Birthmark based identification of software piracy using Haar wavelet

Shah Nazir[a]*, Sara Shahzad[b], Rahmita Wirza[c], Rohul Amin[d], Muhammad Ahsan[a], Neelam Mukhtar[b], Iván García-Magariño[e], Jaime Lloret[f]

[a]Department of Computer Science, University of Swabi, Swabi, Pakistan
[b]Department of Computer Science, University of Peshawar, Peshawar, Pakistan
[c]Department of Multimedia, University Putra, Sardang, Malaysia
[d]Department of Mathematics, University of Peshawar, Peshawar, Pakistan
[e]Department of Computer Science and Engineering of Systems, Instituto de Investigación Sanitaria Aragón, University of Zaragoza, c/ Atarazana 2, 44003, Teruel, Spain
[f]Instituto de Investigación para la Gestión Integrada de Zonas Costeras, Universitat Politècnica de València, València, Spain

Abstract
Piracy of software is an increasing problem of modern day software industry. Piracy of software is the unlawful use of software or part of it without proper permission as described in license agreement. Software piracy is a serious crime but not taken seriously by most people. Preventing software piracy is very important for the growing software industry. Efforts are being made to prevent and detect software piracy. Several techniques have been developed most important of which is software birthmark. The birthmark of a software is the intrinsic properties of software. A recent research shows that a features based software birthmark can be used as a strong mechanism to detect piracy of a software and how much piracy performed has been performed on it. An objective measure is needed to overcome this problem and to compare features based birthmark of a software which efficiently and precisely detect piracy in reproduction of software. The proposed study presents Haar wavelet collocation method for software features (birthmark) to detect piracy. The proposed method gives an exclusive solution for the features based birthmark of software and is then further used for comparisons of birthmark. The results of the proposed study show the effectiveness in term of accuracy and efficiency to compare the features based software.

Keywords: Software piracy detection, Software birthmark, Haar wavelet, partial differential equation.

1 Introduction

Piracy of software has turned out to be a key concern and is an increasing problem of modern day software industry. Piracy of software is the unlawful use of software or part of software without proper permission as enforced in the license and agreement. Most of the people don't know about the software piracy which they are doing it is a serious crime. Preventing software piracy is important for the growing of industry, as software development involves huge effort of creating ideas and implementing it in real world applications. Many of the techniques are used for prevention of software piracy. Several versions of advanced techniques for theft detection is available such as software watermarking [1–7], finger prints [8, 9], software cloning [10, 11] plagiarism detection [12–16], and software birthmark [17–28]. Software watermark is used for the

*The authors to whom all the correspondence should be addressed. Email: snshahnzr@gmail.com

ownership of program, fingerprints is used for identification intellectual property, software clone detection is used to detect piracy in code. These techniques have some limitations, due to which it is not of more in use. For example watermark can be erased from a document using advanced technique of code obfuscation. Among these theft detection techniques, software birthmark is one of the prominent technique. Birthmark of a software is the intrinsic properties of software which is used to detect the theft in a software. Similarities in birthmarks of softwares shows piracy among them.

A topical study shows that features based software birthmark provide a right and strong mechanism for detection of piracy and or level of piracy carried out in a software. To overcome this drawback an objective measure is needed for the comparison of features based birthmarks of software which accurately and resourcefully detect the piracy in software.

Different research work has been conducted by the research community and has presented several methodologies for the identification of different types of birthmarks of a software [20, 24–34]. S. Nazir et al. [29] have presented features based design of software birthmark and then presented a proper estimation process of birthmark in [30]. The birthmark of software has been extensively discussed in literature from different perception in the region of software piracy and detection of theft. Still there is lack of an objective measure to efficiently compare software birthmarks for detection of piracy.

This paper discusses a Haar Wavelet collocation method for software features based birthmark to detect piracy for the work presented in [29]. The other software (pirated) can be evaluated for comparison of features based birthmark with the original software and this comparison will eventually confirm the piracy performed in software. The proposed model gives a best fit solution for the features based software birthmark and is then further used for comparison of birthmarks of software. The results of the proposed model show the effectiveness in terms of accuracy and efficiency for the comparison of features based birthmark of software.

The reminder of the paper is organized as follows. Section 2 presents the details of the literature related to the proposed research. Section 3 describes the details of the Haar wavelet that is used as a tool for modelling of the proposed research. Section 4 discusses the governing equations. The proposed scheme for the governing equations are briefly discussed in section 5. The numerical results are discussed in section 6. The paper concludes at section 7.

2 Related work

Software industries are faced with extremely high loss due to piracy of software. Pirates gain huge amount of money from the business of software piracy which they are doing. According to report of 2013 Business Software Alliance (BSA) [35], 43% of software configured on computers in the globe were not appropriately licensed. The values of such software were approximately to 62.7 billion dollars. According to G. Myles and C. Collberg [28] there are three major threats to the industry of software which includes unlawful reselling of the lawful software, tampering in software, and malicious reverse engineering. Different mechanism is already in use for the detection and identification of theft of a software. Such techniques include software watermark, fingerprint, clone detection, plagiarism detection, and Software birthmark. Among these techniques only the technique of software birthmark is quite reliable, as in the rest of the techniques the code of software can be removed by using techniques of advanced code obfuscation.

Tamada et al. [36] suggested the first birthmark which consists of constant value in field variable, sequence of method calls, inheritance structure, and used classes. The suggested birthmark is effectively in use in the industry for detection of software theft. Y. Zeng et al. [37] presents an

abstract interpretation framework for software birthmark based on semantic. G. Myles and C. Collberg [28] presents a mechanism of "Whole Program Path Birthmarking" which is based on the complete control flow of program of a software. Y. Mahmood et al. [38] suggested a birthmark technique for software that is named as method based similarity level. By the help of this method the code elements and their properties can be traced. S. Choi et al. [23] analysed the static API based birthmark of software for the executable of Windows binary. In their research they did comparison of 49 Windows executable and showed that the birthmark can distinguish and detect the theft done in the form of copies of software. The birthmark is further compared with the Windows dynamic birthmark and presented that it is more appropriate for the application of GUI. H. Park et al. [24] used static API trace birthmark for theft detection of Java based programs. The results of the method show that the static API birthmark can detect related components of two different packages whereas the other birthmark technique(s) fails to do so. G. Myles and C. Collberg [39] conducted an experimental analysis of the K Gram based birthmark of software by analysis of 111 programs of java. X. Xie et al. [40] suggested a static birthmark for k-gram and their weights. The weight is computed by analyzing rate of change in frequency of k-gram of the original and modified program. Apart from this, different studies [16, 20, 24–29, 34, 36, 37, 41, 42] also presents types of birthmark, their analysis and evaluation. Z.Tian et al. [43] presented two approaches based on dynamic birthmark. The first approach takes out key instructions, and the second approach takes out system calls. These two approaches consider the effect of thread scheduling on computing of the software birthmarks. Their experimental study shows that the two approaches can successfully detect plagiarism of multi-thread programs and show evidence of strong resilience to preserving transformations. D. Kim et al. [44] developed an intelligent software filtering system based on software similarity. The proposed system measures the similarity of properties of original program and suspicious program. After the similarity measuring it shows that whether the suspicious software is a cracked copy of the original software. The system they proposed can handle fresh program by grouping using method of machine learning. Several experiments has been carried out to show the effectiveness of the system. Dong-Kyu Chae et al. [45] proposed authority histogram that can satisfy the three essential requirements for a good birthmark in term of resilience, credibility and scalability. Given an original and suspicious program, it first constructs A- CFGs for each program. Then generate Ahs from each A-CFG and compute their similarity. Authority histogram not only reflects the frequency of APIs, but also their call order. The proposed method works even in large program. By performing several experiments the method verify that both the credibility and resilience of authority histogram exceed those of existing birthmarks, so the authority histogram provide enhanced accuracy in plagiarism detection. The present study is a step toward birthmark identification in terms of comparison of softwares based on Haar wavelet collocation.

3 Haar Wavelet

The term wavelet represents the diminutive wave. A classical behaviour of wavelets is that it oscillates on finite interval and is zero everywhere outside the interval. Fourier transforms can approximate stationary signals but cannot give good approximation for non-stationary signals. Fourier transforms are localized in the frequency domain. Due to this reason frequency-amplitude representation cannot give information that at which time what frequency component exists in given interval of time. The Wavelet transforms on the other hand are capable to give time-frequency representation and provides information about existence of frequency components in interval of our interest. When some changes occur in the frequency it does not make any changes

in the time domain using wavelet transforms [46]. Haar wavelet is the simple family of wavelet. This wavelet forms orthogonal basis in $L^2(R)$. Any function $f(x)$ which is square integrable can be written as the linear combination of infinite number of Haar basis functions $\tilde{\mathcal{H}}_p(x) : p \in \mathbb{Z}^+$:

$$f(x) = \sum_{i=1}^{\infty} a_i \tilde{\mathcal{H}}_p(x). \tag{1}$$

For approximation purposes the above series is terminated at a finite number of terms. Idea of Haar transformation is based on multi-resolution analysis. Scaling and translating parameters make it possible to resolve time domain in accordance to their scales at different levels of resolution. Haar wavelet is a piecewise constant function along with some restraints. Mathematically first function in Haar wavelet family, called the scaling function, is defined on interval $[a, b)$ as [47]:

$$\tilde{\mathcal{H}}_p(x) = \begin{cases} 1 & \text{for} \quad x \in [a_1, a_2), \\ -1 & \text{for} \quad x \in [a_2, a_3), \\ 0 & \text{elsewhere}, \end{cases} \tag{2}$$

where

$$a_1 = a + (b-a)\frac{s}{r}, \quad a_2 = a + (b-a)\frac{s+0.5}{r}, \quad a_3 = a + (b-a)\frac{s+1}{r}. \tag{3}$$

In the above definition integer $r = 2^j$, $j = 0, 1, \ldots, J$, represents the level of the wavelet and integer $s = 0, 1, \ldots, r-1$ is the translation parameter. Maximum level of resolution is J. The index p in Eq. (2) is calculated using the formula $p = r + s + 1$. In case of minimal values $r = 1$, $s = 0$, we have $p = 2$. The maximal value of p is $2M = 2^{J+1}$. We define the following notations for integrals of the Haar wavelets;

$$\tilde{\mathcal{H}}_p^1(x) = \int_0^x \tilde{\mathcal{H}}_p(x)dx'$$

So using Eq.(2) we get

$$\tilde{\mathcal{H}}_p^1(x) = \begin{cases} x - a_1 & \text{for} \quad x \in [a_1, a_2), \\ a_3 - x & \text{for} \quad x \in [a_2, a_3), \\ 0 & \text{elsewhere}. \end{cases} \tag{4}$$

For Haar wavelet collocation method, the computational domain $[a, b]$ is discretized using the following collocation points:

$$x_j = a + (b-a)\frac{j-0.5}{N} \quad j = 1, 2, \ldots, N. \tag{5}$$

The Haar wavelet has several applications, such as [48];

- Haar wavelet belongs to the family of square functions (i.e. they may acquire only the values $0, +1, -1$). Such wavelets are simple mathematically when compared with the other wavelet families.
- The Haar matrices contains many zeros; this makes the Haar transform faster than for the other wavelet functions.
- The method is very suitable for solving boundary value problems, since the boundary conditions are taken into account automatically.

- Numerical prediction of the shock formation by Haar wavelet method is accompanied only by local oscillations in the vicinity of the shock, whereas e.g. in the Fourier method, the oscillations are presented through the whole domain.
- In the case of resonance, the Haar method describes accurately the sharp peaks, while for the finite element method solution a broad specter of forcing frequencies appear (the signal is smeared out to some region).

4 Governing equations

The proposed method for comparison of suggested feature based software birthmark is mathematically modelled for the facilitation of the birthmark comparison on the basis of the features defined [29]. This comparison of feature based birthmark suggests the similarities among software applications. Four features were previously identified in [29]. Here, we considered the three main features from the previous identified features. These features include input features, nonfunctional features and functional features. The category of pre conditional features has been ignored. The pre-conditional features include program availability, runnable, identification of components. The pre-conditional feature is a general category which was ignored due to the reason that these features exists in all types of software while checking the piracy. The governing equations for the features based software birthmark are given below;

$$\begin{aligned} U_x(x,y) &= f(x,y) \\ U_y(x,y) &= g(x,y) \end{aligned} \tag{6}$$

with the boundary conditions

$$\begin{aligned} U(0,y) &= f_o(y) \\ U(x,0) &= g_o(x). \end{aligned} \tag{7}$$

5 Haar wavelets scheme for the governing equations

The following subsections provide descriptions of the methodology section of the paper.

5.1 Software features identification

The features of a software system contain all the necessary and important information and are the static attributes in the software. The software features are interlinked with each other performing diverse operations. A clear perception of these features of software and their organization into logical grouping is additional step toward the understanding of the code of a program. This understanding of a precise code of a program can eventually help in identifying the similarities among software applications. The requirements of software are specified in the field of domain engineering. Defining the important features of a software is a way to easily understand what exists in the software. The concept of identifying and defining different features of a software(software birthmark)has been proposed [29]. To provide an understandable definition of birthmark the defined features are categorized into different types. The research work groups related features of software program under four broad categories. These categories are pre-conditional software features, input software features, functional software features and non-functional software features. The features (termed as sub features belonging to each category) collectively define the features based software birthmark.

5.2 Haar wavelet scheme for birthmark based identification of software piracy

The comparison of birthmark can take part in accepting the effectiveness of a birthmark by comparing all the essential features of software. There is a need for a study that can formally compare the features of a birthmark for a software. This features based comparison of software birthmark will smooth the progress of software industry in detecting software theft and piracy. The comparison of features with high accuracy helps in detecting and identifying software theft and piracy. Birthmarks comparison is essential for inspecting the similarity of software programs. If the birthmarks of both of the software are similar, at the end the software programs are similar. The software program features are considered to be a birthmark and can be compared with features of other software programs (duplicated or pirated copy) to investigate the extent of originality and similarity of the software program. The comparison of feature based birthmark of software [29] is modelled mathematically to help the comparison process of defined features of birthmark. This comparison of feature advises the similarities among copies of the software. Several mathematical models are used by practitioners to model different real life phenomenon. Some of these techniques include separable variable methods, exact equations, linear equations, solution by substitution and numerical methods. Such methods are used for solving the first order differential equations [49]. The features based software birthmark has been identified [29]. The goal of presenting features based software birthmark is to have a detailed and cooperative identity of a software which can effectively be used for software piracy and theft detection. The technique of software features based birthmark estimation and the idea of birthmark estimation has been firstly presented [30]. S. Nazir et al. [50, 51] evaluate and classified the software birthmark. They used the concept of estimation of software birthmark and execute the system using Fuzzy logic [52]. The proposed method for comparison of suggested feature based software birthmark is mathematically modeled for facilitation of the comparison of birthmark on the basis of the defined features [29], and Haar wavelet scheme is used. The features based comparison shows the similarities among copies of software. The proposed study classifies features in four main categories that were identified previously [29]. The features are pre conditional features, input features, functional features and non-functional features. The category of pre-conditional features further divided into three sub-features that are availability of program, identification of components and runnable. These are the important features which can be checked initially for each type of program, due to which this category is ignored in the proposed research work. The rest of three categories were considered in this study.

To construct Haar wavelet scheme, we approximate the derivative by Haar series as;

$$U_x(x,y) = \sum_{q=1}^{2M}\sum_{p=1}^{2M} \alpha_{pq}\tilde{\mathcal{H}}_p(x)\tilde{\mathcal{H}}_q(y) \tag{8}$$

and

$$U_y(x,y) = \sum_{q=1}^{2M}\sum_{p=1}^{2M} \beta_{pq}\tilde{\mathcal{H}}_p(x)\tilde{\mathcal{H}}_q(y) \tag{9}$$

Integrating Eq. (8) with respect to x from 0 to x we get

$$U(x,y) = U(0,y) + \sum_{q=1}^{2M}\sum_{p=1}^{2M} \beta_{pq}\tilde{\mathcal{H}}^1_p(x)\tilde{\mathcal{H}}_q(y) \tag{10}$$

Putting Eq. (8) and Eq. (10) in Eq. (6) we get a system of algebraic Eqs.

$$\begin{aligned}\sum_{q=1}^{2M}\sum_{p=1}^{2M}\alpha_{pq}\tilde{\mathcal{H}}_p(x)\tilde{\mathcal{H}}_q(y) &= f(x,y)\\ \sum_{q=1}^{2M}\sum_{p=1}^{2M}\beta_{pq}\tilde{\mathcal{H}}_p(x)\tilde{\mathcal{H}}_q(y) &= g(x,y)\end{aligned} \tag{11}$$

Eq. (11) can be solved by any mathematical software for the unknown coefficients α_{pq} and β_{pq}. We have used MATLAB for our calculations. By finding these unknown and putting back in Eq. (10) we can obtained the numerical solution of the governing equations. The experimental rate of convergence of Haar function is 2, while in the proposed case study of different software such as java calculator, timer convertor, Photo illustrator, password generator and so on, the experimental rate is calculated, which is approximately equal to 2. The confirmation of the theoretical results is given in Table 1.

6 Numerical results and discussion

First of all we can implemented Haar on such problem which has exact solution. After successful implementation we will analyze if the error is significant, and then we will apply it to our proposed governing equation. We also computed the experimental rate of convergence $R_c(N)$ which is defined as

$$R_c(N) = \frac{\log[L_\infty(N/2)/L_\infty(N)]}{\log 2} \tag{12}$$

Case Study 1. In this first case, we simulate a real birthmark of the literature for existing Windows binary executables [54] with the following function similarity;

$$sim_f(f_1, f_2) = \frac{2|F_1 \cap F_2|}{|F_1| + |F_2|} \tag{13}$$

where $|F_1|$ and $|F_2|$ are the number of API calls in programs F_1 and F_2, and $|F_1 \cap F_2|$ is the number of common API calls in F_1 and F_2.

In this context, we have taken the following type of equation for the illustration;

$$U_x = e^{x+y}, \quad U_y = e^{x+y}, \tag{14}$$

with boundary conditions

$$U(0,y) = e^y, \quad U(x,0) = e^x. \tag{15}$$

The exact solution is

$$U(x,y) = e^{x+y} \tag{16}$$

In Table 1 we have presented the maximum absolute error L_∞ to check the performance of the present Haar wavelets method. The table shows that by increasing the collocation points the accuracy of the method increases. The maximum absolute error depends on the collocation points, as we increase these points the error decreases. The collocation points are 2, 8, 16, 64, 128, and so on. The efficiency of the method is given in terms of CPU time (seconds) and is shown in Table 1. The accuracy is shown in term of L_∞ while the efficiency is presented in term

Table 1: **The maximum absolute errors for different number of collocation points are given with CPU time for Case Study 1 of present method.**

M	L_∞	$R_c(N)$	CPU time (Present method)
1	1.149×10^{-1}	—	0.167
2	3.617×10^{-2}	1.6675	0.207
4	1.015×10^{-2}	1.8333	0.297
8	2.688×10^{-3}	1.9169	0.329
16	6.917×10^{-4}	1.9583	0.910

of CPU time. The rates of convergence is also calculated which is approximately equal to 2, which confirms the theoretical results.

The comparison of Haar Wavelet and exact solution are depicted in Figure 1, where we can see that the numerical results match with the results of exact solution. The HW solutions obtained a minimum of 1.0 and a maximum of 2.77, with an average result of 1.86. The surface plot of the numerical results is shown in Figure 1. The profile of absolute error is shown in Figure 2 up to an accuracy of 10^{-4}, which is a good achievement. The Comparison of exact and approximate solution by proposed method are shown in Fig. 1 (left) while three dimensional graphs of the approximate solution for $2M = 16$ are shown in Fig. 1 (Right). In this particular latter case, the maximum solution result was 7.32, the minimum was 0.33, and the average was 2.82, when considering a range of [0,1]x[0,1]. It is worth mentioning that the minimum, maximum and averages may vary regarding the ranges, since maximum and minimum were located at the limits of the figure and consequently these would be different for extended limits.

The maximum absolute errors are shown in Fig. 2. The figure shows the convergence and accuracy of the method. The most relevant feature of the Figure 1 is that a good result was obtained from proposed method as compared to the exact solution, also the figure gives an objective measures and can efficiently compare the software(s) birthmark for the purpose of piracy detection. The minimum absolute error was $0.72 \cdot 10^{-4}$, the maximum absolute error was $7.5 \cdot 10^{-4}$ and the average absolute error was $3.07 \cdot 10^{-4}$. Hence the proposed research makes the process of detecting software piracy and theft more clear and easy. Previously, no accurate mathematical model exits for identification of software piracy based on some features and birthmark. From the accuracy and efficiency shown in Table 1, it is clearly mentioned that the proposed method is an objective measure and is validated by the help of examples. The results obtained from the proposed collocation method was enough sufficient accordingly to our expectation. Such method can help in identification and prediction of piracy of software and theft detection purposes. Furthermore, the software industry can get benefits of the method applied for their business purposes.

Case Study 2. In this second case study, we used the existing dynamic birthmark for Java [55], defined as the union of all k-long call sequences during the execution of program P and input I:

$$B(P, I, k) = \bigcup_{o} S(T(o), k) \quad where\ class(o) \in API \tag{17}$$

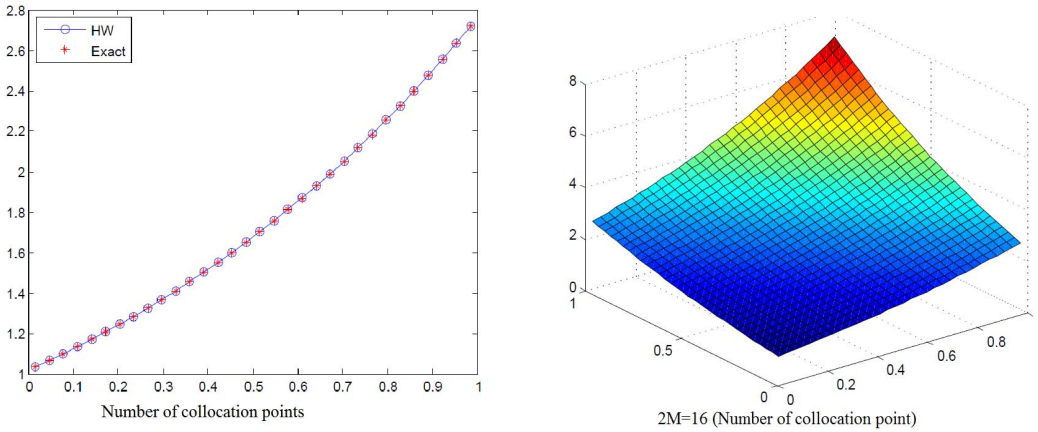

Figure 1: Comparison of exact and HW solution (left) and surface plot of HW solution (Right) at $M = 16$ for Case Study 1.

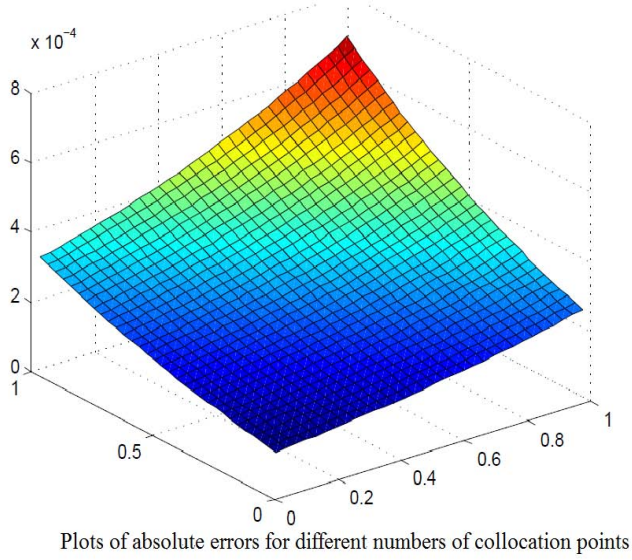

Figure 2: Surface plot of absolute error at $M = 16$ for Case Study 1.

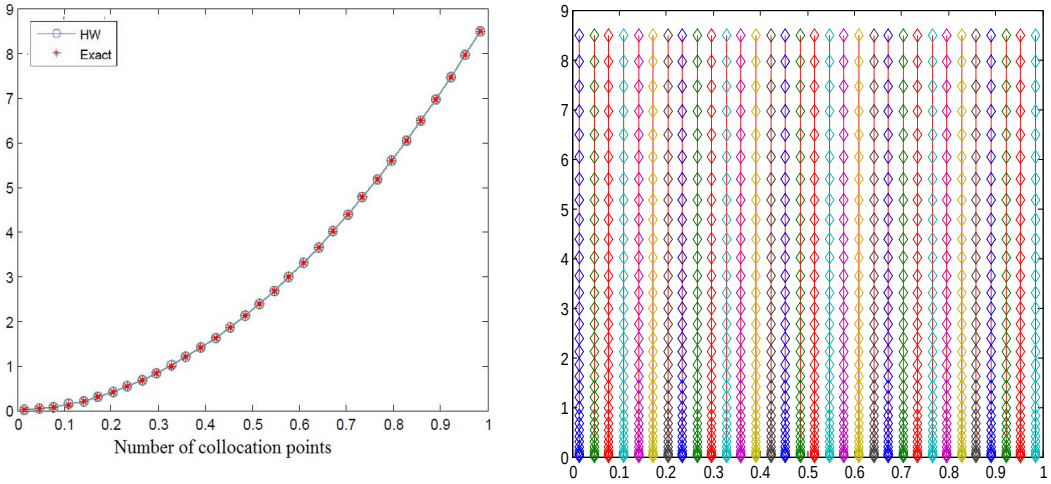

Figure 3: Haar wavelet Solution (left) and coefficient matrix (right) for Case Study 2.

In this context, we considered the governing equation in which $f(x, y) = 17x + 16y$ and $g(x, y) = 16x + 17y$ with the boundary conditions

$$\begin{aligned} U_x &= 17x + 16y, \quad U_y = 16x + 17y, \\ U(0, y) &= \frac{17}{2}y^2, \quad U(x, 0) = \frac{17}{2}x^2. \end{aligned} \tag{18}$$

The profile of Haar wavelet solution of the governing solution is captured in Figure 3. The Haar wavelet solutions ranged from a minimum of 0.00 to a maximum of 8.56 and average of 3.96. The three dimensional plot of approximate solution by proposed method are shown in Figure 4. In this three dimensional solutions, the results ranged from a minimum 0.00 to a maximum of 32.61 with an average of 8.74 considering the range of [0, 1]x[0, 1] A better accuracy can be obtained by increasing number of collocation points. The Comparison of exact and approximate solution by proposed method are shown in Fig. 3 (left). The coefficient matrix are obtained from solving the linear system which are shown Fig. 3 (right). The three dimensional graphs of the approximate solution for $2M = 16$ are shown in Fig. 4. From figures, we see that the proposed method solution are near the exact solution and the numerical results exhibit that the technique is simple and effective. In order to get more accuracy, there is a need of larger number of nodal points that is the solitary drawback to this technique because of the choice of large number of nodal points result expanded computational cost because inversion of $N \times N$ matrix.

The disadvantage of the proposed method is the size of the matrix when it is very large ($N = 256$, where $N = 2M$ is the number of collocation points). The system says out of memory (it depends on the computer RAM). The Haar function uses constant box function and due to this we need a large number of collocation points in order to achieve better accuracy. The error decreases as we increases the collocation points which means that the accuracy increases while the efficiency is presented in term of CPU time given in Table 1. It has already been discussed in Case Study 1.

Below input values in Table 2 were taken as example from [53]. The following calculations were performed for accuracy, F-Measure, resiliency, credibility, and scalability Different calculations

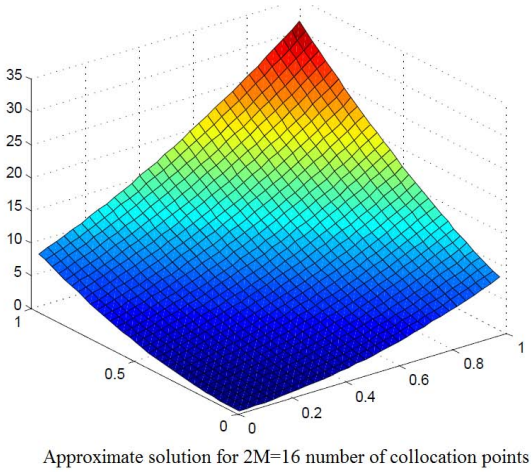

Figure 4: Surface plot of Solution for Case Study 2.

Table 2: **Input values as example** [53]

Java calculator	0.7 0.7 0.7 0.7 0.8 0.7 0.8 0.9 0.8 0.9 0.7 0.9 0.7 0.8 0.7 0.8 0.9 0.8 0.9 0.7 0.8 0.9 0.9 0.8 0.7 0.8 0.9 0.9 0.9 0.8 0.8 0.8 0.9
Timer convertor	0.5 0.5 0.6 0.7 0.8 0.7 0.6 0.6 0.8 0.9 0.1 0.3 0.2 0.3 0.2 0.1 0.2 0.5 0.6 0.1 0.3 0.2 0.3 0.2 0.7 0.2 0.2 0.1 0.1 0.3 0.2 0.3 0.2
Photo illustrator	0.1 0.1 0.3 0.2 0.3 0.2 0.1 0.2 0.2 0.1 0.1 0.3 0.2 0.3 0.2 0.1 0.2 0.2 0.1 0.1 0.3 0.2 0.3 0.2 0.1 0.2 0.2 0.1 0.1 0.3 0.2 0.3 0.2
Password generator	0.5 0.5 0.6 0.7 0.8 0.7 0.6 0.6 0.8 0.9 0.7 0.6 0.7 0.6 0.7 0.8 0.6 0.5 0.6 0.7 0.6 0.5 0.6 0.8 0.7 0.2 0.2 0.1 0.1 0.3 0.2 0.3 0.2

were carried out in term of; accuracy, F-Measure, resiliency, credibility, and scalability.

$$Acc = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN}, \quad Acc = 0.98 \tag{19}$$

$$Precision = \frac{TP}{TP + FP} \quad , Precision = 1 \tag{20}$$

$$Recall = \frac{TP}{TP + FN} \quad , Recall = 0.25 \tag{21}$$

$$F - Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad , F - Measure = 0.4 \tag{22}$$

According to Tamada et. al. [36] birthmark can satisfy the two properties given below.
Resilience: Let p' be the programs obtained from p by applying semantic preserving transformation T, and we say f is resilient to T if $f(p) = f(p')$. Due to the fact that it is the inherited property of software that is not been dependent upon any extra code added to the software, resilience was supported.
Credibility:Let p, q be two independently written programs which achieve the same task, and then f is credible if $f(p) \neq f(q)$. The credibility of the proposed system was checked and the system was found enough credible for different values of solution for Case Study 1 and Case Study 2 which have been plotted in figure 1(right) and figure 4.
Scalability- The accuracy increases as the size of the system increases. Such as for M=1, the maximum error is 1.149×10^{-1}, while for $M = 16$, the maximum error is 6.917×10^{-4}, as given in Table 1.

7 Conclusion

The aims of the proposed research is to present a method of Haar wavelet based approach to compare birthmarks of software based on features identified by S. Nazir et al. [29]. These feature based software birthmark is divided into three categories of input software feature, functional software features and non-functional software features. The combination of these features is called as birthmark of the software. The proposed research work in this paper presents a Haar wavelet based system which is method for birthmark based features of software which in turn helps comparing software birthmarks to be tested for piracy detection purpose. The proposed approach based on Haar wavelet for the birthmark based features categories gives an objective measures and can efficiently compare the software(s) birthmark for the purpose of piracy detection. Hence, the proposed research makes the process of detecting software piracy and theft more clear and easy.

The proposed research work can be extended to a more efficient and accurate way by following some advanced technique(s) of computational mathematics. In addition, we plan to apply the current approach for detecting malicious software in the vehicles with Internet, since these can be vulnerable to some kinds of hijacking and their repercussions could provoke traffic accidents.

References

[1] Thabit, R. and B.E. Khoo, "Robust reversible watermarking scheme using Slantlet transform matrix," Journal of Systems and Software, vol. 88, pp. 74-86, (2014).

[2] Zeng, Y., F. Liu, X. Luo, and C. Yang, "Software Watermarking Through Obfuscated Interpretation: Implementation and Analysis," Journal of Multimedia, vol. 6, 2011).

[3] Liu, F., B. Lu, and X. Luo, "A Chaos-Based Robust Software Watermarking," in Information Security Practice and Experience. vol. 3903, ed: Springer Berlin Heidelberg, pp. 355-366 (2006).

[4] Collberg, C. and T.R. Sahoo, "Software watermarking in the frequency domain: Implementation, analysis, and attacks," Journal of Computer Security, vol. 13, pp. 721-755, (2005).

[5] Myles, G. and C. Collberg, "Software Watermarking Through Register Allocation: Implementation, Analysis, and Attacks," in Information Security and Cryptology - ICISC 2003. vol. 2971, ed: Springer Berlin Heidelberg, pp. 274-293 (2004).

[6] Collberg, C., E. Carter, S. Debray, A. Huntwork, C. Linn, and M. Stepp, "Dynamic path-based software watermarking," in In ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 04), pp. 1-10 (2004).

[7] Arboit, G.e., "A method for watermarking java programs via opaque predicates," in The Fifth International Conference on Electronic Commerce Research (ICECR-5), pp. 1-8 (2002).

[8] Pieprzyk, J., "Fingerprints for Copyright Software Protection," in Information Security. vol. 1729, ed: Springer Berlin Heidelberg, pp. 178-190 (1999).

[9] Collberg, C.S., C. Thomborson, and G.M. Townsend, "Dynamic graph-based software fingerprinting," ACM Trans. Program. Lang. Syst., vol. 29, p. 35, (2007).

[10] Baxter, I.D., A. Yahin, L. Moura, M. Sant'Anna, and L. Bier, "Clone Detection Using Abstract Syntax Trees," presented at the Proceedings of the International Conference on Software Maintenance, (1998).

[11] Rattan, D., R. Bhatia, and M. Singh, "Software clone detection: A systematic review," Information and Software Technology, vol. 55, pp. 1165-1199, (2013).

[12] Aiken, A., "Moss: A system for detecting software plagiarism," University of California-Berkeley. http://www. cs. berkeley. edu/aiken/moss. html (2005).

[13] Whale, G., "Identification of program similarity in large populations," Computer, vol. 33, pp. 140-146, (1990).

[14] Schleimer, S., D. Wilkerson, and A. Aiken, "Winnowing: Local algorithms for document fingerprinting," in Proceedings of 2003 SIGMOD Conference (2003).

[15] Tian, Z., Q. Zheng, T. Liu, M. Fan, X. Zhang, and Z. Yang, "Plagiarism detection for multithreaded software based on thread-aware software birthmarks," presented at the Proceedings of the 22nd International Conference on Program Comprehension, Hyderabad, India, (2014).

[16] Tian, Z., Q. Zheng, T. Liu, and M. Fan, "DKISB: Dynamic Key Instruction Sequence Birthmark for Software Plagiarism Detection," in IEEE International Conference on High Performance Computing and Communications and IEEE International Conference on Embedded and Ubiquitous Computing, pp. 619-627 (2013).

[17] al, H.T.e., "Detecting the theft programs using birthmarks," Graduate School of Information Science, Nara Institute of Science and Technology, November 2003).

[18] Lim, H.-i., "Customizing k-Gram Based Birthmark through Partial Matching in Detecting Software Thefts," in IEEE 37th AnnualComputer Software and Applications Conference Workshops (COMPSACW), pp. 1-4 (2013).

[19] Xin, Z., H. Chen, X. Wang, P. Liu, S. Zhu, B. Mao, et al., "Replacement attacks: automatically evading behavior-based software birthmark," International Journal of Information Security, vol. 11, pp. 293-304, (2012).

[20] Park, H., H.-i. Lim, S. Choi, and T. Han, "Detecting common modules in Java packages based on static object trace birthmark," Computer Journal, vol. 54, pp. 108-124, (2011).

[21] Chan, P.P.F., L.C.K. Hui, and S.M. Yiu, "Dynamic Software Birthmark for Java Based on Heap Memory Analysis," in Communications and Multimedia Security. vol. 7025, ed: Springer Berlin Heidelberg, pp. 94-107 (2011).

[22] Mahmood, Y., S. Sarwar, Z. Pervez, and H.F. Ahmed, "Method based static software birthmarks: A new approach to derogate software piracy," in 2nd International Conference on Computer, Control and Communication, pp. 1-6 (2009).

[23] Choi, S., H. Park, H.-i. Lim, and T. Han, "A static API birthmark for Windows binary executables," Journal of Systems and Software, vol. 82, pp. 862-873, (2009).

[24] Park, H., S. Choi, H.-i. Lim, and T. Han, "Detecting Java Theft Based on Static API Trace Birthmark," in Advances in Information and Computer Security. vol. 5312, ed: Springer Berlin Heidelberg, pp. 121-135 (2008).

[25] Park, H., S. Choi, H.-i. Lim, and T. Han, "Detecting code theft via a static instruction trace birthmark for Java methods," in 6th IEEE International Conference on Industrial Informatics, pp. 551-556 (2008).

[26] Lim, H.-i., H. Park, S. Choi, and T. Han, "Detecting Theft of Java Applications via a Static Birthmark Based on Weighted Stack Patterns," IEICE - Trans. Inf. Syst., vol. E91-D, pp. 2323-2332, (2008).

[27] Yang, J., J. Wang, and D. Li, "Detecting the Theft of Natural Language Text Using Birthmark," in Proceedings of the 2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 1-4 (2006).

[28] Myles, G. and C. Collberg, "Detecting Software Theft via Whole Program Path Birthmarks," in Information Security. vol. 3225, ed: Springer Berlin Heidelberg, pp. 404-415 (2004).

[29] Nazir, S., S. Shahzad, Q.U.A. Nizamani, R. Amin, M.A. Shah, and A. Keerio, "Identifying Software Features as Birthmark," Sindh University Research Journal (Science Series), vol. 47, pp. 535-540 (2015).

[30] Nazir, S., S. Shahzad, S.A. Khan, N.B. Ilyas, and S. Anwar, "A novel rules based approach for estimating software birthmark," Scientific World Journal, vol. 2015, pp. 1-8, (2015).

[31] Lee, D., Y. Choi, J. Jung, J. Kim, and D. Won, "An Efficient Categorization of the Instructions Based on Binary Executables for Dynamic Software Birthmark," International Journal of Information and Education Technology, vol. 5, pp. 571-576, (2015).

[32] Nazir, S., S. Shahzad, and S.B.S. Abid., "Selecting software design based on birthmark," Life Science Journal, vol. 11, pp. 89-93, (2014).

[33] Choi, J., Y. Han, S.-j. Cho, HaeYoungYoo, and J. Woo, "A Static Birthmark for MS Windows Applications Using Import Address Table," in Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, pp. 129-134 (2013).

[34] Park, H., H.-i. Lim, S. Choi, and T. Han, "Detecting common modules in Java packages based on static object trace birthmark," Computer Journal, vol. 54, pp. 108-124, (2009).

[35] BSA, "The Compliance Gap BSA GLOBAL SOFTWARE SURVEY," Business Software Alliance2014).

[36] Tamada, H., M. Nakamura, and A. Monden, "Design and evaluation of birthmarks for detecting theft of Java programs," in Proceedings of IASTED International Conference on Software Engineering pp. 569-575 (2004).

[37] Zeng, Y., F. Liu, X. Luo, and S. Lian, "Abstract interpretation-based semantic framework for software birthmark," Computers and Security, vol. 31, pp. 377-390, (2012).

[38] Mahmood, Y., Z. Pervez, S. Sarwar, and H.F. Ahmed, "Similarity Level Method Based Static Software Birthmarks," in High Capacity Optical Networks and Enabling Technologies, pp. 205-210 (2008).

[39] Myles, G. and C. Collberg, "K-gram based software birthmarks," presented at the Proceedings of the 2005 ACM symposium on Applied computing, Santa Fe, New Mexico, (2005).

[40] Xie, X., F. Liu, B. Lu, and L. Chen, "A Software Birthmark Based on Weighted K-gram," in IEEE International Conference on Intelligent Computing and Intelligent System (ICIS), pp. 400-405 (2010).

[41] Fukuda, K. and H. Tamada, "A Dynamic Birthmark from Analyzing Operand Stack Runtime Behavior to Detect Copied Software," in 2013 14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, pp. 505-510 (2013).

[42] Bai, Y., X. Sun, G. Sun, X. Deng, and X. Zhou, "Dynamic k-gram based software birthmark," presented at the IEEE ASWEC 2008 19th Australian Conference, (2008).

[43] Tian, Zhenzhou, et al. "Plagiarism detection for multithreaded software based on thread-aware software birthmarks." Proceedings of the 22nd International Conference on Program Comprehension. ACM, 2014.

[44] Kim, Dongjin, et al. "An effective and intelligent Windows application filtering system using software similarity." Soft Computing 20.5 (2016): 1821-1827.

[45] Chae, Dong-Kyu, et al. "Credible, resilient, and scalable detection of software plagiarism using authority histograms." Knowledge-Based Systems 95 (2016): 114-124.

[46] Daubachies, I. Ten lectures on wavelets, Vol. 61, SIAM, 1992.

[47] Siraj-ul-Islam, I. Aziz, B. Sarler, The numerical solution of second-order boundary-value problems by collocation method with the Haar wavelets, Math. Comp. Model. vol. 52 , pp. 1577-1590 (2010).

[48] U. Lepik, H. Hein, Haar wavelets with applications, Math. Eng. Springer, New York London, 2014.

[49] ZILL, D.G. and M.R. CULLEN, Differential Equation s with boundary Value Problem, 7 ed.: Brooks/Cole Cengage Learning, (2009).

[50] Nazir, S., S. Shahzad, I. Zada, and H. Khan, "Evaluation of software birthmarks using fuzzy analytic hierarchy process," in Proceedings of the Fourth International Multi-topic Conference, pp. 171-175 (2015).

[51] Nazir, S., S. Shahzad, and L.S. Riza, "Birthmark-Based Software Classification Using Rough Sets," Arabian Journal for Science and Engineering, vol. 42, pp. 1-13, (2016).

[52] "MATLAB," 7.10.0 ed. Natick, Massachusetts: The MathWorks Inc, 2010.

[53] S. Nazir, S. Shahzad, R. B. Atan, H. Farman, "Estimation of software features based birthmark", Cluster Computing,(2017),1-14.

[54] Choi, S., Park, H., Lim, H. I., and Han, T. "A static API birthmark for Windows binary executables." Journal of Systems and Software 82.5 (2009): 862-873.

[55] Schuler, D., Dallmeier, V., and Lindig, C. (2007, November). "A dynamic birthmark for Java." In Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering, pp. 274-283 (2007).