Decentralized Digital Content Exchange and Copyright Protection via P2P Networks

D. Tsolis
Department of Computer Engineering and Informatics
University of Patras
26504, Patras, Greece
dkt@hpclab.ceid.upatras.gr

S. Sioutas, A. Panaretos, I. Karydis and K. Oikonomou
Department of Informatics
Ionian University
49100, Corfu, Greece
(sioutas, alex, karydis, okon)@ionio.gr

***Abstract*—This paper presents the use of a novel Peer to Peer (P2P) infrastructure in order to provide rapid broad digital content exchange, digital rights protection and efficient transaction management through watermarking technologies. Copyright owners use digital watermarking techniques so as to encrypt copyright information to the content. This information is represented by multiple watermarking keys used, amongst other reasons, for proof of ownership, unique identification and transaction management. Especially for transaction management the watermarking keys used are constantly changing following the content's route from user to user. The challenges for this research are a) the use of P2P technologies for efficient Digital Rights Management (DRM), b) to apply a robust watermarking algorithm to digital content which successfully embeds and detects multiple keys for each use case and c) to successfully maintain consistent the P2P system when watermarking keys are changing during content's transaction. This paper concludes that DRM and P2P can be quite complementary and not always contradictory.**

***Keywords*-Computer networks; copyright protection; peer to peer networks; digital image processing**

I. Introduction

Peer to Peer networking is supported by suitable software which enables a computer to locate a content file (text, image, video, sound, software etc.) on another networked device and copy the encoded data to its own hard drive. P2P technology often is used to reproduce and distribute copyrighted content without authorization of rights owners. Except for digital music and video the P2P infrastructure is also used to make and distribute illegal copies of digital content which lies under the protection of the Intellectual Property Rights (IPR) legislation. For this reason the short history of P2P technology and software has been one of constant controversy by many in the content industry. The content owners are feeling even more threatened by the broad and unregulated exchange of digital content in P2P environments [10].

As a general protection measure for copyright violations through digital technologies including P2P, copyright owners often uses digital watermarking techniques to encrypt and watermark content or otherwise Digital Rights Management technologies to restrict access, totally blocking digital content to be accessed through the Internet and the P2P software infrastructure.

This paper claims that watermarking, Digital Rights Management (DRM) and P2P can be quite complementary. Specifically, a P2P network infrastructure is presented which allows broad digital content exchange while on the same time supports copyright protection and management through watermarking technologies. In brief, the platform is functioning mainly for digital images and is tracking all the watermarked image files which are distributed and copied through the P2P network. The challenge is the algorithmic complexity of detecting multiple watermarking keys in the P2P network effectively and quickly, especially when thousands of image files are concerned. This is managed by a novel decentralized lookup algorithm which allows effective watermarking key detection in optimal number of hops. The complexity is even higher during digital content's transaction management which forces multiple watemarking keys to change over time and after earch transaction.

Equivalent systems, which combine watermarking, DRM and P2P technologies do not yet exist in practice but only in theory. Certain methodologies and strategies have been proposed for exploiting P2P technologies in DRM and vice versa [9]. The proposed system is setting a new basis for the close cooperation of the two different scientific areas of DRM and P2P aiming at exploiting the distributed computing nature of P2P networks for efficient digital rights protection and management.

II. Protection, Watermarking and Keys

In this section the copyright protection part of the P2P infrastructure is presented which is mainly based on a watermarking algorithm for digital images which produces the correspondent watermarking keys distributed within the P2P environment.

A. Copyright Protection through Watermarking

The copyright protection system's main objectives are to provide an appropriate information infrastructure which supports rights management for the digital content and for the transactions taking place and on the same time protects the copyright of the digital images though robust watermarking techniques. The watermarking techniques are playing a very important role in such systems mainly because they provide

the protection means for proving the identification of the copyright owner and detecting unauthorized use of digital content [12], [14]. Towards this functionality, watermarking algorithms are casting keys to the digital content (in most of cases invisible keys) which when detected prove the copyright ownership of the digital content [1].

In case of digital content transactions a very large number of digital images are being exchanged through networks and the Internet for which the legality of their future use is highly improbable. The situation is even more difficult in P2P network infrastructures through which digital content is being exchanged based on specialized stand alone applications which exchange digital files of all kinds (and not only images). A proposed solution is to apply a watermarking algorithm which produces sufficient information which is distributed to the P2P nodes. This information consists mainly of the watermarking key and other data relating to the digital image itself.

B. Generating Keys with the Watermarking Algorithm

Generally, a watermark is a narrow band signal, which is embedded to the wide band signal of a digital image [4]. In our case spread Spectrum techniques are being used and are methods by which energy generated at one or more discrete frequencies is deliberately spread or distributed in time or frequency domains.

In particular, this technique employs pseudorandom number sequences (noise signals) to determine and control the spreading pattern of the signal across the allotted bandwidth. The noise signal can be used to exactly reconstruct the original data at the receiving end, by multiplying it by the same pseudorandom sequence: this process, known as ”de-spreading”, mathematically constitutes a correlation of the transmitted pseudorandom number sequence with the receiver’s assumed sequence [5]. Thus, if the signal is distorted by some process that damages only a fraction of the frequencies, such as a band-pass filter or addition of band limited noise, the encrypted information will still be identifiable. Furthermore, high frequencies are appropriate for rendering the watermarked message invisible but are inefficient in terms of robustness, whereas low frequencies are appropriate with regards to robustness but are useless because of the unacceptable visual impact [3], [11], [13].

In our case, the embedding of a robust multibit watermark is accomplished through casting several zero-bit watermarks onto specified coefficients. The image watermark, a random sequence of Gaussian distribution in our case, is casted multiple times onto the selected coefficients preserving the same sequence length but shifting the start point of casting by one place. Actually the final watermark that is embedded into the image is not a single sequence but many different sequences generated with different seeds. These sequences are casted, one after the other, on the mid coefficients of the image, using the additive rule mentioned above and begging from successive starting points. If all sequences where to be casted, beginning from the same starting point, then, besides the severe robustness reduction resulting from the weak correlation, the possibility of false positive detector response would dramatically increase, since every number that has participated as a seed during the sequence generation procedure, will be estimated by the detector as a valid watermark key. Shifting the starting point by one degree for every sequence casting ensures that the false positive rate will remain in very small level due to the artificial desynchronisation introduced. Every single random sequence of Gaussian distribution is generated using a different number as the seed for the Gaussian sequence generator. It is important to differentiate the sequences in order not to mislead the detection mechanism, since it is based on the correlation between the extracted sequence and the sequence produced with the watermark key. The watermark key is responsible both for the generation of the first sequence and the construction of a vector, containing the rest of the numbers that will serve as the corresponding seeds. The placement of several Gaussian sequences into the image content can model, under specific conventions, a multi-bit watermark. The detection of a zero-bit watermark is interpreted as if the bit value of the specified bit is set to one. On the contrary, failure of the detector to detect the zero-bit watermark leads to the conclusion of a zero bit value. Thus, in order for a message to be casted into the image content, it is initially encoded using the binary system and applied afterwards in the sense of zero-bit watermarks using the embedding mechanism and according to the derived bit sequence. Some important remarks regarding the novelty of the proposed schema are addressed below.

Data payload: The reason that most of the proposed robust watermarking systems are zero-bit, is highly related to the data payload. Data payload is the amount of information encoded into the image during the watermark procedure. In other words, it is the number of coefficients modified according to the additive rule. The performance of the correlation function adopted by the detector is increased when a strong statistical dependency is present. On the other hand, the statistical dependency requires a significant sequence length in order to fulfill the requirements of the correlation function. In addition, the position and the amount of coefficients modified, affects directly the resulting image quality. This is on of the most important tradeoffs that the designer of a watermarking system has to balance. Casting multiple sequences will maximize the problem of image distortion. In that sense, the maximum number of bits allowed for encoding the watermark message is crucial. In the proposed scheme a total number of 16 bits were selected. The first bit indicates the existence of a watermark. If the response is positive the detector continues with the following zero-bit watermarks, otherwise the mechanism outputs a negative response. This is a useful shortcut saving the detector of

valuable time and processing power. The second bit serves as a flag important for the decoding operation. The role of this bit flag is described in detail in the following paragraph. The next 14 bits are dedicated to the encoding of the watermark message. Under the aforementioned conventions the system is capable of embedding 214 different messages.

Seed Vector Generation: The watermark key is a positive integer value playing a vital role in the overall watermarking procedure. It corresponds to the private information that must be shared between the embedder and the detector of the watermark. One of the basic principles of private watermarking is that the encryption of the information to be embedded is performed according to a private key. Thus, if an image is watermarked using a specified key, it is impossible for the detector to detect the watermark unless provided with the same key. The encryption is accomplished by using the private key as the seed for the pseudorandom sequence of Gaussian distribution generator. In our case, there is the necessity of 15 extra numbers, one for each sequence. Thus, the private key except from its basic operation as a pseudorandom generator seed is also used as the seed for producing a vector containing 15 numbers. It is important for every private key to produce a different vector of numbers, in order to avoid undesirable statistical dependencies between different watermarks. A pseudorandom generator provided by any compiler is capable of applying this one-way relationship between the private key and the produced vector of numbers.

Flag bit operation: Under the convention, that for every one-bit-value we cast a zero-bit watermark and for every zero-bit-value we don't do anything except moving to the next starting point, the number of zero-bit watermarks to be casted is dictated by the bit sequence. It is obvious that a bit sequence containing only a single one-bit-value is preferable from a sequence consisted of 14 aces. Both for, processing power and watermark's imperceptibility purposes, a bit reversal trick is required for optimizing the embedder's performance.

Thus, after acquiring the binary representation of the message, a counter scans the bit sequence counting the zeros and the aces. If the number of aces is grater than the number of zeros a bit reversed sequence is generated. The zerobit watermarks casting is now performed according to the newly generated sequence. In that case, the flag bit is set to one serving as an indicator to the detector that the extracted sequence is bit-reversed. As a consequence, the decoder, equipped with the appropriate information, can easily decode a message represented by 14 aces binary sequence, even though the embedder had casted only two zero-bit watermarks. The benefit of using the specified trick is that even though a 16-bit watermark is supported, we only need to cast 8 zero-bits watermarks in the worst case.

The detector used in the proposed information system reveals the existence of 11 watermarks. Three of them correspond to the three zero-bit schemes while the rest 8 positive responses are used for the encoding of the fingerprint. The detector has succeeded in detecting all eleven watermarks without any confusion or misleading, resulting in a capability of facilitating proof of ownership for the digital content, copy control, unique identification and transaction tracking at the same time [3].

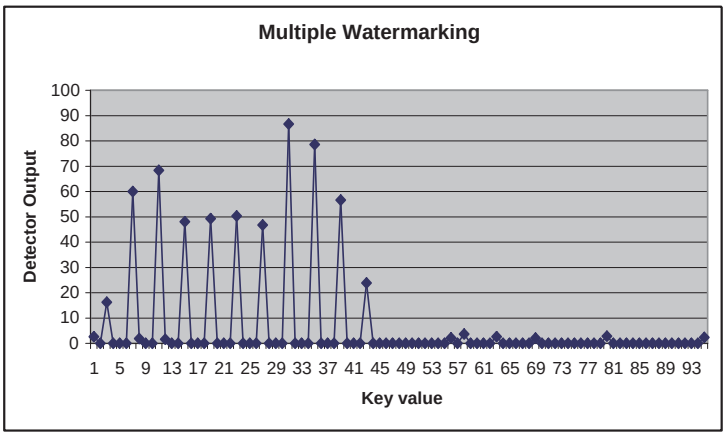

Figure 1. Multiple Watermarking Keys per Image

C. Intermediate Conclusions

In this section a watermarking algorithm has been presented which is robust enough to facilitate copyright protection and management for the digital images while at the same time produces sufficient information which is distributed and stored to the P2P nodes. This information consists mainly of the watermarking key. Taking into consideration that for each digital image a set of watermarking keys are being used for copyright protection, the next step towards an efficient P2P environment which supports digital rights management is to use these keys as an information for retrieving the copyright status of each image transacted through the P2P network. For this reason, the watermarking keys are being stored in the independent network Peers. The copyright owner can use the watermarking key as query information to track down its digital images and their use. The issue is how quickly and efficiently the Peer that contains the under inspection key is being located taking into account that thousands of digital images could exist in the P2P network and multiple watermarking keys could exist in a digital image. The complexity is even higher in the case of transaction management during which the multiple watermarking keys change when each user share and distribute digital content. The solution proposed is a scalable and robust data indexing structure, the so-called ART p2p Hierarchical scheme.

III. ART P2P NETWORK: AN OVERVIEW

ART [16] provides a tree-like structure for the P2P network upon which watermarking key-based searching can be performed. ART focuses on exact-match and range query processing on large-scale, typically distributed infrastructures and outperforms the most popular decentralized structures, including Chord (and some of its successors), BATON (and its successor) and Skip-Graphs. ART supports the join/leave and range query operations in $O(\log \log N)$ and $O(\log_b^2 \log N + |A|)$ expected w.h.p number of hops respectively, where the base b is a double-exponentially power of two, N is the total number of peers and $|A|$ the answer size (for exact-match queries, $|A| = 1$).

For comparison purposes, Table 1 presents a qualitative evaluation with respect to elementary operations between ART, Skip-Graphs, Chord and its newest variations (F-Chord(α) [15], LPRS-Chord [17]), BATON [7] and its newest variation BATON* [8]. It is noted that c is a big positive constant.

Existing structured P2P systems can be classified into three categories: distributed hash table (DHT) based systems, skip list based systems, and tree based systems (for details see the survey book [2]). The available solutions for architecting such large-scale systems are inadequate, since at the envisaged scales (trilions of watermarking-keys at millions of nodes) the classic logarithmic complexity (for point queries) offered by these solutions is still too expensive. And for range queries, it is even more disappointing. Art outperforms related work *with respect to all major operations, such as lookup (insert/delete), join/leave and to the required routing state that must be maintained in order to support these operations*. Specifically, ART achieves a sub-logarithmic complexity for all the above! ART is an exponential-tree structure, which remains unchanged w.h.p., and organizes a number of fully-dynamic cluster_peers in efficient way.

One of the basic components of the final ART structure is the LRT **L**evel **R**ange **T**ree) structure. LRT will be called upon to organize collections of peers at each level of ART.

A. The LRT structure:

LRT is built by grouping nodes having the same ancestor and organizing them in a tree structure recursively. The innermost level of nesting (recursion) will be characterized by having a tree in which no more than b nodes share the same direct ancestor, where b is a double-exponentially power of two (e.g. 2,4,16,...). Thus, multiple independent trees are imposed on the collection of nodes. Figure 2 illustrates a simple example, where $b = 2$.

The degree of the overlay peers at level $i > 0$ is $d(i) = t(i)$, where $t(i)$ indicates the number of peers at level i. It holds that $d(0)$=2 and $t(0)$=1. Let n be w-bit keys. Each peer with label i (where $1 \leq i \leq N$) stores ordered keys that belong in the range $[(i-1)\ln n, i \ln n$–1], where $N = n/lnn$ is the number of peers. Each peer is also equiped with a table named *Left Spine Index* (LSI), which stores pointers to the peers of the left-most spine (see pointers starting from peer 5). Furthermore, each peer of the left-most spine is equipped with a table named *Collection Index* (CI), which stores pointers to the collections of peers presented at the same level (see pointers directed to collections of last level). Peers having the same father belong to the same collection.

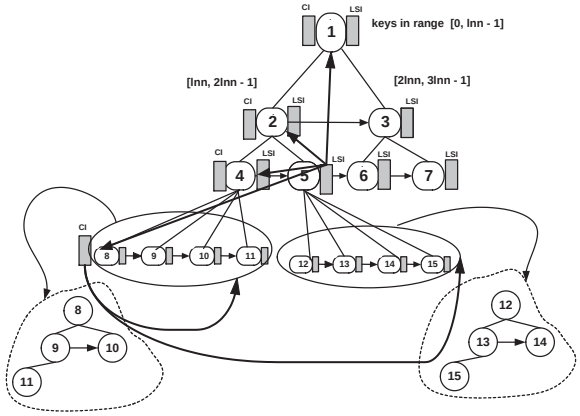

Figure 2. The LRT structure

Lookup Algorithm: Assume we are located at peer s and seek a key k. First, the algorithm finds the range where k belongs. If $k \in [(j-1)\ \ln n, j \ln n - 1]$, it has to search for peer j. The first step of algorithm is to find the LRT level where the desired peer j is located. For this purpose, it exploits a nice arithmetic property of LRT. This property says that for each peer x located at the left-most spine of level i, the following formula holds:

$$label(x) = label(father(x)) + 2^{2^{i-2}} \quad (1)$$

For each level i (where $0 \leq i \leq \log \log N$), it computes the value x of its left most peer by applying Equation (1). Then, it compares the value j with the computed value x. If $j \geq x$, it continues by applying Equation (1), otherwise it stops the loop process with current value i. The latter means that node j is located at the i-th level. Then, it follows the i-th pointer of the LSI table located at peer s. Let x the destination peer, that is the leftmost peer of level i. Now, the algorithm must compute the collection in which the peer j belongs to. Since the number of collections at level i equals the number of nodes located at level $(i-1)$, it divides the distance between j and x by the factor $t(i-1)$ and let m the result of this division. Then, it follows the $(m+1)$-th pointer of the CI table. Since the collection indicated by the CI[m+1] pointer is organized in the same way at the next nesting level, it continues this process recursively.

Analysis: Since $t(i) = t(i-1)d(i-1)$, it gets $d\,(i) = t\,(i) = 2^{2^{i-1}}$ for $i \geq 1$. Thus, the height and the maximum number of possible nestings is $O(\log \log N)$ and

P2P architectures	Lookup, Insert, Delete key	Maximum Size of routing table	Join/ Depart peer
CHORD	$O(\log N)$	$O(\log N)$	$O(\log N)$ w.h.p.
H-F-Chord(a)	$O(\log N/\log\log N)$	$O(\log N)$	$O(\log N)$
LPRS-Chord	$O(\log N)$	$O(\log N)$	$O(\log N)$
Skip Graphs	$O(\log N)$	$O(1)$	$O(\log N)$ amortized
BATON	$O(\log N)$	$O(\log N)$	$O(\log N)$ w.h.p.
BATON*	$O(\log_m N)$	$O(m\log_m N)$	$O(m\log_m N)$
ART-tree	$O(\log_b^2 \log N)$	$O(N^{1/4}/\log^c N)$	$O(\log\log N)$ expected w.h.p.

Table I
PERFORMANCE COMPARISON BETWEEN ART, CHORD, BATON AND SKIP GRAPHS.

$O(\log_b \log N)$ respectively. Thus, each key is stored in $O(\log_b \log N)$ levels at most and the whole searching process requires $O(\log_b \log N)$ hops. Moreover, the maximum size of the CI and RSI tables is $O(\sqrt{N})$ and $O(\log\log N)$ in worst-case respectively.

B. The ART structure:

The backbone of ART is exactly the same with LRT. During the initialization step the algorithm chooses as cluster_peer representatives the 1st peer, the $(\ln n)$-th peer, the $(2\ln n)$-th peer and so on. This means that each cluster_peer with label i' (where $1 \leq i' \leq N'$) stores ordered peers with keys belonging in the range $[(i'-1)\ln^2 n, \ldots, i'\ln^2 n - 1]$, where $N' = n/\ln^2 n$ is the number of cluster_peers. ART stores cluster_peers only, each of which is structured as an independent decentralized architecture. Moreover, instead of the **L**eft-most **S**pine **I**ndex (LSI), which reduces the robustness of the whole system, ART introduces the **R**andom **S**pine **I**ndex (RSI) routing table, which stores pointers to randomly chosen (and not specific) cluster_peers (see pointers starting from peer 3). In addition, instead of using fat CI tables, the appropriate collection of cluster_peers can be accessed by using a 2-level LRT structure.

Load Balancing: The join/leave of peers inside a cluster_peer were modeled as the combinatorial game of bins and balls presented in [9]. In this way, for a $\mu(\cdot)$ random sequence of join/leave peer operations, the load of each cluster_peer never exceeds $\Theta(\text{polylog } N')$ size and never becomes zero in expected w.h.p. case.

Routing Overhead: The 2-level LRT is an LRT structure over $\log^{2c} Z$ buckets each of which organizes $\frac{Z}{\log^{2c} Z}$ collections in a LRT manner, where Z is the number of collections at current level and c is a big positive constant. As a consequence, the routing information overhead becomes $O(N^{1/4}/\log^c N)$ in the worst case (even for an extremely large number of peers, let say N=1.000.000.000, the routing data overhead becomes 6 for $c = 1$).

Lookup Algorithms: Since the maximum number of nesting levels is $O(\log_b \log N)$ and at each nesting level i the standard LRT structure has to be applied in $N^{1/2^i}$ collections, the whole searching process requires $O(\log_b^2 \log N)$ hops. Then, the target peer can be located by searching the respective decentralized structure. Through the polylogarithmic load of each cluster_peer, the total query complexity $O(\log_b^2 \log N)$ follows. Exploiting now the order of keys on each peer, range queries require $O(\log_b^2 \log N + |A|)$ hops, where $|A|$ the answer size.

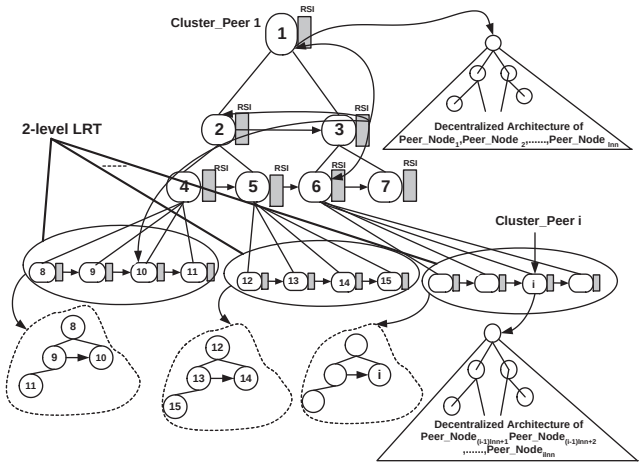

Figure 3. The ART structure

Join/Leave Operations: A peer u can make a join/leave request at a particular peer v, which is located at cluster_peer W. Since the size of W is bounded by a $polylogN$ size in expected w.h.p. case, the peer join/leave can be carried out in $O(loglogN)$ hops.

Node Failures and Network Restructuring: Obviously, node failure and network restructuring operations are according to the decentralized architecture used in each cluster_peer.

Performance Evaluation: The source code of the whole evaluation process, which showcases the improved performance, scalability, and robustness of ART is publicly available at http://code.google.com/p/d-p2p-sim/.

IV. TRANSACTION MANAGEMENT: TRANSACTION STATUS DETECTION VIA ART P2P SYSTEM

In the following we briefly present how the copyright status of each digital image can be retrieved and evaluated rapidly via the

ART p2p system.

Algorithm 1 Transaction_Status_Detection(s,wk,R_{wk},TS)

1: Input: s, wk (we are at peer s and we are looking for watermarking-key wk)
2: Output: idW (the identifier of the cluster-peer W, which stores the key wk, R_{wk} (the record or vector associated to wk key), TS(Transaction Status)
3: BEGIN
4: We compute idS:the identifier of Cluster_peer S, which contains peer s;
5: We compute idW:let j be the identifier of target Cluster_peer W, which stores the wk key;
6: Let T the basic ART structure of cluster-peers;
7: W=ART_Lookup(T, S, idS, W, idW); {call of the basic routine}
8: Linear_Scan of Cluster_peer W until we find the wk;
9: R_{wk}=the Record associated to wk key;
10: TS=Retrieve from R_{wk} record the Transaction Status Field;
11: END

The *Transaction_Status_Detection(s,wk,R_{wk},TS)* routine (Algorithm 1) gets as input the peer s in which the query is initiated and the respective watermarking-key wk and returns as output the id of the cluster_peer S, which contains peer s as well as the cluster_peer W in which the key wk belongs in. Then, it calls the basic *ART_Lookup(T, S, idS, W, idW)* routine (for further details see [16]), in order to locate the target peer responsible for key wk. Finally, it detects the Record R_{wk} associated to wk key from which it retrieves the Transaction Status (TS) Field.

V. Conclusion

In this paper we focused on a P2P network infrastructure which allows broad digital content exchange while on the same time supports copyright protection and management through watermarking technologies. In brief, a watermarking algorithm casts watermarking keys to the digital images and the same time the watermarking keys are being stored in the independent network Peers of ART system. The watermarking key detection process withih the P2P framework is very efficient and outperforms the most popular infrastructures used directly for many solutions for P2P information discovery. The key detection process is very important for the copyright owner because when successful the copyright status of each digital image can be retrieved and evaluated. The future applicability of the proposed infrastructure is strong as it could be used for the creation of P2P environments, supported by GUIs, with which a user could exchange digital files while copyright protection occurs at the same time.

Acknowledgment

We thank E. Magkos for several interesting discussions.

References

[1] M. Barni, F. Bartolini, V. Cappellini, A. Piva, "A DCT-domain system for robust image watermarking", Signal Processing, "Special Issue on Watermarking", (66) 3 (1998), pp. 357-372.

[2] J. F. Buford, H. Yu, and E. K. Lua, *P2P Networking and Applications*, Morgan Kaufman Publications, California, 2008.

[3] Ingemar J. Cox, Matthew L. Miller and Jeffrey A. Bloom, Digital Watermarking. Morgan Kaufmann Publishers 2002.

[4] Randall Davis, "The Digital Dilemma", Communications of the ACM, Volume 44, February 2001, pp.80.

[5] V. Fotopoulos, A.N. Skodras, "A Subband DCT Approach to Image Watermarking", Proc. X European Signal Processing Conference (EUSIPCO-2000), Tampere, Finland, 5-8 Sept. 2000.

[6] House of Representatives. (1998, october). Digital Millennium Copyright Act.

[7] Jagadish H.V., Ooi B.C. and Vu Q.H.: "Baton: a Balanced Tree Structure for Peer-to-peer Networks", *Proceedings 31st International Conference on Very Large Data Bases (VLDB)*, pp.661-672, Trondheim, Norway, 2005.

[8] Jagadish H.V., Ooi B.C., Tan K.L., Vu Q.H. and Zhang R.: "Speeding up Search in P2P Networks with a Multi-way Tree Structure", *Proceedings ACM International Conference on Management of Data (SIGMOD)*, pp.1-12, Chicago, IL, 2006.

[9] A.Kaporis, C.Makris, S.Sioutas, A.Tsakalidis, K.Tsichlas, and C.Zaroliagis, Improved bounds for finger search on a ram, In Proceedings 11th Annual European Symposium on Algorithms(ESA), pages 84–89. 325-336, September 2003.

[10] Computer Science and Telecommunications Board, National Research Council. (1999). The Digital Dilemma: Intellectual Property in the Information Age (pp. 2-3). Washington: National Academy Press.

[11] N. Nikolaidis and I. Pitas, Robust image watermarking in the spatial domain, Signal Processing, Elsevier, vol. 66, no. 3, pp. 385-403, 1998.

[12] Nikos Nikolaidis, Ioannis Pitas: Digital Image Watermarking: An Overview. ICMCS, Vol. 1 1999: 1-6.

[13] N. Nikolaidis, S. Tsekeridou, A. Nikolaidis, A. Tefas, V. Solachidis and I. Pitas, Applications of chaotic signal processing techniques to multimedia watermarking, Proceedings of the IEEE workshop on Nonlinear Dynamics in Electronic Systems, pp. 1-7, Catania Italy, May 18-20 2000.

[14] P. Wayner, Disappearing Cryptography - Information Hiding: Steganography and Watermarking (Second, pp. 291-318). (2002). Morgan Kaufmann.

[15] Stoica I., Morris R., Liben-Nowell D., Karger D.R., Kaashoek M.F., Dabek F. and Balakrishnan H.: "Chord: a Scalable Peer-to-peer Lookup Protocol for Internet Applications", *IEEE/ACM Transactions on Networking*, Vol.11, No.1, pp.17-32, 2003.

[16] Spyros Sioutas, George Papaloukopoulos, Evangelos Sakkopoulos, Kostas Tsichlas, Yannis Manolopoulos, Peter Triantafillou: Brief announcement: ART–sub-logarithmic decentralized range query processing with probabilistic guarantees. PODC 2010: 118-119.

[17] Zhang H., Goel A. and Govindan R.: "Incrementally Improving Lookup Latency in Distributed Hash Table Systems", *SIGMETRICS*, pp.114-125, San Diego, CA, 2003.