一种面向密文基因数据的子序列外包查询方法

王占兵 宋 伟 彭智勇 杨先娣 崔一辉 申 远
(武汉大学计算机学院 武汉 430070)

摘 要 精准医疗是一种强烈依赖病人基因组分析结果的医疗模式,而子串检索是执行基因组分析的重要方法。近年来,基因数据的数据量急剧增长,其存储代价和处理复杂度已远超医疗方可承受的范围。于是,利用云服务提供商廉价的存储设备和强大的计算能力,将基因数据托管至云服务提供商成为切实可行的解决方案。考虑到云服务提供商并不完全可信,在数据上传至云端之前执行数据加密是保证数据安全性和隐私性的有效方法。然而,如何基于加密数据执行序列检索成为亟待解决的问题。针对这一问题,对基因数据处理和密文检索领域进行调研,提出采用q-gram技术对序列数据的定长窗口创建前缀签名的方案,并在执行查询时在每个窗口中完成前缀查询的解决方案。在子序列查询过程中,云端并不能获取用户数据明文。最后通过实验验证了所提方案具有较好的性能和存储开销,例如当窗口大小为100且q取6时,对100000长序列串执行构建索引耗时15.06 s。与GPSE相比,所提方法的性能更优。

关键词 精准医疗,子序列检索,密文查询,全文检索
中图法分类号 TP309.2 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.06.009

Subsequence Outsourcing Query Method over Encrypted Genomic Data

WANG Zhan-bing SONG Wei PENG Zhi-yong YANG Xian-di CUI Yi-hui SHEN Yuan
(School of Computer, Wuhan University, Wuhan 430070, China)

Abstract Precision medicine is a medical model that relies heavily on patient genome analysis. The subsequence search plays an important role in performing genome analysis. Recently, the amount of genomic data are increasing dramatically, and the storage cost and processing complexity of them have been far beyond the capacity of hospitals. So, utilizing the powerful cloud computing capability to analyze and process such massive genomic sequence data is becoming popular. Considering that cloud service provider is not completely trusted, encrypting genomic data before uploading is a straightforward and effective solution to guarantee the privacy and security of DNA sequence data. However, how to perform queries over the encrypted genomic sequence data becomes another difficult problem. To address this problem, this paper made a detailed survey on genomic data processing and full-text retrieval fields. It constructed indexes on fix-length windows of the genomic sequence using q-gram mapping, and performed queries in every window. If the query sequence is the prefix of any window in genomic sequence, the query hits. Throughout all the processes, cloud service provider stores indexes and performs subsequence query, without obtaining any privacy details. Moreover, this paper set up the system model and several security assumptions, and proved their security. Experiments were carried out to evaluate the performance of scheme on a public dataset. The results show that the proposed solution achieves better performance in time cost and storage cost, i.e. when w is 100 and q is 6, the building index algorithm costs 15.60 s for sequence of 100000 length. Compared with GPSE, the proposed solution has higher execution efficiency in performing queries.

Keywords Precision medicine, Subsequence query, Ciphertext query, Full-text query

1 引言

精准医疗(Precision Medicine)是一种以个人基因组信息为基础,为病人定制最佳治疗方案,从而达到治疗效果最大化和副作用最小化的先进医疗模式。该医疗模式依赖于对病人基因组的分析结果。传统解决方案中,DNA序列数据由医疗方管理;然而,随着海量DNA序列数据的不断增长,其存储代价和处理复杂度已远超出医疗方可承担的范围。另一方

到稿日期:2017-03-11 返修日期:2017-07-04 本文受国家自然科学基金(61232002,61572378)资助。
王占兵(1992-),男,硕士生,主要研究方向为云安全,E-mail:bingo711x@whu.edu.cn;**宋 伟**(1978-),男,副教授,主要研究方向为云安全、大数据安全、应用密码学等,E-mail:songwei@whu.edu.cn(通信作者);**彭智勇**(1963-),男,教授,主要研究方向为数据库等,E-mail:peng@whu.edu.cn;**杨先娣**(1974-),女,副教授,主要研究方向为可信数据管理等,E-mail:xiandiy@whu.edu.cn;**崔一辉**(1981-),男,博士生,主要研究方向为可信数据管理等,E-mail:cuiyihui@whu.edu.cn;**申 远**(1983-),男,博士生,主要研究方向为隐私保护等,E-mail:shenyuan@whu.edu.cn。

面，云计算技术发展趋于成熟，外包数据至云服务提供商作为一种先进的 DaaS(Database-as-a-Service)解决方案，已成为工业界的大趋势。于是，医疗方托管数据至云服务提供商(Cloud Service Provider，CSP)，并授权 CSP 代替医疗方存储和分析基因序列数据成为切实可行的方案之一。然而，由于 CSP 并不完全可信，因此存在用户的基因数据的安全性和隐私性问题。

基因组信息作为生物体唯一性的标识而包含诸多隐私信息，例如亲子关系、疾病信息和遗传特征等。DNA 序列信息一旦被泄露将面临来自工作、保险、婚姻等各方面的歧视。由于 DNA 信息无法被修改，一旦发生泄露其后代也将永远面临此类问题。例如：为了保护 DNA 信息免于泄露和滥用，美国国会于 2008 年通过反基因信息歧视法案(GINA)，截止 2010 年，侵权案例数目已高达 201 例。总而言之，保护基因数据免于滥用和泄露极其重要。

在个性化医疗场景中，最频繁执行的基因处理操作是致病基因片段检索。例如：病人 a 被查出患有淋巴癌，由于淋巴癌有多种类型(如：CD5＋，TdT＋等几十种)，医生需要根据病人基因组是否包含指定致病基因来完成确诊，从而确定病人 a 的用药方案。医生对病人 DNA 序列进行致病基因片段检索，客观上产生了基因片段查询的需求。

考虑到外包基因数据的安全性和隐私性以及客观致病基因片段检索的必要性，找出一种基于加密 DNA 序列数据执行致病基因片段检索的密文查询方案是解决问题的关键。本文提出了基于 q-gram 哈希映射和签名向量实现子串检索的方案来解决以上问题，并通过实验验证本方案的各项相关指标。

本文第 2 节对当前研究现状进行总结；第 3 节对问题进行定义，并根据问题定义与分析规定系统模型和安全假设；第 4 节对本文提出的基于 DNA 序列数据的子序列密文检索解决方法进行了详细阐述；第 5 节进行实验展示并分析实验结果；最后总结全文。

2 相关工作

针对在加密 DNA 序列数据上执行密文子序列检索的问题，本节对当前在基因序列检索与密文检索领域的相关研究进行了总结。

考虑到基因数据高相似性的特点(人与人之间的基因相似度高达 99.9%)，Wheeleer 等人[1]首次提出了一种序列压缩算法，大大降低了数据存储空间的开销。该方法采用高相似性序列数据中的特定序列作为基准，其他序列存储自身和基准的编辑距离，从而达到数据压缩的目的(压缩比高达 1000∶1)。基于该思想，产生了很多研究成果[2-5]。这些研究成果可以依据索引类型分为两类。第一类成果[2,4-5]基于整体压缩的方法对数据进行索引，从而支持直接在压缩数据上进行查询。例如：Ferrada 等人[4]基于 LZ77 提出一种快速索引方法，Schneeberger 等人[3]通过在 Lempel-Ziv 压缩结构上构建高速查询索引来改善查询效率；Wang[2]总结了之前的研究，并基于 BWT 压缩方法对算法实现了改进，从而使查询效率增加。第二类成果[6-7]通过 q-gram 进行快速查询定位，然后对命中位置进行并发查询。例如：Claude 等人[6]基于序列数据构建 q-gram 签名从而实现快速查询；Yang 等人[7]采用 q-gram 倒排索引全速定位，构建了 BST 实现高效查询。以上方法可以解决数据的压缩存储和高效查询问题，然而，在新的云存储场景下，采用公开基准序列易导致数据的泄露问题。

另一方面，有一些研究成果聚焦于序列数据的安全性和隐私性，这些研究等人[8-13]对上述问题面临的挑战进行了阐述。例如：Ayday 等人[8]首次提出了基因数据外包子序列查询问题并指出其面临的安全与隐私威胁；Kang 等人[14]基于云处理框架设计了一种基因数据，并基于该框架提出了一种 3EGSM 解决方案。以上研究成果主要对该领域处理方案的基本规则与主要安全威胁进行了定义，为后续研究工作提供了指导。Kobori 等人[15]提出一种通过局部位图的频率直方图来评估两个 DNA 序列相似性的算法。

在加密数据的全文检索领域，研究[16-19]具有较好的效果。Wang 等人[19]提出了 GPSE 加密数据广义模式字符串搜索算法，该算法实现了带通配符的密文字符串检索；Song 等人[17]设计了一种基于布隆过滤器的树索引结构，解决了海量加密云数据的全文检索问题；Wang 等人[18]提出了一种基于倒排索引的公钥可搜索加密模式。以上研究方案在云存储领域具有较好的效果，但是对于基因数据的密文检索并不能获得很好的效果。

3 基本理论

3.1 问题定义

令 $\Sigma=\{A,G,C,T\}$ 为碱基对的取值空间。令 $s=s[1]s[2]s[3]\cdots s[|s|](s[i]\in\Sigma)$ 代表待处理的基因序列，其中 $|s|$ 表示基因序列 s 的长度，$s[i]$ 表示序列 s 中第 i 个碱基(从 1 开始计数)。令 s 代表基序列，sp 代表检索序列，c 代表对 s 执行加密操作后的密文，e 代表执行加密的算法。

$$s\rightarrow sc=\{\langle si,i\rangle\mid si=s[i,i+m-1],i=1,2,\cdots,|s|-m+1\} \tag{1}$$

定义 1(碱基序列精确检索) 对于查询 $\langle sp,scope\rangle$，其中，sp 为检索串，$scope$ 为检索区间，s 为被检索对象，若存在 $i<j(i,j\in scope)$ 满足 $s[i,j]=sp$，则返回开始位置 i，否则返回 FALSE。

定义 2(密文序列检索) 给定数据明文 m，若存在一组加密模式 $\langle e,query\rangle$，使得执行加密 $c=e(m,key)$(m 为明文，key 为加密密钥，c 为密文)后，满足 $query(c,sp)$ 与定义 1 返回的结果一致，则称 $\langle e,query\rangle$ 为支持密文序列检索的加密算法模式。

定义 3 令 p,s 是两个 DNA 序列，令 SC 是 s 中长度为 m 的定长窗口 si 的集合，若 p 是 s 的子串$\Rightarrow\exists si\in SC$ 使得 p 是 si 的前缀。

定义 4 令 p,s 是两个基因序列，L_p^q,L_s^q 分别是 p,s 产生的 q-gram 序列，如果 p 是 s 的前缀，可得以下结论：

1) $\forall g\in L_p^q\Rightarrow g\in L_s^q$；

2) $\forall g\in L_p^q\Rightarrow count_p(g)\leqslant count_s(g)$，其中，$count_p(g)$ 和 $count_s(g)$ 分别表示 $gram$ 在 L_p^q 和 L_s^q 中对应出现的次数；

3) $\forall g\in L_p^q\Rightarrow pos_p(g)=pos_s(g)$，$pos_p(g)$，$pos_s(g)$ 分别表示 g 在 L_p^q，L_s^q 中首次出现的位置。

3.2 基本方法

q-gram 分割：q-gram 是一项应用于自然语言处理与相似

性文本检索领域的技术。q-gram 是指序列数据中连续 q 项生成的最小单元，记为 *GramCreate*。该算法有两个输入参数：DNA 序列串（si 或 sp）和 q；其输出为 q-gram 的切割有序集，其中每个元素的长度为 q。

例如：对于序列串 s=AGCAGTTA，其 q=2 的结果集为{AG，GC，CA，AG，GT，TT，TA}，记为 L_s^q。

查询窗口：基因序列是典型的流数据，直接基于加密序列实现子序列检索存在以下问题：1）若序列数据采用块加密，则由于人为的块划分将无法保证序列数据的连续性。例如：检索序列为 ACTC，*block*=4，那么检索 CTC 时采用同样的加密方法得到的密文将无法完成匹配。2）若采用确定性同态算法，由于 98%的人类 DNA 序列都是相同的，那么根据密文容易恢复出明文。3）若采用非确定性同态算法，则检索时无法构建高效索引，导致检索效率低下；另一方面由于相同的明文会被加密成不同的密文，必然导致匹配不准确的问题。鉴于以上问题，本文采用定长窗口产生窗口签名，并基于签名完成索引构建与检索查询，若判断窗口序列满足：sp 是定长 m 窗口的前缀，则返回 TRUE，否则返回 FALSE。

本文将子串查询问题转化为序列前缀匹配问题，即：若 sp 是 s 的任意窗口序列 si 的前缀，则 sp 是 s 的子串。

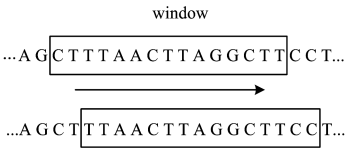

图 1　基因流数据窗口化

Fig. 1　Window of gene flow data

3.3　系统模型和安全假设

本文考虑一个基于云的基因数据分析和查询系统场景，如图 2 所示。系统主要由 3 部分组成：数据拥有者（病人）、云服务提供商（CSP）和数据分析者。考虑到数据安全的威胁来源，如外部有目标的网络黑客攻击和内部 DBA 的操作不当或恶意窃取，系统保证数据一旦流出用户，将始终以密文形式存在，服务器端不能执行解密操作。本系统假设 CSP 为诚实而好奇的，并采用将原始数据加密后上传至 CSP 的基本思想。

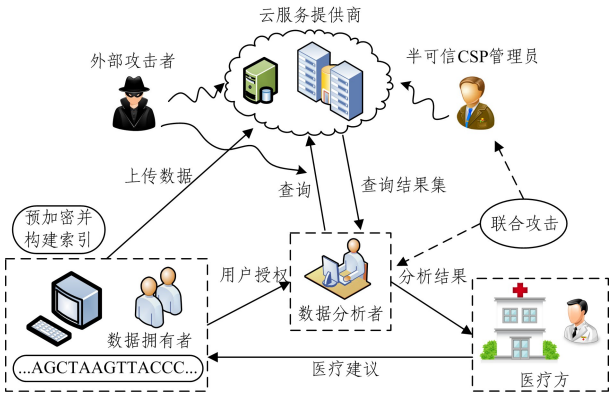

图 2　密文基因序列数据查询处理系统模型图

Fig. 2　Genomic subsequence query processing system model

安全假设 1（诚实而好奇）：本文假设 CSP 为半可信服务方，即 CSP 可以保证外包数据的稳定性和可用性，并忠实地提供用户需要的服务；但对用户数据保持好奇，可能存在 CSP 未经授权就尝试获取用户隐私数据的情况。

数据拥有者（病人）：拥有大量的 DNA 序列数据，并将数据外包至 CSP；考虑到 CSP 并不完全可信，系统先在本地 client完成数据预加密与索引构建，再将其上传至 CSP。

云服务提供商（CSP）：提供稳定的数据存储和计算服务；承诺提供服务的稳定性，但是存在数据泄露的风险。

数据分析者：接收来自数据拥有者的授权，向 CSP 提交查询，并返回结果供医疗方给出准确的医疗判断。

4　窗口签名前缀检索模式

在数据管理中，数据的安全性和处理效率往往不能兼顾，因此通常考虑在可接受的范围内寻找二者的折中。本文的主要思想是将子序列串检索问题转化为前缀匹配问题，也就是说，如果检索串 sp 是 s 的某个窗口序列串 si 的前缀，则 sp 一定是 s 的子串。

4.1　主要思想

给定 DNA 序列（查询序列 sp 或窗口序列 si），采用相同的处理方案产生签名向量。任一序列都可以被其分割产生的 q-gram 和所处序列内的位置来唯一标识。以 si 为例：$si \Leftrightarrow \{\langle g,i\rangle \mid g \in L_{si}^q, L_{si}^q(i)=g\}$，其中 L_{si}^q 是一个由 si 产生的 q-gram 的有序列。基于 L_{si}^q 抽取计数信息和位置信息 $\langle g, count, position\rangle$。本文首先基于窗口序列数据（$sp$ 和 si）计算签名向量 V_{sp} 和 V_{si}，两个向量采用相同维度，记为 d。每个用户持有 hk（哈希参数）对相同的 q-gram 执行映射计算时，相同的 q-gram 会被映射至签名向量的相同位置。如何产生签名向量将在 4.2 节详细介绍。

基于 q-gram 的前缀匹配算法的执行过程基于签名向量 V_{sp} 和 V_{si}。为了判断 sp 是否为 si 的前缀，引入对比向量 V_{sp} 和 V_{si}，$\Omega(sp,si)=dis(V_{sp},V_{si})$，该部分将在 4.3 节详细阐述。

4.2　签名向量抽取生成算法 SVE

签名向量抽取算法（SVE）的输入参数为处理序列 si 和 sp；输出参数为签名向量 V。下面以 si=AGCAGTTA 为例来阐述算法 SVE。

1）设定初始化算法参数 q，m 和 d。

2）以 $step$=1 对输入序列 si 进行切割，产生长度为 m 的窗口，使用算法 GramCreate 对每个窗口序列 si 进行 q-gram 处理，从而产生有序集合。若 q=2，则结果为{AG，GC，CA，AG，GT，TT，TA}。

3）分别计算每个 gram 的 *position* 和 *count* 信息，依据式（2）和式（3）产生的统计结果集合，si 产生的统计结果为{⟨AG，1，2⟩，⟨GC，2，1⟩，⟨CA，3，1⟩，⟨GT，5，1⟩，⟨TT，6，1⟩，⟨TA，7，1⟩}。

4）使用算法 GramMap 映射每个统计元组至签名向量 V 中对应的位置，如式（1）所示，并填充⟨*position*，*count*⟩至对应位置。

5）保存当前签名向量 V，并跳转至步骤 2）继续执行其他窗口的 V 向量生成，直至 s 结尾处结束循环。

$$Position(g)=\min_{g\in L_{si}^q}\{j \mid L_{si}^q(j)=g, j\in N^*\} \tag{2}$$

$$Count(g)=\sum_{i=1}^{len} check(g_i==g), g_i\in L_{si}^q \tag{3}$$

其中，$Position(g)$代表 q-gram 元素 g 于 L_{si}^{q} 首次出现的位置；$Count(g)$代表 g 在 L_{si}^{q} 中出现的总次数；$check(\cdot)$代表真值函数，若输入 TRUE 则返回 1，若输入 FALSE 则返回 0。

$$GramMap(hk,g)=\mathcal{R}_{hk}(g)\bmod d, g\in L_{si}^{g} \tag{4}$$

其中，hk 为映射函数密钥，在系统初始化时为每个用户生成密钥并分发至用户，由数据拥有者保存；$\mathcal{R}$ 为带参哈希函数(参数为 hk)，对数据拥有者的原始数据执行向量映射；d 为签名向量 V 的维度。

SVE 算法如算法 1 所示。

算法 1 Signature Vector Extraction (SVE)

Input：窗口序列或者查询序列 si 或 sp，执行 GramCreate 函数的切割粒度 q，向量 V 的维度 d

Output：使用 SVE 算法产生的签名向量 V

```
1. function SVE(si;q;d)
  # every element of GramInfo 〈gram;pos;cnt〉
2. Set V to 0
3. Set GramInfo to 0
4. Set GramArray to GramCreate(si;q)
5.   for i=1 to len(GramArray) do
6.     element=GramArray[i]
7.       if GramInfo.exist(element) then
8.       GramInfo[element].count++
9.     else
10.  GramInfo.add(〈element;i;1〉)
11. end if
12. end for
13. while GramInfo.isNotEmpty() do
14.  cell=GramInfo.popElement()
15.  index=GramMap(cell;gram)
16.  V[index]=〈cell:pos;cell:count〉
17. end while
18. return V
19. end function
```

4.3 签名向量匹配算法 SVM

签名向量匹配算法 SVM 用于匹配 si 与 sp，SVM 算法的输入参数为两个签名向量，二者是在 SVE 算法中由 sp 和 si 生成。该算法的匹配条件为定义 4 中的条件 2)和条件 3)，从而判断 sp 是否为 si 的前缀。

对于给定的 V_{si} 和 V_{sp}，遍历 V_{si} 中所有的非零位并匹配 V_{sp} 中对应的位置元素，如式(5)所示，标记 V_{sp} 中的非零位置为 $Ne=\{n_1,n_2,\cdots,n_l\}$，其中 $\forall n_i\in Ne(i=1,2,\cdots,l)$，$V_{sp}(n_i)\neq 0$。

$$\begin{aligned}&\Omega(sp,si)=SVM(V_{sp},V_{si})=\prod_{i\in Ne}check(c1\&\&c2)\\&c1:V_{sp}(i)[pos]==V_{si}(i)[pos]\\&c2:V_{sp}(i)[cnt]\leqslant V_{si}(i)[cnt]\end{aligned} \tag{5}$$

SVM 算法如算法 2 所示。

算法 2 Signature Vector Match (SVM)

Input：窗口序列 si 产生的签名向量 V_{si}，查询序列 sp 产生的签名向量 V_{sp}

Output：True 或 False(如果 sp 是窗口序列 si 的前缀则返回 True，即 sp 是 si 所在序列的子串，否则返回 False)

```
1. function SVM(Vsi;Vsp)
2. for i=1 to d do
3.   c1=Vsi[i]:pos!=Vsp[i]:pos
4.   c2=Vsi[i]:cnt<=Vsp[i]:cnt
5.   if c1 &&c2 then
6.     return False
7.   end if
8. end for
9. return True
10. end function
```

4.4 算法分析

空间开销：普通的文本数据将每个词作为天然的窗口进行处理，对每个词产生一个签名向量，无冗余产生。然而对于基因序列数据，为满足子串查询必须在加密的同时保证其连续性，需要人为地将其切割成互相覆盖的子串，该操作带来了较大的数据冗余。本方案中的空间计算式如下：

$$ISC(s)=\frac{1}{n}\sum_{i=1}^{n-m+1}(1+\pi(m,q)(bit_{\text{loc}}+bit_{\text{cnt}}+bit_{\text{pos}})) \tag{6}$$

其中，m 表示窗口长度，q 表示执行 GramCreate 切割的粒度，n 表示整个序列 s 的长度。式(6)中每个长度为 m 的窗口产生 $\pi(m,q)$个互不相同的 q-gram。ISC 作为衡量存储开销的参数，在设计时采用了对 V 进行非零元素抽取的方法，且仅存储非零元素。q 不变时，随着 m 的增大，$\pi(m,q)$递减。

准确度分析：所提方法采用定义 4 中的条件 2)和条件 3)作为判断依据，执行子串匹配时，所提结果存在假阳性。引起假阳性的情况可以分为以下两种：

1)$g\in L_{sp}^{q}$ 且 g 并非首次出现，满足 g 在 sp 中的出现次数不大于 g 在 si 中的出现次数，算法设置可通过适当增加 q 值避免出现过多哈希碰撞。

2)哈希函数映射冲突，即 $\mathcal{R}_{hk}(g)\bmod d$ 将不同的 gram 映射至同一位置。引发假阳性至少需要 q 个 gram 映射错误。

$$P(si)=\sum_{i=0}^{\lfloor |si| \rfloor}\prod_{j=\max\{i-m+1,0\}}^{\min\{i,m-q+1\}}(1-P(\xi=j)) \tag{7}$$

其中，$P(si)$表示发生假阳性的概率，$P(\xi=j)$表示在 L_{si}^{q} 中第 j 个 gram 首次出现的概率。

5 实验结果分析

5.1 实验设计

本文实验使用 C 语言单线程实现了签名向量生成和签名前缀匹配算法。实验的操作系统为 windows 7，主机参数为：Core-2320 3.0GHz CPU，4GB RAM。实验使用 PizzaChili 公开数据集(http://pizzachili.dcc.uchile.cl)，本方案对 DNA 序列数据长度没有限制，为了便于展示，所使用的碱基序列数据长度为 100000。当执行查询时，检索字符串 sp 是随机产生的。最后，采用相同的硬件环境对比所提算法与现存的匹配算法[19]的匹配效率，以验证方案的高效性。

5.2 实验结果分析

本节对所提方案进行实验对比，包括构建索引的开销、构建索引的时间开销与 m 和 q 的关系。

1)时间开销对比(SVE)

如图 3 所示，通过实验评估 SVE 算法的时间开销。实验中分别设置窗口长度为 100 和 500，然后分别使用 SVE 算法测试构建时间，重复实验，取时间开销的平均值。本文方法的主要时间开销是执行带参哈希运算的开销。q 越大，gram 的总数越少，对应的开销越大，故相同的窗口长度下，随着 q 的

增大，时间开销曲线的中间位置应该会有一个回落点（如图 3 中 $w=100, q=5$ 与 $w=500, q=6$ 的点），实验结果也再次验证了这一点。

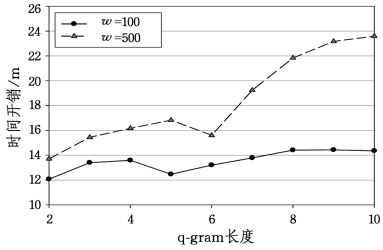

图 3　构建索引的时间开销

Fig. 3　Time overhead of constructing indexing

2）空间开销参数 ISC（SVE）

如图 4 所示，分别以 q 和 $window$ 长度为变量对比 ISC 的变化趋势。ISC 的决定因素为窗口序列 si 所生成 L_{si}^{q} 中的互不相同的 gram 的个数（如式(6)所示）。若窗口长度一定，q 越大，生成 grm 空间越大，但并非是无限制地增多，当所有 gram 均互不相同时即为 ISC 上限；当 q-gram 一定时，显然 si 越长产生互不相同 gram 的概率越高，故 w 为 500 时 ISC 的平均值高于 w 为 100 时 ISC 的平均值。

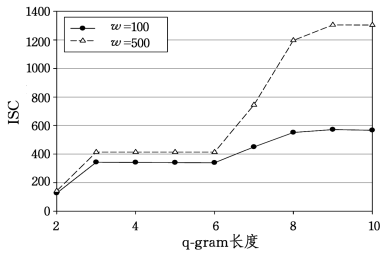

图 4　ISC 开销与 m 和 q 的关系

Fig. 4　Relationship between ISC overhead with different m and q

3）检索效率对比

针对检索效率，将所提算法与现有方案 CPSE 进行对比。相对于长度为 n 的序列串 si，SVM 以算法复杂度 O(N) 扫描整个索引集，时间开销依赖目标序列签名向量的个数。另一方面，对于签名向量比对，仅执行〈$position$, $count$〉（如式(5)所示），复杂度较低。对比实验中设置 $w=500, q=6$。实验结果如图 5 所示，可知本文方案执行子串匹配时具有较好的执行效率。

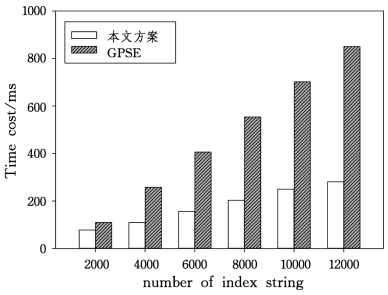

图 5　本文方案与 GPSE 的时间开销对比

Fig. 5　Comparison of time overhead between proposed scheme and GPSE

结束语　在当前海量的基因数据处理场景中，数据一旦被上传至云端将会完全脱离用户的物理控制，也将面临来自外部有目的的黑客攻击和不可信管理员的恶意窥探。针对这一问题，采用数据加密（块加密）可有效保护数据隐私，但同时也会使数据丧失可检索的特性，从而降低数据的可用性。因此，本文首先基于该场景提出处理框架和基本安全假设，基于模型与假设提出采用 q-gram 散列映射技术进行窗口签名向量构建的算法，并通过签名向量进行窗口化查询。本文详细阐述了 SVE 和 SVM 算法的细节，最后基于所提方法进行实验分析，验证了方法的有效性和匹配算法的高效性。

参 考 文 献

[1] WHEELER D A, SRINIVASAN M, EGHOLM M, et al. The complete genome of an individual by massively parallel DNA sequencing[J]. Nature, 2008, 452(7189): 872-876.

[2] WANG J Y, WANG B, YANG X C. Efficient compressed genomic data oriented query approach[J]. Journal of Software, 2016, 27(7): 1715-1728. (in Chinese)
王佳英，王斌，杨晓春. 面向压缩生物基因数据的高效的查询方法[J]. 软件学报，2016，27(7)：1715-1728.

[3] SCHNEEBERGER K, HAGMANN J, OSSOWSKI S, et al. Simultaneous alignment of short reads against multiple genomes [J]. Genome biology, 2009, 10(9): 98.

[4] FERRADA H, GAGIE T, HIRVOLA T, et al. Hybrid indexes for repetitive datasets[J]. Philosophical Transactions of the Royal Society a Mathematical Physical and Engineering Sciences, 2013, 372(372): 20130137.

[5] KOBORI Y, MIZUTA S. Similarity Estimation Between DNA Sequences Based on Local Pattern Histograms of Binary Images [J]. Genomics, Proteomics & Bioinformatics, 2016, 14(2): 103-112.

[6] CLAUDE F, FARINA A, MARTÍNEZ-PRIETO M A, et al. Compressed q-gram indexing for highly repetitive biological sequences[C]// IEEE International Conference on Bioinformatics and Bioengineering. 2010: 86-91.

[7] YANG X, WANG B, LI C, et al. Efficient direct search on compressed genomic data[C]// IEEE International Confernece on Data Engineering. 2013: 961-972.

[8] AYDAY E, HUBAUX J P. Privacy and Security in the Genomic Era[C]// Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016: 1863-1865.

[9] AZIZ A, MOMIN M, HASAN M Z, et al. Secure and efficient multiparty computation on genomic data[C]// Proceedings of the 20th International Database Engineering & Applications Symposium. ACM, 2016: 278-283.

[10] BANERJEE S S, ATHREYA A P, MAINZER L S, et al. Efficient and Scalable Workflows for Genomic Analyses[C]// Proceedings of the ACM International Workshop on Data-Intensive Distributed Computing. ACM, 2016: 27-36.

[11] CERI S, KAITOUA A, MASSEROLI M, et al. Data Management for Next Generation Genomic Computing[C]// EDBT. 2016: 485-490.

[12] FRIZZO-BARKER J, CHOW-WHITE P A, CHARTERS A, et al. Genomic Big Data and Privacy: Challenges and Opportunities for Precision Medicine[J]. Computer Supported Cooperative Work (CSCW), 2016, 25(2-3): 115-136.

[13] QIN Y,YALAMANCHILI H K,QIN J,et al. The current status and challenges in computational analysis of genomic big data [J]. Big Data Research,2015,2(1):12-18.

[14] KANG S,AUNG K M M,VEERAVALLI B. Towards Secure and Fast Mapping of Genomic Sequences on Public Clouds[C]// Proceedings of the 4th ACM International Workshop on Security in Cloud Computing. ACM,2016:59-66.

[15] KOBORI Y,MIZUTA S. Similarity Estimation Between DNA Sequences Based on Local Pattern Histograms of Binary Images [J]. Genomics,Proteomics & Bioinformatics,2016,14(2):103-112.

[16] CLAUDE F,FARINA A,MARTÍNEZ-PRIETO M A,et al. Compressed q-gram indexing for highly repetitive biological sequences[C]// IEEE International Conference on Bioinformatics and Bioengineering. 2010:86-91.

[17] SONG W,WANG B,WANG Q,et al. A privacy-preserved full-text retrieval algorithm over encrypted data for cloud storage applications[J]. Journal of Parallel and Distributed Computing,2017,99:14-27.

[18] WANG B,SONG W,LOU W,et al. Inverted index based multi-keyword public-key searchable encryption with strong privacy guarantee[C]// IEEE Conference on Computer Communiations. 2015:2092-2100.

[19] WANG D,JIA X,WANG C,et al. Generalized pattern matching string search on encrypted data in cloud systems[C]// IEEE Conference on Computer Communiations. 2015:2101-2109.

(上接第 40 页)

结束语 本文提出了一种基于差分隐私的 FP-tree 发布方法,该方法具有很高的安全性,是准确性和效率的均衡。实验结果表明,本文提出的差分隐私 FP-tree 发布方法在效率方面优于许多已有的关联规则挖掘方法,具有较低的假阳性率,是可行且高效的方法。当数据拥有者过多时,噪音会逐渐增大,进而影响假阳性率,该问题需要进一步解决。

参 考 文 献

[1] AGRAWAL R,SRIKANT R. Privacy-preserving data mining [C]// ACM Sigmod International Conference on Management of Data. ACM,2000:439-450.

[2] CHANDRAMOULI B,GOLDSTEIN J,QUAMAR A. Scalable progressive analytics on big data in the cloud[J]. Proceedings of the VLDB Endowment,2013,6(14):1726-1737.

[3] CHANDRAMOULI B,GOLDSTEIN J,DUANS. Temporal analytics on big data for web advertising[C]// International Conference on Data Engineering. IEEE Computer Socieyt,2013:90-101.

[4] LI B,MAZUR E,DIAO Y,et al. A platform for scalable one-pass analytics using mapreduce[C]// ACM SIGMOD International Conference on Management of Data. ACM,2011:985-996.

[5] JOHNSON A,SHMATIKOV V. Privacy-preserving data exploration in genome-wide association studies[C]// Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM,2013:1079-1087.

[6] BONOMI L,XIONG L. Mining frequent patterns with differential privacy[J]. Proceedings of the VLDB Endowment,2013,6(12):1422-1427.

[7] XU S,SU S,CHENG X,et al. Differentially private frequent sequence mining via sampling-based candidate pruning[C]// 2015 IEEE 31st International Conference on Data Engineering (ICDE). IEEE,2015:1035-1046.

[8] LI N,QARDAJI W,SU D,et al. Privbasis:Frequent itemset mining with differential privacy[J]. Proceedings of the VLDB Endowment,2012,5(11):1340-1351.

[9] ZENG C,NAUGHTON J F,CAI J Y. On differentially private frequent itemsetmining[J]. Proceedings of the VLDB Endowment,2012,6(1):25-36.

[10] BHASKAR R,LAXMAN S,SMITH A,et al. Discovering frequent patterns in sensitive data[C]// Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM,2010:503-512.

[11] WONG K S,KIM M H. Privacy-preserving frequent itemsets mining via secure collaborative framework[J]. Security and Communication Networks,2012,5(3):263-272.

[12] NANAVATI N R,JINWALA D C. A novel privacy - preserving scheme for collaborative frequent itemset mining across vertically partitioned data[J]. Security and Communication Networks,2015,8(18):4407-4420.

[13] DWORK C,ROTH A. The algorithmic foundations of differential privacy[J]. Foundations and Trends in Theoretical Computer Science,2014,9(3/4):211-407.

[14] DWORK C,MCSHERRY F,NISSIM K,et al. Calibrating noise to sensitivity in private data analysis[C]// Theory of Cryptography Conference. Springer Berlin Heidelberg,2006:265-284.

[15] GIANNOTTI F,LAKSHMANAN L V S,MONREALE A,et al. Privacy-preserving mining of association rules from outsourced transaction databases[J]. IEEE Systems Journal,2013,7(3):385-395.

[16] MCSHERRY F D. Privacy integrated queries:an extensible platform for privacy-preserving data analysis[C]// Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data. ACM,2009:19-30.

[17] ROY I,SETTY S T V,KILZER A,et al. Airavat:Security and Privacy for MapReduce[C]// Usenix Symposium on Networked Systems Design and Implementation(NSDI 2010). San Jose,CA,USA,2010:297-312.

[18] HAN J,PEI J,YIN Y. Mining frequent patterns without candidate generation[C]// ACM SIGMOD International Conference on Management of data. ACM,2000:1-12.

[19] XIONG P,ZHU T Q,WANG X F. A survey on differential privacy and applications[J]. Chinese Journal of Computers,2014,37(1):101-122.(in Chinese)
熊平,朱天清,王晓峰. 差分隐私保护及其应用[J]. 计算机学报,2014,37(1):101-122.

[20] BLAKE C L,MERZ C J. UCI Repository of machine learning databases [OL]. http://www.ics.uci.edu/~mlearn/MLRepository.html.