PHYLOGENETIC ANALYSIS OF MULTIMEDIA CODEC SOFTWARE

Sebastiano Verde, Simone Milani, Giancarlo Calvagno

Department of Information Engineering, University of Padova, Padova, Italy
email: {sebastiano.verde,simone.milani,giancarlo.calvagno}@dei.unipd.it

ABSTRACT

The phylogenetic analysis of multimedia contents (texts, images, videos and audio tracks) has been extensively investigated during the last years. Nevertheless, no research works have considered reconstructing the evolution and the dependencies among different releases of a given software. The current paper presents a first attempt in this direction considering different image and video codecs.

The proposed solution codes a set of input images and videos, builds a dissimilarity matrix from the resulting coding artifacts, and then estimates the corresponding Software Phylogenetic Tree (SPT) using a minimum spanning tree algorithm. Experimental results show that the obtained accuracy is quite promising for different configurations.

Index Terms— phylogeny, video coding, image coding, software identification, multimedia forensics

1. INTRODUCTION

The online sharing of a digital content (multimedia data, datasheet, software, etc.) gives rise to a mutation process operated by different users or software (e.g., social bots or crawlers). The uploaded data can be downloaded, edited, and re-uploaded multiple times: as a result, after a short time lapse, multiple copies or similar versions (near-duplicate or ND) of the initial content are available online. Such evolution presents many similarities with the biological history of mutations that generated the different living organisms, and therefore it can be reconstructed by using phylogenetic algorithms operating on the similarity/dissimilarity relations between each content. To this purpose, several phylogenetic tree estimation algorithms have been proposed during the last years, focusing on different types of contents (images [1, 2], videos [3], and audio tracks [4]).

To the best of our knowledge, the current paper presents a first attempt to perform a phylogenetic analysis of codec software. The main motivation for such an approach relies on the need for reliable tools that are able to compute dependencies between different codecs whose identity and origin are completely unknown. This permits understanding whether one software is a near-duplicate of another one and parameterizing the impact of operated modifications. Moreover, given a set of ND software codecs, the reconstruction of the software phylogenetic tree (SPT) permits recovering their evolution, which proves to be extremely useful in their authentication and, consequently, in the detection of plagiarisms or patent violations.

In this work we focused on reconstructing the SPT for image and video codec software. Multimedia codecs are characterized by many non-normative parts. Image and video coding standards define the architecture of decoders, allowing the developers to implement the encoder as they like, provided that the generated bitstream is compliant with the standard specifications (i.e., readable by a standard decoder). As a result, the different codec vendors entail different coding strategies (e.g., fast motion estimation algorithms or rate-distortion optimization) in order to make the characteristics of the implemented coder more efficient or suitable for a specific device (i.e., improving the quality of the reconstructed image/sequence or reducing the required computational complexity). These differences generate different coding artifacts on the reconstructed sequence, as well as a different coding efficiency in the bitstream generation; it is possible to exploit these features in reconstructing the SPT of a given software and in distinguishing different codecs.

The proposed algorithm aims at reconstructing the software phylogenetic tree for different releases of a given multimedia codec. At first, a dissimilarity matrix is generated by computing the differences between the coding artifacts generated by each software version. Then, a minimum spanning tree (MST) strategy is used to reconstruct the correct dependency graph. Experimental results show that the proposed solution is able to converge to the correct phylogenetic structure with a limited amount of input data.

The rest of the paper is organized as follows. Section 2 overviews the recent scientific literature on phylogenetic analysis. Section 3 presents the proposed strategy, where Subsection 3.1 describes the adopted dissimilarity metrics and Subsection 3.2 covers the algorithms for estimating the software phylogenetic tree. Section 4 reports the experimental results and Section 5 draws the final conclusions.

This work has been partially supported by the University of Padova project Phylo4n6 prot. BIRD165882/16.

ISBN 978-90-827970-1-5

2. RELATED WORKS

During the last decade, several solutions targeted the problem of phylogenetic analysis of a set of near-duplicate documents [5]. Later work focused on sets of ND images, i.e., pictures that differ only by mean of affine transformations or compression [1]. Such strategies parameterize the difference between each pair of images via a dissimilarity metric [6]; such measurements are used to create a complete dissimilarity graph which is then processed by a minimum spanning tree algorithm [7]. Other solutions refine the dissimilarity measurements by employing a denoising strategy [8] or introducing an aging metric [9]. Better results can be obtained composing different dissimilarity metrics [10, 11] or localizing the acquisition point for each image [2]. Additional work has been carried out on video phylogeny [12, 3, 13, 14] and audio phylogenetic analysis has been considered as well, for both digital [4] and analogic contents [15].

As for software, previous work have been mainly focusing on malicious software detection and identification. Some of the proposed solutions compute a set of characterizing features from the sequence of code instructions [16]. In order to overcome the problem of obfuscation (which is largely employed for malware and viruses), it is possible to adopt a dynamic feature synthesis [17], while others process software outcomes like register errors [18].

The strategy of analysing software outcomes proves to be extremely robust to obfuscation, re-engineering and partial alterations, which make the authentication and plagiarism detection extremely difficult as the complexity of the software increases. As a matter of fact, the proposed solution characterizes the dissimilarity between different image and video codecs relying on the generated coding artifacts. The following sections will describe this process in detail.

3. PHYLOGENETIC ANALYSIS OF CODEC SOFTWARE

Similarly to other phylogenetic approaches, the analysis of a given set $\mathcal{S} = \{s_i(\cdot)\}$ of N different codecs can be divided into two main stages.

In the first stage, dissimilarity measurements $d_{i,j}$ need to be computed for all the $N \cdot (N-1)$ pairs $(s_i(\cdot), s_j(\cdot))$. Such measurements can be organized in a $N \times N$ dissimilarity matrix $D = [d_{i,j}]$, which is then processed with a MST estimation algorithm that outputs the most likely SPT tree. Whenever the set $\mathcal{S}$ includes releases of different codecs, a preliminary clustering of the rows of D permits separating $\mathcal{S}$ into subsets on which the MST algorithm is then applied.

In this analysis we assume that the executables of $s_i(\cdot)$ are available (coder and decoder), although they could have been obfuscated and the source codes are not known. The following subsections will describe the dissimilarity computation and the MST reconstruction in detail.

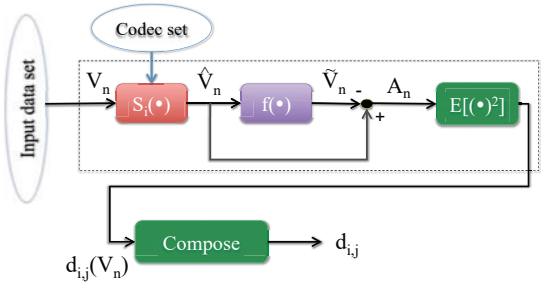

Fig. 1. Block diagram for dissimilarity computation.

3.1. Dissimilarity metrics

Every phylogenetic approach starts parameterizing the differences between object pairs in the analysis set. In order to accomplish this task, a dissimilarity or similarity metric needs to be defined. In the previous sections we highlighted the fact that coding artifacts permits distinguishing one codec from another [19]. As a matter of fact, it is possible to compare them in order to parameterize the dissimilarity level.

Given an input image or video sequence V, the proposed strategy compresses and reconstructs it using each s_i in $\mathcal{S}$. Naming the reconstructed image/sequence $\hat{V}_i$ after coding V with s_i, it is possible to generate a denoised version $\tilde{V}_i = f(\hat{V}_i)$ in order to remove most of the high-pass components introduced by block-based compression. In our implementation we adopted the denoising solution in [20], which was applied on each frame independently. This operation enables the extraction of coding artifacts,

$$A_i = \hat{V}_i - \tilde{V}_i. \tag{1}$$

We designed and tested two different dissimilarity metrics. The first one operates on the pixel-domain,

$$d^{(1)}_{i,j}(V) = \mathrm{MSE}\left(A_i, A_j\right), \tag{2}$$

where MSE denotes the pixel-wise mean squared error. The second one operates on the bitstream,

$$d^{(2)}_{i,j}(V) = \left\| h(\hat{V}_i) - h(\hat{V}_j) \right\|_2, \tag{3}$$

where $\|\cdot\|_2$ denotes the Euclidean vector norm and $h(V)$ is the byte-wise histogram (256 bins) of the bitstream of V. We observed experimentally that $d^{(1)}$ provides better results for video sequences and $d^{(2)}$ for images. The final dissimilarity metric is obtained as

$$d_{i,j}(V) = \begin{cases} d^{(1)}_{i,j}(V) & \text{if } V \text{ is a video sequence,} \\ d^{(2)}_{i,j}(V) & \text{if } V \text{ is an image.} \end{cases} \tag{4}$$

Previous work on multimedia phylogeny showed that these metrics can not be properly referred to as 'distances' in

ISBN 978-90-827970-1-5

Table 1. Releases and commits of different image and video codecs.

Ref.	H.265/HEVC rel.	Ref.	Thor com.
H1	8.0	T1	18/07/15
H2	9.2	T2	18/01/16
H3	10.1	T3	21/03/16
H4	12	T4	30/08/16
H5	13.0	T5	09/12/16
H6	15.0	T6	31/01/17
H7	16.15	T7	22/03/17
		T8	09/11/17
		T9	17/01/18

Ref.	JPEG2000 rel.	Ref.	Google Guetzli com.
J1	1.5.2	G1	16/01/17
J2	2.0	G2	21/03/17
J3	2.1	G3	02/06/17
J4	2.1.1	G4	21/08/17
J5	2.1.2		
J6	2.2.0		
J7	2.3.0		

mathematical terms as they lack symmetry, i.e., $d_{i,j} \neq d_{j,i}$, in general. This peculiarity proved to be extremely useful in estimating the associated phylogenetic tree since $d_{i,j} < d_{j,i}$ implies that the j-th element is more likely to be a derivation of the i-th one than viceversa.

However, note that the dissimilarities proposed in eq. (2) and (3) are symmetric, and therefore the SPT reconstruction algorithm can not easily infer whether $i \rightarrow j$ or $j \rightarrow i$ is more probable. As a matter of fact, we need to collect multiple dissimilarity measurements in order to obtain a satisfying accuracy.

To this purpose, a set of images or video sequences $\mathcal{V} = \{V_n\}$ is processed and the resulting values are composed in a global dissimilarity measure,

$$d_{i,j} = \frac{1}{|\mathcal{V}|} \sum_{V_n \in \mathcal{V}} d_{i,j}(V_n). \tag{5}$$

The whole dissimilarity computation process is reported in the block diagram of Fig. 1. The obtained dissimilarity matrix $D = [d_{i,j}]$ is then processed by the MST estimation strategy reported in the following subsection.

3.2. Software Phylogenetic Tree Reconstruction

The second step of a phylogenetic analysis approach usually takes the estimated dissimilarity matrix and processes it with a MST algorithm in order to reconstruct the desired structure. In the proposed strategy, we perform a two-step preprocessing before running the reconstruction algorithm.

First, we scan the dissimilarity matrix D looking for identical rows/columns. This scenario occurs when one or more releases produce no differences in the encoded file, thus yielding $d_{i,j} = 0$. If this happens, we collapse all identical nodes of the dissimilarity graph into a single 'meta-node' and the related rows and columns are removed from D.

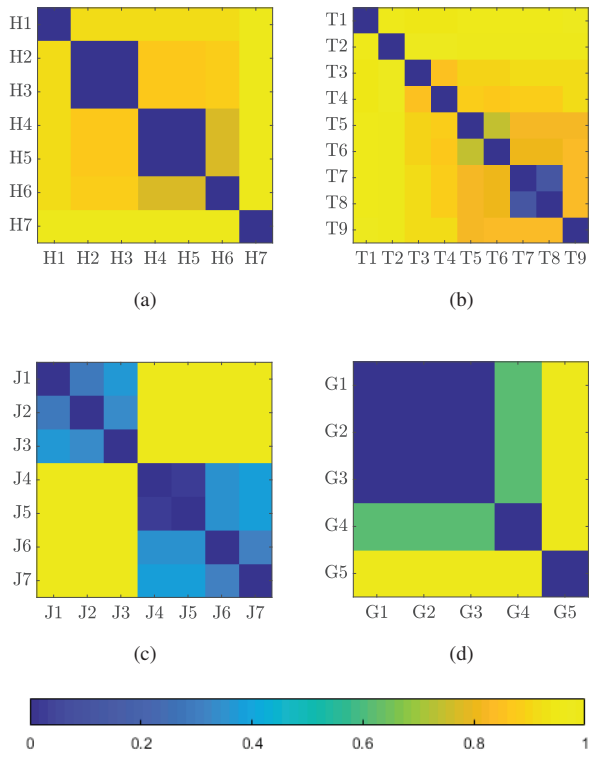

Fig. 2. Dissimilarity matrices D for HM (a), Thor (b), JPEG2000 (c) and Guetzli (d). Values are normalized to $[0, 1]$ with respect to the maximum element of each matrix.

Secondly, a clustering operation is performed on D by considering the i-th row $d_{i,:}$ as a feature vector for software $s_i(\cdot)$. If an evident separation is present, D is split in the respective sub-matrices, which are then processed independently by the reconstruction algorithm. This allows to deal with the scenario in which different codecs are present in the analysed dataset. In the proposed strategy, clustering operation was performed with a k-means algorithm.

Finally, the phylogenetic tree (or forest, in the case of matrix clustering) is estimated via the Kruskal MST algorithm.

4. EXPERIMENTAL RESULTS

In order to evaluate the effectiveness of the SPT estimation strategy, we considered different versions and releases of four different multimedia codecs.

The first software is the HM [21] implementation of the H.265/HEVC video coding standard [22]. A second coding

ISBN 978-90-827970-1-5 © EURASIP 2018

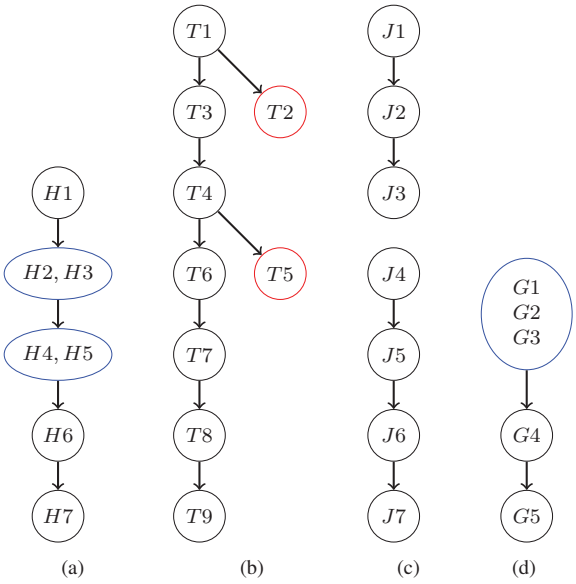

Fig. 3. Estimated phylogenetic trees for HM (a), Thor (b), JPEG2000 (c) and Guetzli (d).

software is the Thor video codec [23], whose implementations are available at the repository [24]. In order to build the dissimilarity matrices, raw video sequences (`crew`, `soccer`, `ducks_takeoff`, `ice`) at 4CIF resolution were coded with GOP IBBP of 8 frames. The final dissimilarity matrices in Fig. 2 (a,b) were obtained by averaging the different ones obtained for each sequence.

In addition to these two video codecs we considered an implementation of the JPEG2000 image coder available at [25] and the Guetzli image coder available at [26]. For the generation of the dissimilarity matrices in Fig. 2 (c,d), images from Kodak dataset [27] were used.

For each software, we downloaded different releases and commits, reported in Table 1.

Figure 2 reports the computed dissimilarity matrices for HM (a), Thor (b), JPEG2000 (c) and Guetzli (d). Matrices (c) and (d) are reported in logarithmic scale for a better visualization. It is possible to note the presence of two evident clusters in matrix (c), which is possibly due to a significant implementation change between releases J3 and J4 of JPEG2000.

Figure 3 shows the estimated phylogenetic trees or forests for the considered codecs. Note that most of the trees are list-shaped, since the release process of different software versions is usually linear. In Figure 3a and in Figure 3d it is possible to see the effect of the node collapsing operation described in Section 3.2. In fact, releases H2 and H4 of HM software produce the same outputs of releases H3 and H5, respectively, and the same happens for releases G1, G2 and G3 of Guetzli software. Therefore, the duplicate nodes are collapsed by the algorithm and form a single node in the estimated structure. In Figure 3b is reported the estimated tree

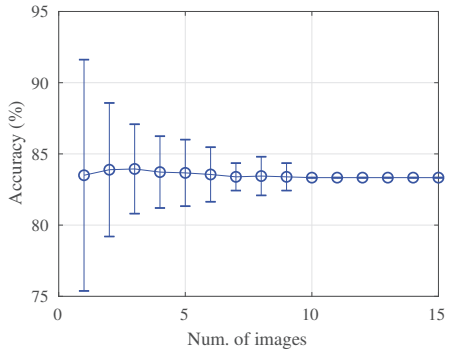

Fig. 4. Reconstruction accuracy and its standard deviation vs. number of processed images for JPEG2000 phylogenetic analysis.

for Thor software, which contains two reconstruction errors related to T2 and T5 releases. The rest of the chain is estimated in the correct order. Finally, in Figure 3c it is possible to observe the effect of the clustering operation described in Section 3.2, which brings to the reconstruction of a phylogenetic forest.

Final tests were devoted to analyze how many input images/videos are needed in order to obtain a reliable estimation. To this purpose, phylogenetic analyses on JPEG2000 software were performed considering subsets of the image dataset with varying cardinality $|\mathcal{V}|$. Reconstruction was run on multiple subsets and the accuracy of the resulting SPT was measured considering the percentage of correctly estimated edges. We evaluated the average reconstruction accuracy and its standard deviation, which are reported in Fig. 4. It is possible to notice that, in this case, $|\mathcal{V}| = 7$ is enough in order to minimize the variability of the reconstruction. For video codecs, using sets of 4 sequences proved to be enough, as the amount of processed information per sequence is higher.

5. CONCLUSIONS

In this paper we presented a first approach to the phylogenetic analysis of multimedia codec software. The proposed strategy employs a double dissimilarity metric which is then processed by a clustering and a minimum spanning tree algorithm in order to reconstruct a phylogenetic tree or forest. The solution was validated with multiple releases of different implementations of video and image codec software available online. Experimental results show that the proposed strategy is able to reconstruct with good accuracy the underlying phylogenetic structure.

Future work will focus on the analysis of codecs presenting more complex phylogenetic trees (e.g. with branches) and on the improvement of the dissimilarity metric with the aim of introducing asymmetry in the estimated matrix, allowing the reconstruction of directed graphs.

ISBN 978-90-827970-1-5

6. REFERENCES

[1] Z. Dias, A. Rocha, and S. Goldenstein, "First steps toward image phylogeny," in *Proc. of IEEE WIFS 2010*, 2010.

[2] S. Milani, P. Bestagini, and S. Tubaro, "Phylogenetic analysis of near-duplicate and semantically-similar images using viewpoint localization," in *Proc. of IEEE WIFS 2016*, Dec 2016, pp. 1–6.

[3] Z. Dias, A. Rocha, and S. Goldenstein, "Video Phylogeny: Recovering near-duplicate video relationships," in *Proc. of IEEE WIFS 2011*, 2011.

[4] S. Verde, S. Milani, P. Bestagini, and S. Tubaro, "Audio phylogenetic analysis using geometric transforms," in *Proc. of IEEE WIFS 2017*, Dec 2017, pp. 1–6.

[5] L. Kennedy and S.-F. Chang, "Internet image archaeology," in *ACM international conference on Multimedia (ACMMM)*, 2008.

[6] A. De Rosa, F. Uccheddu, A. Costanzo, A. Piva, and M. Barni, "Exploring image dependencies: A new challenge in image forensics," in *SPIE Conference on Media Forensics and Security*, 2010.

[7] Z. Dias, A. Rocha, and S. Goldenstein, "First steps toward image phylogeny," in *Information Forensics and Security (WIFS), 2010 IEEE International Workshop on*, dec. 2010, pp. 1 –6.

[8] A. Melloni, P. Bestagini, S. Milani, M. Tagliasacchi, A. Rocha, and S. Tubaro, "Image phylogeny through dissimilarity metrics fusion," in *Proc. of EUVIP 2014*, Dec 2014, pp. 1–6.

[9] S. Milani, M. Fontana, P. Bestagini, and S. Tubaro, "Phylogenetic analysis of near-duplicate images using processing age metrics," in *Proc. of IEEE ICASSP 2016*, March 2016, pp. 2054–2058.

[10] N. Le Philippe, W. Puech, and C. Fiorio, "Phylogeny of JPEG images by ancestor estimation using missing markers on image pairs," in *Proc. of IPTA 2016*, 2016.

[11] M. Oikawa, Z. Dias, A. Rocha, and S. Goldenstein, "Manifold Learning and Spectral Clustering for Image Phylogeny Forests," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 11, pp. 5–18, 2016.

[12] John R. Kender, Matthew L. Hill, Apostol Natsev, John R. Smith, and Lexing Xie, "Video genetics," in *ACM Conference on Multimedia*, 2010.

[13] F. Costa, S. Lameri, P. Bestagini, Z. Dias, S. Tubaro, and A. Rocha, "Hash-based frame selection for video phylogeny," in *Proc. of IEEE WIFS 2016*, 2016.

[14] S. Milani, P. Bestagini, and S. Tubaro, "Video phylogeny tree reconstruction using aging measures," in *Proc. of EUSIPCO 2017*, Aug 2017, pp. 2181–2185.

[15] S. Verde, N. Pretto, S. Milani, and S. Canazza, "Stay true to the sound of history: Philology, phylogenetics and information engineering in musicology," *Applied Sciences*, vol. 8, no. 2, 2018.

[16] A. Palisse, A. Durand, and J.-L. Lanet, "Malware'O'Matic a platform to analyze Malware," in *French Japanese workshop on CyberSecurity*, Rennes, France, Sept. 2016, Inria.

[17] F. Biondi, S. Josse, and A. Legay, "Bypassing Malware Obfuscation with Dynamic Synthesis," *ERCIM News*, , no. 106, Sept. 2016.

[18] G. E. Dahl, J. W. Stokes, L. Deng, and D. Yu, "Large-scale malware classification using random projections and neural networks," in *Proc. of ICASSP 2013*, May 2013, pp. 3422–3426.

[19] P. Bestagini, S. Milani, M. Tagliasacchi, and S. Tubaro, "Codec and gop identification in double compressed videos," *IEEE Transactions on Image Processing*, vol. 25, no. 5, pp. 2298–2310, May 2016.

[20] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259 – 268, 1992.

[21] Heinrich Hertz Institute, "HM software repository," `https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware`, Dec. 2017.

[22] P. Bordes, G. Clare, F. Henry, M. Raulet, and J. Viéron, "An overview of the emerging HEVC standard," in *Proc of ISIVC 2012*, jul. 2012.

[23] G. Björntegaard, T. Davies, A. Fuldseth, and S. Midtskogen, "The Thor video codec," in *Proc. of DCC 2016*, March 2016, pp. 476–485.

[24] Cisco System, "Thor Video Codec Git Hub Repository," `https://github.com/cisco/thor`, 12 2017.

[25] ISPGroup (UCL), "OpenJPEG: An open-source JPEG 2000 codec," `https://openjpeg.org`, 2017.

[26] Google, "Guetzli: Perceptual JPEG Coder," `https://github.com/google/guetzli`, 2017.

[27] Kodak, "Kodak Lossless True Color Image Suite," `http://r0k.us/graphics/kodak`, 2017.

ISBN 978-90-827970-1-5 © EURASIP 2018