一种基于动态图编码的软件水印方案

刘嘉怡 燕雪峰
(南京航空航天大学计算机科学与技术学院 南京 211106)

摘 要 软件水印能够证明软件的相关信息。目前,软件水印算法大多基于经典的动态图软件水印算法——CT算法。该算法将水印分解为水印片段后,通过编码方案实现水印片段的嵌入。针对扩展的平面环路树(ExtendPPCT)编码方案改变了原平面环路树(PPCT)编码结构、水印隐蔽性较差以及节点易被删除攻击破坏的缺点,提出一种新的基于平面环路树(PPCT)和排序图的混合编码方式来实现成组地表达同余方程的模数和余数:PPCT枚举表达模数,对PPCT的叶子节点进行排序编码表达余数。这种成组编码方式使得嵌入软件中的水印片段减少了一半,对嵌入水印的程序的性能影响较小,水印的隐蔽性更强;并且这种编码方式不改变原PPCT的唯一外部回路,同时可抗击删减攻击。

关键词 排序编码,平面环路树,动态图软件水印,混合编码,防篡改
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.09.026

Software Watermarking Scheme Based on Dynamic Graph Coding

LIU Jia-yi YAN Xue-feng
(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

Abstract The software watermark could prove the related information of software. At present, most of the watermarking algorithms are based on the classical dynamic image watermarking algorithm——CT algorithm. After the watermark was decomposed into watermark segments, the watermarks were embedded by the coding scheme. Coding scheme of ExtendPPCT has changed the structure of PPCT, watermarked concealment is poor and nodes are easy to be removed. A new hybird encoding based on planted plane cubic tree and rank order encoding was put forward. The PPCT expresses modulus, and the leaves of PPCT sort coding expresses the remainder. This way on pair of encoding makes its segments of the watermark embedded software cut in half which have little effect on the embedded watermark and watermark hiding ability is stronger. The encoding did not change the only external loop of the original PPCT, at the same time this type of encoding could resist against reducing attack type.

Keywords Rank order coding, Planted plane cubic tree, Dynamic garph based software watermark, Hybird encoding, Tamper-resistant

1 引言

软件水印是数字水印的一个分支,是一种软件产品版权保护技术,既能用来标识软件所有者、发行方、使用时限等信息,又可以鉴别出盗版、非法复制等违法行为。随着Internet的发展,人们可以更加方便快捷地通过互联网获得需要的数字产品,但是由于法律的不完善以及监管不到位,非法获得、使用、复制等违法行为严重侵害了软件版权所有者的权益。因此,人们越来越重视软件水印的保护。Collberg和Thomborson提出了经典动态水印算法——CT算法[1],CT算法的关键思想是在程序运行时动态地在建立的拓扑图结构中嵌入水印信息。该算法先将水印信息构建成拓扑结构,再将其划分为多个子拓扑图结构,从而将子拓扑图结构隐藏在程序中,并在程序动态建立的过程中生成拓扑图从而实现水印的嵌入。由于在CT算法中别名分析和指针的引用非常困难,使得构造水印的代码不易被分析,因此可以有效防止语义保持攻击。

在CT算法中将水印分解为子水印,利用编码方案进行拓扑图的构建是算法中较为关键的部分。目前常用的编码方式有PPCT[2]、PPT[3]、排序图编码[4]、K-基数编码[5]等,排序图编码方式和PPCT编码方式均是动态图拓扑结构中的主要编码方式。二者具有以下两个特征:1)图的表示是唯一的,不存在二义性[6];2)图拓扑结构中任意两节点之间都是相通的。除以上共同点外,二者也有区别:排序图的编码效率高,但结构不稳定,容易被攻击者破坏;PPCT编码结构稳定,抗攻击能力强,被篡改后容易被及时发现,但其编码效率较低。

针对以上两种方式的优缺点,本文利用中国剩余定理分解水印,获得水印片段,即多个同余方程组,借助排序图编码

到稿日期:2016-10-28 返修日期:2017-01-17 本文受十三五重点基础科研项目(JCKY2016206B001),江苏省六大人才高峰项目(XXRJ-004),软件新技术与产业化协同创新中心资助。

刘嘉怡(1989—),女,硕士生,主要研究方向为信息安全,E-mail:651981862@qq.com;燕雪峰(1975—),男,博士,教授,主要研究方向为软件工程方法论、系统建模与仿真等。

及 PPCT 编码各自的原理及特点，提出了一种基于排序图编码及 PPCT 编码的混合编码方式，相较于传统编码方式中一个编码结构对应表达一个整数，该混合编码实现了一个编码结构表达两个整数(本文指同余方程的余数和模数)，从而实现了更快的拓扑效率和更强的检错性、容错性、抗攻击性。

2 现有编码

在介绍排序图及 PPCT 混合编码之前，先分别介绍 3 种编码方式。

2.1 排序图编码

排序图编码方式是将一组自然数序列的各种排列方式与自然数相对应，即一组序列对应一个自然数，形成一对一映射，这组序列又与一个循环链表相对应。如序列〈1,2,3,4〉对应 0，序列〈1,2,4,3〉对应 1，由全排列字典顺序依次排序，可以得到序列〈2,4,1,3〉对应 10。其对应的循环链表如图 1 所示。

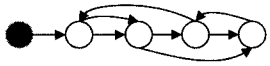

图 1 排序编码

排序图编码中，第一个节点为生成节点，生成节点只有一个指针。生成节点指向一个循环链表。链表中每个节点有左、右两个指针(最后一个节点只有左指针)，右指针指向下一个节点，左指针指向当前节点位置序号经过全排列以后对应的节点位置序号。如图 1 所示，1 号位置经全排列后对应 2，则 1 号节点左指针指向节点 2，依次类推。每个叶子节点的出度为 2，入度为 3。通过对叶子节点出、入度数目的判断实现检错。

2.2 PPCT 编码

PPCT 编码与二叉树类似，每一个节点都有两个指针域，即左、右指针，同时在根节点上添加一个生成节点。叶子节点的左指针指向自己左边最近的叶子节点，左起第一个叶子节点的左指针指向生成节点，生成节点的左指针指向右起第一个叶子节点，所有叶子节点的右指针指向节点本身。从而生成节点及叶子节点构成了一个循环链表。图 2 示出了具有 4 个叶子节点的 PPCT 枚举编码[7]。

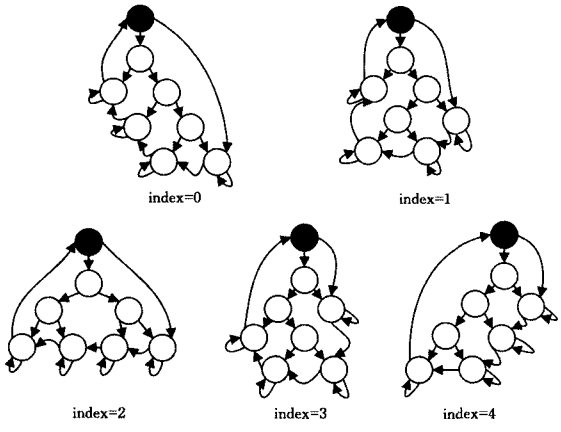

图 2 4 个叶子节点的 PPCT 枚举编码

具有 n 个叶子节点的 PPCT 可以表示的整数范围为 0～C(n)－1。假设 int(T)表示 PPCT 结构 T 的 CLOC 索引，T. left表示 T 的左子树，T. right 表示 T 的右子树，L 表示左子树的叶节点数，R 表示右子树的叶节点数。Yong He 对 Palsberg J 的公式做了修正，PPCT 解码为水印数据的正确公式为：

$$\min\_int(L,R)=\sum_{1<i<L}(C(L-i)\times C(R+i))$$

叶子节点的入度为 3，出度为 2。由图 2 可知，PPCT 具有两个纠错属性[8]：1)可检测的二维性。对于任意内部节点，该节点的右子树的最左孩子的左指针链接到其左子树的最右孩子。因此，其可对只有一个结点的追加/删除攻击进行成功检测并准确纠错。2)抗边翻转攻击能力。PPCT 具有唯一的外部环路，该环路在线性时间复杂度内可以遍历，针对边翻转攻击，PPCT 可以及时感知并修复[9]。

2.3 ExtendPPCT 与 K-基数混合编码

ExtendPPCT 混合编码的基本思想是：将 ExtendPPCT 树结构中的所有节点按层次遍历一一映射成自然数，并运用指针对自然数序列重排列，从而实现混合编码。如图 3 所示，用 ExtendPPCT 与 K-基数编码实现了编码整数 196223。

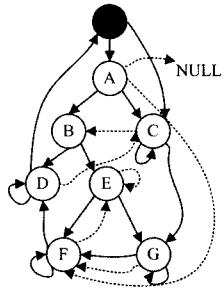

图 3 ExtendPPCT 编码

如图 3 所示，按照层次遍历二叉树的方法将 A～G 一一映射成 1～7，即对应 K-基数中的节点顺序，按照 K-基数编码规则对整数 196223 进行编码，有：$196223=317\times619=0\times8^6+5\times8^5+7\times8^4+7\times8^3+1\times8^2+7\times8^1+7\times8^0$。当叶子节点数为 n 时，编码范围是 $0\sim2n^{(2n-1)}-1$，这种编码方式在 PPCT 结构的基础上将每一个节点都赋予第三个指针域 next，用来实现 K-基数编码[10]。

3 排序图与 PPCT 混合编码

由上文的编码方式可知，ExtendPPCT 与 K-基数混合编码方式将原本不参与编码的非叶子节点进行了编码，当某一节点受到增加、删除、篡改攻击时，无法通过节点的入度、出度数目判断该节点是否受到攻击，这导致该混合编码鲁棒性较差。同时，传统的编码方式中一个拓扑子图仅能编码一个整数，当水印 W 为较大整数时，会造成分解的水印片段过多，必然导致嵌入软件中的拓扑子图增多，水印的隐蔽性降低[11]。针对以上缺点，提出 PPCT 与排序图的混合编码方式。

3.1 水印分解

经典 CT 算法利用中国剩余定理将水印 W 分解为 W_1，W_2，…，W_n[12]，水印的嵌入与检测过程复杂，本文主要对经典 CT 算法的前三步进行研究，即：1)水印 W 的选取；2)某个双射 f 将 W 与某个拓扑结构对应；3)将拓扑图划分为适合的拓扑子图。在构造拓扑图的编码方式的选择上提出了自己的方法并加以验证[13]。由 CT 算法可知，只要获得足够多的正确

的片段即可恢复 W。W 只由合法的版权拥有者所拥有，因而只有提取出正确的水印并与嵌入者所拥有的水印相吻合时，才能证明软件的归属。水印分解算法主要在水印 W 的选取及分解上做了详细的说明。

算法具体如下[14]：

选取大整数 W，同时选择 m 个素数 $p_1, p_2, \cdots, p_m$，其中 $p_1, p_2, \cdots, p_m$ 均小于 W。

将 W 分解成 $[m\times(m-1)]/2$ 个，其中 p_i, p_j 为从 m 个素数中随机选取的两个素数，计算得出 $[m\times(m-1)]/2$ 个余数，这样可以获得 $[m\times(m-1)]/2$ 个形如 $x_k \bmod p_i\times p_j$ 的式子。构造的同余方程组如下所示：

$$\begin{cases} W\equiv x_1 \bmod p_{11}\times p_{12} \\ \quad\vdots \\ W\equiv x_k \bmod p_{ki}\times p_{kj} \end{cases}$$

每一个 $x_k \bmod p_i\times p_j$ 对应一个拓扑子图，利用下节所提混合编码方式实现拓扑子图的构建。提取水印时获取每一个拓扑子图并还原成相对应的 $x_k \bmod p_i\times p_j$，当获取足够多的式子后即可由中国剩余定理计算还原出水印 W。

由以上分析可以看出，提取水印时并不需要获得并还原所有的同余方程，只要保证所还原的方程中包含全部的素数 p 即可。最好的情况是还原 $[m/2]$ 个同余方程即可获得全部的素数 p。此外，如果某个同余方程所对应的拓扑子图被篡改，即 p_i, p_j 的乘积无法拆分成两个素数的乘积，此时发现拓扑子图已被篡改，从而舍弃该同余方程，因此即便某些水印片段丢失或被篡改，依然有机会获得正确的水印。因此该算法具有自动剔除被篡改的水印片段的能力。

水印分解算法实现了将水印 W 分解成片段。下一节将阐述拓扑子图的构建方法。

3.2　PPCT 与排序图混合编码

假设水印 W 的一个片段的同余方程是 $x_k \bmod(p_i\times p_j)$，此时需要通过一个编码结构实现同时存储一组整数，即一个拓扑子图存储一组数 $(x_k, p_{ki}\times p_{kj})$[15]。规定：以 PPCT 原结构作为混合编码结构基础，该 PPCT 结构对应表达一组整数 $(x_k, p_{ki}\times p_{kj})$ 中的 x_k，即同余方程中的余数。同样，对 PPCT 的叶子节点进行排序图编码来表达整数 $(p_{ki}\times p_{kj})$，即同余方程组中的模数。从而便在一个编码结构中实现了同时表达一组整数。对混合编码节点进行如下具体修改：

(1)每一个节点有 3 个指针域，分别是 left，right 和 next。

(2)对于非叶子节点，left 和 right 分别指向该节点的左、右孩子节点。next 指向 null。

(3)对于叶子节点，left 指针指向其左边最靠近该点的叶子节点。right 指针用于排序图编码。next 指针指向叶子节点本身，表示该节点为叶子节点，生成节点左指针指向右起第一个叶子节点，右指针指向根节点，next 指向 null。

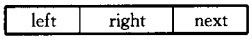

图 4　混合编码节点的数据结构

同余方程的余数和模数分别用 PPCT 编码及排序图编码方式进行编码。

需要注意的是，因为所要表达的水印片段实际上是一对整数，所以在构建编码结构时必须考虑分别用相应编码方式所能表达的整数范围需满足这对整数中的任意一个。即假设 a, b 分别表示余数和模数按照相应编码方式编码所需的最少叶子节点数[16]。max 则表示 (a, b) 中较大的一个。

当取值为 (a, b) 中的较大值时，有时会出现如下情况：PPCT 编码表示余数所需的叶子节点数 a 大于排序编码表示模数所需叶子节点数，这会导致在叶子节点进行排序编码时有多余的叶子节点不会被使用，规定排序编码时不被使用的叶子节点右指针为空[17]。

举例：$4238\equiv 4 \bmod 2\times 3$，由 PPCT 枚举表示整数的范围可知，$a\geqslant 4$，取 $a=4$。由排序图表示整数范围可知，$b!-1\geqslant 6$ ($b!$ 表示 b 的阶乘)，解得 $b\geqslant 4$，取 $b=4$。因此 max$=4$。根据以上编码规则，4 mod(2×3)的混合编码如图 5 所示(由于非叶子节点 next 指针均为 null，故本图省略非叶子节点的 next 指针)。

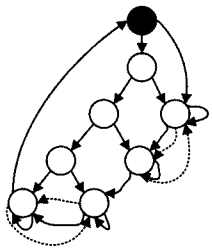

图 5　4 mod(2×3)的 PPCT 与排列图混合编码

4　实验性能对比分析

4.1　抗攻击性

为验证本文混合编码的抗删减攻击能力，借助 SandMark 平台进行仿真实验。选取已嵌入水印的 Calculator 和 Notepad++进行删减攻击实验，同时将 ExtendPPCT 与 K-基数混合编码方案作为对照组。

模拟攻击者在对程序代码进行删减攻击时，每次删除一定数量的水印片段，统计 1000 次实验中恢复正确水印的百分比。

如图 6 所示，横坐标表示删除水印片段占总水印片段数量的百分比，纵坐标表示在进行 1000 次删除攻击的实验中，能够根据剩余的水印片段正确提取原始水印的次数占总实验次数的百分比。图 6 中有两组对比实验，$m=20$ 和 $m=40$ 分别表示水印片段较少和较多两种情况。

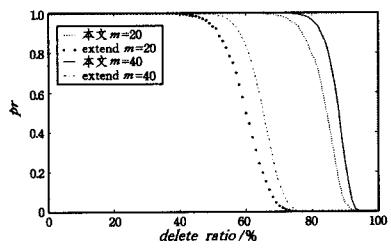

图 6　两种编码方式被删减攻击攻击后提取水印的情况

对比可知，随着删除水印片段比例的增加，两种编码方案提取水印的可能性均呈现逐渐下降的趋势，在删除 70%左右的水印片段后，extend 混合编码基本上已经没有恢复原始水印的可能，但本文编码方案在删除 80%的水印片段时依然有

较大的可能性提取到正确的原始水印。因此，本文编码方案抗击删除攻击的能力明显优于 extend 混合编码。究其原因，如下所述。

将水印 W 分解为若干水印片段 $W_1, W_2, W_3, \cdots, W_n$ 后，以拓扑子图的方式嵌入到软件中，由于水印片段由模数、余数两部分构成，采用传统编码方式时需要按照一定的关联性将二者嵌入程序，一旦攻击者破坏了该关联性，删减了某一表示模数或余数的水印片段，就会使得在提取水印的过程中无法实现二者匹配，则水印片段被破坏[18]。本文编码方案实现成组地存储水印片段，使一对模数、余数不再由两个相关联的拓扑子图表达，从而避免了两个水印片段之间的关联性被破坏后导致无法被正确提取的缺点。同时，水印分解算法本身已经加入了冗余水印片段，当部分水印片段被攻击者删除时，并不影响整个水印的恢复[19]。

对于某一水印片段的篡改攻击，篡改指针将导致提取模数出现错误。下面举例说明过滤错误片段的过程。

已知提取的部分水印片段如下：

$$\begin{cases} 25 \equiv 1 \bmod 6 \\ 25 \equiv 5 \bmod 10 \\ 25 \equiv 12 \bmod 1 \end{cases}$$

由于是从素数集合中任意选择两个数并将二者乘积作为模数，因此当提取水印时模数必然且只能拆分成唯一的一对素数的乘积。由此可以断定第三组同余方程模数提取错误，拓扑子图被篡改，因此舍弃该方程。

4.2 构建拓扑的效率

目前的编码方式侧重扩大编码范围，但却忽视了将水印 W 分解成众多水印片段后势必导致需要建立更多的拓扑子图[20]，从而使得建图过程的耗时增加而且不利于隐藏。

如图 7 所示，本文通过混合编码实现成对地表达水印片段。采用中国剩余定理分解水印，余数和模数总是成对出现，若余数个数为 t 时，余数、模数总个数为 $2\times t$，此时需要建立 $2\times t$ 个拓扑子图来分别表达每一个余数和模数，并且二者之间必须建立某种关联，以便在提取水印时能将二者恢复为原水印[21]。本文中的混合编码通过一个拓扑子图表达一对整数，从而将拓扑子图数量减少一半。同时各个拓扑子图之间无序，即便被攻击者扰乱顺序仍然可以提取出正确的水印信息[22]。

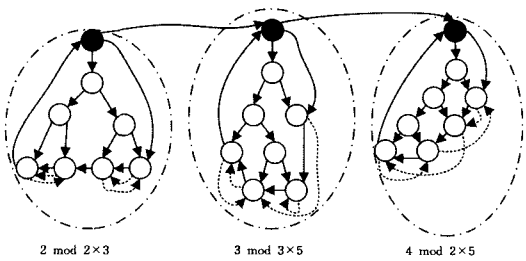

图 7 多个拓扑子图连接(next 指针未标明)

利用 Java 语言实现 ExtendPPCT 和本文提出的混合编码方案。如图 8 所示，横坐标为构造拓扑子图的数目，第一个折线图的纵坐标表示构造当前数目拓扑子图所需时间，第二个折线图的纵坐标表示两种编码方式拓扑时间的比值(本文编码方案/ExtendPPCT)，实验数据为 1000 次实验所得的平均值。

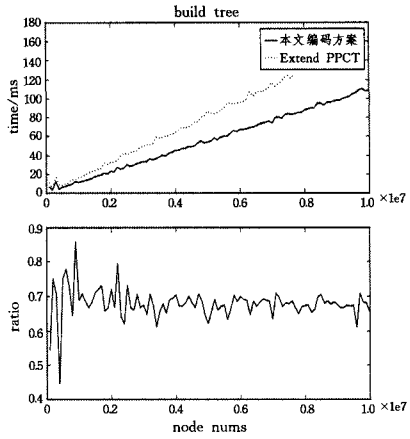

图 8 构造相同节点时两种编码所用时间及比值

由图 8 可知，随着拓扑图数目的增加，本文编码所需时间明显少于 ExtendPPCT 所需时间，并且二者之间的差值呈增大趋势。由第二个折线图可知，二者时间比值在 0.4～0.9 之间波动比较大，但随后趋于稳定，并且两个阶段的比值均在 0.6～0.8 之间。这表明本文的混合编码方式可以有效减少编码时间，提高编码效率，并且随着编码数目的增加，二者比值趋于稳定。

4.3 性能过载分析

水印的嵌入必然会对原程序产生影响，包括程序对运行空间的影响以及程序对执行时间的影响[23]，当这种影响在用户可接受的范围之内时，则认为该水印方案是一种好方案；否则，这种方案不可取。本文将通过实验数据来对比分析水印嵌入前、嵌入后对程序性能产生的影响。

在实验中为了便于分析，分别选取 Calculator 和Notepad++ 两个不同大小的程序进行实验，其中水印对程序执行时间的影响取 10 次实验的平均值。实验数据如表 1、表 2 所列。

表 1 水印对程序大小的影响

测试程序	原始大小/kB	嵌入水印后的大小/kB	增长比例/%
Calculator	1.20	1.26	5.00
Notepad++	80.00	83.00	3.70

表 2 水印对程序执行时间的影响

测试程序	原始执行时间/ms	嵌入水印后的执行时间/ms	增长比例/%
Calculator	2.2	2.5	13.6
Notepad++	14.0	14.6	4.2

通过以上实验数据可知，PPCT 与排序图混合编码方案对原程序产生的影响在可接受范围内[24]。

结束语 通过充分研究各种编码方案，提出了一种基于排序图和 PPCT 的混合编码方案，该方案在原 PPCT 的基础上对叶子节点进行排序图编码，实现了成对地表达水印片段；利用中国剩余定理分解水印，同时加入冗余片段。当攻击者对某个水印片段对应的拓扑子图中某一节点进行攻击时，可以借助 PPCT 的稳定性及生成节点和叶子节点构成的循环链表的检错和纠错能力实现自我修复；当攻击者对部分水印片段进行删除攻击时，仍可以由水印分解算法的冗余性及纠错

能力进行修正，从而恢复正确的原始水印；同时，嵌入水印对原程序性能产生的影响在用户可接受的范围内。但由于叶子节点数需要同时满足两种编码方案，当两种编码方案所需叶节点不同时，可能会造成叶子节点的冗余。在后续的研究中将对其进行改进。

参 考 文 献

[1] YU T, YANG J. An Encoding Scheme of Dynamic Software Watermarking[J]. Computer & Communications, 2006, 24(1): 76-79. (in Chinese)
虞涛，杨杰. 一种动态图软件水印技术编码方案[J]. 交通与计算机，2006，24(1)：76-79.

[2] WANG H J, SHA Z L, HAN A C. Modeling Hybrid Scheme Based on PPCT and Cardin k [J]. Computer Engineering and Applications, 2010, 46 (25): 109-111. (in Chinese)
王慧娇，沙宗鲁，轩爱成. 基于 PPCT 和基数 k 的动态图混合编码方案[J]. 计算机工程与应用，2010，46(25)：109-111.

[3] LIU J R, QIN Z, PENG C. Improved dynamic watermarking coding scheme [J]. Application Research of Computers, 2011, 28 (2): 720-723. (in Chinese)
刘建蓉，秦拯，彭程. 改进的动态图水印技术编码方案[J]. 计算机应用研究，2011，28(2)：720-723.

[4] CHEN Y L, YANG Q X, CHEN G X. Tamper-resistant Dynamic Watermarking Scheme Based on Improved KPPCT Coding [J]. Microelectronics and Computer, 2013(11): 35-38. (in Chinese)
陈艳琳，杨秋翔，陈够喜. 基于改进的 KPPCT 编码的防篡改动态图水印方案[J]. 微电子学与计算机，2013(11)：35-38.

[5] LI B. Research on Watermarking Algorithm Based on Anti-tampering Dynamic Graph [D]. Zhengzhou: Zhengzhou University, 2013. (in Chinese)
李斌. 基于防篡改的动态图软件水印算法研究[D]. 郑州：郑州大学，2013.

[6] HAN J J. Research on Software Watermarking Based on Dynamic Graphs[D]. Jilin: Jilin University, 2012. (in Chinese)
韩敬敬. 基于动态图的软件水印研究[D]. 吉林：吉林大学，2012.

[7] CHENG J, SONG Y. Dynamic map based on PPCT structure software watermark protection[C]// World Automation Congress (WAC). IEEE, 2012: 133-136.

[8] YIN K, LIU Y, SUN H, et al. Concurrently controlled multithreads based dynamic graphical software watermarking scheme[J]. International Review on Computers & Software, 2012, 67(28): 23-26.

[9] WANG Y. Improved PPCT hybrid coding scheme[J]. Computer Engineering & Applications, 2012, 49(26): 39-44.

[10] WANG G. Software Copyright Protection Based on Quadratic Residue[J]. Computer Engineering, 2008, 34(18): 196-198.

[11] ZHOU Q L, BIN L I. Double Software Watermark Scheme Based on Tamper-proofing[J]. Computer Engineering, 2013, 132 (64): 66-68.

[12] CHEN Y L. Based on the dynamic map software watermarking technology research [D]. Taiyuan: North University, 2014. (in Chinese)
陈艳琳. 基于动态图的软件水印技术研究[D]. 太原：中北大学，2014.

[13] HURD B H, MAC C, JOEL S, et al. WaterRPG: A Graph-based Dynamic Watermarking Model for Software Protection [J]. Eprint Arxiv, 2014, 40(1): 129-148.

[14] ZHANG D, ZHOU Q L. Watermark recoverable software watermarking scheme [J]. Application Research of Computers, 2014, 12(2): 104-106.

[15] MCKINLEY T J, SEDER P A, RODRIGUEZ T F. Watermark systems and methods[OL]. http://www.google.com/patents/US7123740.

[16] CHENG C, ZENG R. Improved Watermarking Scheme Based on PPCT Coding Software [J]. Computer Knowledge and Technology, 2016, 12(12): 62-69. (in Chinese)
程成，曾嵘. 一种改进的基于 PPCT 编码软件水印方案[J]. 电脑知识与技术，2016，12(12)：62-69.

[17] WANG G. A hybrid software watermarking based on quadratic residue [J]. Journal of Qingdao University (Engineering & Technology Edition), 2007, 22(3): 38-43. (in Chinese)
王刚. 一种基于二次剩余的混合软件水印研究[J]. 青岛大学学报(工程技术版)，2007，22(3)：38-43.

[18] YANG J L. Recent Development of Software Watermark[J]. Computer Engineering, 2007, 33(17): 168-170.

[19] ZHANG H C. Software watermarking algorithm research and implementation [D]. Nanjing: Nanjing University of Aeronautics and Astronautics, 2012. (in Chinese)
张海超. 软件水印算法的研究与实现[D]. 南京：南京航空航天大学，2012.

[20] WANG Y M, CHE S B. An overview of software watermarking and its research [J]. Journal of Computer Applications and Software, 2015, 12(4): 6-10. (in Chinese)
王叶茂，车生兵. 软件水印及其研究现状概述[J]. 计算机应用与软件，2015，12(4)：6-10.

[21] LI S Z, ZHOU S Y, LIU J H. Dynamic watermarking software watermarking scheme based on Chinese remainder theorem [J]. Software Journal, 2008, 24(8): 191-193. (in Chinese)
李淑芝，周诗源，刘京华. 基于中国剩余定理的动态图软件水印方案[J]. 软件导刊，2008，24(8)：191-193.

[22] SONG G W, DUAN Y F, PAN F. A New technology and implementation of stealth coding based on water marking[J]. Journal of Computer Applications and Software, 2017, 42(2): 226-230. (in Chinese)
宋广为，段云飞，潘锋. 一种新的基于水印技术的隐形编码技术及实现[J]. 计算机应用与软件，2017，24(2)：226-230.

[23] TAN Y K. Research and Implementation of Software Watermarking Based on Dynamic Graphs [J]. Jilin: Jilin University, 2011. (in Chinese)
谭永坤. 基于动态图的软件水印研究与实现[D]. 吉林：吉林大学，2011.

[24] ZHANG X J. Improved dynamic software watermarking study [D]. Wuhan: Wuhan University of Technology, 2014. (in Chinese)
张学佳. 一种改进的动态图软件水印研究[D]. 武汉：武汉理工大学，2014.