the queried segment and the time interval of the segment in those video files.

B. Experimental Setup and Procedure

In the experiments, the robustness and discriminating capability of our proposed fingerprint method are evaluated under the common processing in P2P networks. For this purpose, we have extracted the mean sequences of 337 films, which are assumed as the shared video by the peers in the network (800 kbps-2 Mbps, 52.682.104 frames, ~585 hours, (300-400) x (700-800) frame sizes). Then a query video set is derived from this shared video: Bitrate is decreased to 128 kbps, frame sizes are reduced to half in both directions, and a



(a) L=16 (b) L=64

Fig. 3 Histogram of Hamming distance for different M values; S=0 (perfect alignment).

different GOP structure is applied. These degradations surpass the typical operations in P2P, such as changes in bitrates and display formats.

In order to test the detection performance and false positive rates, we extract the fingerprint vectors of the shared video, $\{\dots v_{n-1}, v_n, v_{n+1}\dots\}$ and the query set, $\{\dots v'_{n-1}, v'_n, v'_{n+1}\dots\}$. Then, we collect the statistics of the *distance* between v_n and v'_k under two *hypotheses*, H_0: video segments for v_n and v'_k are copies of each other ($n=k$), and H_1: they are not copies ($n\neq k$). The design parameters in the tests are M (subsampling period), L (length of the fingerprint vectors) and S (misalignment (shift) during sampling in the query video with respect to original). Hamming distance is used as the distance measure between fingerprints.

C. Experiments on Design Parameters

Figure 3 (a) and (b) shows the histograms of Hamming distance under H_0 and H_1 for different M, when $L=16$ (a) and $L=64$ (b) and $S=0$ (perfect alignment) in both figures. As the sampling period M increases, the difference between consecutive mean values becomes higher. Consequently, the number of bit errors between the original and distorted perceptual tags decreases. This can be verified with the increase in H_0 for 0 distance (no error) both in Figure 3 (a) and (b). On the down side, the increase in M also increases the granularity (i.e. MxL frames).

For $M=100$, histogram values on H_0 for '0' distance are about 94 % and 77 % for (a) and (b), respectively. The probability of having a fingerprint without any bit errors is increasing as L decreases. Note that this fact will be used in the distribution of fingerprints (in Section VII) as the percentage of the fingerprints without any bit errors is crucial for correct mapping of the fingerprints. On the other hand, the larger L values increase the gap between H_0 and H_1. This provides a better thresholding to achieve smaller false positive (FP) rates. For L=64, the theoretical FP rates computed from the model are in the range of ($\sim 2^{-64}$) as the distribution under H_1 is a binomial distribution with $p=0.5$.

As a last point, histogram value for H_0 is about 19 % for one bit hamming distance (error) when $M=100$ and $L=64$. If one bit neighbors of a given fingerprint are also queried in the network, more than 95 % of the clips can be detected. As a trade-off, this requires extra L (# of one bit neighbors) queries for a DHT based search. Searching also two or more bit neighbors requires much more queries over hundreds. The proposed method detects a high percentage of the video clips without needing such a high number of queries.

V. Synchronization of Query Video

In a practical case, the subsampling positions of frames for the shared video in the network are unknown; there are M alternative positions for a query video to begin the sampling. A naive approach to solve this synchronization problem would be to try all alternative positions and to find minimum of them, as in sequential search methods [16]-[18]. However, this is computationally infeasible and causes an overload of queries in the network.

A. Proposed Synchronization Scheme

Our proposed solution is to find a robust reference point on the mean sequence of the video segments of MxL frames to align the sampling pattern during fingerprint extraction. For this purpose, we take the difference of the consecutive frame means and denote the position of the maximum in the resulting sequence as the reference point. In order to verify the robustness of this approach, we find the shift in the position of the reference points for the video pairs, one of which is the derived version of other (as described in section III B for the query video set).

Table 4 shows the ratio of the number of video pairs with a particular shift to the total number of video pairs. The maxima are at the same position (0 shift) in more than about 98 % of the video pairs in different durations, which indicates the high robustness of the reference position to the typical processing in P2P.

With such a reference point, fingerprint extraction changes as follows (Fig. 4):

- For a given mean sequence of a video, divide it into segments of MxL frames with an overlapping of $Mx(L-1)$ frames.
- For each segment of MxL frames, find the reference point.

- Extract the fingerprint for each segment as in Section 3.1 such that the subsampling pattern passes from the reference point. To do that, take modulo M of reference point and begin the subsampling from that frame in the first M frames.

This modification has no effect on the distribution of H_1. Regarding the detection rate (H_0), a given query of MxL frames with an arbitrary position (Fig. 4) corresponds to either v_n or v_{n-1} if the maximum of the query is at the same point with one of the fingerprints. Considering the overlapping regions between the query fingerprint and two consecutive fingerprints, it can be shown that the probability of such an occurrence is higher than $1-(1/L^2)$ (Appendix).The detection rate is affected with such a factor as the worst case.

B. Experimental Results with Synchronization

In order to test the synchronization scheme, we collect the statistics under H_0 and H_1 by introducing random delays between 0 and M-1 to the query set. Figure 5 (a) shows the histograms with and without the synchronization scheme ($S=0$) for L=16 and M=100. The distribution under H_0 is a binomial distribution for both cases. Figure 5 (b) gives the detection rates computed from the histograms with a threshold of 1 bit from L=16 to 64. There is a decrease of about 2-4 % in the detection rates with the synchronization scheme. This decrease is mostly due to the distortions on the query set which slightly change the positions of reference point (Table 4). The decrease originating from the overlapping strategy of the scheme itself is in small levels as the probability of missing both consecutive fingerprints is very negligible for the utilized values of L (Appendix). The proposed synchronization scheme achieves to reduce the number of searches from M to 1 with a slight loss in the detection rate as a trade-off.

VI. Hierarchical Reference Point Finding

Durations of the main operations in the proposed method are presented in Table 5 for a 2.2 GHz computer (using MATLAB 7.1). These durations are much lower than existing methods [10]-[19], as can be expected due to the simpler features and distance metrics utilized in the proposed method. The main computational load for fingerprint extraction is decoding of every frame to find the reference point. Although 8.5 min. of decoding time can be accepted for a video shared for a number of days, 37 sec. of waiting time for a query is not tolerable for a user. Therefore, we proposed a method to avoid decoding of every frame.

A. Proposed Method for Hierarchical Reference Point Finding

In scale invariant interest point detection [38]-[39], an interest point on an image is found directly on its down-sampled version. Inspired by this result, we propose a hierarchical approach to find the reference point in 1-D frame mean sequence (Figure 6). Given a downsampled version of the mean sequence, $I(Kn)$, first a Gaussian filter of $g(Kn)$ is applied to the sequence for smoothing. Position of the reference point (n_o^*) is found in the resulting sequence by determining the maximum after taking the difference of

Table 4: Ratio of the number of video pairs with a particular shift

# of Pairs	482173	503741	509133	514525	519917
Dur(~min.)	8.5 min	4 min	3 min	2 min	1 min
# Frame	128x100	64x100	48x100	32x100	16 x100
0 shift	**98.64** %	**98.61** %	**98.48** %	**98.27** %	**97.88**%
1 shift	0 .25 %	0.23 %	0 .22 %	0.21 %	0.21 %
2 shift	0 %	0 %	0 %	0 %	0.02 %

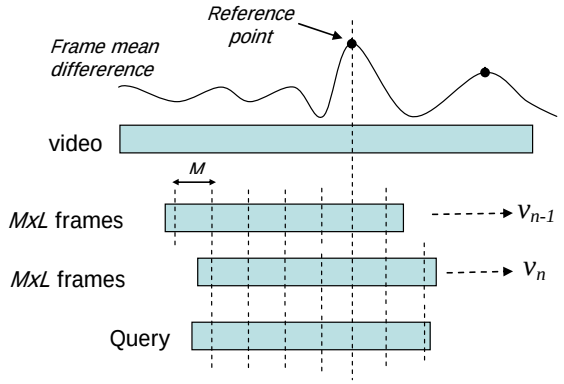


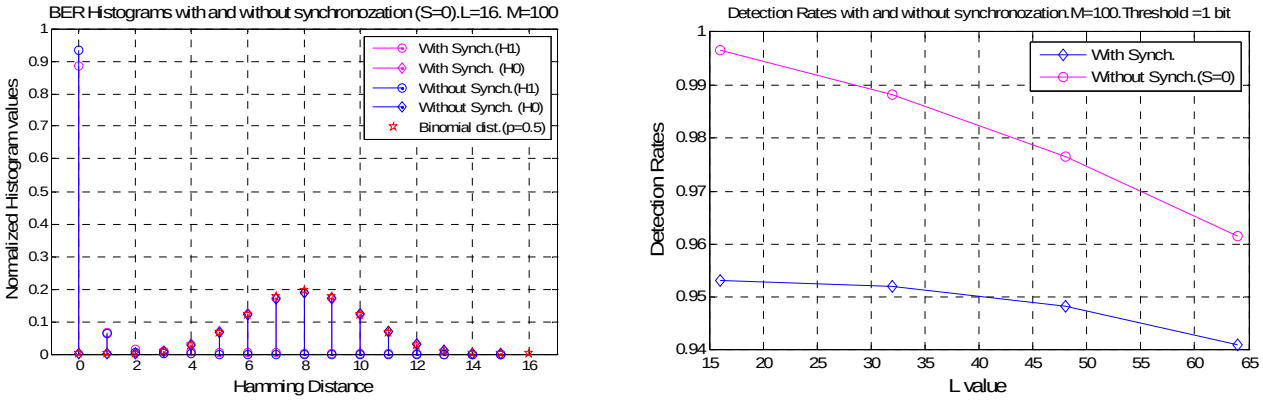
Fig. 4: Illustration of synchronization scheme. Subsampling patterns for v_n and v_{n-1} pass from the illustrated point for the given example.

consecutive samples. This gives a coarse estimate of the reference position. Then, $2F$ frames around this position, as F is the support of the filter $g(n)$, are decoded and passed from $g(n)$. Finally, the accurate reference point (n_o) is found in the selected signal.

For the robustness tests, we generate the video pairs one of which is the distorted version of other. We use the hierarchical approach in one of the pair and the previous approach in the other, and compare the shifts in the reference position. Table 6 shows the percentage of video pairs that gives the same position in the tests and the percentage of decoded frames. While the performance of reference point finding decreases to 90 % levels, the number of the decoded frames decreases down to 20 % levels with the proposed hierarchical method.

B. Adaptation of the Method to MPEG Coded Video

Although finding the reference point from a down-sampled version of the mean sequence gains a lot in terms of the number of decoded frames, a more efficient method for an MPEG coded video would be to select the down sampling positions as the position of *Intra* frames. However, the *group of pictures (GOP)* size of the query video can be different than the down sampling rate (K), which should be fixed for a generic implementation. Second, GOP size can show slight variations during the coding to achieve the targeted bitrate with a better encoding quality [40]. Considering these problems, we propose to use the linear interpolation of the means of the neighbor *Intra* frames as the value of the mean sequence at the subsampling position.



(a) (b)

Fig. 5: (a) Histogram of Hamming Distance with and without synchronization (*S*=0); *L*=16; *M*=100, (b) Detection rates for different L values with and without synch.(S=0); M=100; Detection Threshold: 1 bit.

Table 7 (a) shows the detection rates with and without such a modification in the hierarchical method. The video sequences are coded with 128 kbps while frame sizes are reduced to half in both directions. Typical GOP sizes (GS) and GOP structures different than the originals are selected during MPEG coding. The down sampling rate (K) is kept higher than the GOP size to avoid the use of the same Intra frame in the consecutive down sampling positions. The detection performances are more than 94 % levels. There is a small decrease for the higher GOP size as the average distance between the neighbor Intra frames and the down sampling position is increasing.

An interesting result in the table is the slightly better performance in the MPEG adapted approach compared the previous case. In other words, linear interpolation of mean values of Intra frames is better tracking the changes of the mean sequence in the coarse level compared to the mean values extracted directly from inter frames at subsampling positions. This can be linked with the better precision of mean values for Intra frames and also with the MPEG mechanism which introduces new *Intra* frames if the change in the scene cannot be coded with the motion estimation in *Inter* frames.

Table 7 (b) gives the approximate durations of three main decoding stages in the modified approach: decoding of the intra frames (D1), decoding of the 2*F* frames around the found reference point (D2) and decoding of the every *M*'th frame for fingerprint extraction (D3). As MPEG decodes the frames in a GOP successively, we decode all the frames of the GOP for the current implementation if the M'th frame corresponds to an inter frame during the calculation of D3. The proposed approach achieves to decrease the fingerprint extraction from a query of *MxL* frames (M=100, L=64) into less than 10 seconds. A slight trade-off between the performance and extraction duration with respect to GOP sizes can be observed from the tables. The given durations in the proposed method can be further decreased by directly computing the frame means from the DC coefficients of the coded stream and hence avoiding the full decoding of the frames.

Table 5: Duration of the main operations for a one hour video (V), and a query video of *MxL* frames (Q). (*M*=100, *L*=64).

Operation	Duration	
	V	Q
Decoding of the frames (by *ffmpeg* [41])	8.5 min	37 sec.
Computing frame means	1.63 sec	0.12 sec.
Fingerprint extraction	0.38 sec	0.001 sec
Search of a query of *MxL* frames in 40 hour video	0.28 sec	

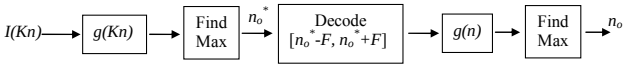


Fig. 6 Hierarchical decoding scheme to find the reference point

Table 6: Performance of reference point finding and the number of decoded frames (%, %) for the hierarchical scheme.

(%, %)	{Variance, Support} of Gaussian filter *g(Kn)*		
Downsample Rate: *K*	{1, 9}	{2, 17}	{3, 25}
5	**(86.56, 21.41)**	**(92.29, 22.66)**	**(94.42, 23.91)**
10	**(83.86, 12.81)**	**(89.22, 15.31)**	**(93.98, 17.81)**
20	**(80.96, 10.63)**	**(86.81, 15.63)**	**(92.82, 20.62)**

VII. ACHIEVING THE UNIFORM DISTRIBUTION

In the distribution of fingerprints, instead of using all the bits of the fingerprints (L bits) to map the fingerprints to the peers, our proposal is to use only a part of them for two reasons: (1) the number of the peers in existing P2P networks is in the ten thousand ranges [2], [29]. For a network of about 64000 peers, only 16 bits are sufficient to partition the fingerprint space among the peers; (2) if the number of bits to map a fingerprint to a zone of virtual coordinate space is decreased, the probability of finding the zone correctly without a single bit error in the utilized bit word for mapping increases (Figure 3

(a) and (b)). This decreases the need for extra searches to find the erroneous versions of a fingerprint in P2P.

Assuming a network of about (2^{16}) peers and fingerprint vectors of 64 bits, the utilized 16 bits for mapping are formed by regularly taking 1 bit from every 4 bits. Some other strategies based on majority voting or *xor* ciphering in every four bits can also be considered for the generation of 16 bits. In order to justify the uniform distribution, we find the number of occurrences for each bit word used for mapping, extracted from our 585 hours of video, for increasing values of *M*. The histograms in Figure 7 indicate that uniform distribution is approximately achieved for the values of M greater than 20. For smaller values, the steady increasing or decreasing behaviors in the frame mean sequence are generating repetitive bits that breaks the uniform characteristics. For a selection of M=100, min. and max. of the histogram is 0.0043 and 0.0036 respectively. None of the peers have an indexing load more than about 120 % of any others, which achieves a quite fair decentralized sharing.

With such a distribution of fingerprints, computation of the detection rates returns into a two-level process: (1) finding the correct zone (peer) in the virtual coordinate space storing the requested key (i.e. an error in the utilized 16 bits for mapping results in a wrong zone to send the query); (2) finding the correct fingerprint of *L*=64 bits that is stored at that zone. Multiple bit errors during the matching of *L* bit fingerprint are possible for this case as a sequential search is performed in the targeted peer. Detection rates in a network can then be computed as a product of these two detection rates. Other detections that can originate from false peer identification are ignored due to their very small probability.

In table 8, detection rates are given by using the two graphs in Figure 3 for *L*=16 (peer identification) and *L*=64 (fingerprint identification) for different M values. The proposed method achieves to detect more than 94 % of the fingerprints by performing totally 17 searches with one-bit neighbors of utilized 16-bit word for mapping. Compared to the detection rates for the case of using all bits for mapping (section IV-c, L=64, M=100), decreasing the number of bits has enabled the identification of shorter video durations (L=64, M=25) with a lower number of searches while achieving the similar detection performances.

VIII. Selecting Design Parameters For P2P

We have shown the trade-offs for different aspects of our proposed distributed fingerprint system in the previous sections by selecting some specific values for *M* and *L*. In this section, we discuss these selections for a P2P application considering the fingerprint size, and false positive and detection rates. Table 9 shows fingerprint sizes for different *M* and *L* values. With the given *M* and *L* values, the sizes are reduced to kilobytes similar to the crypto-hash sizes (Table 3). The proposed method reduces the load of fingerprint storage and dissemination to the levels of existing applications in P2P with the flexibility provided in its design parameters.

Table 9 also indicates the detection rates (D) and average number of false positives (# FP) for the threshold of zero bit and one bit error in the utilized bit word for mapping (T_1) for the given *M* and *L* values. The proposed synchronization and distribution schemes are included for this ultimate case during the detection. For all *L* values, 16 bits of the fingerprints are used for mapping. The threshold during the matching operation in the targeted peer (T_2) is selected as the value covering at least 95 % of all detections for the lowest *M* value,

Table 7: (a) Performance of hierarchical reference point finding with linear interpolation (I) and without linear interpolation (II); (b) durations of the main decoding operations with the modified approach for a MPEG coded query video of MxL frames (Bitrate: 1140 kbps, 25 frames/sec., CIF size); Variance and Support of Gaussian filter g(Kn): (3,25); Down sample Rate (*K*)=20; M=100; *L*=64.

(a)

GS (set)	Average *GS*	Variance *GS*	Perf. I (%)	Perf. II (%)
5	4.9883	0.0358	95.32	93.01
15	14.8942	1.0786	94.46	92.82

(b)

GS	D1 (sec.)	D2 (sec.)	D3 (sec.)	Total (sec.)
5	5.7	2.5	0.8	9.0
15	1.9	2.5	2.5	6.9

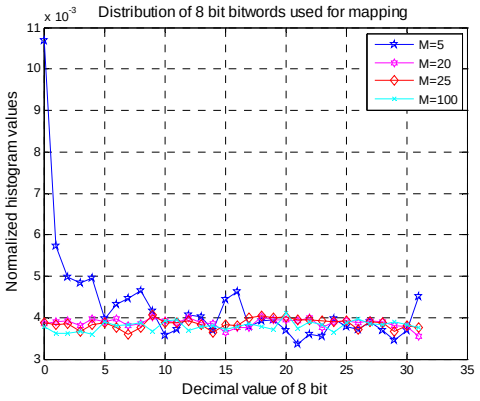


Fig. 7: Histogram of utilized bits for mapping. (The first 32 elements of the histograms for 8 bit bitwords are shown for visual clarity.)

Table 8: Detection rates (%) for different *M* values after the distribution of fingerprints. Threshold for *L*=16 (T_1) is 0 bit (query only the utilized 16-bit word for mapping) or 1 bit (query also one bit neighbors of the16-bit word). Threshold for *L*=64 (T_2) is 6 bits. Perfect synchronization is assumed.

T_1	# of searches	*M*		
		25	*50*	*100*
0 bit	*1*	75.88	88.38	93.51
1 bit	*17*	94.00	98.77	99.66

Table 9: Fingerprint sizes (FS) for a 60 minutes video, Detection rates (D) and Average number of False Positives (# FP) with the proposed distribution and synchronization schemes for different *M* and *L* values. *G*: Granularity (MxL frames); *#V*: Number of fingerprints in a peer sharing the fingerprint index of 40 hours of video; T_1: Threshold for the utilized 16 bits for mapping; *NS*: Number of Searches for the query; T_2: Threshold for the L bit fingerprint matching in the targeted peer.

L	*M*	*G* (min:sec)	*FS* (KByte)	T_2 *(bits)*	*D (%)* (T_1 *= 0 bit; NS*=1)	*D (%)* (T_1 = 1 bit; *NS*=17)	*# V (~)*	*# FP* (~) (T_1=1 bit; *NS*=17)
32	25	0:32	14.06	4	71.34	89.72	1.44e+005	3.1286
	50	1:04	7.03	4	82.22	92.67	7.2e+004	1.5643
	100	2:08	3.52	4	87.33	93.51	3.6e+004	0.7821
64	25	1:04	28.12	6	72.05	90.81	1.44e+005	1.1019e-006
	50	2:08	14.06	6	83.51	94.15	7.2e+004	5.5097e-007
	100	4:16	7.03	6	89.93	96.19	3.6e+004	2.7549e-007
128	25	2:08	56.25	10	72.05	91.85	1.44e+005	1.4814e-019
	50	4:16	28.12	10	85.66	96.25	7.2e+004	7.4072e-020
	100	8:32	14.06	10	91.39	97.61	3.6e+004	3.7036e-020

which is 25. This threshold is kept fixed for other *M* values given for each *L* value in the table. Utilizing the binomial model for H_1, we approximately compute the average number of false positives resulted from a single query in the network in terms of number of searches for the query and the number of fingerprint vectors (or matching operations) in the targeted peer sharing a fingerprint index of 40 hours of video in average [31].

For all *L* values, there is an increase in the detection rates for increasing values of M. The disadvantage is the increasing granularity with M. The detection rates for different M values are getting closer when one-bit neighbors are also searched.

For fixed M values, detection rates are highest for *L*=128. As there is a higher overlapping regions between the consecutive fingerprints for higher values of *L*, synchronization scheme is performing better for *L*=128. FP rates are also the lowest for *L*=128 due to the binomial distribution. However, the disadvantage is comparatively higher granularity, which is about 2 minutes for the lowest case.

Decreasing *L* to 32 decreases the granularity to about 32 seconds. However, this also causes a significant decrease in FP rates. In our opinion, *L*=64 and *M*=25 is the most convenient selection with its high detection and suitable FP rates, for a fingerprint system in P2P that identifies video segments of about 1 minutes.

IX. Conclusions

We point out the main requirements for a distributed fingerprint based video identification system in P2P networks. The core part of the paper is to realize that it is not possible to implement current fingerprint methods in existing P2P networks due to their high storage size and complexity. We propose to increase granularity from frames to minutes. This increase faces us with the synchronization problem as the query can come from random intervals. We propose a novel method for synchronization by finding robust reference points on 1-D frame mean sequence, which also avoids the decoding of every video frame. The present results indicate the well suitability of the method for P2P networks. Future work will investigate the statistical modeling of the method, security and privacy aspects of the proposed system and the implementation in real P2P networks.

APPENDIX

Without loss of generality consider the illustrations of the positions of a query fingerprint and two consecutive fingerprints in Figure A.1 with the following notations:

v_1 and v_2: Fingerprints corresponding to two consecutive blocks of *MxL* frames with an overlapping of *Mx(L-1)* frames.

v_q: Fingerprint corresponding to the query video of *MxL* frames between these two fingerprints.

P_{12}: Position of the maximum (reference point) for *Mx(L+1)* frames corresponding to v_1 and v_2 together.

P_{q1}: Position of the maximum for all frames corresponding to v_q and v_1 together.

P_{q2}: Position of the maximum for all frames corresponding to v_q and v_2 together.

P_1, P_2 and P_q: Positions of the maximum corresponding to v_1, v_2 and v_q, respectively.

Query fingerprint matches with one of two consecutive fingerprints if its maximum is at the same point with the maximum of one of two fingerprints. Assuming that the query video is generated from the original video without any

distortions, the detection occurs in the following cases (Figure A.1):

Case I: If $P_{12} \in (R_2 \cup R_3 \cup R_4) \Rightarrow P_q$ matches P_1 or P_2. Probability of such an event is:

$$\Pr(P_{12} \in (R_2 \cup R_3 \cup R_4)) = LM/(L+1)M. \quad (A.1)$$

Case II: If $P_{12} \in R_1 \Rightarrow P_q$ does not match P_1, but matches P_2 if $P_{q2} \in (R_3 \cup R_4)$. Probability of such an event is:

$$\Pr(P_{12} \in R_1).\Pr(P_{q2} \in (R_3 \cup R_4)) = \frac{a}{(L+1)M}\frac{(L-1)M+a}{LM+M-a}. \quad (A.2)$$

Case III: If $P_{12} \in R_5 \Rightarrow P_q$ does not match P_2, but matches P_1 if $P_{q1} \in (R_2 \cup R_3)$. Probability of such an event is:

$$\Pr(P_{12} \in R_5).\Pr(P_{q1} \in (R_3 \cup R_4)) = \frac{M-a}{(L+1)M}\frac{LM-a}{LM+a}. \quad (A.3)$$

Assuming that the query video can come from any random interval, the detection probability (P_D) can be expressed as follows:

$$P_D = \Pr(Case\ I) + \Pr(Case\ II) + \Pr(Case\ III) = \frac{LM}{(L+1)M} + \frac{1}{M}\sum_{a=0}^{M-1}\frac{a}{(L+1)M}\frac{(L-1)M+a}{LM+M-a} + \frac{1}{M}\sum_{a=0}^{M-1}\frac{M-a}{(L+1)M}\frac{LM-a}{LM+a}. \quad (A.4)$$

Figure A.2 shows the analytical and experimental computations for the missed detection probability (i.e. $P_M = 1-P_D$) for different L values. For the experimental computations, we repetitively generate a random vector of size $Mx(L+1)$, and a query vector of MxL is extracted from this vector with a random shift. Then, we compare the positions of maximums for each case and determine the number of detections. The analytical and experimental computations show a good match. The missed detection rate is much smaller than 1 % for $L \geq 16$ and always smaller than $1/L^2$.

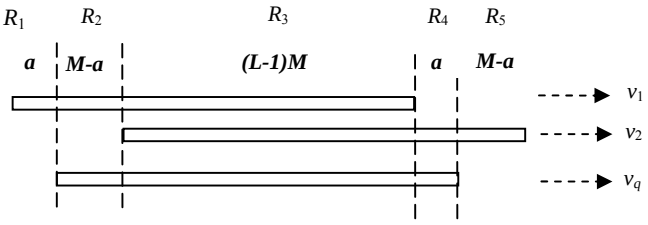


Fig. A. 1: Illustration of the positions of two consecutive fingerprints and a query fingerprint between them.

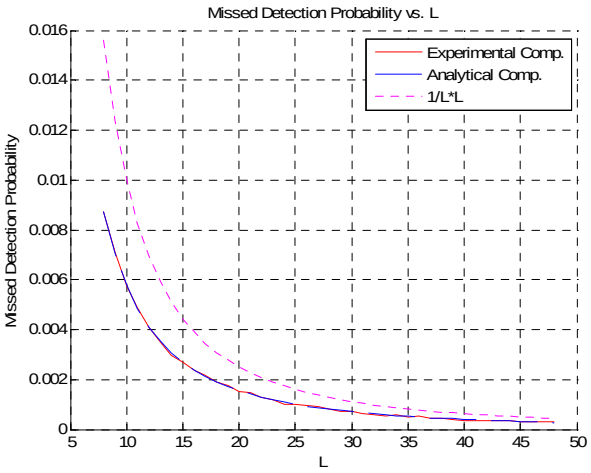


Fig. A. 2: Analytical and experimental computations for the missed detection probabilities.

REFERENCES

[1] B. Cohen, "The BitTorrent Protocol Specification", [Online]. Available: http://www.bittorrent.org/

[2] J.A. Pouwelse, P. Garbacki, D.H.J. Epema, H.J. Sips, "The Bittorrent P2P File-sharing System: Measurements and Analysis", IPTPS'05, Feb 2005.

[3] Hendrik Schulze, Klaus Mochalski, "Ipoque Internet Study Report 2008/2009", http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009

[4] T. Kalker, D.H.J. Epema, P.H. Hartel, R.L. Lagendijk, M. van Steen, "Music2Share—Copyright-Compliant Music Sharing in P2P Systems", PIEEE, 92(6):961-970 (2004).

[5] D. Kundur, Z. Liu, M. Merabti, and H. Yu, "Advances in Peer-to-Peer Content Search", in *Proc. IEEE Int. Conf. Multimedia and Expo*, 2007, pp. 404-407.

[6] J.A. Pouwelse, P. Garbacki, D.H.J. Epema, H.J. Sips, "Pirates and Samaritans: A decade of measurements on peer production and their implications for net neutrality and copyright", *Telecommunications Policy*, vol. 32, pp. 701-712, 2008.

[7] A. Jantunen, S. Peltotalo, J. Peltotalo, "Peer-to-Peer Analysis : State-of-the-art", Technical Report, February 2006, Tampere University of Technology, [Online]. Available: http://delco.cs.tut.fi/doc/other/p2p_analysis_v01.pdf

[8] Emule [Online]. Available: http://www.emule-project.net

[9] J. Law-To et al, "Video Copy Detection: A Comparative Study", in *Proc. 6th ACM Conference on Image and Video Retrieval - CIVR* , pp. 371-378, 2007.

[10] B. Coskun, B. Sankur, and N. Memon, "Spatio-temporal transform-based video hashing", IEEE TMM, 8(6):1190–1208, 2006.

[11] A. Joly, O. Buisson, and C. Frelicot, "Content-based copy detection using distortion-based probabilistic similarity search", IEEE TMM, 9(2): 293-306 (2007).

[12] J. Law-To, O. Buisson, V. Gouet-Brunet, N. Boujemaa, "Robust voting algorithm based on labels of behavior for video copy detection", ACM Multimedia 2006: 835-844.

[13] S. Chen, J. Wang, Y. Ouyang, B. Wang, Q. Tan and H. Lu, "Multi-Level Trajectory Modeling for Video Copy Detection", in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing,* 2010, pp. 2378-2381.

[14] L. Chaisorn, J. Sainui and C. Manders, "Fast and Efficient Video Signature Generation and Matching for Online Video Sharing", in *Proc. IEEE Int. Conf. Image Processing,* 2010, pp. 2129-2132.

[15] J. Oostveen, T. Kalker, J. Haitsma, "Feature Extraction and a Database Strategy for Video Fingerprinting", VISUAL 2002: 117-128.

[16] *R. Mohan, "Video Sequence Matching," IEEE ICASSP'98, Vol. 6, pp. 3679-3700, 1998.*

[17] L. Chen and F. W. M. Stentiford, "Video sequence matching based on temporal ordinal measurement," *Pattern Recognition Letters (2008)*.

[18] C. Kim and B. Vasudev, "Spatiotemporal sequence matching techniques for video copy detection", IEEE TCSVT, 1(15):127–132, Jan. 2005.

[19] Y. Uchida, M. Hashimoto, and R. Kawada, "Fast and Robust Content-based Copy Detection based on Quadrant of Luminance Centroid and Adaptive Feature Comparison", in *Proc. IEEE Int. Conf. Image Processing,* 2010, pp. 1021-1024.

[20] S. Ratnasamy, P. Francis, M. Handley, R. M. Karp, S. Shenker, "A scalable content-addressable network", SIGCOMM 2001: 161-172.

[21] I. Stoica et al., "Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications", IEEE/ACM Transactions on Networking, Vol. 11, No. 1, pp. 17-32, February 2003.

[22] A. Rowstron and P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large Scale Peer-to-Peer Systems", *in Proc. 18th IFIP/ACM Int. Conf. on Distributed Systems Platforms*, 2001, Germany.

[23] P. Shrestha and T. Kalker, "Audio Fingerprinting In Peer-to-peer Networks", ISMIR 2004.

[24] G. Tzanetakis, J. Gao, and P. Steenkiste, "A Scalable Peer-to-Peer System for Music Content and Information Retrieval", *in Proc. of 4th International Conference on Music Information Retrieval*, Baltimore, Maryland, 2003.

[25] Napster [Online]. Available : www.napster.com

[26] A. Oram et al., "Listening to Naspter", *Chapter 2* in *Peer-to-Peer: Harnessing the Power of Distruptive Technologies*, 2001, O'Reilly.

[27] Gnutella [Online]. Available: http://rfc-gnutella.sourceforge.net/

[28] Kazaa [Online]. Available : www.kazaa.com

[29] J. Liang, R. Kumar and K.W.Ross, "The Kazaa Overlay: A measurement Study", Computer Networks (Special Issue on Overlays), 2005

[30] Grokster [Online]. Available: http://www.groskter.com

[31] P. Garbacki, D.H.J. Epema, M. van Steen, "Optimizing Peer Relationships in a Super-Peer Network", in Proc. of 27th International Conference on Distributed Computing Systems (ICDCS), pp. 31, 2007.

[32] K. Sripanidkulchai, B. Maggs, and H. Zhang, "Efficient Content Location using Interest-Based Locality in Peer-to-Peer Systems", in Proc. IEEE INFOCOM, vol. 3, pp. 2166-2176, 2003.

[33] R. Zhang and Y. C. Hu, "Assisted Peer-to-Peer Search with Partial Indexing", IEEE Trans. on Parallel and Distributed Systems, vol. 18, pp. 1146-1158, August 2007.

[34] Tribler [Online]. Available: http://www.tribler.org/

[35] J. Pouwelse *et al.* "Tribler: A Social-Based Peer-to-Peer System", Concurrency and Computation: Practice and Experience, pp. 127-138, 2008.

[36] MAGNET-URI project website, [online]. Available: http://magnet-uri.sourceforge.net/

[37] Magnet URI scheme, [online]. Available: http://en.wikipedia.org/wiki/Magnet_URI_scheme

[38] K. Mikolajczyk, C. Schmid, "Indexing Based on Scale Invariant Interest Points", in *Proc. of IEEE Int. Conf. Computer Vision*, pp. 525-531, 2001.

[39] Tony Lindeberg, "Discrete Scale-Space Theory and the Scale-Space Primal Sketch", Ph. D. Thesis, Computational Vision and Active Perception Laboratory (CVAP), Royal Institute of Technology, Sweden, May 1991.

[40] The MPEG home page [online]. Available: http://mpeg.chiariglione.org/

[41] FFmpeg, [online]. Available: http://www.ffmpeg.org