[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 1/00 (2006.01)

[12] 发明专利说明书

专利号 ZL 03808579.8

[45] 授权公告日 2007 年 3 月 14 日

[11] 授权公告号 CN 1304914C

[22] 申请日 2003.5.14 [21] 申请号 03808579.8
[30] 优先权
[32] 2002. 5.15 [33] US [31] 10/147,741
[86] 国际申请 PCT/GB2003/002065 2003.5.14
[87] 国际公布 WO2003/098406 英 2003.11.27
[85] 进入国家阶段日期 2004.10.15
[73] 专利权人 国际商业机器公司
地址 美国纽约
[72] 发明人 拉科什·阿格拉瓦尔
杰拉德·基耶尔宁
[56] 参考文献
WO 01/59705 A2 2001.8.16
CN 1386341 A 2002.12.18
审查员 李 科

[74] 专利代理机构 中国国际贸易促进委员会专利商标事务所
代理人 康建峰

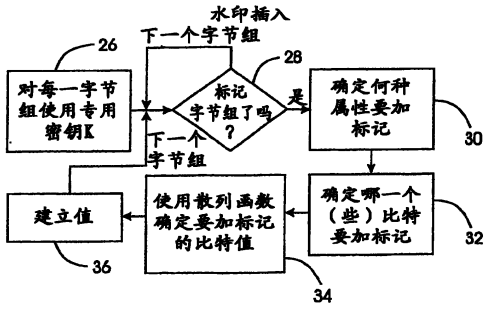
权利要求书 2 页 说明书 13 页 附图 2 页

[54] 发明名称

用于数据存储库的数字水印的系统和方法

[57] 摘要

一种使用秘密信息建立比特模式增强数据库安全的方法和系统，只有使用秘密信息，才能在数据库的复制本(许可的或未许可的)中检测该模式建立的水印。



1. 一种计算机，用于对数据存储库加水印，该数据存储库有字节组，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该计算机包括至少一个执行包括下述的方法动作的处理器：

根据保密信息，对至少一些字节组的至少一种属性的至少一个比特位置建立水印值，以便建立水印，其中，要加标记的字节组、字节组内要加标记的属性、属性中要加标记的比特位置、和特定的比特水印值中的至少两个是根据保密信息确定的，其中通过设置所述要加标记的属性的所述比特值为所述水印值，来建立水印。

2. 按照权利要求 1 的计算机，其中只有数值属性中的比特位置是水印的一部分。

3. 按照权利要求 1 的计算机，其中的水印值至少部分根据所述保密信息的单向散列函数建立。

4. 一种计算机，用于确定数据存储库中是否有水印存在，该数据存储库有字节组，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该计算机包括至少一个执行包括下述的方法动作的处理器：

接收有字节组的测试数据结构；和

至少部分使用保密信息确定测试数据结构中是否存在水印，包括：

对每一字节组确定该字节组是否已经加标记；

对每一已经加标记的字节组中至少一种属性确定该属性是否已经加标记；

对每一已经加标记的属性中至少一个比特确定该比特是否已经加标记；和

对每一已经加标记的比特确定水印值，以便确定该测试数据结构是否包含水印。

5. 按照权利要求 4 的计算机，其中所述的确定测试数据结构中是否

存在水印是在预定概率内确定测试数据结构中是否存在水印。

6. 一种对数据存储库加水印的方法，该数据存储库有字节组，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该方法包括如下步骤：

根据保密信息，对至少一些字节组的至少一种属性的至少一个比特位置建立水印值，以便建立水印，其中，要加标记的字节组、字节组内要加标记的属性、属性中要加标记的比特位置、和特定的比特水印值中的至少两个是根据保密信息确定的，其中通过设置所述要加标记的属性的所述比特值为所述水印值，来建立水印。

7. 按照权利要求 6 的方法，其中建立水印值的步骤还包括：

根据保密信息，确定该数据存储库中要加标记的多个字节组；

在每一要加标记的字节组中，确定至少一种要加标记的属性；

在每一要加标记的属性中，确定至少一个要加标记的比特；和

对要加标记的至少每一非无效比特建立水印值，以便建立水印。

8. 一种用于确定数据存储库中是否有水印存在的方法，该数据存储库有字节组，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该方法包括如下步骤：

接收有字节组的测试数据结构；和

至少部分使用保密信息确定测试数据结构中是否存在水印，包括：

对每一字节组确定该字节组是否已经加标记；

对每一已经加标记的字节组中至少一种属性确定该属性是否已经加标记；

对每一已经加标记的属性中至少一个比特确定该比特是否已经加标记；和

对每一已经加标记的比特确定水印值，以便确定该测试数据结构是否包含水印。

用于数据存储库的数字水印的系统和方法

技术领域

本发明涉及数据存储库的安全性。

背景技术

诸如软件、图像、视频、音频、和文本等数字资产的非法复制，长期以来是这些资产拥有者关心的问题。这些资产的保护通常是以插入数据中的水印为基础。水印软件把小的错误引入被加水印的对象中。这些故意的错误被称为标记，而所有这些标记共同构成水印。这些标记必须对数据的使用没有显著冲击，且它们的安排应使恶意的用户在不让数据失去效用的情况下不能破坏它们。因此水印不阻止复制，但它通过提供一种建立重新分布的复制本的原始所有权手段，制止非法复制。

在"防火墙后数据处理"之外的应用程序中，数据库使用的增长，正在建立类似的、对数据库加水印的需求。例如，在半导体工业，半导体零件的参数数据主要由三家公司提供：Aspect、IHS、和 IC Master。他们雇佣大量人力，从数据表中人工抽取部分技术要求，并建立参数数据库。然后，他们以高价向设计工程师发许可证。像 Acxiom 那样的公司，已经汇编了大量消费者和商业数据。在生命科学工业，诸如 Celera 那样的公司的主要资产是生物信息数据库。互连网正在向这些数据供应商施加巨大的压力，要他们建立服务（常常称为 e 实用程序或网络服务）允许用户远程搜索并接入数据库。虽然这种趋势对终端用户是种恩惠，但它却使数据供应商遭受数据窃贼的威胁。

就数据库方面，制作水印提出挑战，因为对多媒体数据加水印的技术不一定存在，大多数水印在开始时是对静态图像研发的，其后推广到视频和音频源。两种应用程序的差别就本文的理解，包括如下方面：

1. 多媒体对象包括大量比特，有颇大的冗余度。因此，水印有大的

覆盖区可以隐藏。相反，数据库关系由字节组构成，每一字节组代表分离的对象。水印必须散布在这些分离的对象上。

2. 多媒体对象各个片段的空间/时间定位，一般不改变。相反，某种关系的字节组构成一个集合，其它们之间不蕴涵次序。

3. 多媒体对象各部分在不引起对象中感觉上变化的情形下，不能丢失或置换。但是，关系的非法复制能够简单地使一些字节组丢失，或以另一种关系取代它们。

由于这些差别，为多媒体数据研发的技术，不能直接用于水印关系。同样，用于文本的水印技术，利用格式化文本的特殊性质，不能容易地应用于数据库。此外，对软件加水印的技术已经取得有限的成功，因为计算机程序中的指令常常能够重新排列而不改变程序的语义。但是，这种重新排序能够破坏水印。

发明内容

因此，本发明提供一种计算机，用于对有字节组(tuple)的数据存储库添加水印，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该计算机包括至少一个处理器，该处理器执行的方法的作用包括：根据保密信息，对至少一些字节组的至少一种属性的至少一个比特位置建立水印值，以便建立水印。

本发明优选用于识别数据的非法复制。

能够加水印的数据库关系最好有如下的属性，它们的一些值的改变不影响该应用程序。真正的数据集能容许少量错误而不使它们的可用性降质。例如，用于建立天气预报模型的 ACARS 气象数据，有风矢量和温度精度估计，分别在 1.8 m/s 和 0.5 °C 之内。按照一个优选实施例的本发明，认识到水印引进的误差，能够限制在该数据的测量容差之内。作为另一个例子，考虑实验上获得的基因表达数据集，是用各种数据开发技术分析的。还有，按照一个优选实施例的本发明，认识到数据收集和分析技术的本性，在于少量数据值的改变，不可能影响结果。类似地，如果补充数据的外部供应商从少量交易中增加或减去一些量，消费者商

品公司的顾客程序分段结果将不受影响。最后，考虑上述半导体零件的参数数据。对许多参数，因水印引进的误差，可以限制在测量容差之内。

数据存储库例如可以是文件系统、数据库、或其他记录存储器。计算机最好根据保密信息，对一些字节组一些属性中的比特位置建立水印值，以便建立定义水印的比特模式。

在一个优选实施例中，只有数值属性中的比特位置必须是水印的一部分。水印值最好至少部分根据至少保密信息的单向散列函数建立。

按照一个优选实施例，该计算机能够确定，被怀疑已经从有水印的数据库中被复制的测试数据库，是否事实上含有该水印。使用保密信息，计算机最好在预定的概率内确定，该水印是否存在于该测试数据结构中。

最好提供通用的计算机，用于确定有字节组的测试数据存储库中，是否存在水印。最好用保密信息来确定该水印是否存在于测试数据结构中。

在一个实施例中，要加标记的字节组、字节组内要加标记的属性、属性中要加标记的比特位置、和特定的比特水印值，以上至少一种是根据保密信息确定的。

在一个实施例中，要加标记的字节组、字节组内要加标记的属性、属性中要加标记的比特位置、和特定的比特水印值，以上至少两种是根据保密信息确定的。

在一个实施例中，要加标记的字节组、字节组内要加标记的属性、属性中要加标记的比特位置、和特定的比特水印值，以上所有各个都是根据保密信息确定的。

最好是，只有数值属性中的比特位置是水印的一部分。

最好是，水印值至少部分根据至少保密信息的单向散列函数建立。

最好是，能够接收测试数据结构；并至少部分使用保密信息，确定该水印是否存在于测试数据结构中。

最好能在预定概率内确定，该水印是否存在于测试数据结构中。

在一个实施例中，要加标记的字节组、字节组内要加标记的属性、属性中要加标记的比特位置、和特定的比特水印值，以上至少一个、至

少两个、或所有各个，是根据保密信息确定的。

按照一个方面，本发明提供一种计算机，用于确定有字节组的数据存储库中是否存在水印，其中，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该计算机包括至少一个处理器，该处理器执行的方法的作用包括：接收测试数据结构；并至少部分使用保密信息确定该水印是否存在于测试数据结构中。

最好是在预定的概率内确定，该水印是否存在于该测试数据结构中。

最好是，对导出测试数据结构的数据存储库，根据保密信息对一些字节组一些属性的至少一些比特位置建立水印值。

按照一个优选实施例，提供一种计算机程序装置，该计算机程序装置包括：可由数字处理设备读出的计算机程序存储装置；和在该程序存储装置上的程序与可由数字处理设备执行的指令，以便改进数据存储库的安全性，该程序包括：根据保密信息确定多个要加标记的字节组的装置；确定每一个要标记的字节组中至少要标记的一种属性的装置；确定每一个要标记的属性中至少要标记的一个比特的装置；和对至少每一个要标记的非无用比特建立水印值，以便建立水印的装置。

按照一个优选实施例，提供一种提高数据存储库的安全性的方法，包括：使用秘密信息建立遍及数据存储库的比特值的模式。

最好是，要标记的数据存储库中的多个字节组根据保密信息确定。

最好是，确定要标记的每一字节组中至少一种要标记的属性。

最好是，确定要标记的每一种属性中至少一个要标记的比特。

最好是，对至少每一个要标记的非无用(non-null)比特建立水印值，以便建立水印。

按照一个优选实施例，是提供一种计算机程序装置，该计算机程序装置包括：可由数字处理设备读出的计算机程序存储装置；和在该程序存储装置上的程序与可由数字处理设备执行的指令，该程序包括：接收有字节组的测试数据结构的装置；对每一字节组确定该字节组是否已经根据保密信息标记的装置；至少对已经标记的每一字节组中一种属性，确定该属性是否已经标记的装置；对已经标记的每一属性中至少一个比

特，确定该比特是否已经标记的装置；和对已经标记的每一比特确定水印值，以便确定测试数据结构是否包含水印的装置。

最好是，只有数值属性中的比特位置被作为水印的一部分测试。

最好是，能够保持对校正的比特值计数，并根据该计数确定，该水印是否存在于测试数据结构中。

最好是，水印值至少部分根据至少保密信息的单向散列函数确定。

按照一个优选实施例，提供一种方法，包括：使用秘密信息确定，是否在测试数据结构中存在某种值的模式。

最好是，对测试数据结构中每一字节组确定，该字节组是否已经根据保密信息标记。

最好是，对每一已经标记的字节组中至少一种属性确定，该属性是否已经标记。

最好是，对每一已经标记的属性中至少一个比特确定，该比特是否已经标记。

最好是，对已经标记的每一比特确定水印值，以便确定测试数据结构是否包含水印。

在一个实施例中，至少一些属性有主密钥(primary key)，且用某一属性的主密钥来建立水印值。

在一个实施例中，至少第一属性没有主密钥，且用分割第一属性中的比特来建立水印值。

在一个实施例中，至少一些属性有主密钥，且用某一属性中的主密钥来建立比特值。

按照一个方面，是提供一种对有字节组的数据存储库加水印的方法，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该方法包括如下步骤：根据保密信息，对至少一些字节组至少一种属性至少一个比特位置，建立水印值，以便建立水印。

按照另一个方面，是提供一种计算机程序，用于确定有字节组的数据存储库中，是否存在水印，每一字节组与一个或多个属性相关，每一

属性有一个或多个比特位置，至少一些比特位置有比特值，该计算机程序包括程序码单元，当该程序码在计算机上执行时，实施如下步骤：接收测试数据结构；并至少部分使用保密信息，确定该水印是否存在于测试数据结构中。

按照另一个方面，是提供一种方法，用于确定有字节组的数据存储库中，是否存在水印，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该方法包括如下步骤：接收测试数据结构；并至少部分使用保密信息，确定该水印是否存在于测试数据结构中。

按照另一个方面，是提供一种计算机程序，用于在有字节组的数据存储库中制作水印，每一字节组与一个或多个属性相关，每一属性有一个或多个比特位置，至少一些比特位置有比特值，该计算机程序包括程序码单元，当该程序码在计算机上执行时，实施如下步骤：根据保密信息，对至少一些字节组至少一种属性至少一个比特位置，建立水印值，以便建立水印。

按照另一个方面，是提供一种数据存储库，该数据存储库包括：多个字节组，每一字节组至少有一种属性，每一属性至少有一个有比特值的比特；根据秘密信息建立的比特之间的值模式。

附图说明

现在将参照以举例的方式说明本发明的优选实施例。

图 1 按照本发明的一个优选实施例，画出一种系统的示意图；

图 2 按照本发明的一个优选实施例，画出在数据库中用于建立水印的逻辑的流程图；和

图 3 按照本发明的一个优选实施例，画出在测试数据结构中用于确定是否存在水印的逻辑的流程图。

具体实施方式

现在首先参考图 1，图上画出一种系统，一般以 10 表示。如图所示，

系统 10 包括计算机 12，该计算机接入诸如文件系统的数据结构或存储库，或如图 1 举出的实施例所示，接入关系数据库 14。因此，计算机 12 能作为关系数据库管理系统的主机。水印应用程序 16 能够在计算机 12 上执行，承担图 2 和 3 所示的逻辑。

在一个示例的非限制性实施例中，水印应用程序 16 可以是用户定义的用 Java™写的函数。计算机可以是 Windows NT® Version 4.00 工作站，用 Java™ Database Connectivity(JDBC™连通性)执行 DB2® Universal Database™(UDB) Version 7。其他类型的计算机，包括不受主机架限制的计算机、膝上型、桌面型、和笔记本型计算机，均可使用，包括其他类型的数据库。（Java，全部基于 Java 的商标和 JDBC 都是 Sun Microsystem Inc 在美国、其他国家的注册商标; Windows NT 是 Microsoft Corporation 在美国、其他国家的注册商标;DB2 和 DB2 Universal Database 是 International Business Machine Corporation 在美国、其他国家的注册商标）。

为说明的目的，图 1 画出的数据存储库（如数据库 14）包含字节组 18，每一字节组 18 包含一种或多种属性 20。而每一属性 20 又可以包括一个或多个数据比特 22，每一数据比特有值 24，该值例如不是“0”便是“1”（或在一些情形中，是无用比特）。在该优选实施例中，利用秘密信息建立一种值的模式，诸如比特值的模式，或较不可取的数据库属性值的模式，该秘密信息在此后不能被检测到，除非通过该秘密信息接入。

借助以上概述的本体系结构，应当了解本逻辑是在如图 1 所示的体系结构中，按照下面讨论的流程图执行的。本文的流程图说明，优选实施例的逻辑结构，体现在计算机的程序软件中。本领域熟练人员应当明白，流程图说明的逻辑单元结构，诸如计算机程序码单元，或电子逻辑电路，按照本优选实施例起作用。最好是用一种机器部件，指令数字处理设备（即计算机）实施一系列与图示对应的那些功能步骤，再现该逻辑单元。

换句话说，该逻辑可以用计算机程序体现，该计算机程序由处理器作为一系列计算机可执行指令执行。这些指令，例如可以驻留在 RAM 或

硬盘或光盘上，或者，这些指令可以存储在磁带、电子只读存储器、或其他适当的数据存储装置上。在本发明的一个示例性实施例中，该计算机可执行指令可以是多行 Java 或汇编 C^{++}兼容码。

现在参考图 2，可以从图中看到数据库 14 中用于建立水印的逻辑。流程从方框 26 开始，对每一字节组 18 的循环语句进入数据库 14，在该循环语句中使用了保密信息。虽然优选的保密信息可以是"密钥"，如一对公用密钥-专用密钥的专用密钥，但任何合适的保密信息，诸如一串机密的号码，也可以使用。

为较好地了解图 2，首先给出下面的定义。

假定数据库 14 能够被认为是数据库关系 R，它的模式是 $R(P,A_0,\ldots,A_{v-1})$，这里 P 是主密钥属性。为了说明，假定所有 v 种属性 $A_0,\ldots,A_{v-1}$ 都是作标记的候选者。就是说，数据库 14 有 η 字节组，而 v 是关系中可用于作标记的属性数，以 ξ 为一种属性中可用于作标记的最低有效比特数。还有，$1/\gamma$ 粗略代表将要作标记的字节组的百分数，ϖ 代表在图 2 逻辑之后已被标记的字节组数，α 代表为确定测试数据结构中是否存在水印时测试的显著水平，关于该测试数据结构，下面还要结合图 3 说明，最后，如下面在图 3 公开的，τ 是指示存在水印所必需的正确标记的字节组最小数。最好是，所有用于标记的候选属性都是数值属性，且它们的值在 ξ 最低有效比特中的变化，对它们全部都是难以察觉的。符号 $r.A_i$ 是指属性 A_i 在字节组 $r\in R$，即可用属性集合中的值。

除上述定义外，下面给出某些功能的解释。目前优选的非限制性实施方案，使用单向散列函数 H 操作任意长度的输入消息 M，且 H 返回固定长度的散列值 h，即 h=H(M)。优选的散列函数的附加特征是，i)给定 M，容易计算 h，ii)给定 h，难以从 H(M)=h 计算 M，和 iii)给定 M，难以找到另一个消息 M'，使 H(M)=H(M')。Message Digest 5 和 SHA 是对 H 的好的选择。

消息鉴别码（MAC）是依赖于密钥的单向散列函数。假定 F 是一 MAC，它使字节组 r 主密钥属性 r.P 随机化并返回一大范围的整数值。F 能够作为种子，以专用密钥 K 播下，该专用密钥 K 只有数据库 14 的拥

有者了解。在该优选实施例中，可以用如下的安全 MAC:
F(r.P)=H(K○H(K○r.P))，其中的○表示并置。

记住上面的定义，图 2 的说明可以结束。逻辑移动到判定菱形 28，其中，在一个优选的、非限制性实施例中确定，被测试的字节组是否应如下所述作标记。如果 F(r.P)modγ 等于 0，该字节组被指定用于标记。显而易见，只有具备秘密信息 K 知识的用户，才容易知道什么字节组要作标记，从而知道什么字节组已经作了标记。

如果对标记的测试在判定菱形 28 不被满足，循环回到获得下一个字节组。否则，如果该字节组已被标记，逻辑前进至方框 30，在方框 30 确定字节组的什么属性被作标记。为此，在一个优选的、非限制性实施方案中，选择第 i 种属性用于标记，这里，属性索引(attribute_index) i=F(r.P)mod v。然后，移动到方框 32，对选择的属性确定哪一个（些）比特要作标记。为此，在一个优选的、非限制性实施方案中，选择属性的第 j 个比特用于标记，这里，比特索引(bit_index) j=F(r.P)mod ξ。没有必要使用连续 ξ 个最低有效比特作标记。例如，那些比特值分布被歪斜的比特位置，必要时可以略去。

之后，在方框 34，用散列函数确定选择的比特值。该值可以称为“水印值”。在一个优选的、非限制性实施方案中，如果 H(K○pk)为偶，则第 j 个最低有效比特的值被置为“0”，否则该比特值是“1”，其中的“pk”是属性的主密钥。

下面的伪码是上面逻辑的一个例子。应当指出，该伪码是按简化公开的内容的形式，而不是按计算上最有效的形式写的。

```
//保密密钥(private key) K 仅对于数据库的拥有者是公知的。
//参数γ、v、和 ξ 对于拥有者也是保密的。
1) for each tuple r∈R do
2)       if (F(r.P)mod ã  equals 0) then // 标记字节组
3)            attribute_index i= F(r.P)mod V /标记属性 A_i
4)                 bit_index j= F(r.P)mod ξ // 标记第 j 比特
```

```
5)              r.A_i = mark(r.P, r.A_i, j)
6)  mark(primary key pk, number v, bit index j) return number
7)   first_hash = H(K○pk)
8)   if (first_hash is even) then
9)      set the j^th least significant bit of v to 0
10)  else
11)     set the j^th least significant bit of v to 1
12)  return V
```

根据以上内容，现在可以明白，作标记是把某一属性的一些值降低，同时增加一些其他的值并保留一些不变。数据库提出允许属性采取无效值。如果遇到一个无效属性值的同时，对一字节组作标记，该标记最好不加在无效值上，保留它不变。无论如何，在一个优选的、非限制性实施方案中，在方框 36，可以用 SQL 更新函数把水印值插入该比特中。然后，逻辑取来下一个字节组，并循环回到判定菱形 28。

如在一个非限制性实施例中所给出的，不论字节组是被标记或与它的主密钥属性无关。结果是，能够插入一字节组，无需考察任何其他字节组的标记。类似地，可以简单地删除一个字节组。当更新一字节组的主密钥属性时，在把该字节组存储进数据库前，可以重新计算它的标记。当更新非主密钥属性时，如果算法没有选择该属性用于标记，那么什么事都不必做。相反，如果该属性是标记的候选者，则最好在属性值存储进数据库之前加上该标记。

可能有这样的情形，在要加水印的关系中不存在主密钥。如果是这样，并假定该关系 R 包括单个数值属性 A，则属性 A 的比特能够分割为两组。值 r.A 的“X”比特用作字节组“r”的“主密钥属性”，而剩余的 ξ 比特用作标记。如果该关系有多于一个属性，其中一个可以用作替代者，其余则用作标记。选择有最小重复的属性作为主密钥的替代者，该替代者还能够在多于一个的属性上展开，以便降低重复性。

现在参照图 3，从图上可见，通过该逻辑，一种测试数据结构能够确

定，例如数据库的全部还是一部分已经被复制。下面将看到，该逻辑是统计性的。

流程在方框 38 开始，由于有保密信息 K 可供使用，进入循环语句用于测试数据结构中的每一字节组。作为鉴别水印的一部分，判定菱形 40 确定，待测试的字节组是否已经加标记。为此，在一个优选的、非限制性实施方案中确定，F(r.P)modγ 是否等于 0。如果是，该字节组应当已经被加标记，它已经通过图 2 的算法处理。如果在判定菱形 38，对标记的测试没有被满足，方法循环回到获得下一个字节组。

否则，如果该字节组已经被加标记，逻辑前进至方框 42，在方框 42 确定，字节组的哪一种（些）属性已经被加标记，图 2 的加水印已经实施。为此，在一个优选的、非限制性实施方案中，指定第 i 种属性已经被选择用于标记，这里，属性索引(attributr_index) i=F(r.P)mod v。还有，指定属性的第 j 个比特已经被选择用于标记，这里，比特索引(bit_index) j=F(r.P)mod ξ。

于是前进到判定菱形 44，方框 42 选择的比特的实际值，与用保密信息 K 加了水印的数据结构应当有的值比较。如果在方框 46 是符合的，计数递增。在判定菱形 48 确定，最后的字节组是否已经测试。如果否，逻辑循环回到判定菱形 40，检索下一个字节组。否则，循环前进至判定菱形 50，确定计数是否超过 Õ。如果否，在方框 52 发回“未发现水印”。否则，在方框 54 发回“已发现水印（怀疑非法复制）”。

下面的伪码表明一种水印检测逻辑的非限制性实施方案。

```
//K、γ、v、ξ 具有与水印插入所用的值相同的值。
//α 为检测器预选的测试显著度。
1)    totalcount=matchcount=0
2)    foreach tuple s in the test data structure S do
3)     if (F(s.P)mod γ equals 0)then // 标记字节组
4)            attribute_index i= F(s.P)mod v // 标记属性 A_i
5)            bit_index j=F(s.P)mod ξ //标记第 j 比特
```

6) totalcount= totalcount+1

7) matchcount= matchcount+ match(s,P,s.A_i,j)

8) τ=threshold(totalcount,α)//

9) set the j^{th} least significant bit of v to 0

10) if(matchcount ≥ τ)then suspect piracy

11) first_hash=H(K○pk)

12) if(first hash is even)then

13) return 1 if the j^{th} least significant bit of v is 0 else return 0

14) else

15) return 1 if the j^{th} least significant bit of v is 1 else return 0

图 3 的检测逻辑能够用用户定义的集合函数，按照图 3 所示逻辑实现，该检测函数如果发现水印，发回“1”，否则发回“0”。

在上面的伪码中，第 8 行使用的阈值子程序，可以说明如下。假定图 3 的逻辑从“疑似”（测试）数据库中测试 ω 个字节组。可以认为这些测试是 Bernoulli 试验，该逻辑在特定比特位置成功发现查找值的概率是 1/2。第 8 行的子程序发回一极小值 τ，使得 ω 次试验至少 τ 次成功而发回正确结果的概率小于α。α 的显著性确定该系统没有命中有多少依据，α 越小，不正确地把无辜的测试数据库识别为有水印的机会越小。

在图 3 的逻辑中，如果已经作标记的字节组属性被省略，该字节组就被忽略。同样，如果发现一字节组本应作标记，但其属性有无效的值，该字节组就被忽略。无论如何，匹配计数和总计数不受影响。

应当指出，检测算法是盲目(blind)的，因为它简单地从数据抽取信息比特，不要求接入原始数据或水印来作出判定。这一点对数据库关系是重要的，因为关系常常更新。不用该盲目的检测，关系的每一版本必须保持，因为要求检测原来版本的水印。

在一个特定的非限制性实施方案中，如上所述，借助选择表中指定的属性 P、A，能够通过首先检索 R 的字节组插入水印。选择的陈述包含附加的子句“对 A 的更新”，该子句能使数据库引擎了解选择的“r”字节组

将被更新。对这样取数据的每一字节组“r”，如果水印算法确定该“r”不在间隙之中并需要改变 r.A 的值，那么可以向标记 r.A 发出更新的陈述。该更新陈述(statement)可以有“当前光标”的子句，该子句能使数据库引擎了解要更新的字节组是 r。

前面已经公开，水印检测可以用选择的陈述实现，以便取出怀疑的数据库关系 S 的字节组，指定选择表中的属性 P、A。如果得到的字节组“s”不在间隙中，“总计数”的计数递增。如果“s”包含标记，“匹配计数”的计数也递增。当已经处理了所有的字节组，检测算法确定，在“总计数”字节组中发现“匹配计数”标记的概率是否在显著水平之内。如果是，则已经检测到水印。

与数据能允许的误差有关，选择所谓“间隙”γ 的值，迫使不小心的复制者在试图擦除水印中犯下大的错误，令该复制者的数据不合用。间隙 γ 是个控制参数，它以 $\omega \approx \eta/\gamma$ 确定标记的字节组数。可以定出一种 γ 对 ξ 的折衷方案，该方案确定引进属性值的范围。就是说，如果作标记的字节组越少，则被标记属性值的改变范围越大。还有，选择 ξ，使不小心的复制者猜测该值太高，会导致大的误差，而如果他低估该值，他破坏水印的成功机会则降低。类似地，v 的建立是为挫败不小心复制者的攻击。

概括地说，α 值的降低，降低了命中失败的机会，但在复制、可能的数据库改变中，增加了水印丢失的机会。降低 γ 值，增加系统对抗攻击的牢固度，但增加了水印数据中的数据错误。增加 v 和 ξ 值，则增加牢固度，但对 ξ 来说，也增加数据中的错误。

图1

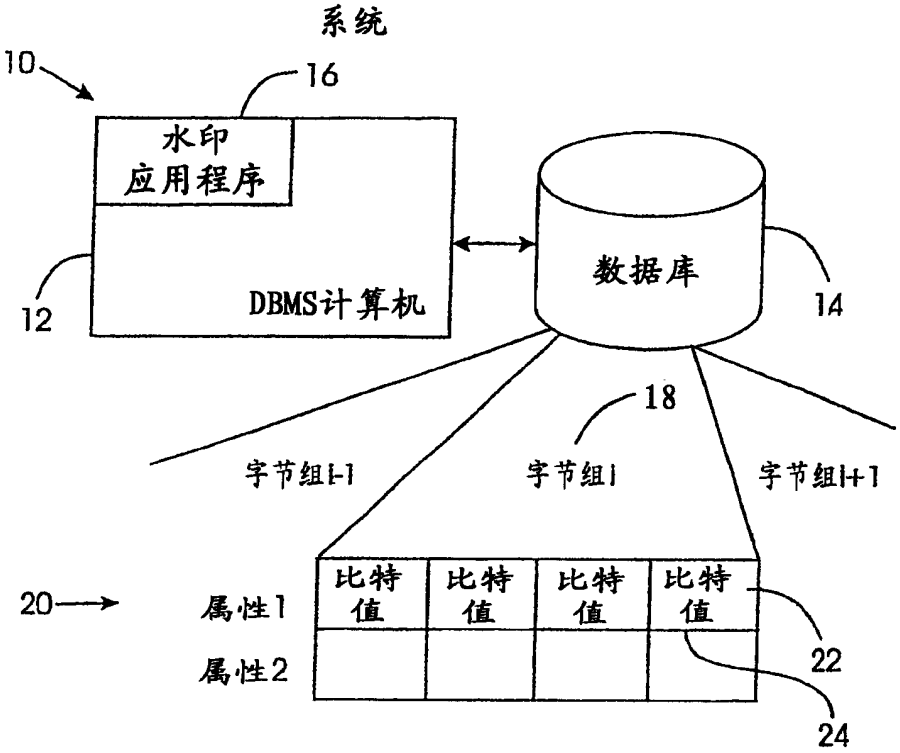

系统
10
16
水印
应用程序
DBMS计算机
12
数据库
14
18
字节组I-1
字节组I
字节组I+1
20
属性1
属性2
比特值
比特值
比特值
比特值
22
24

图2

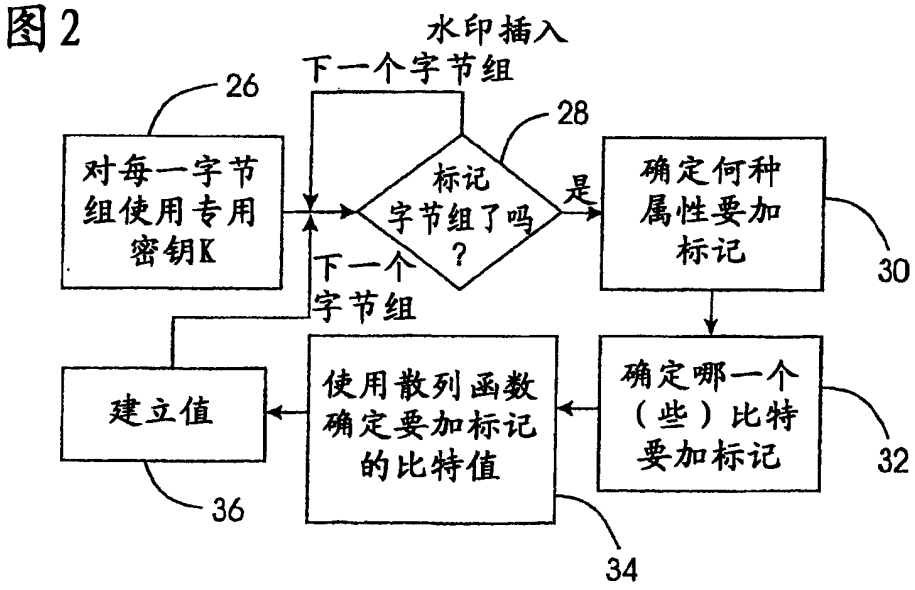
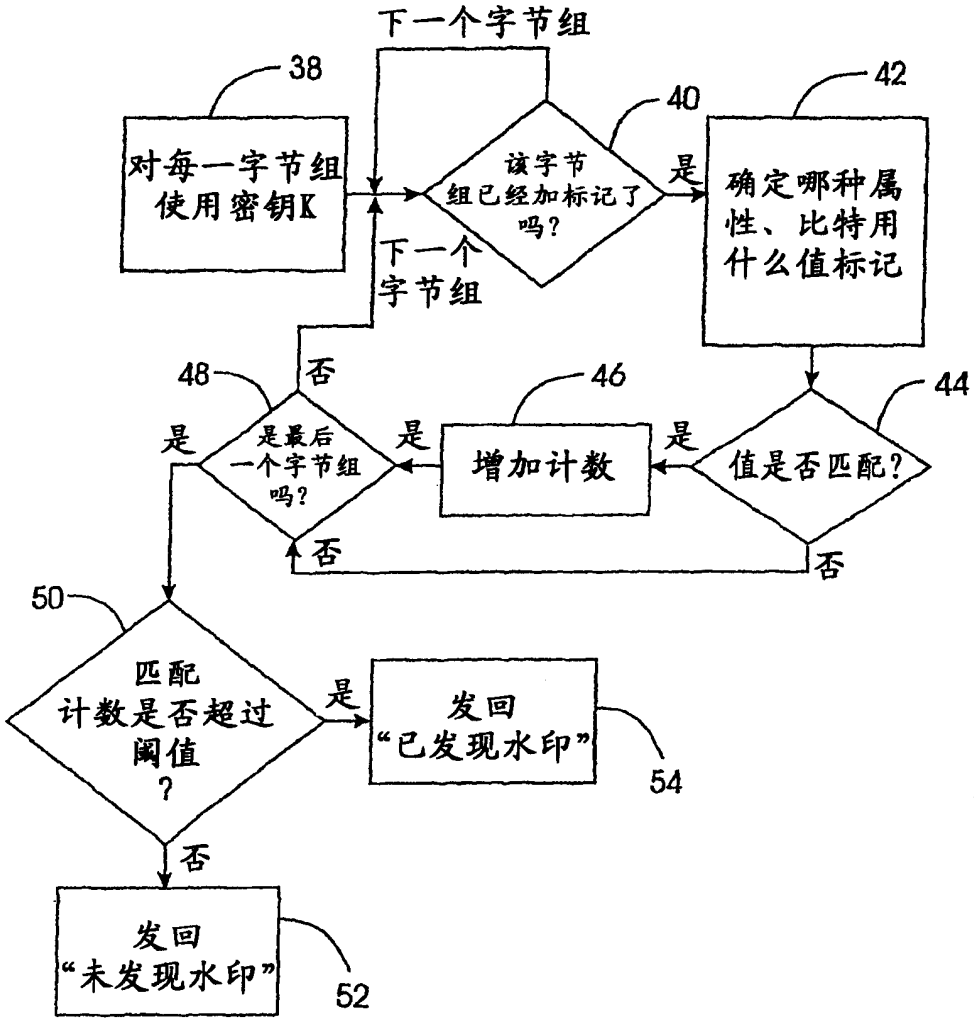

水印插入
下一个字节组
26
对每一字节组使用专用密钥K
28
标记字节组了吗？
是
确定何种属性要加标记
30
下一个字节组
确定哪一个（些）比特要加标记
32
使用散列函数确定要加标记的比特值
34
建立值
36

图3
水印检测


下一个字节组
38
对每一字节组使用密钥K
40
该字节组已经加标记了吗?
是
42
确定哪种属性、比特用什么值标记
下一个字节组
否
48
是最后一个字节组吗?
是
46
增加计数
是
44
值是否匹配?
否
否
50
匹配计数是否超过阈值?
是
发回"已发现水印"
54
否
发回"未发现水印"
52