学号 2017282110198

密级 非密

海量存储技术专业课结课论文

NoSQL 数据库重复数据删除技术

院（系）名 称：计算机学院

专 业 名 称 ：计算机技术

学 生 姓 名 ：朱晓南

指 导 教 师 ：何水兵副教授

二〇一七年十二月

摘　要

随着数字信息的爆炸式增长，企业数据库规模不断增大，管理数据的成本也越来越高，研究发现，应用系统所保存的数据中高达 60%是冗余的，而且随着时间的推移越来越多。此外，在独立管理的数据库中，不仅数值，而且数据的结构，语义和基本假设也可能不同，减少合并后数据库的数据冗余也是一项难题。传统关系型数据库很难处理海量数据，以 NoSQL 数据库为代表的新型数据库得以重视，而 NoSQL 数据库中也存在数据冗余，目前现有的重复数据删除技术该如何应用于 NoSQL 数据库。

关键词：重复数据删除技术；NoSQL；海量数据存储

ABSTRACT

With the explosion of digital information, the size of corporate databases is growing and the cost of managing data is getting higher and higher. The study found that as much as 60% of the data held by application systems is redundant, and as time goes on more and more. In addition, in an independently managed database, not only values, but also the structure, semantics, and underlying assumptions of the data may be different, reducing data redundancy in the consolidated database is also a challenge. The traditional relational database is hard to deal with the huge amount of data, the new database represented by NoSQL database is valued, and the data redundancy exists in NoSQL database. How to apply the existing deduplication technology to NoSQL database.

Key words: Data deduplication technology; NoSQL; Mass data storage

目　录

1 重复数据检测

数据库在当今的信息技术经济中发挥着重要的作用。许多行业和系统依靠数据库的准确性来进行操作。因此，存储在数据库中的信息（或缺少信息）的质量可能对依靠信息来运行和开展业务的系统具有重大的成本影响。在一个完全清洁数据的无错系统中，数据的综合视图通过连接操作，合并段上的两个或多个表格。然而，数据缺乏一个独特的全局标识符来允许这样的操作。而且，这些数据既没有严格控制质量，也没有以不同数据来源的一致方式进行定义。因此，数据质量往往受到很多因素的影响，包括数据输入错误（例如 Microsft 而不是 Microsoft），缺少完整性约束（例如，允许像 EmployeeAge = 567 这样的条目）以及用于记录信息的多个约定（例如 44W 第四街与西四街 44 号）。更糟糕的是，在独立管理的数据库中，不仅数值，而且数据的结构，语义和基本假设也可能不同。

通常，在整合来自不同来源的数据来实施数据仓库时，开发者会意识到潜在的系统性差异或冲突。这些问题归结于总体数据异质性。

重复记录检测是识别涉及一个独特的真实世界实体或对象的不同或多个记录的过程。典型地，重复检测的过程之前是数据准备阶段，在该阶段中数据条目以统一的方式存储在数据库中，至少部分解决结构异质性问题。数据准备阶段包括分析，数据转换和标准化步骤。这些步骤改善了流入数据的质量，使数据具有可比性，并且更加可用。

解析是数据准备阶段的第一个关键组件。解析定位，识别和隔离源文件中的单个数据元素。解析使得更容易纠正，标准化和匹配数据，因为它允许比较单个组件，而不是长时间复杂的数据串。例如，将适当的名称和地址组件解析成一致的信息包是数据清理过程中的关键部分。数据转换指的是可以应用于数据的简单转换，以使它们符合其相应域的数据类型。换句话说，这种类型的转换重点是一次操作一个字段，而不考虑相关字段中的值。标准化最常见的形式是将数据元素从一种数据类型转换为另一种数据类型。当遗留或父应用程序将数据存储在数据中时，通常需要这种数据类型转换。

2 相同数据检测技术

相同数据检测技术是将数据进行划分，找出相同的部分，并且以指针取代相同数据的存储。

2.1 完全文件检测技术

WFD 技术是以文件为粒度查找重复数据的方法。如图 1 所示，首先对整个文件进行 hash 计算，然后将该值与已存储的 hash 值进行比较，如果检测到相同的值，则仅将文件用指针替换，不进行实际存储，否则存储新文件。研究者将 hash 算法引入到重复数据删除技术中，是利用了 hash 值可以唯一地表征特定的数据实体，通过 hash 技术，数据被标志为一个固定大小的值，比较该值，就可以判别数据的重复性。目前 MD5 和 SHA1 是应用最广泛的 Hash 算法，两者的抗冲突性都比较好。

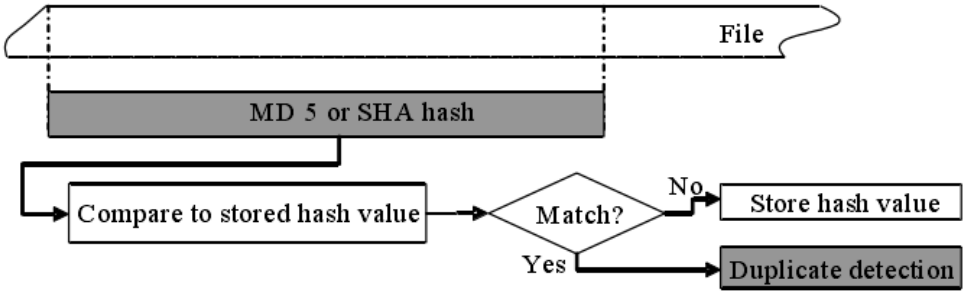

图 1 WFD 技术

Windows 2000 的 SIS(single instance storage)应用该技术对具有 20 个不同 Windows NT 映像的服务器进行测试,结果表明总共节省了 58%的存储空间。

基于 hash 算法的完全文件检测技术具有两个优势：(1)在普通硬件下计算速度很快；(2)可以检测到所有完全相同的文件，节省存储空间较大。但是，该方法也有两个主要缺点：(1)对于较大的数据集，需要比较的范围大，耗时多；(2)不能检测不同文件内部的相同数据。

2.2 基于 FSP 算法的块检测技术

完全文件检测技术不能用于文件内部的重复数据查找,因此有研究者提出了更细粒度——块级别的重复数据检测。基于固定尺寸划分算法(FSP)的相同数据

块检测技术是使用固定大小的分块策略在存储系统中识别相同数据的方法。如图2所示，该方法分3个步骤：(1)提供一个已经预先定义好的块的大小(该值独立于所存取的数据内容)，所有文件均按照这个固定的块大小进行划分；(2)每个划分好的数据块均通过哈希算法(MD5或SHA1)得到一个指纹值；(3)将该值与已存储的块指纹值进行比对，如果检测到相同的值，则删除其代表的数据块，否则存储新的数据块。

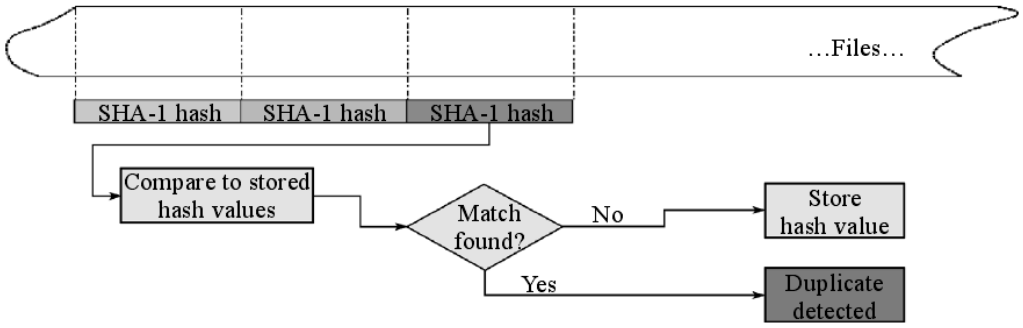

图2 FSP技术

基于FSP算法的相同数据块检测技术已应用在很多领域，并具有如下两个特征：(1)缩减存储空间；(2)减少网络传输的数据量。

此外，基于FSP算法的相同数据检测技术还可以提供很高的处理速度，适合于在交互性的环境中应用。但是它也具有一定的局限性：对编辑和修改的序列很敏感，对于插入问题(在原来的数据流中某处插入少量新字节，其他部分不变)和删除问题(在原来的数据流中某处删除少量新字节)处理十分低效，不能智能地根据文件自身内容的变化和文件之间的关联关系进行调整与优化，基于此，研究者们提出了可变块大小划分的检测技术。

2.3 可变分块检测技术

2.3.1 基于CDC算法的检测技术

CDC算法是应用Rabin指纹将文件分割成长度大小不一的分块策略。与固定分块策略不同的是，它对文件进行块划分的方法是基于文件内容的，因此数据块大小是可变的，如图3所示，其过程有两步：(1)一个文件按照CDC算法分为若干数据块。CDC算法首先从文件头开始，将固定大小(互相重叠)的滑动窗口中的数据看成组成文件的各个部分。在窗口的每个位置，该窗口中数据的一个指纹

(Fingerprint)被计算出来。在实际中，鉴于 Rabin 指纹计算的高效性及 Rabin 指纹函数的随机性(对任意数据呈现出均匀分布)，研究者们多使用其计算滑动窗口内容的指纹值。当指纹满足某个条件时，如当它的值模某个指定的整除数为 0 时，则把此时窗口的位置作为块的边界。重复这个过程，直到整个文件数据都被分为块。(2)划分出的每个块用 hash 函数(MD5，SHA-1 或更高的 SHA 标准函数)计算出它的指纹值并与已存储的数据块进行对比，如果检测到相同的指纹值，则删除其代表的数据块，否则存储新的数据块。

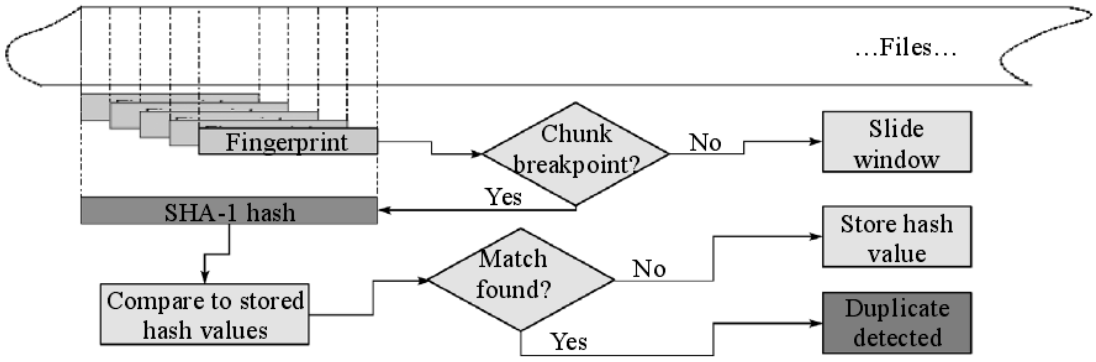

图 3 基于 CDC 算法的检测技术

当前可变分块检测技术已应用在 P2P 文件系统 Pasta、Pastiche 备份系统、基于值的网络缓存系统、DeepStore 归档存储系统和低带宽网络系统 LBFS 中。在 LBFS 中，系统针对数据分块可能出现的病态现象(滑动窗口数据的指纹值不能符合条件，因此块的边界不能确定，导致块过大)进行了改进，将块大小进行了限定，给出数据大小分块的上、下限。

综上，根据 CDC 算法的特性，无论是插入还是删除一小部分字节，都只会影响一到两个块，其余的块保持不变，即 CDC 方法在两个相似对象(只相差几个字节)之间可以检测出更多的冗余。

但是，CDC 算法也存在一定的局限性，它划分的粒度绝大部分取决于期望块的设定。如果该值设置得较小，那么，虽然粒度较细，重复数据查找较为精确，但是额外存储每块信息的开销很大；反之，如果该值设置得较大，则粒度过粗，重复数据删除的效果不好。所以，如何权衡精确查找和额外开销是一个难点。

2.3.2 基于 fingerdiff 算法的检测技术

为了弥补 CDC 算法额外存储空间开销大的缺陷，研究者提出了 fingerdiff

算法，其核心思想是将没有变化的块尽可能地合并，以减少各个数据块的元数据占用的存储空间。如图 3 所示，应用 fingerdiff 算法进行相同数据块检测过程包含 3 个步骤：(1)一个文件按照 CDC 算法进行子块的划分。(2)每个子块按照 fingerdiff 设置的最大子块数进行合并。(3)每个块用 hash 函数计算出它的指纹值，然后对比已存储的数据块指纹值，如果检测到相同的指纹值，则删除其对应的数据块；否则将大块进行拆分，找到最小的不同数据块进行存储，其余块仍然保持合并状态。

以上 CDC 算法和 fingerdiff 算法的检测技术存在两个不足之处：(1)在可变分块检测过程中，对于一个文件对间较小的随机改变，效果不好，两种算法的适应性很差；(2)两种技术中数据块的划分都是根据内容可变的，但该值在很大程度上取决于算法中期望块大小的设定，而期望块大小的选择依赖于文件相似性程度和文件修改的位置，这对相同数据检测的性能影响很大。因此，可以通过深入挖掘数据特征规律，自适应地调整数据块的大小，选择符合数据特征和性能指标的最佳块大小。

2.4 滑动块检测技术

滑动块检测技术结合了固定块大小检测技术和可变块大小检测技术的优点，块大小固定，管理简单。文献通过测试发现，对大的簇，CDC 的重复数据检测性能较好，而滑动块技术对细粒度匹配更适用。基于滑动块技术的相同块检测过程有 4 步，如图 4 所示：(1)一个文件用 rsync 求和校验(checksum)函数和固定块大小的滑动窗口来计算文件对象的每个重叠块的求和校验值。(2)对于每个块，比较求和校验值与先前存储的值。(3)若匹配，则利用更严格的 SHA-1 算法对块进行 hash 计算，并将 SHA-1hash 值与先前存储的值进行比较，从而进行冗余检测。如果检测到数据冗余，将其记录后，滑动窗口越过这个冗余块继续前移。而且先前已经被划分的块和最近被检测到冗余之前的这个碎片需要被记录并且存储。(4)如果求和校验值或 hash 值不能匹配，则滑动窗口继续前移。如果滑动窗口已经移动了一个块大小的距离，但是仍然无法匹配到任何已经被存储的块，则需要对这个块进行求和校验和 SHA-1hash 计算，并存储在各自的表中，用作以后块的比较对象。

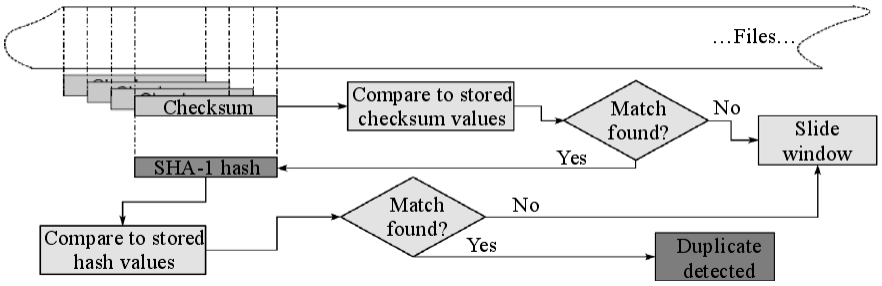

图 4 滑块分块检测技术

滑动块检测技术的特点是对插入问题和删除问题处理高效，并能检测到更多的冗余数据。(1)插入问题：如果一小部分字节被插入到一个对象中，则只有周围的块会改变，其他的块仍将被识别出来并被匹配，并且一个长度等于插入的字节数的碎片会产生出来。(2)删除问题：删除一小部分字节也会产生一个长度等于块大小减去被删除部分字节长度的碎片，其他块不受影响。但滑动块检测技术也存在一个不足：在插入和删除问题中都会引入碎片，如何能够准确识别改变的数据，不影响匹配数据块，从而少产生额外的碎片将是一个研究的难点。

3 NoSQL 数据库

大数据存储与管理要用存储器把采集到的数据存储起来，建立相应的数据库，并进行管理和调用。重点解决复杂结构化、半结构化和非结构化大数据管理与处理技术。大数据存储及管理技术的重点内容是开发可靠的分布式文件系统(DFS)、能效优化的存储、计算融入存储、大数据的去冗余及高效低成本的大数据存储技术；突破分布式非关系型大数据管理与处理技术，异构数据的数据融合技术，数据组织技术；突破大数据索引技术，突破大数据移动、备份、复制等技术。目前出现了几类大数据存储和管理数据库系统，NoSQL 数据库就是其中之一。

NoSQL（Not Only SQL）是一项全新的数据库革命性运动。数据库分为关系型数据库、非关系型数据库以及数据库缓存系统。其中非关系型数据库主要指 NoSQL 数据库，当前主要有以下四种：键值存储数据库、列存储数据库、文档型数据库和图形数据库。NoSQL 数据库在以下几种情况下较适用：（1）数据模型比较简单；（2）需要灵活性更强的 IT 系统；（3）对数据库性能要求较高；（4）不需要高度的数据一致性；（5）对于给定 key，比较容易映射复杂值的环境。

NoSQL 数据库运用非关系式的方法解决传统数据库无法解决的问题，而并非要取代现在广泛应用的传统关系式数据。Nosql 遵守 CAP 原则和 BASE 思想，CAP 原则，指的是在分布式系统中，只可以同时满足 Consistency（一致性）、Availability（可用性）、Tolerance（区分容错性）其中的两种要求，不能三种兼顾，因此，不同的 NoSQL 数据库会根据自身的开发目的选择满足哪些要求，比如，Mongodb 满足 CP 要求。BASE 是基本可用（Basically Available）、软状态（Soft state）、最终一致性（Eventually consistent）三个术语的缩写，基本可用性是指在分布式系统出现故障时，同意系统部分失去可用性，保证核心部分的可用性，软状态是指同意系统不同节点同步有延时，最终一致性系统所有数据在最后能达到一致的状态的性能。大部分 NoSQL 数据库都遵循 BASE 思想，舍去高一致性得到可用性和可靠性。

NoSQL 数据库种类繁多，如果只用一个 NoSQL 标签来代表所有 NoSQL 数据库就太笼统了，比如 memcached 和 mongodbt 这两种数据，尽管在都是 NoSQL 数据

库的同类比较下彼此也会显示出很大的区别，所以，NoSQL 数据库大致可以分为以下几类：

键值型数据库，该数据库会使用哈希表，数据以键值的形式存放，一个或多个键对应一个值。键值型数据库处理速度最快，但是必须通过匹配完全一致的键查询数据。

列存储数据库，以列为单位存放数据。目前大部分关系型数据库是以行为单位存放数据，当面对大量数据时，以行为单位的数据库操作会更加困难，写入速度降低。对以列存储的数据库来说，可以对大量数据进行读取，具有高扩展性，但因为思维方式与传统型数据库多有不同，应用困难。

文档型数据库，将数据封装存储到未严格定义的以 JSON、XML 等类型的文件中，虽然它与键值型数据库有相似点，每个文档存储一个或多个键值，但不同的是，其中值可以是文件类型。在文档型数据库中，即使没有提前定义数据表结构，也可以继续使用。键值型数据库必须通过匹配完全一致的键查询数据，文档型则可以通过复杂的查询条件进行操作。但是文档型数据库没有事务处理能力。

图形数据库，起源于欧拉公式和图论，应用图论的节点、关系、属性三个基本要素存放数据之间的关系信息，将点、线、面等基本图形元素按照一定结构排列的数据集合，在此类数据库中，程序员可以任意添加属性、节点、关系且不影响系统的初始状态，适用于处理复杂的、相互交叉的数据，解决复杂的图形问题。

4 NoSQL 数据库的重复数据删除

NoSQL 数据库通常遵循一个简单的数据模型，对数据布局和表单进行动态控制。与高度规范化的关系数据库相比，由于其设计更简单，往往会有大量的重复数据。在使用 NoSQL 数据库的分布式环境中需要冗余，但通常通过复制来以受控的方式完成冗余。然后将这些重复的数据进一步传播到备份中，从而增加存储需求。虽然在关系数据库管理系统的数据库级别上使用重复数据删除是没有意义的，但是在 NoSQL 数据库中使用它却很有意义。使用键值数据库作为其 NoSQL 数据库重复删除实现的代表结构，它是目前使用最广泛的 NoSQL 数据库类型，尽管具有最复杂的数据模型。

方法可以分为两大类：硬件重复数据删除方法和软件重复数据删除方法。对于软件级别的备份系统，可以根据以下位置来区分另外两个类别：源重复数据删除在要备份的服务器上标识重复数据，在通过网络发送数据之前，或在将重复数据删除提供给备份服务器的目标作为 Network Attached Storage（NAS）共享或虚拟磁带库（VLT）。

现有的重复数据删除技术降低了存储和网络流量的成本，使其价格更便宜，但是许多公司仍然在努力定期完成备份。数据丢失的情况与遭遇飓风类似，留下了一个可能无法恢复的企业。由于所有数据大部分都是以文件形式组织的，因此现有技术大多使用文件和子文件重复数据删除策略。各种组块策略提供更高或更低的重复数据删除比率，具体取决于数据类型以及策略对内容的了解程度。但是这已经被证明是不够的。使用元数据信息来帮助识别重复数据，在数据库层面进行重复数据删除的想法可以与现有的重复数据删除方法并行使用，并将增加重复数据删除比率。

4.1 NoSQL 数据库数据模型

在许多学科中，表格是组织和表示数据的最常用的约定之一，而数据库则是“格式”。与关系数据库相比，NoSQL 数据库与此角度相差无几。根据它们的设计目的，区别在于它们的物理布局。面向行的表示将行值存储在一起，而面向列的

表示将列值存储在一起。它们的逻辑表示仍然可以作为具有列和行的表，尽管一个单元也可以是具有其自己的结构的非常复杂的对象。在他们的逻辑表示中，根据他们设计的需求，NoSQLDB 具有一列或多列，有时更复杂的列组等等。

键值数据库具有非常简单的数据模型，并按行存储它们的数据。条目在大型散列表中作为键值对存储。数据域与关系数据库表类似，但没有定义特定的模式。键是任意的，而值是大的大对象。数据域之间没有明确的关系。因此，为了使这种灵活性水平反映出来，因为没有关于数据库中的数据的结构信息本身就要求应用程序具有这种知识。隐含的是，在数据库级缺乏结构会导致大量重复的数据。

4.2 NoSQLDB 元数据

基于块的数据重复数据删除是减少主文件系统和数据备份空间要求的主导技术。这种技术在文件和子文件级别接近重复数据删除。这个重复数据删除技术涉及两个步骤：分块-将数据分成不重叠的数据块-重复检测-每个块与所有其他存储的块进行比较，以检测它们是否具有完全相同的内容。为了更好地识别重复的数据，已经开发了各种精细的分块技术，并且以合理的结果应用于数据文件。在数据库上下文中，利用关于数据结构（元数据）的信息对应于“分块”技术。

4.2.1 重复检测

代表找到重复值并用指针替换它们。重复的检测步骤可以分成两个子步骤：（1）为每个数据块计算散列；（2）比较数据块的散列值以检测重复。为了避免在文件/子文件重复数据删除过程中用作比较单元的文件块之间的混淆，将键值存储的比较单位称为“数据集”。例如，元组{f，g}可以是一个数据集，这意味着存储在该元组位置和格式中的数据将跨表行进行比较。比较准确性的一个重要方面是选择抗碰撞散列函数。这使得两个不同输入产生相同输出的概率很低，以至于假设每个块具有唯一的散列值是可行的。这项研究使用 MD5（消息摘要算法），这是一种广泛使用的加密散列函数，可生成 128 位（16 字节）散列值。为了研究的目的，丢失数据的风险没有重大的后果，并且该算法被认为是可接受的。

4.2.2 重复数据删除比

重复数据删除比或“空间缩减比”由分数表示：

比率=字节输入/字节输出

这表示输入到数据重复数据删除进程的字节数除以输出的字节数，通常描述为“ratio：X”。例如，如果 100GB 的数据消耗 10GB 的存储容量，则空间缩减比率为 10：1。一个直观的假设是关于数据结构的信息越多，可以使用的比较单位越小意味着识别重复数据的可能性越高。由于各种合法原因，NoSQL 数据库的结构非常有限。数据库层面的结构信息缺乏可通过将其纳入应用程序来弥补。根据应用需要，该信息可以具有不同的粒度，对于具有所有的信息根本没有任何信息。

4.2.3 数据重复数据删除与 MapReduce

DDNSDB 的体系结构共享了 MapReduce 的大部分设计原则。它遵循主从设计的主人节点负责管理作业，即启动工作节点，并分配映射/规约任务。每个工作人员可以在任何时候进行映射或规约任务。作业执行通过分割输入数据并将其分配给各个映射任务开始。当工作人员完成执行映射任务时，它将映射结果存储为内存中的中间键值表。每个映射任务的中间结果都分配给现有的规约员工。规约任务从所有映射输出中检索相应的中间结果开始，然后应用规约函数。