基于正交编码的大数据纯文本水印方法

李兆璨[1,2] 王利明[2] 葛思江[2,3] 马多贺[2] 秦 勃[1]

(中国海洋大学信息科学与工程学院 山东 青岛 266100)[1] (中国科学院信息工程研究所 北京 100093)[2]

(中国科学院大学网络空间安全学院 北京 100049)[3]

摘 要 数据泄露是大数据应用面临的重要挑战之一。数字水印技术是实现数据追踪和版权保护的有效手段。当前的数字水印方法主要针对终端用户的多媒体文件流转场景,如图像、音视频等,缺少面向大数据环境的文本数据泄露防护的数字水印研究。文中提出了一种基于正交编码的大数据纯文本水印方法,该方法通过编码将明文水印转换为二进制字节流,设计基于行散列值和基于行序置换的正交编码水印方法。首先对二进制水印串分段,按照每行内容的散列值计算待嵌入水印段号,将对应水印段按照自定义规则转换为不可见字符串后嵌入到文本行末;再调整行序,使得每行内容的散列值与加入标志位的二进制水印串对应,以此将水印嵌入大数据纯文本中。水印提取方法为嵌入方法的逆过程。所提方法能够抵抗大数据环境下复杂数据行序变换运算等操作对水印的破坏,同时通过嵌入脆弱水印来达到文本篡改检测的效果。基于所提方法设计并实现了一个大数据纯文本水印系统,采用 Spark 分布式处理架构来解决海量文本的水印嵌入和提取性能问题,达到了对数据泄露快速追踪溯源的目的,提高了大数据的安全性。实验和理论分析证明,该方法具有较好的水印容量性能和良好的隐蔽性,同时能够抵御多种内容攻击;由于纯文本没有格式,格式攻击对该方法无效,其具有良好的鲁棒性。

关键词 数字水印,纯文本,正交,大数据,追踪溯源

中图法分类号 TP309.2 **文献标识码** A **DOI** 10.11896/jsjkx.181001972

Big Data Plain Text Watermarking Based on Orthogonal Coding

LI Zhao-can[1,2] WANG Li-ming[2] GE Si-jiang[2,3] MA Duo-he[2] QIN Bo[1]

(College of Information Science and Engineering, Ocean University of China, Qingdao, Shandong 266100, China)[1]

(Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China)[2]

(College of Cyberspace Security, University of Chinese Academy of Sciences, Beijing 100049, China)[3]

Abstract Data leakage is one of the biggest challenges for big data applications. Digital watermarking is an effective way for data tracking and copyright protection. However, the current digital watermarking method is mainly focus on multimedia file, such as images, audio and video files. There are little digital watermarking methods for data protection in the big data environment. Therefore, this paper proposed a plain text watermarking method based on orthogonal coding for big data. First, the plain text watermark is converted into a binary byte stream by coding. The orthogonal watermarking method based on row hash value and row-sequence permutation are designed. The binary watermark string is divided into segments and numbers. The watermark segment number to be embedded is calculated according to the hash value of each line of content, and the corresponding watermark segment is converted into an invisible string which is embedded to the end of line. Then, the line order is adjusted so that the hash value of each line corresponds to the binary watermark string with the flag added, which achieves the embedding of the watermark. Watermark extraction method is the inverse process of the embedding method. It can resist the destruction of watermark by operations such as replacement operation for row order in big data environment, and achieve the effect of text tampering detection by embedding fragile watermarks at the same time. Based on the proposed method, a big data watermarking system was designed and implemented. Spark was adopted to solve the problem of watermark embedding and extraction performance of massive texts, which can quickly trace the source of data leakage and improve the security of big data. Experimental and theoretical analysis prove that the proposed method has better watermark capacity performance and good concealment. At the same time, it has strong robustness since it can resist multiple content attacks and format attacks.

Keywords Digital watermark, Plain text, Orthogonal, Big data, Traceability

---

到稿日期:2018-10-23 返修日期:2019-04-26 本文受国家重点研发计划(2017YFB1010000)资助。

李兆璨(1994-),女,硕士生,主要研究方向为大数据安全;**王利明**(1978-),男,博士,副研究员,CCF 会员,主要研究方向为大数据安全、网络安全等,E-mail:wangliming@iie.ac.cn(通信作者);**葛思江**(1994-),女,硕士生,主要研究方向为云安全;**马多贺**(1982-),男,博士,副研究员,CCF 会员,主要研究方向为云安全、网络与系统安全等;**秦 勃**(1964-),男,博士,教授,主要研究方向为图形图像处理、高性能计算、机器学习。

1　引言

随着全球信息化的不断推进以及互联网的高速发展，数据作为重要的生产因素已经渗透到各个领域。人们对海量数据的挖掘和运用，标志着大数据时代的到来。大数据平台中的文件可能包含用户隐私、敏感数据、商业机密等信息，如果文件被恶意用户窃取后传播到公开网络或平台，则会暴露用户隐私，泄露商业机密，引起版权纠纷，从而造成严重后果，因此通过对平台中的文件进行隐私保护并保证文件泄露后的可追溯性成为研究的焦点。

2010 年，Apache Hadoop 组织将大数据定义为："普通的计算机软件无法在可接受的时间范围内捕捉、管理、处理规模庞大的数据集。"大数据的常见特点包括：巨大的数据体量(Volume)、繁多的数据类型(Variety)、快速的数据处理(Velocity)和超低的价值密度(Value)。其中，繁多的数据类型是指，相比易于存储的结构化数据，大数据平台中的非结构化数据和半结构化数据(如文本、图像、多媒体等信息)的占比越来越大。

随着大数据技术的不断发展，许多传统的信息安全技术受到了挑战。在大量的数据产生、收集、存储和分析的过程中，既会涉及一些传统的安全问题，也会涉及一些新的安全问题，并且这两类问题会因数据规模、处理过程、安全要求等因素而被不断放大。大量事实表明，若大数据未被妥善处理，则会对用户的隐私造成极大的侵害。依据传统安全策略，数据经过匿名化处理后就可以被公开发布了。但在实际的互联网应用中，用户数据的收集、存储、管理和使用等均缺乏规范标准，未能得到有效监督，仅靠企业的自律是不够的，用户对自己隐私信息的用途无从得知[1]。

为了解决数据的隐私保护和溯源追踪问题，数字水印(Digital Watermarking)作为一种重要的技术手段，一直以来被学术界和工业界广泛研究和使用。数据水印技术是将一些标识信息(即数字水印)直接嵌入数字载体(包括多媒体、文档、软件等)中或是间接表示(修改特定区域的结构)，且不影响原载体的使用价值，也不容易被探知和再次修改。数字水印是保护信息安全、实现防伪溯源和版权保护的有效方法，是信息隐藏技术研究领域的重要分支。

当前，数字水印的研究大多集中在图像数据、音频数据、视频数据，对文本水印的研究相对较少。而大数据平台中的很大一部分数据是由文本形式存储的，如日志文件、交易记录、推荐记录等，因此，大数据环境下的文本水印方案引起了越来越广泛的关注，成为了信息安全的热门研究领域。

在大数据平台和大数据计算中，大量的数据为纯文本数据，而在纯文本中嵌入水印较为困难，这主要是因为纯文本有其特殊性。

(1)广义的文本是由内容和格式构成的，由于对文档内容的表现方式不同，因此文本文档的格式也不尽相同，纯文本有 TXT 和 CSV 等多种格式。

(2)各种格式的文件通常可以互相转换，甚至可以直接抽取文件中的纯文本内容，这样原来嵌入在格式中的水印信息便荡然无存了。

(3)WORD 和 PDF 等富文本由内容和结构组成，可以将水印嵌入结构中；而纯文本没有文档结构，仅由字符组成，每个字符都有固定的编码，没有可以嵌入水印信息的多余空间。

(4)图像、音频、视频对象中存在大量的冗余位，在不引起知觉变化的前提下可以将冗余位删除或者进行任意的修改替换，水印可以隐藏其中。而对于文本，需要结合文本内容进行删除或修改才不会改变文章原意。

针对大数据纯文本的上述特点，本文提出了一种基于正交编码的大数据纯文本水印方法。该方法通过在行尾嵌入不可见字符及改变行序的方法，将水印嵌入纯文本中，其具有不依赖原始记录的排列顺序，所需的水印空间较小，适用于水印空间较小的纯文本数据等优点，同时正交编码方法兼顾了水印的隐蔽性和健壮性。基于 Spark 平台实现的本文系统，满足大数据性能需求和分布式处理要求。

2　相关研究

当前大数据安全面临的挑战性问题主要有大数据隐私保护和数据泄露防护等。针对上述安全挑战，产生了大数据安全关键技术，数字水印技术是其中一个重要的研究方向。

近年来，数字水印技术研究取得了很大的进展，典型的数字水印技术主要包括空域算法、频域算法、压缩域算法等。这些算法主要针对图像数据、音频数据、视频数据，针对文本的数字水印算法相对较少。目前常用的文本水印技术主要包括以下几类。

(1)基于文档结构的文本水印方法。这类算法主要适用于具备设置文本排版格式功能的文档，如 WORD，PPT 和 PDF 等。Brassi 等[2-3]提出了 3 种著名的文本水印编码技术，通过文档自身的特点进行文档结构微调编码，包括文本的行间距编码、字间距编码和特征编码，从而实现水印信息的嵌入。文献[4]提出通过改变 WORD 中字体颜色的 RGB 分量低位数值来嵌入水印。除此之外，还可以改变文本字体或字符流形来嵌入水印信息[5]。通过这类方法嵌入的水印很难用肉眼发现，并且嵌入空间较大，但是针对文本的部分拷贝或修改可能致使水印信息不完整，无格式拷贝或格式修改将会导致水印完全被去除[6]。

(2)基于自然语言的文本水印方法。语言存在多种表述方式，通过自然语言处理技术，可以在不改变原文意思的情况下，对部分文本实施等价信息替换和语序修改等操作，从而实现水印信息的隐藏[7]。目前，自然语言文本数字水印方法主要有基于句法结构和基于语义两类[8-9]。文献[10]和文献[11]分别是基于句法结构和基于语义的典型水印方法。此类方法具有极高的隐藏性，不依赖文本格式和编码，虽然最新的技术已经可以还原原文[11]，但实现过程极为复杂。此外，文本的语言多种多样，包括英文、汉语、阿拉伯语等，目前的许多文本水印算法都是针对英文或汉语的特有语法设计而生，不具有语言的普适性，而适用于任何语言文本的数字水印方法的研究还尚未展开[12]。

(3)不可见编码算法。不可见编码可细分为不可见字符编码、附加空格编码等方案，适用于非格式化文本，即纯文本。不可见字符编码通过在文本中插入不可打印的 ASCII 字符或者不可见 Unicode 控制字符等来存储水印信息。文献[13-

14]提出了在网页文本中嵌入零宽度控制字符来嵌入水印信息。张震宇等[15]在其基础上提出在文本句号前嵌入不可见Unicode控制字符的方法,该方法适用于任意Unicode编码格式的文本。附加空格编码算法一般将信息编码隐藏在字处理系统的断行处,行尾或字间是否有空格和制表符在视觉上难以察觉[16-17],在提取时可通过不可见编码的有无、种类及数目进行解码。文献[18]通过改变空格的字号及上下标属性来表示二进制序列,从而将水印嵌入WORD文档中。

现有的数字水印方案能够为大数据提供可信性保护,但大数据平台的分布式数据处理机制,使得在对大数据文本文件进行处理后无法保证数据顺序与原有顺序一致。目前已有的水印嵌入与提取方法严格依赖原始数据的记录顺序,不适用于大数据平台数据的水印处理,同时削弱了水印的隐蔽性。与数据顺序无关的水印方法需要在水印记录中添加指示信息,增加了水印容量开销,不适用于冗余空间有限的纯文本数据。因此,满足数据量巨大的水印方案还需进一步研究与探讨。

本文提出的文本水印方法,针对大数据环境下的大批量纯文本文件进行水印嵌入,嵌入方法不依赖于原始记录的行序,适用于大数据水印处理,且不在水印记录中添加指示信息,一定程度地减小了水印容量。

3 基于正交编码的大数据纯文本水印方法

纯文本是大数据平台存储和计算中使用得最广泛的数据类型,常见的有以txt和csv为后缀的文件以及无后缀的log文件等,纯文本之间可以相互转换。其中,逗号分隔值(Comma-Separated Values,CSV)文件以纯文本形式存储表格数据(数字和文本)。CSV文件示意图如图1所示,其由任意数目的记录组成,记录间以某种换行符分隔;每条记录由字段组成,字段间的分隔符是其他字符或字符串,最常见的是逗号。通常,所有记录都有完全相同的字段序列。CSV文件由于自身的内容不依赖于行序,符合大数据平台的分布式数据处理机制,因此是一类常见的大数据纯文本格式。本文以CSV文件为例来分析介绍水印算法。

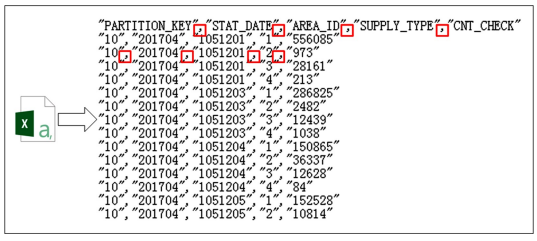

图1 CSV文件格式样例

Fig. 1 Sample format of CSV file

3.1 基于正交编码的水印方法模型

本文针对大数据平台纯文本文件的无序性,提出了一种基于正交编码的大数据纯文本水印方法。正交编码是指本文采用的两种水印嵌入和提取方法(基于行散列值和基于行序置换的水印方法)是相互正交的。其中,基于行散列值的水印嵌入方法在纯文本文件的文本行末嵌入不依赖于行序的水印段,相当于添加了一列内容,类比为对列的改变;基于行序置换的水印嵌入方法的结果为改变行序。这两种方法是解耦合的,不会相互影响,即两种方法相互正交,称为正交编码方法。

本文的水印方法模型可以使用传统水印模型表示,定义六元组(XS,WS,KS,G,E,D),其中,XS代表所要保护的纯文本文件XP的集合,WS代表所有可能水印信号W的集合,KS是水印密钥K的集合,G表示水印生成算法,E表示水印嵌入算法,D表示水印提取算法。

$$G:W=G(I,K) \tag{1}$$

式(1)表示水印生成算法G。将潜在泄密者即下载文件的人提供的个人信息、系统自动生成的时间戳信息串联成固定长度的明文字符串I,并采用ASCII编码或Unicode编码对该明文字符串进行编码后生成由0和1构成的二进制水印字符串。

$$W=\{w(k)\mid w(k)\in B,k\in W^d\} \tag{2}$$

其中,W^d表示维数为d的水印信号域,水印信号可以是二值形式$B=\{0,1\}$,或$B=\{-1,1\}$,也可以是高斯噪声形式。本文算法采用$B=\{0,1\}$的形式。

$$E:XS\times WS\rightarrow XS,XP_w=E(XP_0,W) \tag{3}$$

其中,XP_0代表原始的数字产品,XP_w代表嵌入水印后得到的数字产品。

式(3)表示将生成的二进制水印信号W嵌入纯文本文件XP_0中的嵌入算法E,水印嵌入模型如图2所示。本文采用正交编码方法实现水印嵌入,基于行散列值的水印嵌入方法在文本行末嵌入对应不可见水印段,基于行序置换的水印嵌入方法,根据随机数生成器生成的伪随机数选取起始行号进行行置换,从而完成水印嵌入。

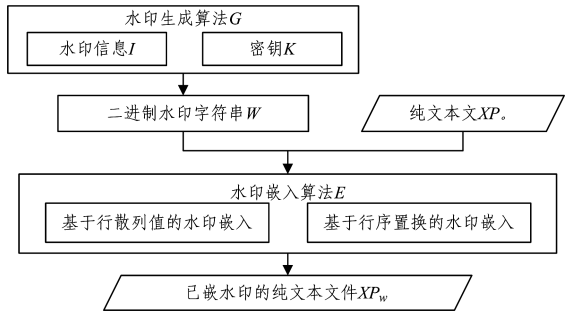

图2 水印嵌入模型

Fig. 2 Watermark inserting model

$$D:XS\times KS\rightarrow\{0,1\}$$

$$D(XP,K)=\begin{cases}1, & \text{如果 } XP \text{ 中存在 } W(H_1)\\ 0, & \text{如果 } XP \text{ 中不存在 } W(H_0)\end{cases} \tag{4}$$

式(4)表示水印提取算法D,其中H_1和H_0代表二值假设,分别表示水印的有和无。提取过程为嵌入的逆过程,水印提取模型如图3所示。

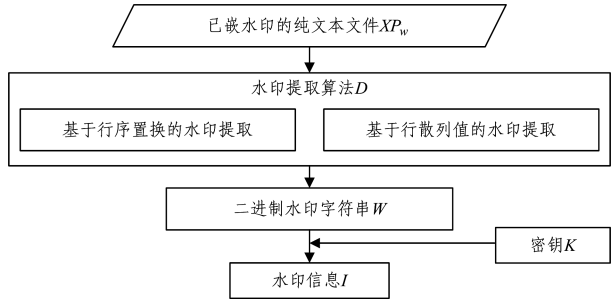

图3 水印提取模型

Fig. 3 Watermark extracting model

3.2 核心算法设计

本文采用正交编码方法实现水印嵌入。基于行散列值的水印嵌入方法，先将生成的二进制水印字符串按照每行固定嵌入的水印长度等分，再对载体文件中每行固定长度的内容进行哈希和模运算，将所得结果作为水印序列号信息，采用不可见字符代替二进制水印流，在每行末尾嵌入不可见水印字符串；基于行序置换的水印嵌入方法，根据随机数生成器生成的伪随机数选取起始行号进行行置换，使得行的散列值代表起始标志位、二进制水印编码值和结束标志位，以此来完成水印嵌入。水印嵌入的具体步骤如下：

步骤 1　将潜在泄密者即下载文件的用户的个人信息以及系统自动生成的当前时间戳信息串联成固定长度的明文字符串，对该明文字符串进行 AES 对称加密以及 BASE64 编码，将生成的字符串转换为由 0 和 1 构成的二进制水印字符串 WM。

步骤 2　将水印容量等分为 K 份，计算待嵌入水印份数 $K=L_{wm}/l$，其中 L_{wm} 为二进制水印字符串长度，l 为每行嵌入的固定水印长度，定义可变参数 n 为水印的冗余度，则水印总容量 $N=L_{wm}*n$。

步骤 3　逐行读取原始文件，计算每行固定位置文本内容的 MD5 值 h_{MD5}，利用该行内容的 MD5 值对水印份数 K 进行取模运算，得到结果值 $k=h_{\mathrm{MD5}}\%K$。

步骤 4　将第 k 份二进制水印串 WM_k 按照式(5)所示的规则映射为不可见字符串，并将不可见字符串嵌入该行，回到步骤 3 继续读取下一行，直至整个文件读取完成。

$$WM_k(i)=\begin{cases}\text{空格}, & \text{第 } i \text{ 个字符为 } 0\\ \text{制表符}, & \text{第 } i \text{ 个字符为 } 1\end{cases} \tag{5}$$

步骤 5　随机数生成器生成伪随机数 i，则起始行号为 i。

步骤 6　进行行置换，将起始行 i 到 $i+m$ 行的 $HASH\oplus 2$ 值全部变为 1，从 $i+m+1$ 行到 $i+m+N$ 行的 $HASH\oplus 2$ 值为二进制水印编码值，从 $i+m+N+1$ 行到 $i+2m+N+1$ 行的 $HASH\oplus 2$ 值全部变为 0，其中 m 为设定的标志位长度。结束水印嵌入。

水印嵌入算法如算法 1 所示。

算法 1　水印嵌入

```
输入：待嵌入水印文件 file，水印明文信息 inf
输出：嵌入水印后的文件 wm_file
1. function Encode(inf)
2. wm<- Info2Binary(inf)
3. return wm
4. end function
5. function RowHash(file,wm)
6. n<- len(wm) / l
7. for i in range(0,n)
8. wm[i]<- Invis(wm[i*l:(i+1)*l])
9. for line in file
10. k<- MD5(line[0:x]) % n
11. line<- line+wm[k]
12. wm_file.writelines(line)
13. end for
14. return wm_file
15. end function
16. function RowSwap(file,wm)
17. N<- len(wm)
18. i<- random(0,row-2m-N-1)
19. make each value of HASH⊕2 of line i to i+m equal 1
20. make each value of HASH⊕2 of line i+m+1 to i+m+N equal wm
21. make each value of HASH⊕2 of line i+m+N+1 to i+2m+N+
    1 equal 0
22. wm_file.writelines(line)
23. return wm_file
24. end function
25. function main()
26. wm<- Encode(inf)
27. RowSwap(RowHash(file,wm),wm)
28. end function
```

水印提取过程为嵌入的逆过程。首先，通过基于行序置换的水印提取方法计算纯文本文件中每行的散列值，当检测到起始标志位时开始记录计算后的散列值，直到检测到结束标志位时停止，将记录下的散列值作为待比较的二进制字符串。通过基于行散列值的水印提取方法，将载体文件中每行固定长度的内容进行哈希和模运算，将得到的结果值作为水印序列号信息，将行末的不可见字符转换为二进制字符，统计各段水印出现的次数，将次数最多的每段水印排序并合并为待比较的二进制字符串。将两串二进制序列进行比较，采用对称密码算法对上述二进制水印字符串进行解密，将解密后的明文字符串进行反编码后得到明文水印要素信息，该信息包括泄密者的个人信息，以及嵌入水印时的时间戳信息。水印提取的具体步骤如下：

步骤 1　逐行读入文本文件内容，如果检测到连续 m 行 $HASH\oplus 2$ 的值为 1，则从下一行开始将 $HASH\oplus 2$ 的值记录下来，直到检测到连续 m 行 $HASH\oplus 2$ 的值为 0 时停止检测，将记录下的二进制字符串记为 watermark1。

步骤 2　在逐行读入文本文件内容的同时，检测行末是否存在不可见字符，若存在则将该行的文本内容部分和末尾的不可见水印字符部分分割，对该行文本内容部分的固定位置计算 MD5 值 h_{MD5}，计算 $k=h_{\mathrm{MD5}}\%K$，其中 $K=L_{wm}/l$，L_{wm} 为二进制水印字符串长度，l 为每行嵌入的固定水印长度。将本行的不可见水印字符部分转换为二进制串，并将其作为第 k 份水印存储，直至整个文件读取完成。

步骤 3　统计第 k 份水印中各个字符串出现的次数，将出现次数最多的字符串作为该份水印记录下来，按顺序对每份水印进行排序并组合，使其成为完整的二进制水印字符串，记为 watermark2。

步骤 4　对比 watermark1 与 watermark2，若两者相同且长度为 L_{wm}，则将 watermark1 输出；若两者不同，则将长度为 L_{wm} 的字符串输出；若两者长度都不为 L_{wm}，则水印提取失败。

步骤 5　对照 ASCII 码表或 Unicode 码表对输出的二进制水印字符串进行反编码，得到原始的明文水印要素信息，其中包括泄密者个人信息以及时间戳信息。

3.3 系统实现

本文方法基于 Spark 设计实现，处理场景为针对大批量文件的处理。针对多个文件的分布式批处理流程如下：

(1)使用 sparkContext 的 wholeTextFiles 接口从 HDFS 上读取整个目录的文件，得到键为文件名、值为文件内容的弹性分布式数据集 RDD。

(2)对 RDD 的每个键值对进行处理,执行加水印或者提取水印操作,得到键为文件名、值为处理结果的 RDD。

(3)使用 saveAsHadoopFile 接口,保存 RDD 到 HDFS 中,重写 OutputFormat,键为文件名,输出时只输出其值。

3.4 系统架构

基于本文方法,设计并实现了一个大数据纯文本水印系统,系统架构如图 4 所示,水印嵌入模块和水印提取模块基于 Spark 分布式处理平台开发。其中,水印嵌入模块包括水印生成单元和水印嵌入单元。水印生成单元主要进行水印信息的预处理工作,将水印信息串转换为二进制水印字符串;水印嵌入单元利用基于正交编码的水印嵌入方法将二进制水印字符串嵌入到原始文件中。水印提取模块包括水印提取单元和水印恢复单元。其中,水印提取单元是水印嵌入模块中的水印嵌入单元的逆过程,利用基于正交编码的水印提取方法将嵌入文件的二进制水印字符串提取出来;水印恢复单元主要将二进制水印字符串转换为水印信息串,得到水印文件的明文信息。大数据纯文本文件存储在 HDFS 分布式文件系统中,水印信息数据库存储待嵌入水印信息及订单信息等。本文开发了相应的系统界面,供用户进行水印嵌入、提取及文件的上传、下载等操作。

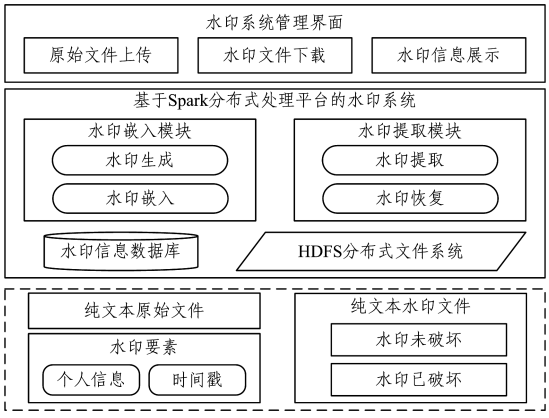

图 4 系统架构

Fig. 4 System architecture

4 实验与分析

数字水印技术应该满足水印的固有特性,例如鲁棒性、隐蔽性、水印容量和算法性能等[19-20],因此按照水印的固有特性进行水印技术评估。本文的水印方法测试主要分为以下 4 个部分:1)水印方法性能测试,包括水印容量测试和针对多个文件进行大规模批量水印处理的测试;2)隐蔽性测试,针对嵌入水印前后的文件视觉效果进行对比分析;3)鲁棒性测试,针对嵌入水印后的文件进行删减、添加、拷贝、替换攻击,检测提取水印的正确性;4)对比测试,将本文算法与 3 种文本水印算法进行对比分析,记录测试指标,包括处理时间、水印提取成功率。

4.1 实验环境

(1)硬件与集群配置

硬件配置为 E5-2603 v2 @ 1.80 GHz CPU、8 GB 内存。Spark 集群由 4 个节点组成,即 1 个主节点和 3 个从节点。

(2)软件配置

实验环境的软件配置如表 1 所列。

表 1 实验环境的软件配置

Table 1 Software configuration of experimental environment

软件名称	版本
操作系统	Ubuntu 16.04
Java	1.8.0
Spark	2.1.0
Scala	2.12.2

4.2 性能测试

水印方法性能测试分为水印容量测试和针对多个文件进行大规模批量水印处理测试。

根据对水印嵌入算法的分析,由于基于行散列值的水印方法的水印容量与文本大小关系不大,因此本节重点对基于行序置换的水印方法进行水印容量的测试与分析。

假设文本有 N 行,水印标志位长度为 m,则理论上文本中最多能嵌入的二进制水印字符串长度为 $L_w=N-2m$,而水印信息中英文采用 ASCII 编码,中文采用 Unicode 编码,一个英文字母由 8 位二进制序列表示,一个汉字由 16 位二进制序列表示,因此理论上能嵌入的水印容量如下:

$$R_{英}=L_w/8=(N-2m)/8$$

$$R_{中}=L_w/16=(N-2m)/16$$

上述分析证明,本文算法具有较好的水印容量性能。

针对多个小文件进行大规模批量水印处理,分别提交嵌入和提取任务,记录任务执行时长,并计算单个文件的平均处理耗时。

针对多个 CSV 文件的批量水印嵌入处理摘要如图 5 所示,使用 1000~6000 个单个大小为 700 kB 的测试文件来测试水印嵌入任务计算的总耗时。

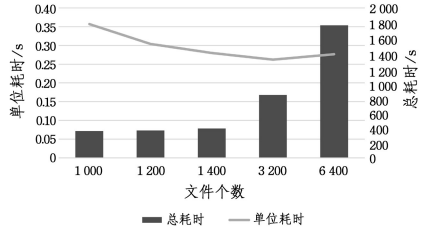

图 5 Spark 集群正交编码水印嵌入算法的多文件处理时间

Fig. 5 Multi file processing time under spark cluster of watermark inserting based on orthogonal coding

针对多个 CSV 文件的批量水印提取处理摘要如图 6 所示,使用 1000~6000 个单个大小为 700 kB 的测试文件,测试水印嵌入任务计算的总耗时。

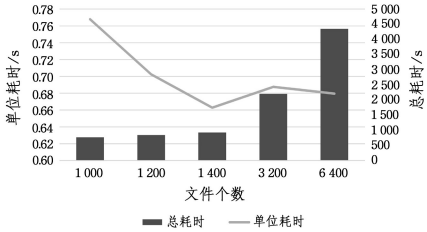

图 6 Spark 集群正交编码水印提取算法的多文件处理时间

Fig. 6 Multi-file processing time under Spark cluster of watermark inserting based on orthogonal coding

本测试在局域网环境、配置等因素作用下,时间数据可能

会在不同时期略有出入。性能测试结果表明,随着文件数量的增加,水印处理时间延长,但整体处理时间能够满足用户的心理期望。随着批处理文件数量的增加,每个文件处理的单位耗时缩短,表明当文件数量较少时,任务初始化所占时间相对较长。因此,本文方法更适用于文件数量庞大的情形。

4.3　隐蔽性测试

图 7 为嵌入水印前的原始 CSV 文件与嵌入水印后的 CSV 文件的对比。

All-seasons.csv

```
Season,Episode,Character,Line
10,1,Stan,"You guys, you guys! Chef is going away."
10,1,Kyle,"Going away? For how long?"
10,1,Stan,"Forever."
10,1,Chef,"I'm sorry boys."
10,1,Stan,"Chef said he's been bored, so he joining a group called the Super Adventure Club."
10,1,Chef,"Wow!"
10,1,Mrs. Garrison,"Chef?? What kind of questions do you think adventuring around the world is gonna answer?!"
10,1,Chef,"What's the meaning of life? Why are we here?"
10,1,Mrs. Garrison,"I hope you're making the right choice."
10,1,Cartman,"I'm gonna miss him.  I'm gonna miss Chef and I...and I don't know how to tell him!"
10,1,Stan,"Dude, how are we gonna go on? Chef was our fuh...f-ffriend."
10,1,Mayor McDaniels,"And we will all miss you, Chef,  but we know you must do what your heart tells you.."
10,1,Jimbo,"Bye-bye!"
10,1,Gerald,"Good-bye!"
10,1,Mr. Mackey,"So long!"
10,1,A Man,"So long, Chef!"
10,1,A Sign-Holder,"Good-bye, Chef!"
10,1,Randy,"Good-bye, Chef! Have a great time with the Super Adventure Club!"
10,1,Chef,"Good-bye! .."
10,1,Kyle,"Draw two card, fatass."
```

All-seasons-we-ed.csv

```
Season,Episode,Character,Line
10,1,Stan,"You guys, you guys! Chef is going away."
10,1,Kyle,"Going away? For how long?"
10,1,Stan,"Forever."
10,1,Chef,"I'm sorry boys."
10,1,Stan,"Chef said he's been bored, so he joining a group called the Super Adventure Club."
10,1,Chef,"Wow!"
10,1,Mrs. Garrison,"Chef?? What kind of questions do you think adventuring around the world is gonna answer?!"
10,1,Chef,"What's the meaning of life? Why are we here?"
10,1,Mrs. Garrison,"I hope you're making the right choice."
10,1,Cartman,"I'm gonna miss him.  I'm gonna miss Chef and I...and I don't know how to tell him!"
10,1,Stan,"Dude, how are we gonna go on? Chef was our fuh...f-ffriend."
10,1,Mayor McDaniels,"And we will all miss you, Chef,  but we know you must do what your heart tells you.."
10,1,Jimbo,"Bye-bye!"
10,1,Gerald,"Good-bye!"
10,1,A Sign-Holder,"Good-bye, Chef!"
10,1,Randy,"Good-bye, Chef! Have a great time with the Super Adventure Club!"
10,1,Chef,"Good-bye! .."
10,1,Mr. Mackey,"So long!"
10,1,A Man,"So long, Chef!"
```

图 7　嵌入水印前后的 CSV 文件的对比

Fig. 7　Comparison of CSV files before and after inserting watermark

嵌于行尾的空格和制表符在视觉上具有很好的隐蔽性,如果不对比原始文件,嵌入水印后的文件中行序的改变也不会造成视觉上的影响,因此算法具有很强的隐蔽性。

4.4　鲁棒性测试

鲁棒性测试也称为抗攻击性测试。对文本数字水印算法的抗攻击性测试可以概括成两种类别:格式攻击和内容攻击。格式攻击主要针对 DOC 和 PDF 等文件格式,包括对字体、字体颜色、字号、行间距、字间距等的修改。内容攻击的对象包括所有的文件格式,攻击形式包括删减、添加、混淆、拷贝、替换、转码、复合攻击等,其中,删减攻击是指删除文件中的部分内容,添加攻击是指在文件中添加一些内容,混淆攻击是指对文件行序进行打乱重排,拷贝攻击是指将文件内容拷贝到其他文件中,替换攻击是指将文本中的某些内容替换成其他内容,转码攻击是指对文件编码格式进行转换,复合攻击是指多种攻击相结合的攻击方式。由于纯文本不具备格式,因此格式攻击对其是完全无效的。

现设计如下测试方法,以检验基于正交编码的大数据纯文本水印方法的抗攻击性。

(1)选取若干测试文件,使用本文方法进行水印嵌入操作。

(2)将嵌入水印后的文件分组,分别对其进行删减、添加、混淆、拷贝、替换、转码及复合攻击。

(3)使用本文方法对攻击后的文件进行水印提取操作,记录水印提取成功率。

针对删减攻击,若删减对象是若干字符,则本文算法可以完全抵御;若删减对象为若干行,则只要保留基于行散列值的水印方法嵌入的一份完整水印即可成功提取。若删减的内容包含在基于行序置换的水印嵌入方法的范围内,则可以检测出文本遭到攻击。若对文本进行大量的行删除,将导致文本可用性降低,在保证文本可用性的前提下,本文方法能够完全抵御删减攻击。

针对添加攻击,若添加的位置为文末,则本文算法可以完全抵御;若添加的位置为文中,则在添加对象为行时,由于添加的内容不带有水印段,本文方法可以完全抵御;若添加对象为字符,则只要保留基于行散列值嵌入的一份完整水印不被破坏即可成功提取。若添加的内容包含在基于行序置换的水印嵌入方法范围内,则可以检测出文本遭到攻击。

针对混淆攻击,由于基于行散列值的水印嵌入方法与行序无关,因此本文方法可以完全抵御。而基于行序置换的水印嵌入算法高度依赖行序,因此可以检测出文本遭到攻击。

针对拷贝攻击,若拷贝的范围为全文,则不会影响水印的提取,本文方法能够完全抵御;若拷贝的范围为部分内容,则相当于对文本进行了删减攻击,在保证文本可用性的前提下,本文方法能够完全抵御,并能部分检测出文本遭到攻击;若将文本内容拷贝到一个已有的文本中,则相当于对文本进行了添加攻击,只要保留基于行散列值嵌入的一份完整水印不被破坏即可成功提取,并能部分检测出文本遭到攻击。

针对替换攻击,若替换的为基于行散列值的水印嵌入方法使用的不可见字符,则基于行散列值的水印方法无法成功提取水印,但不影响基于行序置换的水印方法,本文算法可以完全抵御该攻击,并能检测出文本遭到攻击;若替换的是其他内容,本文算法可以完全抵御该攻击,若替换的内容包含在基于行序置换的水印嵌入方法范围内,则可以检测出文本遭到攻击。

针对转码攻击,由于任何编码都支持空格和制表符,转码不影响水印信息的存在,因此本文算法可以完全抵御该攻击。

针对多种攻击相结合的复合攻击,只要文本足够大,在保证文本可用性的前提下进行攻击,被破坏的文件只要保留基于行散列值嵌入的一份完整水印不被破坏即可成功提取,并能通过基于行序置换的水印方法检测出文本遭到攻击。

4.5　对比测试

表 2 从算法的鲁棒性即抗攻击性方面对本文算法和 3 种文本水印算法进行了对比分析。

表 2　抗攻击性的对比

Table 2　Comparison of performances of anti-attack

攻击方式	本文算法	文献[15]算法	文献[16]算法	文献[17]算法
删减	√	√	×	√
添加	√	√	×	√
混淆	√	√	×	×
拷贝	√	√	×	√
替换	√	√	×	×
转码	√	×	√	√

其中，√表示在保证文件可用性的前提下，对嵌入水印后的文件进行攻击，依然可以成功提取水印，×则表示无法正确提取水印。

抗攻击性对比测试结果表明，在保证文件数据可用性的前提下，文献[16]中的算法的抗攻击性较差，文献[15,17]中的算法的抗攻击性较好，而本文算法在文件遭受各种攻击时均能成功提取水印信息，说明本文方法具有较强的鲁棒性，能够有效对数据泄露进行追踪溯源，提高数据安全性。

结束语 本文提出了一种基于正交编码的大数据平台纯文本水印方法，针对大数据平台纯文本的数据记录顺序不固定、冗余空间有限的特点，通过两种正交的水印方法将不可见字符嵌入文本文件。该方法既与数据记录的顺序无关，又能通过改变部分行序来嵌入脆弱水印，以达到文本篡改检测的效果，基于行散列值和基于行序转换这两种方法相辅相成，在所需的水印空间较小的基础上，保证了较强的算法鲁棒性。对于大数据平台中的纯文本，本文方法具有很强的适用性，使用了 Spark 平台对大批量文本数据进行分布式处理，提高了该方法的处理性能。实验和理论分析证明，本文方法能够有效对数据泄露进行追踪溯源，提高数据安全性。

但本文方法嵌入了大量的空格和制表符，增加了文件的大小，并且在文本中嵌入不可见字符，使得文本的统计特性发生改变。后续研究将集中于改进空格编码方法，使得文本统计特性不会发生较大改变，以增强方法的安全性与隐蔽性。

参 考 文 献

[1] FENG D G, ZHANG M, LI H. Big data security and privacy protection[J]. Chinese Journal of Computers, 2014, 37(1): 246-258. (in Chinese)
冯登国，张敏，李昊. 大数据安全与隐私保护[J]. 计算机学报，2014，37(1)：246-258.

[2] BRASSIL J T, LOW S, MAXEMCHUK N F, et al. Electronic marking and identification techniques to discourage document copying[J]. IEEE Journal on Selected Areas in Communications, 1995, 13(8): 1495-1504.

[3] BRASSIL J T, LOW S, MAXEMCHUK N F. Copyright protection for the electronic distribution of text documents[J]. Proceedings of the IEEE, 1999, 87(7): 1181-1196.

[4] CAI F F, LIU Y, YIN X L. Text Watermarking Scheme for Word Documents[J]. Computer Science, 2012(S2): 39-40.

[5] XIAO C, ZHANG C, ZHENG C. FontCode: Embedding Information in Text Documents using Glyph Perturbation[J]. ACM Transactions on Graphics (TOG), 2018, 37(2): 15.

[6] CHEN Q, XING X X. Research on performance evaluation benchmark of formatted text watermarking[J]. Application Research of Computers, 2014, 31(9): 2764-2768. (in Chinese)
陈青，邢晓溪. 格式化文本水印性能评估基准研究[J]. 计算机应用研究，2014，31(9)：2764-2768.

[7] KAUR M, MAHAJAN K. Performance Evaluation of Natural Language Text Watermarking using Encryption Techniques[J]. International Journal of Computer Applications, 2015, 129(3): 22-28.

[8] ATALLAH M J, RASKIN V, CROGAN M, et al. Natural Language Watermarking: Design, Analysis, and a Proof-of-Concept Implementation[C]// International Workshop on Information Hiding. Springer-Verlag, 2001: 185-199.

[9] ATALLAH M J, MCDONOUGH C J, RASKIN V, et al. Natural language processing for information assurance and security: an overview and implementations[C]// The Workshop on New Security Paradigms. ACM, 2001: 51-65.

[10] LI G S, CHEN J P, MA H Y, et al. Method for Text Watermarking Based on Subject-verb Encoding[J]. Computer Science, 2015, 42(S2): 374-377.

[11] LIN X J, TANG X H, WANG J. A Reversible Text Watermarking Algorithm Based on Coding and Synonymy Substitution[J]. Journal of Chinese Information Processing, 2015, 29(4): 151-158. (in Chinese)
林新建，唐向宏，王静. 编码与同义词替换结合的可逆文本水印算法[J]. 中文信息学报，2015，29(4)：151-158.

[12] KAMARUDDIN N S, KAMSIN A, POR L Y, et al. A Review of Text Watermarking: Theory, Methods, and Applications[J]. IEEE Access, 2018, 6: 8011-8028.

[13] MIR N. Copyright for web content using invisible text watermarking[J]. Computers in Human Behavior, 2014, 30: 648-653.

[14] TALEBY A M, DANA M H, TABASI S H. An innovative technique for web text watermarking (AITW)[J]. Information Security Journal: A Global Perspective, 2016, 25(4/5/6): 191-196.

[15] ZHANG Z Y, LI Q M, QI Y. Text watermarking design based on invisible characters[J]. Journal of Nanjing University of Science and Technology, 2017, 41(4): 405-411. (in Chinese)
张震宇，李千目，戚湧. 基于不可见字符的文本水印设计[J]. 南京理工大学学报：自然科学版，2017，41(4)：405-411.

[16] BAI J, XU Y H, YANG Y. An Algorithm of Text Steganography[J]. Application Research of Computers, 2004, 21(12): 147-148. (in Chinese)
白剑，徐迎晖，杨榆. 利用文本载体的信息隐藏算法研究[J]. 计算机应用研究，2004，21(12)：147-148.

[17] FU Y, WANG B B. Extra space coding for embedding Wartermark into text documents and its performance[J]. Journal of Xian Highway University, 2002, 22(3): 85-87. (in Chinese)
傅瑜，王保保. 文本水印附加空格编码方法的实现及其性能[J]. 长安大学学报(自然科学版)，2002，22(3)：85-87.

[18] SUN L. Design of Document Watermarking Algorithm Based on Space Encoding[J]. Science Technology & Engineering, 2007, 7(17): 4504-4507. (in Chinese)
孙利. 基于空格编码的文本数字水印算法设计[J]. 科学技术与工程，2007，7(17)：4504-4507.

[19] TIWARI N. Digital Watermarking Applications, Parameter Measures and Techniques[J]. International Journal of Computer Science and Network Security (IJCSNS), 2017, 17(3): 184.

[20] KAUR B, SHARMA S. Digital watermarking and security techniques: A review[J]. International Journal of Computer Science Technology, 2017, 8(2): 44-47.