可信数据库环境下面向服务的自适应密文数据查询方法

宋 伟 彭智勇 程芳权 李文海 胡文斌 任 毅

(武汉大学计算机学院 武汉 430074)

(武汉大学软件工程国家重点实验室 武汉 430074)

摘 要 实现加密数据的高效安全查询是保证可信数据库安全性和实用性的关键.与目前加密数据查询采用的静态密文分段方法不同,论文基于加密数据的分布和用户查询类型、分布规律,提出了一种自适应加密索引 AEI (Adaptive Encrypted Index),实现面向服务的加密数据查询.AEI 通过分析查询服务对查询性能的影响,根据承载服务特性、密文数据分布、用户查询分布采用自适应的加密索引划分策略,获得更好的加密数据查询性能.基于 AEI 方法可在可信数据库环境下实现密文数据查询,并通过了相关性能测试.实验数据表明,与其它加密数据查询方法相比,AEI 方法具有更好的适应性和更高的加密数据查询效率.

关键词 自适应加密索引;DAS 模型;可信数据库;密文数据查询;查询假阳性率

中图法分类号 TP311 **DOI 号**: 10.3724/SP.J.1016.2010.01324

A Service-Oriented Adaptive Search Method over Encrypted Data in Trusted Database

SONG Wei PENG Zhi-Yong CHENG Fang-Quan LI Wen-Hai HU Wen-Bin Ren Yi

(*Computer School*, *Wuhan University*, *Wuhan* 430074)

(*State Key Laboratory of Software Engineering*, *Wuhan University*, *Wuhan* 430074)

Abstract How to implement an efficient and secure search over encrypted data is crucial for the security and practicability of trusted database. Different with current encrypted data search over static bucket method, this paper proposes an adaptive encrypted index (AEI) based on the distribution of encrypted data and users' accesses to achieve an efficient service-oriented search over encrypted data in trusted database. By analyzing the user accesses' effect on search efficiency, AEI implements an adaptive division on encrypted index based on service features, data and access distributions to achieve a better search efficiency. Moreover, this paper implement encrypted data search in trusted database by AEI and do some experiments to evaluate its performances. The experimental results show that AEI is more scalable and more efficient than others.

Keywords adaptive encrypted index; database as service; trusted database; search over encrypted data; query false positive rate

1 引 言

数据库系统是支撑信息系统的基础软件,它的可信性必须得到保证.数据库的可信性主要强调即使在不可信应用环境下,如数据库管理员不可信、存储介质失窃、黑客攻击等,仍然可以保证数据的安全性.目前还没有一个可信数据库的明确定义,我们认

收稿日期:2010-06-11.本课题得到国家自然科学基金(90718027,60873225)、国家"八六三"高技术研究发展计划项目基金(2007AA01Z403)、中国博士后科学基金(20100471145)、湖北省自然科学基金计划重点项目(2008CDA007)、中央高校基本科研业务费专项资金(6082024)、武汉大学博士生自主科研基金(20082110101000038)资助.**宋 伟**,男,1978 年生,博士后,讲师,研究方向为可信数据管理、对等网络.E-mail: songwei@whu.edu.cn.**彭智勇**,男,1963 年生,博士,教授,博士生导师,研究领域为可信数据管理、Web 数据管理.**程芳权**,男,1983 年生,博士研究生,研究方向为可信密钥管理.**李文海**,男,1979 年生,博士,副教授,研究方向为数据库理论.**胡文斌**,男,1977 年生,博士,副教授,研究方向为工作流管理.**任 毅**,男,1973 年生,博士研究生,研究方向为隐私保护.

为可信数据库是在传统多级安全数据库基础上利用加密和密文查询机制进一步阻止来自内部的恶意访问和窃取，保护数据库敏感数据的私密性和查询过程的安全性[1]. 目前构建可信数据库的方法主要是对数据库数据进行加密处理，但是数据加密后会失去原始数据本身所具有的很多特性，如有序性、相似性等，这给可信数据库中的数据查询带来了很大困难和挑战.

本文分析数据库承载服务的特性及其对密文数据查询的影响，给出了一种自适应加密索引 AEI (Adaptive Encrypted Index)，基于数据和用户访问分布进行自适应密文数据分段划分，实现面向服务的高效、安全密文数据查询服务. 与其它可信加密数据库的查询方法相比，AEI 能更好地适应数据的动态变化和用户访问的不均衡性. 针对 AEI 查询方法的相关性能测试结果，进一步验证了相关分析.

本文第 2 节分析并比较一些其他学者在本研究方向上的相关研究工作成果；第 3 节详细介绍提出的面向服务密文数据查询方法；第 4 节给出模拟实验设计对理论方法进行验证，并对实验结果进行分析；最后在第 5 节总结全文并提出下一步需要继续研究的工作.

2　相关工作

数据库作为信息系统的支撑软件，在各种应用系统中几乎无处不在. 随着应用的日益丰富和复杂，企业为维护数据库需要付出大量人力、物力. 2002 年 Hacı gümüş 等人首先提出了 DAS(Database As Service)服务模式的概念[2]，将数据库作为一种外包服务由第三方提供，这种服务模式减轻了企业的维护代价和运营成本，但是数据存储在第三方也带来了一些安全隐患. 目前数据库数据都以明文形式存储，在 DAS 模型的外包服务模式下，难以保证数据的安全性和私密性，DBA 本身的不可靠，黑客突破 DBMS 访问控制保护以及数据库物理文件的失窃等都给数据库数据安全性带来了极大威胁. 如何解决应用系统将数据放在第三方管理的安全隐患，成为可信数据库的研究热点和关键问题.

DAS 模型提出对数据库中数据进行信息加密，以保护数据隐私安全. 数据加密后会失去原始数据本身所具有的一些特性[3-4]，如有序性、相似性等，难以在密文数据上直接进行查询和运算操作，如果对所有加密数据进行解密，再在明文数据基础上进行查询，由于解密操作开销巨大，会极大影响查询性能，而且对密文数据解密也会引起信息泄漏，破坏可信数据库安全性. 如何实现可信数据库环境下密文数据的高效查询成为近年来的研究热点问题.

目前国内外学者和研究人员在可信数据库加密数据查询方面做了大量研究工作，大致可以分为以下 3 类：

(1) 不解密而直接操作密文数据的方法. 这类方法利用秘密同态、保持有序等加密方法，不对密文数据进行解密，直接在密文数据上进行运算操作. Rivest 等使用秘密同态(privacy homomorphism)函数算法对数据进行加密，无须解密直接可以对加密数据进行算术运算操作[3]. 这种方法提高了加密数据操作性能. 但是在现实中构造一个可行的同态加密函数非常困难，而且同态加密方法本身也存在一定安全隐患. Agrawal 等提出了一种保持有序的加密方法[4]，但加密数据保持有序性，容易遭到选择密文攻击. Ozsoyoglu 等提出了另一种保持有序的加密方法[5]，也存在一定安全隐患，容易遭受已知明文攻击和统计攻击.

(2) 快速解密的方法. 这类方法通过快速解密可信数据库存储的密文数据，提高系统查询性能. 目前的研究成果有子密钥加密、智能卡加密等. David 等基于中国剩余定理提出了子密钥加密技术[6]，这种方法在解密密文数据时，只需进行一次模运算. Ge 等提出了一种轻量级数据加密机制 FCE[7]，Bouganim 等采取智能卡技术实现加密数据快速查询[8]，但是它不能对实数类数据进行范围查询. 这种方法主要对加密算法本身进行改造，使之适应可信数据库应用需求，但加密方法经过改造后存在一定安全隐患，而且数据在服务器端解密，也会破坏可信数据库存储数据的私密性.

(3) 利用密文数据分段过滤的方法. 这类方法中可信数据库服务器进行中间处理得到一个包含所有结果数据的较小中间结果集，由客户端对中间结果集进行解密过滤获得最终结果，从而减少用户解密工作量，提高查询性能. Hacigumus 等在 DAS 模型基础上，对密文数据构建分段索引，缩小解密范围，返回客户端的记录集包含一些不满足查询条件的记录，需要由客户端进行解密过滤处理[9]；Hacigumus 基于这一方法又研究了如何实现加密数据的聚集操作运算[10] 以及密文分段的优化方法[11]. Wang 等提出了一种桶划分的自适应调整方法 STBucket[12] 减小查询的假阳性率.

此外，还有一些其它加密数据查询方法研究，戴

一奇等给出了一种在非同态密文上建立索引的方法完成数据的快速检索[13].这种方法十分适用于单条件密文检索,同时经过处理也能适用于复合条件的查询.文献[14]利用查询重写方法实现属性粒度的数据库加密,降低网络通信量提高系统性能.秘密信息检索 PIR[15-17]保证用户从数据库获得信息,而服务器并不知道用户究竟查询什么信息,保护用户查询的私密性.

可信数据库密文分段查询方法[9]使解密操作只在客户端进行,保证了数据的安全性,但并没有研究密文索引分段对查询效率的影响.文献[11]研究了索引优化方法,但没有考虑用户查询差异带来的影响.目前密文数据分段普遍采用静态划分方法,还无法支持可信数据库数据的更新.文献[12]提出了一种自适应的密文桶划分调整方法,但并没有考虑不同类型查询对密文查询带来的影响.本文首先分析密文数据分布和承载服务特征对密文查询效率的影响,进而提出了一种面向服务的自适应密文分段索引 AEI,与其它研究方法相比,AEI 结合数据分布和用户访问服务特征自适应地进行密文数据分段划分,可以更好地适应数据库数据的更新,同时对不同类型数据查询服务也具有更好的适应性和更高的查询效率.

3 面向服务的自适应密文数据查询方法

3.1 可信数据库环境下的加密数据过滤查询

数据库中存储了不同类型(主要是字符型和数值型)的数据,它们承载的查询服务各不相同,而加密数据的查询差别就更大了,论文主要研究可信数据库中数值型密文数据的高效、安全查询方法.

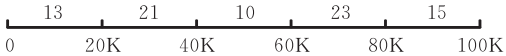
相关研究[2,9,11]对加密属性值划分密文数据分段,服务器收到查询请求后把密文数据分段内的全部密文元组返回用户,这种方法返回的结果集包含所有正确结果,但也有不符合查询条件的内容,由用户解密过滤获得最终结果.如图 1 可信数据库,原始数据加密后的密文元组如图 1(b),*salary* 数据内容采用 $esalary=Map(salary)$ 函数加密存储,*salary* 的密文映射数据分段如图 1(c).当用户发起针对 *salary* 的查询(如 $salary=55K$)时,首先将查询值映射到对应的密文数据分段上,服务器将所有可能元组((40K,60K)分段上的全部密文元组)返回客户端,由客户端解密过滤得到最终查询结果.

id	*Name*	*salary*	*Address*
0012	Tom	75K	Maple
0013	Mary	60K	Main
0014	John	50K	River
0015	Jerry	55K	Hope
0016	Michael	85K	City

(a) 可信数据库原始数据

etuple	*eid*	*eName*	*esalary*	*eAddress*
110101100100…	15	18	23	10
001000110010…	86	11	10	18
100100010110…	47	24	10	14
001011100011…	23	23	10	19
010010011100…	12	17	15	21

(b) 可信数据库密文数据



(c) *salary*加密数据分段

图 1 可信数据库查询

这种查询方法服务器端无需解密,但客户端需要对全部中间结果进行解密过滤,上例服务器端返回(40K,60K)上 3 条密文元组,客户端只有全部解密后才能获得最终的 1 条正确匹配元组.这种查询方式下加解密操作成为决定查询效率的关键.论文利用式(1)所示假阳性率(false positive rate)[11]、误报率即中间结果集的不匹配元组比例,来反映客户端无效解密操作量.针对某查询集合,N_{fit} 为符合查询条件的密文元组数,N_{total} 为返回的全部密文元组规模.如何调整加密数据分段,降低查询假阳性率,减少客户端无效解密操作量,成为论文优化可信数据库查询的主要手段和目标.

$$P=1-\frac{N_{\text{fit}}}{N_{\text{total}}} \tag{1}$$

3.2 面向服务的自适应密文数据查询

不同类型用户查询对加密数据查询方法、效率都有很大影响,论文将用户查询分为以下两类加以研究:(1) *attribute op value* 查询;(2) $attribute_1$ *op* $attribute_2$ 连接查询;其中 *attribute* 为可信数据库某加密属性,*value* 是针对 *attribute* 的查询值,*op* 是查询操作符,可以为“=”、“>”、“<”、“≥”和“≤”等.

3.2.1 *attribute op value* 查询

(1) *op* 为“=”的 *attribute*=*value* 查询操作

针对一系列 *attribute*=*value* 查询,设当前加密属性 A 的密文数据分段为 $Range_1,\cdots,Range_k$ 共 k 段,每个数据分段的密文元组规模为 $M_1,\cdots,M_k$,查询值落在各个密文数据分段内的次数分别为 $Q_1,\cdots,Q_k$,每个密文数据分段返回命中的正确结果总

数为 $H_1,\cdots,H_k$，则查询总假阳性率如式(2).

$$P=1-\frac{\sum_{i=1}^{k}H_i}{\sum_{i=1}^{k}M_iQ_i} \tag{2}$$

论文采用自适应索引划分策略，随着用户查询和数据的变化动态调整密文分段划分. 首先分析密文数据分段划分对查询假阳性率的影响，设将当前第 i 个密文数据分段重新划分为两个新的数据分段，各自包含密文元组数 M_{i1} 和 M_{i2} ($M_{i1}+M_{i2}=M_i$)，承载查询服务量 Q_{i1} 和 Q_{i2} ($Q_{i1}+Q_{i2}=Q_i$)，正确命中查询结果总数为 H_{i1} 和 H_{i2} ($H_{i1}+H_{i2}=H_i$)，在其它密文数据分段不变的前提下，划分前后的假阳性率变化如式(3).

$$\Delta P=\frac{\sum_{t=1}^{k}H_t}{\sum_{t=1}^{i-1}M_tQ_t+M_{i1}Q_{i1}+M_{i2}Q_{i2}+\sum_{t=i+1}^{k}M_tQ_t}-\frac{\sum_{t=1}^{k}H_t}{\sum_{t=1}^{k}M_tQ_t}$$
$$=\frac{(M_{i1}Q_{i2}+M_{i2}Q_{i1})\sum_{t=1}^{k}H_t}{\sum_{t=1}^{k}M_tQ_t\left(\sum_{t=1}^{i-1}M_tQ_t+M_{i1}Q_{i1}+M_{i2}Q_{i2}+\sum_{t=i+1}^{k}M_tQ_t\right)} \tag{3}$$

上式恒大于等于零，因此对较大密文数据段进一步划分可以降低假阳性率，优化查询. 下面分析选择哪个密文数据分段进行划分有可能使优化效果最大. 在确定时间点上 $\sum_{t=1}^{k}H_t$ 和 $\sum_{t=1}^{k}M_tQ_t$ 为常量，则

$$\Delta P\sim\frac{M_{i1}Q_{i2}+M_{i2}Q_{i1}}{\sum_{t=1}^{k}M_tQ_t-(M_{i1}Q_{i2}+M_{i2}Q_{i1})},$$

因此选择划分后 $M_{i1}Q_{i2}+M_{i2}Q_{i1}$ 取得最大值的密文分段可以获得最好的假阳性率优化效果. 不同类型用户查询，AEI 索引结构也不相同，针对 *attribute*=*value* 查询其结构如图 2 所示.

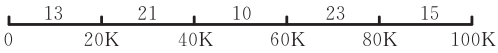
可信数据库中，为了保证查询请求对服务器端保密，服务器只获得针对某个密文数据分段的查询，而并不知道密文数据分段内用户查询值的具体分布，因此服务器端难以确定对哪个密文数据分段进行划分使得 $M_{i1}Q_{i2}+M_{i2}Q_{i1}$ 最大. 如图 2，*Query* 记录每个数据分段承担的查询量，令 $M_{i1}=\alpha M_i$，$Q_{i1}=\beta Q_i$ ($0<\alpha,\beta<1$)，则 $M_{i2}=(1-\alpha)M_i$，$Q_{i2}=(1-\beta)Q_i$，$M_{i1}Q_{i2}+M_{i2}Q_{i1}=(\alpha+\beta-2\alpha\beta)M_iQ_i\sim M_iQ_i$. 因此可信数据库服务器端总是优先选择 M_iQ_i 最大的加密数据分段作为优化目标.

Etuple	*eid*	*eName*	*esalary*	*eAddress*
110101100100…	15	18	23	10
001000110010…	86	11	10	18
100100010110…	47	24	10	14
001011100011…	23	23	10	19
010010011100…	12	17	15	21

(a) 可信数据库加密数据

R_id	*esalary*	*Minimum*	*Maximal*	*Query*	*Joincost*
1	13	0	20000	15	0
2	21	20000	40000	18	228
3	10	40000	60000	58	400
4	23	60000	80000	42	102
5	15	80000	100000	9	null

(b) 可信数据库 *salary* 加密字段 AEI



(c) *salary*加密数据分段

图 2　*attribute op value* 的 AEI 密文分段索引结构

AEI 设置采样时间窗口 *interval* 来统计用户访问量，*Query* 统计 *interval* 内每个密文分段的查询访问量，采样周期结束将访问量清零. 为了使 *interval* 更准确地反映用户访问量的变化，AEI 随着用户访问量的变化调整采样时间间隔(当用户访问量增加时，缩小采样间隔及时地调整可信数据库密文分段，当用户访问量减小时，扩大采样时间间隔，减低服务器端处理量). 设当前采样时间间隔为 $interval_{\text{now}}$，则下一周期的采样间隔 $interval_{\text{next}}$ 如式(4)所示，其中 $Query_{\text{now}}$ 和 $Query_{\text{last}}$ 分别表示当前和上一个时间间隔内的用户访问量.

$$Interval_{\text{next}}=Interval_{\text{now}}\times e^{1-\frac{Query_{\text{now}}}{Query_{\text{last}}}} \tag{4}$$

设系统允许的最大密文分段数为 K，当前密文分段数为 k，每次时间间隔结束，在密文分段总数不超过 K 的前提下，服务器选择至多 $\alpha\times k$ ($0<\alpha<1$) 个 M_iQ_i 最大的密文分段作为待划分的目标. 随着密文分段的划分，直接划分密文分段有可能超出分段总数 K 的限定，为了更好适应用户查询服务变化，有必要对部分数据量和承载服务量较小的密文分段进行合并优化，因此需要衡量划分密文分段和合并之间的代价.

如图 2(b)所示，AEI 以密文分段和右侧相邻密文分段合并后的 $M\times Q$ 作为其合并代价 *Joincost*. 当选择某密文分段 *bucket* 进一步划分超出 K 限制时，将 *bucket* 的优化判定条件 M_iQ_i 与系统中最小的密文分段合并代价 *Joincost* 进行比较，当划分优化效果大于合并代价时，说明划分更有利于优化查询，将 *bucket* 加入待划分列表，合并具有最小合并代价

的两个密文分段并局部更新合并代价，迭代这一过程直至找到 αk 个待划分分段，或划分密文分段的优化判定条件小于最小合并代价为止，选择算法如算法 1 所示. 服务器端最坏情况下需要选择 αk 个待划分的密文数据分段，因此算法最坏情况下的时间复杂度为 $O(\alpha k)$. 如图 2 所示 AEI 索引结构，服务器增加图 2(b)密文索引存储开销与当前密文分段规模 k 成正比为 $O(k)$.

数据库初始状态只有一个数据分段，若单纯利用 AEI 方法调整需要较长时间，不利于服务质量的稳定. 为加快密文分段的优化调整，初始化时可以采用 equi-width、equi-depth 或 QOB 等其它方法划分若干密文分段，再随着数据和服务的加载利用 AEI 方法进行动态调整和优化.

算法 1. *attribute*=*value* 查询，划分数据分段选择算法.

参数：K，最大分段数；α，划分比例；k，当前分段数；
返回：*list_buckets*，待划分密文分段列表；
1. if ($(1+\alpha)k>K$){ //有必要进行合并优化调整
2. for(int $i=1$; $i<=\alpha*k$; i++){
3. *bucket*=剩余密文数据分段中 $M_i\times Q_i$ 最大密文分段；
4. if($i+k<=K$){ //没有超出最大密文分段数
5. *list_buckets.add*(*bucket*)；
6. }else{
7. if (*bucket*'s $M_i\times Q_i<Min(Joincost)$){
8. break; //划分代价更大停止选择
9. }else{
10. *list_buckets.add*(*bucket*)；
11. 合并最小 *Joincost* 的密文数据分段；}}}
12. }else if ($(1+\alpha)k<=K$){ //不超出最大分段数限制
13. 选择 $\alpha\times k$ 个 M_iQ_i 最大的数据分段进入 *list_buckets*；}
14. return *list_buckets*.

确定待划分密文数据分段后，如何进行划分也是必须研究的. 为了保护数据私密性，可信数据库环境下服务器并不能解密存储的数据，无法完成数据分段划分操作. 可信数据库用户在查询过程中会获得加密数据分段上的全部密文元组并对其进行解密过滤，AEI 利用可信数据库的这一服务特点，由可信用户前端完成加密数据分段的划分调整优化.

服务器将密文分段划分请求发送给第一个查询该密文分段的用户，可信用户端在查询解密过滤基础上，基于明文数据元组，选取划分点使得 $M_{i1}Q_{i2}+M_{i2}Q_{i1}$ 取得极大值来划分密文分段. 但可信数据库中服务器端和客户端都不知道用户查询的分布状况，考虑到每个数据分段跨度较小，可以认为用户查询在数据分段内近似服从均匀分布即 $Q\sim Range$，则划分条件可转换为使划分后 $M_{i1}Range_{i2}+M_{i2}Range_{i1}$ 取极大值. 可信前端完成划分后，将数据分段和密文元组分配信息返回服务器，服务器端调整对应密文数据分段，并更新数据分段映射函数，AEI 密文分段优化算法如算法 2 所示.

算法 2. *attribute*=*value* 类型查询，数据分段优化算法.

参数：r，密文分段最小元组数；*list_buckets*，待划分密文分段列表；
服务器端：
1. if (客户端 *client* 查询的密文数据分段 *bucket in list_buckets and bucket* 没有请求其它客户端划分)
2. 服务器端向 *client* 发起调整 *bucket* 请求；
3. while (收到返回的 *bucket* 划分信息){
4. if (*bucket* 可以进行划分)
5. 调整密文数据分段和分段函数；
客户端：
1. if (客户端 *client* 收到划分密文分段 *bucket* 的请求){
2. for (int $i=bucket.Min$, $temp=0$; $i<=bucket.Max$; i++){
3. 以 i 作为分界点试划分；
4. if ($M_1.size()<r$ or $M_2.size()<r$)
5. continue; //不满足最小元组数大于 r 条件
6. elseif ($M_1Range_2+M_2Range_1>temp$)
7. $temp=M_1Range_2+M_2Range_1$；
divide_point=i;}
8. if (*divide_point*==0)
9. return 最小元组 r 限定条件下无法进行划分；
10. else
11. return 新密文数据分段划分信息；}

可信数据库的数据分段与安全性和查询效率都紧密相关，相同情况下数据分段中包含的密文元组数越少，信息泄漏可能性也越大. 为了保证可信数据库安全性，在 AEI 的数据分段过程中保证每个密文数据分段至少包含 r 个元组.

密文数据分段调整算法对于客户端来说，客户端需要比较分段内的所有划分点，其时间开销与密文数据分段跨度 *Range* 成正比，客户端时间复杂度为 $O(Range)$. 对于服务器端，收到每个密文分段的调整信息后，需要调整的密文元组信息与密文分段内元组数 *Size* 成正比，当服务器端确定有 *Count* 个密文数据分段需要调整时，服务器端调整密文数据分段的时间开销为 $O(Size\times Count)$.

AEI 密文分段调整交给对密文分段查询的客户端执行，这样既确保数据对服务器的保密性，而且由于客户端查询过程本身需要解密所有密文元组，

也不会额外增加客户端的解密开销. AEI 加密数据分段函数根据密文数据分段的变化而变化，用户在发起查询请求时需要更新加密数据分段函数.

(2) *op* 为比较操作符的 *attribute op value* 查询操作

op 为比较操作符(“>”,“<”,“≥”或“≤”)，针对图 1 数据查询 $salary>50\text{K}$ 时，服务器端将(40K,60K),(60K,80K),(80K,100K) 3 个数据分段的密文元组都返回给客户端解密过滤获得最终结果. 除了查询值所在密文数据分段存在不匹配数据元组外，其它密文分段返回的都是正确的匹配结果. 和前述分析相同，设属性 A 被划分为 k 个加密数据分段，查询值 *value* 落在每个加密数据分段内时的正确结果总和为 $H_1,\cdots,H_k$. 针对这种查询，需要返回大于或小于方向上的所有加密数据分段的密文元组，首先考虑比较查询符是大于(包括“>”和“≥”)的情况，其总假阳性率如式(5).

$$\sum P=1-\frac{\sum_{t=1}^{k}\left(H_t+Q_t\sum_{j=t+1}^{k}M_j\right)}{\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)}=\frac{\sum_{t=1}^{k}\left(Q_tM_t-H_t\right)}{\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)} \tag{5}$$

设将第 i 个数据分段进行重新划分，划分后密文元组数为 M_{i1} 和 M_{i2}，覆盖查询值次数分别为 Q_{i1} 和 Q_{i2}，对于一个数据分段而言，H_t 和 $Q_{i2}M_{i1}$ 相对于 $Q_t\sum_{j=t+1}^{k}M_j$ 和 $\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)$ 都很小可忽略不计，则假阳性率变化如式(6)所示. 其中 $\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)$ 由当前可信数据库加密数据分段结构和用户查询分布决定并不能改变. 当查询操作符为小于操作时，可以得到类似的假阳性率变化情况.

$$\begin{aligned}\Delta P=&\frac{Q_{i1}M_{i2}\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)+Q_{i2}M_{i1}\sum_{t=1}^{k}\left(Q_t\sum_{j=t+1}^{k}M_j+H_t\right)}{\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)\left(\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)-Q_{i2}M_{i1}\right)}\\ \approx&\frac{Q_{i1}M_{i2}+Q_{i2}M_{i1}}{\sum_{t=1}^{k}\left(Q_t\sum_{j=t}^{k}M_j\right)}\end{aligned} \tag{6}$$

通过上述分析可以发现，与前述 *attribute*=*value* 类型用户查询相同，$Q_{i1}M_{i2}+Q_{i2}M_{i1}$ 也可以作为 *attribute op value* (*op* 为比较操作符)类型用户查询的加密数据分段优化判定条件. 因此 AEI 同样可以基于图 2 的 AEI 数据结构和算法 1、2 实现 *attribute op value* 类型用户查询的优化，本文不进行累述.

3.2.2　*attribute op attribute* 查询

针对加密属性的 *attribute op attribute* 表连接查询，为了获得全部查询结果，需要将所有可能的密文分组连接集合返回给客户端进行解密过滤. 例如：在图 3 所示可信数据库表 A 和表 B 的 *salary* 属性密文分段中，当用户发起 $A.salary=B.salary$ 查询，可信数据库需要把所有可能的加密数据分段组合返回给客户端进行解密过滤，包括((0K,20K),(0K,15K)),((0K,20K),(15K,30K)),((20K,40K),(15K,30K))…，因此针对 *attribute op attribute* 类型查询，可信客户端需要解密过滤的密文数据元组数量是非常大的.

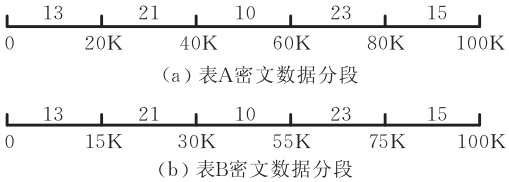


图 3　可信数据库 *attribute op attribute* 密文分段

论文采用一种两级索引方法来优化 *attribute op attribute* 查询. 为降低管理代价，AEI 只在有查询需求的属性对之间构建查询索引. 首先分析两个属性值域之间的关系，图 4 将两个属性的值域分成相离、相交、包含和重叠 4 种相对关系.

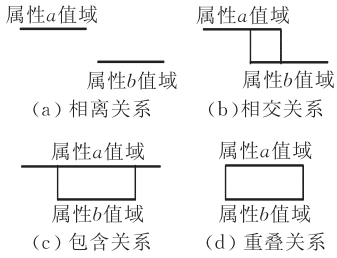


图 4　属性值域间关系

如图 4 可将两个待比较属性的值域 $Range_a(Lower_a,Upper_a)$ 和 $Range_b(Lower_b,Upper_b)$ 分为 5 个子区域：

(1) $Set_{a<b}=(Lower_a,\min(Upper_a,Lower_b))$；

(2) $Set_{a=b}=(\max(Lower_a,Lower_b),\min(Upper_a,Upper_b))$；

(3) $Set_{a>b}=(\max(Lower_a,Upper_b),Upper_a)$；

(4) $Set_{b<a}=(Lower_b,\min(Lower_a,Upper_b))$；

(5) $Set_{b>a}=(\max(Lower_b,Upper_a),Upper_b)$.

两个待比较的密文属性不可能同时具有上述 5 个子集合，如图 4(a) 相离关系就只有 $Set_{a<b}$ 和 $Set_{a>b}$ 两个子集合，其它子集合为空集.

针对一个 *attribute op attribute* 查询，除 $Set_{a=b}$

集合外，其它子集合对于任一查询都是确定的(全部是正确结果，或全部不是)，如针对 $a<b$ 的查询，$Set_{a<b}$ 和 $Set_{b>a}$ 中的密文元组就全是正确匹配的，而 $Set_{b<a}$ 和 $Set_{a>b}$ 则肯定不可能匹配. AEI 设计图 5 所示两级自适应密文分段索引来解决这类查询.

在两级索引中，第 1 级记录了密文属性值所属值域子集，由于属性值域相对固定，第 1 级索引是一个静态索引，针对任何表连接查询，第 1 级索引可以解决除 $Set_{a=b}$ 子集外的所有数据查询，而且不会产生不必要的解密操作，下面研究如何利用第 2 级索引来优化 $Set_{a=b}$ 区域的查询性能.

(1) $attribute=attribute$ 查询

针对 $table1.A=table2.B$ 查询操作，在 $Set_{a=b}$ 区间上，设密文属性 A 划分的密文数据分段为 $A.Range_1,\cdots,A.Range_k$ 共 k 段，每段含有密文元组 $A.M_i$ 条，属性 B 划分为 $B.Range_1,\cdots,B.Range_l$ 共 l 段，每段包含密文元组 $B.M_i$ 条，正确匹配元组数为 H，则 $Set_{a=b}$ 区间产生的假阳性率如式(7)所示.

$$P=1-\frac{H}{\sum_{i=1}^{k}\left(A.M_i\times\sum_{A.M_i\cap B.M_j\neq\varnothing}B.M_j\right)}=1-\frac{H}{\sum_{i=1}^{l}\left(B.M_i\times\sum_{B.M_i\cap A.M_j\neq\varnothing}A.M_j\right)}\tag{7}$$

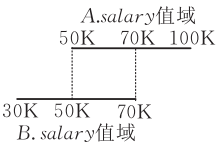
其中 $\sum_{A.M_i\cap B.M_j\neq\varnothing}B.M_j$ 和 $\sum_{B.M_i\cap A.M_j\neq\varnothing}A.M_j$ 的覆盖区域大于等于 $A.M_i$ 和 $B.M_i$ 的覆盖区域，因此在其它条件相同时，在 $Set_{a=b}$ 区间上，保证两属性的二级索引具有相同的密文数据分段划分，如图 5(g)、(h)每个密文分段覆盖区间相同，可以获得更好的查询性能，此时假阳性率为 $P=1-H\Big/\left(\sum_{i=1}^{k}A.M_i\times B.M_i\right)$.

id	Name	salary
012	Tom	75K
013	Mary	60K
014	John	50K
015	Jerry	65K

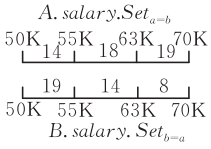
(a) 表 A 原始数据

id	Name	salary
001	Apple	30K
002	Mary	40K
004	Jimmy	60K
005	Michael	50K

(b) 表 B 原始数据



(c) 值域关系



(d) $Set_{a=b}$ 密文数据分段

etuple	Level_1	Level_2
1110…	$Set_{a>b}$	
0010…	$Set_{a=b}$	18
1011…	$Set_{a=b}$	14
0111…	$Set_{a=b}$	19

(e) A.salary 两级加密数据分段

etuple	Level_1	Level_2
0100…	$Set_{b<a}$	
1100…	$Set_{b<a}$	
0101…	$Set_{a=b}$	14
0110…	$Set_{a=b}$	19

(f) B.salary 两级加密数据分段

R_id	esalary	Min	Max	Joincost
1	14	50000	55000	4
2	18	55000	63000	2
3	19	63000	70000	null

(g) A.salary 二级索引

R_id	esalary	Min	Max	Joincost
1	19	50000	55000	4
2	14	55000	63000	2
3	8	63000	70000	null

(h) B.salary 二级索引

图 5 针对 *attribute op attribute* 的两级自适应密文索引

下面分析选择划分两个属性的哪一对密文数据分段对可以获得最好的优化效果. 假设同时对两个属性的第 j 个密文数据分段进行划分，划分后的两个属性密文数据分段仍然相同，存储密文元组数分别为 $A.M_{j1}$，$A.M_{j2}$，$B.M_{j1}$，$B.M_{j2}$，其它数据分段情况相同，则假阳性率变化情况如下.

$$\Delta P=\frac{(A.M_{j1}\times B.M_{j2}+A.M_{j2}\times B.M_{j1})H}{\left(\sum_{i=1}^{k}A.M_i\times B.M_i\right)\times\left[\sum_{i=1}^{k}A.M_i\times B.M_i-(A.M_{j1}\times B.M_{j2}+A.M_{j2}\times B.M_{j1})\right]}\tag{8}$$

由式(8)可知选择划分后 $A.M_{j1}\times B.M_{j2}+A.M_{j2}\times B.M_{j1}$ 最大的密文分段对可以获得更好的假阳性率优化效果，令 $A.M_{j1}=\alpha A.M_j$，$B.M_{j1}=\beta B.M_j$，则 $A.M_{j2}=(1-\alpha)A.M_j$，$B.M_{j2}=(1-\beta)B.M_j$，所以 $A.M_{j1}\times B.M_{j2}+A.M_{j2}\times B.M_{j1}=(\alpha+\beta-2\alpha\beta)A.M_j\times B.M_j\sim A.M_j\times B.M_j$，因此设二级索引当前密文分段数为 k，在二级索引最大分段数 K 限定条件下，AEI 优先选择 $\alpha k(0<\alpha<1)$ 组 $A.M_j\times B.M_j$ 最大的密文数据分段对作为优化划分目标可以获得更大的优化效果. 最坏情况下，密文分段对选择算法需要选择 $\alpha\times k$ 个密文元组乘积最大的密文分段对，因此最坏情况下的密文分段对选择时间复杂度为 $O(\alpha\times k)$. 服务器端优化分段对选择算法类似算法 1，这里不再描述，这时的合并代价 $Joincost$ 是两密文属性密文分段对与右侧密文分段对合并后的密文元组乘积.

由式(8)可知这类查询的假阳性率与用户访问量无关，因此不需要记录用户访问量. 同时由于用户需要解密 $Set_{a=b}$ 上的所有密文元组，为了不额外增加用户解密操作量，AEI 算法由查询 $attribute=$

*attribute*的第一个可信用户端在解密过滤基础上完成所有密文分段的划分工作，分段优化算法如算法 3.

算法 3.　*attribute* A = *attribute* B 二级索引分段优化算法.

参数：r，最小元组数；*list_buckets*，待划分密文分段对；

服务器端：

1. if(客户端 *client* 第一次查询 *attribute* A = *attribute* B)
2.　服务器端向 *client* 发起划分 *list_buckets* 请求；
3. while(收到返回划分信息 *list_info*){
4.　while(*buckets*=*list_info.next*()!=null){
5.　if (*buckets* 可以进行划分)
6.　修改两个属性的密文数据分段信息；}}

客户端：

1. if(客户端收到划分请求){
2. while(*buckets*(A,B)= *list_buckets.next*()!=null){
3.　for(*int* i=$A.Min$, *temp*=0; i<=$A.Max$; i++)
{
4.　以 i 作为分界点试划分；
5.　if ($A_1.size$<r or $A_2.size$<r or $B_1.size$<r or $B_2.size$<r){
6.　continue;//不满足最小元组数条件
7.　}elseif ($A_1.size \times B_2.size + A_2.size \times B_1.size$ > *temp*){
8.　*temp*=$A_1.size \times B_2.size + A_2.size \times B_1.size$();
9.　*divide_point*=i;}}
10.　if(*divide_point*==0){
11.　*list_info.add*(该密文分段对不能进行划分);}
12.　else{
13.　*list_info.add*(密文分段对划分信息);}
14. }Return *list_info*;}

如算法 3 所示，客户端需要对所有待划分密文分段对进行处理，设 $Set_{a=b}$ 区间跨度之和为 $\sum Range$，则客户端的时间复杂度为 $O(\sum Range)$. 而服务器端需要对返回的调整密文分段对元组进行分段调整，设待划分密文分段对的元组总数为 $\sum Size$，则在调整密文分段对时服务器端的时间复杂度为 $O(\sum Size)$. 设两属性密文元组规模为 $Size_1$ 和 $Size_2$，在 $Set_{a=b}$ 子集上的二级密文分段都为 k，为了实现二级 AEI 索引结构，服务器端首先需要存储 $O(Size_1+Size_2)$ 的两级加密数据分段信息，如图 5(e)、(f)，同时需要存储 $2k$ 的二级索引信息，如图 5(g)、(h)，因此整个服务器的存储开销为 $O(Size_1+Size_2+2k)$.

(2) *op* 为比较操作符的 *attribute op attribute* 查询操作

对于 *table*1.A *op* *table*2.B(*op* 为“>”，“<”，“≥”或“≤”)查询，也只会在 $Set_{a=b}$ 区间上产生不匹配返回结果，本节只分析 $Set_{a=b}$ 区间带来的假阳性率，且保持两属性在二级索引上具有一致的密文分段划分，以减少数据分段交叉覆盖引起的大量解密操作. 设属性 A，B 在 $Set_{a=b}$ 区间上都被分成相同的 k 段，包含密文元组数分别为 $A.M_i$ 和 $B.M_i$. 大于和小于比较操作的假阳性率情况并不相同，下面首先研究 *table*1.A<*table*2.B 的查询，设 $Set_{a=b}$ 区间上满足条件的密文元组共 H 条，则假阳性率如下所示.

$$P=1-\frac{H}{\sum_{i=1}^{k}\left(A.M_i\times\sum_{j=i}^{k}B.M_j\right)} \tag{9}$$

设同时对 A，B 二级索引中的第 i 个密文数据分段进行重新划分，分割后新密文数据分段元组数分别为 $A.M_{i1}$，$A.M_{i2}$ 和 $B.M_{i1}$，$B.M_{i2}$，则假阳性率优化情况如下所示.

$$\Delta P_{<}=\frac{H}{\sum_{t=1}^{k}\left(A.M_t\times\sum_{j=t}^{k}B.M_j\right)-A.M_{i2}\times B.M_{i1}}-\frac{H}{\sum_{t=1}^{k}\left(A.M_t\times\sum_{j=t}^{k}B.M_j\right)} \tag{10}$$

由式(10)可知，选择 $A.M_{i2}\times B.M_{i1}$ 最大的密文数据分段进行划分可以获得更好的优化效果，同理当比较操作符为大于时，$\Delta P_{>}$ 优化选择条件为 $A.M_{i1}\times B.M_{i2}$.

考虑到 $A.M_{i1}$，$A.M_{i2}\sim A.M_i$，$B.M_{i1}$，$B.M_{i2}\sim B.M_i$，则有 $\Delta P_{<}$，$\Delta P_{>}\sim A.M_i\times B.M_i$，因此服务器总是优先选择不超过 $\alpha\times k$ 组 $A.M_i\times B.M_i$ 最大的密文数据分段对作为优化目标，优化目标选择算法同样类似算法 1，服务器端的时间复杂度为 $O(\alpha\times k)$.

由于大于和小于比较操作的优化判定条件并不相同，因此为了更好的反映查询服务特性，服务器端记录采样周期 *interval* 内的大于、小于比较查询比例，设分别为 $Prop_{>}$ 和 $Prop_{<}$($Prop_{>}+Prop_{<}=1$)，客户端在划分密文分段对时选择 $Prop_{>}\times(A.M_{i1}\times B.M_{i2})+Prop_{<}\times(A.M_{i2}\times B.M_{i1})$ 最大点作为新的密文数据分段划分点，二级索引分段优化策略类似算法 3，时间和空间复杂度也相同，本节不再累述.

4　性能测试及结果分析

4.1　实验设置

论文采用 JAVA 基于 Totem 数据库实现了面向

服务的加密数据查询方法 AEI,将 AEI 方法与 equi-depth、equi-width[9]、QOB[11] 和 STBucket[12] 等加密数据密文分段查询算法进行对比,比较相同实验环境下的查询总假阳性率性能.模拟实验在一台双核 2.1GHz,内存 2GB 的 PC 机上进行.实验中 AEI 算法通过承载用户查询达到一定密文数据分段数后进行查询结果统计,统计各种算法承载 10000 条用户查询请求的总假阳性率,其它实验设置见表 1.

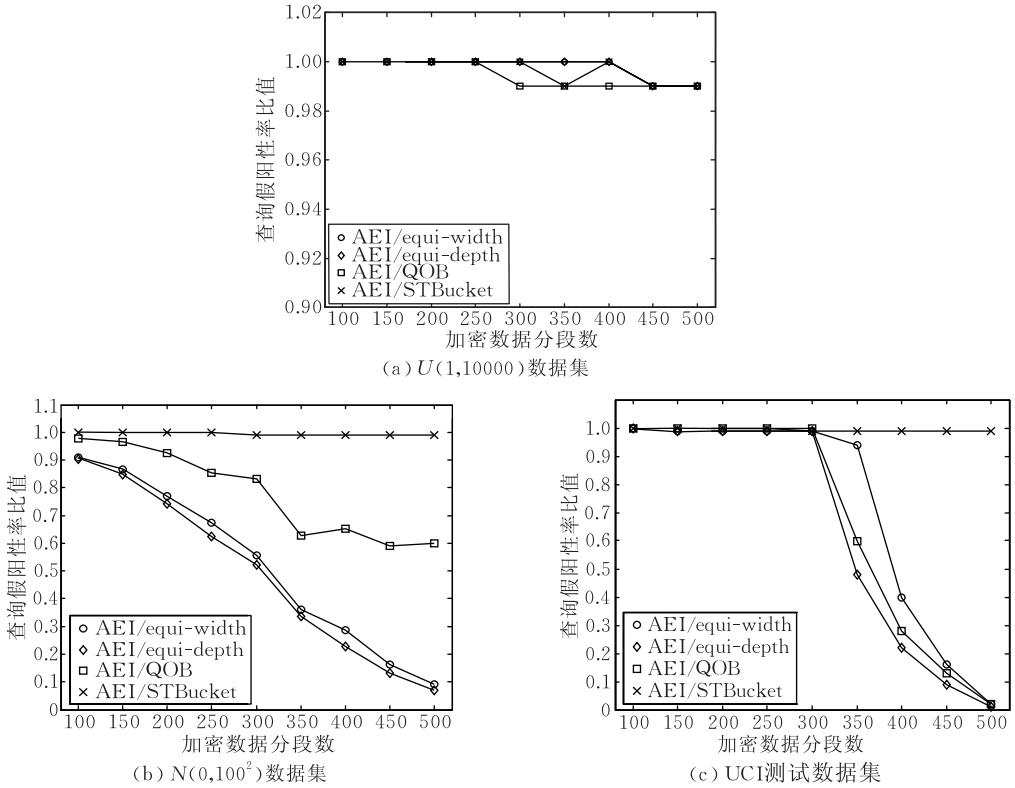
表 1 模拟实验参数设置

	参数意义	取值
r	AEI 密文数据分段的最小元组数	3
$interval_0$	AEI 采用的初始时间间隔	5min
α	划分密文分段比例	0.3

模拟实验采用 3 种不同测试数据集:(1)值域[1,10000]服从 $U(1,10000)$均匀分布规模 10000 的整数数据集;(2)值域$[-\infty,+\infty]$上服从$N(0,100^2)$正态分布取整后规模 10000 的数据集;(3)UCI 的个人信用数据集①中的信用保证金真实数据.

4.2 *attribute*=*value* 类型查询实验

模拟实验设计不同查询测试集对各种密文数据查询算法进行测试,分别是:(1)查询值服从属性值域范围内随机均匀分布的用户查询;(2)考虑到用户查询都应落在属性值域范围内,针对(1)、(2)两种数据源,设计查询值服从 $N(50,100^2)$正态分布查询;针对 UCI 数据源设计查询值服从 $N(5000,100^2)$正态分布的用户查询,实验结果如图 6、图 7 所示.



(a) $U(1,10000)$数据集

(b) $N(0,100^2)$数据集

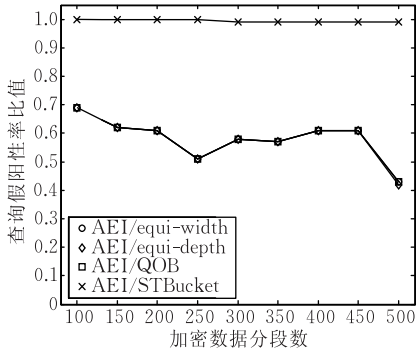
(c) UCI测试数据集

图 6 查询值均匀分布的 *attribute*=*value* 查询假阳性率实验

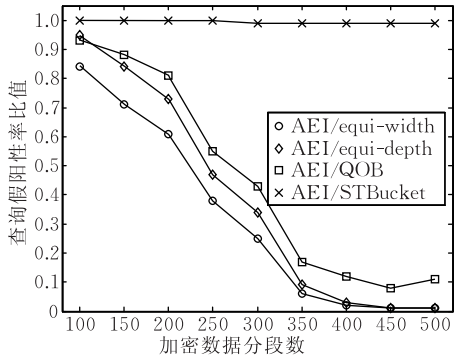
从实验数据可以发现,由于 AEI 和 STBucket 对于 *attribute*=*value* 查询密文数据分段划分策略一致,因此两者的查询假阳性率性能并没有区别.而与其它几种密文查询方法比较,当查询值服从均匀分布时,若不考虑数据分布(都服从均匀分布),如图 6(a),各种密文分段查询方法的查询效率相当.而当数据分布状况发生变化时,如图 6(b)、(c),AEI 密文数据分段方法具有更好的查询性能,且随着数据分段数的增加,AEI 能更好地适应用户查询和数据分布变化,获得更好的查询性能.从实验结果看,实验数据集服从 $N(0,100^2)$分布时,在密文数据分段数 500 情况下,AEI 的查询假阳性率约为 equi-depth 和 equi-width 的 10%,QOB 方法的 60%.对于 UCI 数据集而言,当数据分段较少时,各种方法的查询假阳性率并没有太大区别,随着密文分段数的增加,AEI 方法优化假阳性率效果更为明显,在分段数为 500 时,AEI 的查询假阳性率只有其它方法的约 5%.

在图 7(a)中,即使数据仍然服从均匀分布,当用户查询不均匀时,AEI算法可以更好地适应用户

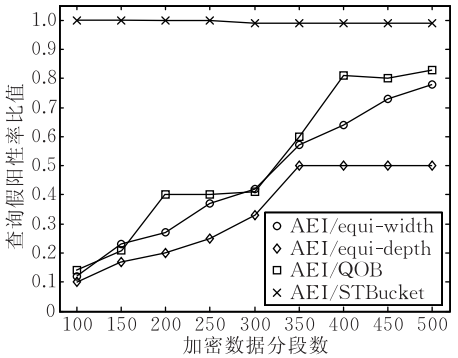
① Statlog (German Credit Data) Data Set. http://archive.ics.uci.edu/ml/datasets.html



(a) $U(1,10000)$数据集



(b) $N(0,100^2)$测试集



(c) UCI测试集

图 7　查询值正态分布的 *attribute*=*value* 查询假阳性率实验

查询的变化，其查询假阳性率约只有其它 3 种方法的 60%. 当用户查询不服从均匀分布时，图 7(b)与图 6(b)相比 AEI 方法的假阳性率性能优势更加明显，更能适应用户查询的变化. 在图 7(c)UCI 数据环境下，AEI 查询方法也具有更好的查询效率，但随着密文数据分段的增加，假阳性率比值会上升，深入分析发现其原因是由于 UCI 测试集数据规模较小，随着密文数据分段的增加，密文分段包含的密文元组数很少，从而使各种查询方法的假阳性率都较低，但 AEI 方法仍然具有最高的查询效率.

4.3　*attribute op value* 类型查询实验

采用 4.2 节中的测试查询值分布集合进行模拟实验，模拟实验过程中保持各种比较操作符(“>”,“<”,“≥”和“≤”)所占比例大致相等，统计各种算法的假阳性率比值，实验结果如图 8、图 9 所示. 与 *attribute*=*value* 查询类似，在 *attribute op value* 类型查询中 AEI 和 STBucket 也具有相同的查询假阳性率性能.

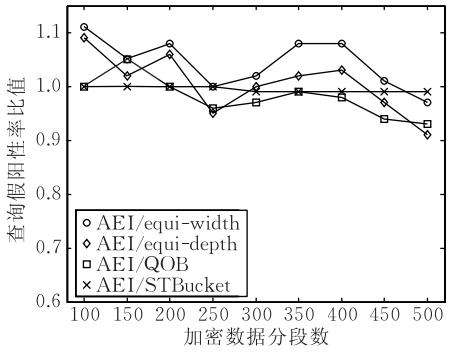
图 8 中的实验结果显示，当数据和用户查询分布都服从均匀分布时，各种查询方法的假阳性率大致相当. 当数据集服从正态分布时，AEI 算法的假阳性率约为 QOB 的 80%，相对于 equi-width 和 equi-depth 方法假阳性率优化效果随密文分段数的增加更加明显，当密文分段足够多时可以节省约 90%以上的无效解密操作. 在 UCI 数据集环境下，AEI 方法的查询假阳性率分别只有 equi-width、equi-depth 和 QOB 方法的约 80%，30%和 45%.

当用户查询值服从正态分布时，与图 8(a)实验结果数据相比，图 9(a)表明 AEI 方法具有更好的用户查询适应性，其查询假阳性率明显小于其它 3 种对比方法，当数据和用户查询值都服从正态分布时，AEI 方法的查询假阳性率只有 equi-width、equi-depth 和 QOB 方法的约 20%、40%和 60%. 在 UCI 数据集环境下 AEI 方法的查询假阳性率分别只有其它 3 种方法的约 35%、30%和 45%.

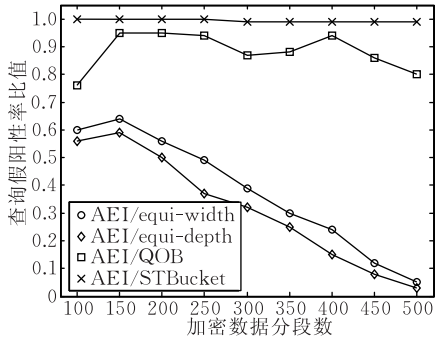
4.4　*attribute*=*attribute* 类型查询实验

AEI 方法中利用两级索引结构优化密文属性之间的查询操作，本节设计模拟实验对密文属性间的 *attribute*=*attribute* 查询假阳性率性能进行评估，实验结果如图 10 所示.

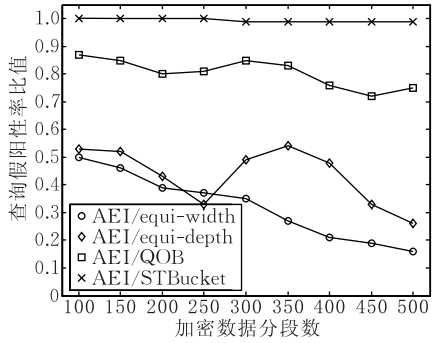
当相同密文数据属性之间进行等值连接查询时，若两个密文属性列数据都服从均匀分布(图 10(a))，AEI 与 equi-width、equi-depth 和 QOB 的查询效率大致相当，而 STBucket 在动态划分密文数据分段时产生了大量的分段值域的重叠，导致针对 *attribute*=*attribute* 的连接查询操作代价大大增加. 而当属性数据不服从均匀分布时(图 10(b)～(c))，AEI 方法显示出更好的适应性，其查询假阳性率也明显优于其它 4 种比较方法.



(a) $U(1,10000)$数据集

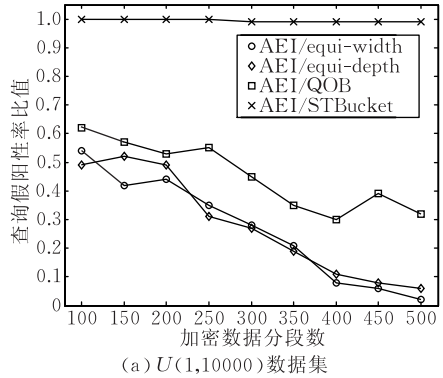


(b) $N(0,100^2)$测试集

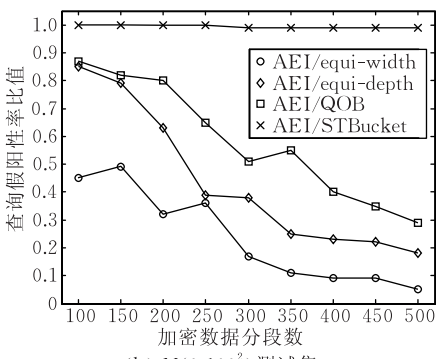


(c) UCI测试数据集

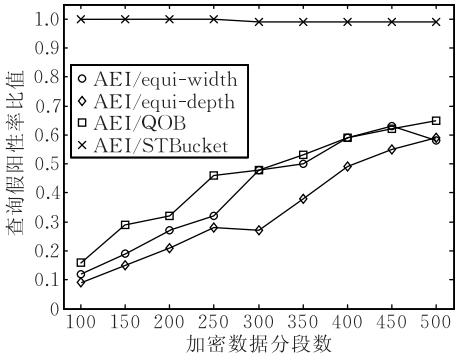
图 8 查询值均匀分布的 *attribute op value* 查询假阳性率实验



(a) $U(1,10000)$数据集



(b) $N(0,100^2)$测试集



(c) UCI测试数据集

图 9 查询值正态分布的 *attribute op value* 查询假阳性率实验

当两个不同密文属性进行等值查询时(图 10(d)～(f)),其它查询方法两个密文属性的数据分段划分都不相同,存在大量的密文数据段交集,使得查询假阳性率大大上升,而 AEI 采用了两级自适应索引方法,很好地解决了密文数据分布和数据段划分带来的影响,其查询假阳性率性能获得了极大的提高,平均假阳性率只有其它方法在相同情况下的不到 50%.

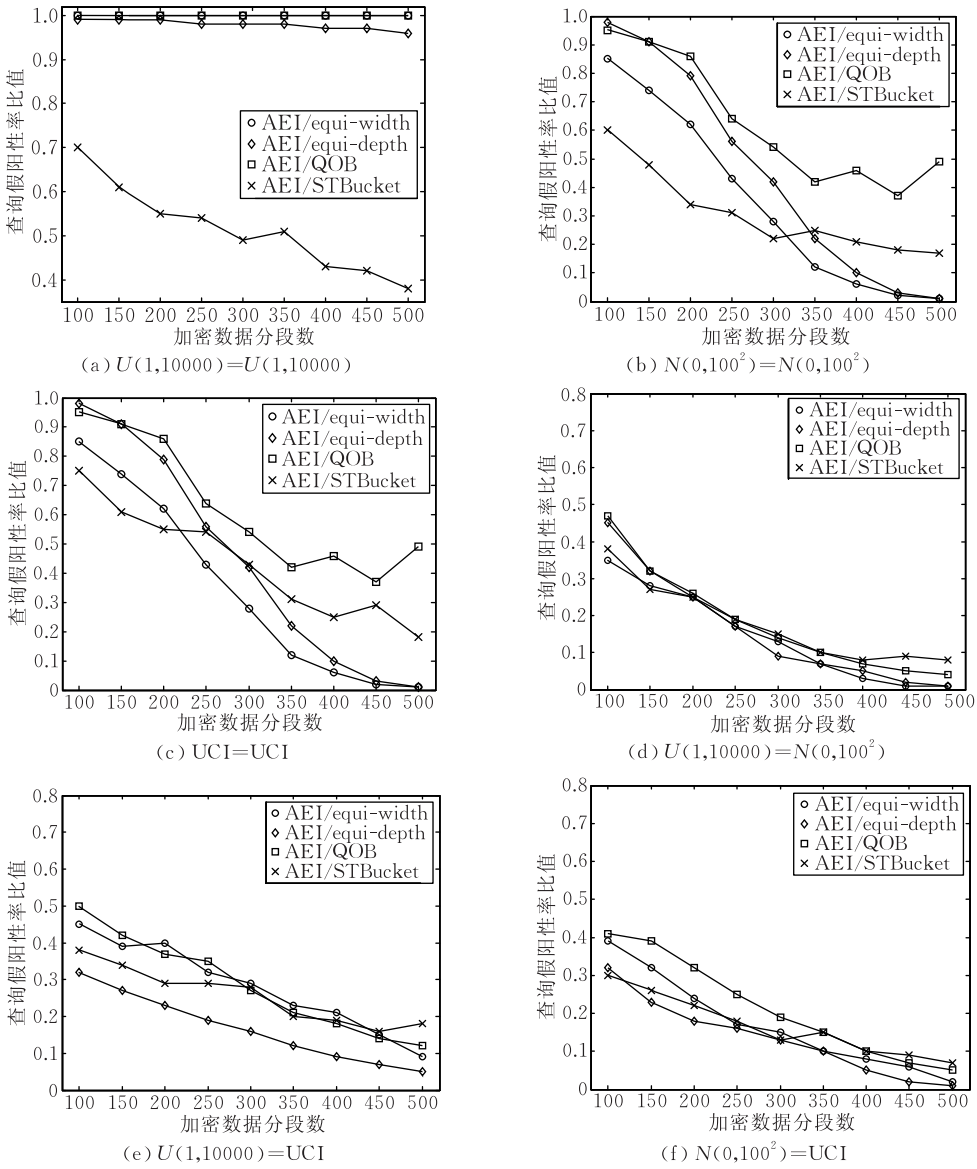


图 10　*attribute* =*attribute* 查询假阳性率实验

4.5　*attribute op attribute* 类型查询实验

本节模拟实验对密文属性之间的 *attribute op attribute*(*op* 为比较操作符)查询假阳性率性能进行评估,模拟实验保持各种比较操作符(“>”,“<”,“≥”和“≤”)比例相当,实验结果如图 11 所示.

从图 11 所示实验结果发现当两属性相同时,当密文属性值服从均匀分布时(图 11(a)),STBucket 方法对密文数据分段进行动态调整,产生部分重叠的密文数据分段对使得查询假阳性率上升,而其它各种密文查询方法的效率相当. 当密文数据分布服从正态分布时(图 11(b)),AEI 方法比其它方法假阳性率性能更高,大约为 QOB 和 STBucket 方法的 40%,equi-width 和 equi-depth 方法的 25%. UCI 数据集相对较小,密文数据分段也较小,AEI 方法相对于 QOB 和 STBucket 方法可以获得约 30%的假阳性率提升,相对 equi-width 和 equi-depth 则分别为 20%和 30%的性能提升.

当两不同密文属性进行比较时,由于两属性的值域、密文数据分段的划分都不相同,AEI 方法显示出更好的适应性和更加明显的优势,从图 11(d)～(f)中可以发现 AEI 密文查询方法的查询假阳性率都要大大优于其它几种方法,且随着密文分段的增加,优势更为明显.

5　结　论

本文分析了查询类型、数据分布和用户查询对可信数据库环境下加密数据查询带来的影响,根据加密数据分布和用户访问情况给出了一种自适应密文索引结构,实现面向服务的加密数据优化查询. 本

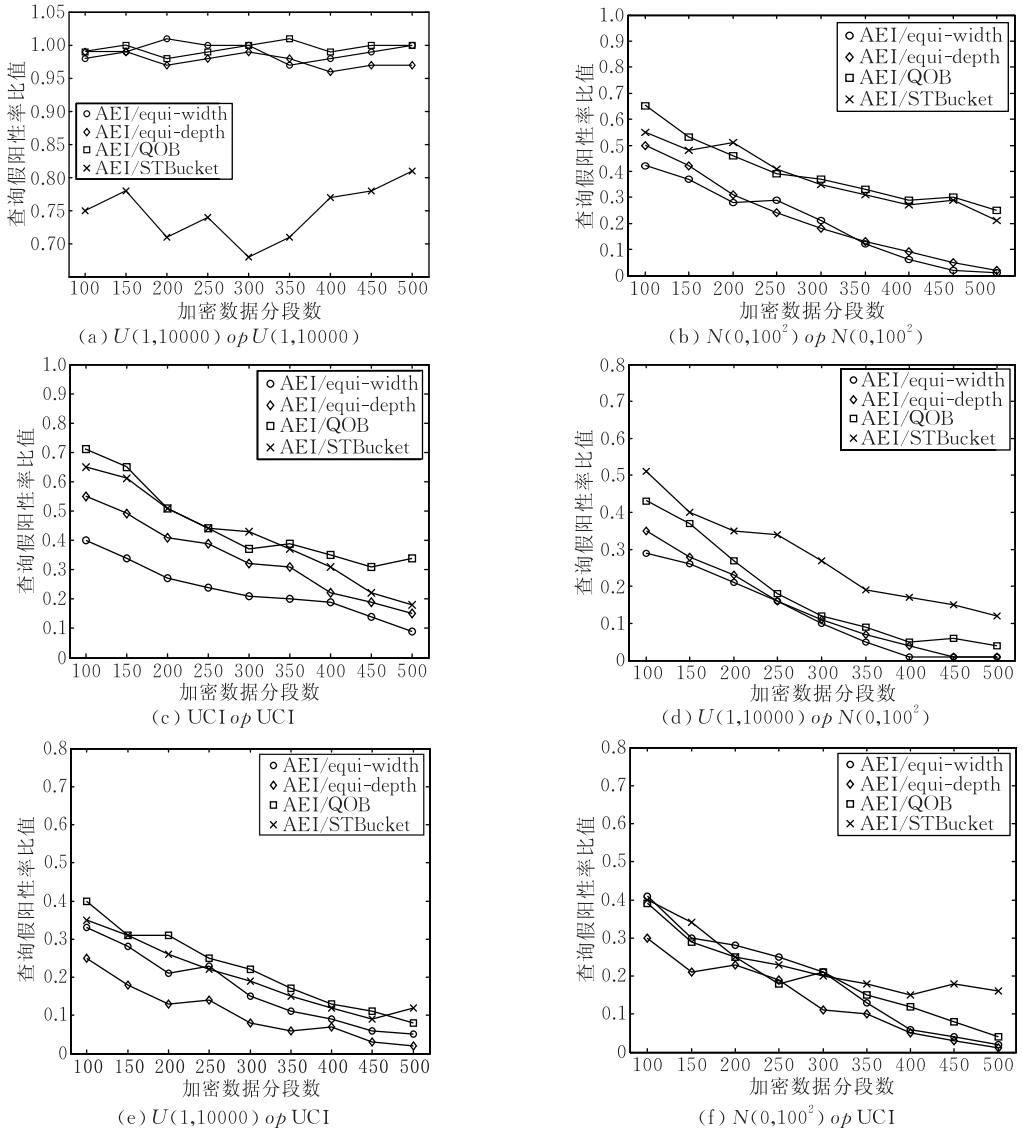


图 11 *attribute op attribute* 查询假阳性率实验

文中的 AEI 算法基于可信数据库客户端需要解密过滤的服务特点将密文分段调整任务交给可信客户端处理既保证了加密数据的安全性,降低了服务器端的处理压力,而且也不会额外地增加客户端解密开销. AEI 采用动态的密文数据分段划分方法可以解决可信数据库数据更新对密文分段带来的影响,同时也可以实现可信数据库的表连接优化查询,这是目前相关研究并没有很好解决的.

基于提出的加密数据查询方法已经实现了一种可信加密 Web 电子邮件系统,本文提出的方法很好地解决了加密电子邮件系统中日期型数据的加密查询操作. 数据库承载的用户查询类型很多,本文研究了两大类主要的可信数据库查询操作,但是数据库需要提供的查询服务种类很多,下一步进一步深入解决可信数据库的其他复杂查询以及应用中面临的各种问题,研究如何实现加密数据的分组、排序以及提供可信数据库基于 AEI 方法的查询接口和标准语言.

参 考 文 献

[1] Peng Zhi-Yong, Yang Ao-Cheng, Ren Yi. Trusted database—Concept, development and challenge. Computer Application, 2008, 28(11): 2741-2744(in Chinese)
(彭智勇,杨鏖丞,任毅. 可信数据库——概念、发展和挑战. 计算机应用, 2008, 28(11): 2741-2744)

[2] Hacı gümüş Hakan, Iye Balar, Mehrotra Shared. Providing database as a service//Proceedings of the 18th International Conference on Data Engineering (ICDE 2002). San Jose, USA, 2002: 29-38

[3] Rivest R L, Adleman L M, Dertouzos M L. On data banks and privacy homomorphisms//DeMillo R A et al eds. Foundations of Secure Computation. New York: Academic Press, 1978: 169-178

[4] Agrawal R, Kirenan J, Srikant R, Xu Yirong. Order-preser-

ving encryption for numeric data//Proceedings of the ACM SIGMOD Conference. Paris, France, 2004: 563-574

[5] Chung S S. Atni-tamper database research: Query encrypted databases [Ph. D. dissertation]. USA: Case Western Reserve University, 2002

[6] David G I, Wells D L, Kam J B. A database encryption system with subkeys. ACM Transactions on Database Systems, 1981, 6(2): 312-328

[7] Ge Tingjian, Zdonik S. Fast, secure encryption for indexing in a column-oriented DBMS//Proceedings of the IEEE 23rd International Conference on Data Engineering (ICDE 2007). Istanbul, Turkey, 2007: 676-685

[8] Bouganim L, Pucheral P. Chip-secured data access: Confidential data on untrusted servers//Proceedings of the 28th International Conference on Very Large Databases (VLDB). Hong Kong, China, 2002: 131-142

[9] Hacigumus H, Lyer B, Li Chen, Mhrotra Sharad. Executing SQL over encrypted data in the database-server-provider model//Proceedings of the ACM SIGMOD. Madison, Wisconsin, USA 2002: 216-227

[10] Hacigumus H, Lyer B, Li C, Mehrotra S. Efficient executing of aggregation queries over encrypted relational database//Proceedings of the Database Systems for Advanced Applications (DASFAA). Jeju Island, Korea, 2004: 125-136

[11] Hore Bijit, Mehrotra Sharad, Tsudik Gene. A privacy-preserving index for range queries//Proceedings of the 30th International Conference on Very Large Databases (VLDB). Toronto, Canada, 2004: 720-731

[12] Wang Haocong, Du Xiaoyong, Wang Jieping, Yang Pingping. STBucket: A self-turning bucket index in DAS paradigm//Proceedings of the 4th ChinaGrid Annual Conference (ChinaGrid). Yantai, China, 2009: 102-109

[13] Dai Yi-Qi, Shang Jie, Su Zhong-Min. Quick index on encrypted database. Journal of Tsinghua University (Science & Technology), 1997, 37(4): 24-27(in Chinese)
(戴一奇，尚杰，苏中民. 密文数据库的快速检索. 清华大学学报(自然科学版)，1997，37(4)：24-27)

[14] Xian He-Qun, Feng Deng-Guo. A query algorithm supporting attribute grain database encryption. Journal of Computer Research and Development, 2008, 45(8): 1307-1314(in Chinese)
(咸鹤群，冯登国. 支持属性粒度数据库加密的查询重写算法. 计算机研究与发展，2008，45(8)：1307-1314)

[15] Yang E Y, Xu J, Bennett K H. Private information retrieval in the presence of malicious failures//Proceedings of the 26th Annual International Computer Software and Applications Conference (COMPSAC 2002). Oxford, England, 2002: 104-121

[16] Kushilevitz E, Ostrovsky R. Replication is not needed: Single database computationally private information retrieval//Proceedings of the 38th Annual IEEE Symposium on the Foundations of Computer Science. Miami Beach, USA, 1997: 364-373

[17] Dinur I, Nissim K. Revealing information while preserving privacy//Proceedings of the Symposium on Principles of Database System (PODS). San Diego, California, USA, 2003: 202-210

SONG Wei, born in 1978, postdoctor, lecturer. His research interests include distributed system, Peer-to-Peer network, and trusted database.

PENG Zhi-Yong, born in 1963, Ph. D., professor, Ph. D. supervisor. His main research interests include Web data management, complex data management, trusted data management.

CHENG Fang-Quan, born in 1983, Ph. D. candidate. His research interests include trusted database and key management in outsourced database.

LI Wen-Hai, born in 1979, Ph. D., associate professor. His main research interests include knowledge discovery, time series database.

HU Wen-Bin, born in 1977, Ph. D., associate professor. His main research interests include workflow management, and software engineering.

REN Yi, born in 1973, Ph. D. candidate. His main research interests include complex data management, privacy protection.

Background

With the development of IT technology, database application is becoming more complex than before. It makes the enterprises have to spend more resources on manage the database in their business information systems. Outsourced database (Database as a Service) architecture enables an enterprise to put their data on a third party database service provider. The database service provider maintains the database and executes the business operations on the enterprise data. The DAS model relieves the enterprise from heavy database management task and makes the enterprise focus on its core business. However, enterprise putting its business data on a third party will take a great security risk. The secure prob-

lems seriously restrict the development of DAS application. So, a trusted database service is significant for the DAS applications.

The trusted database mainly emphasized that even if the database management system is in an untrusted application environment, such as the untrusted DBA, storage media theft, hacker attacks, can still ensure that the stored data is secure. Currently, many researchers have done meaningful work on trusted database. The divided buckets method does not need server to decrypt the data on it. It keeps the data privacy to the third party database server. However, existing methods use static bucket divide method which cannot meet the demands of data update. Moreover, the searching efficiency of existing methods is instability, while the user access is in change.

To address this issue, our paper analyzes how the data distribution and users' access distribution affects the encrypted date searching efficiency. Moreover, we propose an adaptive encrypted index tuning the encrypted data buckets based on the data and access distributions to reduce the query false positive rate and achieve a high searching efficiency.

This work was supported by the National Natural Science Foundation of China (grant Nos. 90718027, 60873225), National High Technology Research and Development Program (863 Program) of China (grant No. 2007AA01Z403), China Postdoctoral Science Foundation under grant No. 20100471145, the Key Program of the Natural Science Foundation of Hubei Province (grand No. 2008CDA007), the Fundamental Research Funds for the Central Universities (grand No. 6082024), self-research program for Doctoral Candidates of Wuhan University (grand No. 20082110101000038).

This project is to make a research on building a trusted database system in an untrusted application environment. Several papers have already been published under this project, mainly on the encrypted data searching, trusted key management, privacy protection, and so on.

With the former work which is mainly on the trusted database architecture, the work done in this paper further give an efficient and secure encrypted data retrieval methods which is useful and significant for trusted database application.