基于字典的压缩加密查询方案研究

田萍芳[1,2],郭万涛[1,2]
(1. 武汉科技大学 计算机科学与技术学院,湖北 武汉 430065;
2. 武汉科技大学 智能信息处理与实时工业系统湖北省重点实验室,湖北 武汉 430065)

摘　要:RDF(资源描述框架,Resource Description Framework)压缩方案可以有效压缩庞大的 RDF 数据集,RDF 加密方案可以有效提高数据的安全性。结合这两种方案,该文提出了一种基于字典的压缩加密查询方案。首先,通过原始的 RDF 数据集构建字典集,实现数据的压缩;然后,通过字典集生成密文 ID 三元组,实现数据的加密。将字典集存放在可信区域,密文 ID 三元组存储在不可信区域。查询时,首先通过字典集将查询语句中的关键词转换成对应的密文关键词,然后在密文 ID 三元组上查询与密文关键词相关的数据,将得到的密文数据传送至可信区域,在可信区域进行解密和解压缩操作,得到最终的明文数据。该方案将查询操作与解密解压缩操作分开,查询只在不可信区域操作,解密解压缩则在可信区域操作,从而进一步提高数据的安全性。最后,在 5 个数据集上与其他 RDF 加密方案进行了对比实验,实验结果证明了该压缩加密查询方案在查询性能上有一定的提升,验证了该方案是可行的和有效的。

关键词:RDF 压缩;RDF 加密;字典;加密查询;资源描述框架
中图分类号:TP391　　**文献标识码**:A　　**文章编号**:1673-629X(2024)02-0120-07
doi:10.3969/j.issn.1673-629X.2024.02.018

Research on Dictionary Based Compression Encryption Query Scheme

TIAN Ping-fang[1,2],GUO Wan-tao[1,2]
(1. School of Computer Science and Technology,Wuhan University of Science and Technology,Wuhan 430065,China;
2. Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System,
Wuhan University of Science and Technology,Wuhan 430065,China)

Abstract:The RDF (Resource Description Framework) compression scheme effectively compresses large RDF datasets,and the RDF encryption scheme enhances data security. Combined these two approaches,we propose a dictionary-based compression and encryption query scheme. In this scheme,a dictionary set is constructed by the original RDF dataset for data compression. Then,using the dictionary set,ciphertext ID triplets are generated to achieve data encryption. The dictionary set is stored in a trusted zone,while the ciphertext ID triplets are stored in an untrusted zone. During querying,the keywords in the query statement are first transformed into corresponding ciphertext keywords using the dictionary set. Then,a query is performed on the ciphertext ID triplets to retrieve data related to the ciphertext keywords. The obtained ciphertext data is transferred to the trusted zone,where decryption and decompression operations are performed to obtain the final plaintext data. By separating the query operations from the decryption and decompression operations,the proposed scheme enhances data security. The querying is conducted only in the untrusted zone,while decryption and decompression are performed in the trusted zone. Finally,a comparison experiment with other RDF encryption schemes is carried out on 5 data sets,and it is proved that the proposed compressed encryption query scheme has a certain improvement in query performance,and verifies that it is feasible and effective.

Key words:RDF compression;RDF encryption;dictionary;encrypted query;resource description framework

0　引　言

随着互联网的快速发展,越来越多的组织和个人开始关注 RDF 数据的组织、管理和共享问题[1]。同时,有许多关于金融、医疗、个人或其他敏感的数据[2],这类数据需要采用数据保护机制,如加密和匿名。为了确保数据的机密性[3],不仅需要在传输过程中加密数据,还需要在静止状态下加密数据。在这种情况下,多个用户对数据的不同部分具有不同的访问权限,用

收稿日期:2023-04-21　　**修回日期**:2023-08-23
基金项目:国家社会科学基金重大项目(11&ZD189)
作者简介:田萍芳(1972-),女,教授,硕士,通讯作者,研究方向为计算机网络、语义网;郭万涛(1996 -),男,硕士研究生,研究方向为语义网、RDF 压缩。

户应该只能访问他们被允许访问的数据。但 RDF 数据使用方必须面对 RDF 数据的存储压力,如果不能有效地管理 RDF 数据集,可能会导致系统性能下降和查询效率降低。随着结构化 RDF 数据的不断扩展,RDF 压缩已成为一个日益重要的研究领域。同样,对于敏感数据数据的安全,如何有效加密以及在密文数据上执行查询也是一个重要的研究领域。

目前国内外一些学者对 RDF 数据压缩和 RDF 加密进行了研究。国外的 Fernández[4] 在 2010 年提出了 HDT 压缩方案,Fernández 通过将 RDF 三元组转换成字典、ID 三元组,将长的资源描述定位符(Uniform Resource Identifier,URI)转换成短的 ID 值,有效减少了 URI 的结构冗余,从而压缩了 RDF 数据集的大小并且支持对压缩数据的访问。随后基于 HDT 的优化方案陆续被学者提出,进一步压缩了数据集的大小,如 HDT – FoQ[5] (HDT Focused on Querying), Waterfowl[6],HDT++[7],RDF–TR[8](RDF Triples Reorganizer)等。Sandra[9] 提出一种 k^2–triples 压缩方案,为 ID 三元组中的每个谓词分配一个矩阵,矩阵中的每个点代表一个(主题,对象),然后用一个四叉树压缩矩阵,实现了 RDF 数据集的有效压缩。Maneth[10] 提出的方案先通过 gRePair 算法预处理 RDF 数据集,处理后的数据集再进一步压缩,实现了 RDF 数据集的有效压缩。Sultana[11-12] 提出的另外两种压缩方案也是基于 gRePair 算法来压缩图的结构,然后再通过 k^2–tree 进一步压缩 RDF 数据集。国内的彭燊[13] 提出的基于关系矩阵的关联数据压缩查询模型 HDVM(Header Dictionary Vector Matrix),将 ID 三元组进行一系列合并,转成主题向量、谓词向量、对象矩阵形式存储,也实现了 RDF 数据集的有效压缩与查询。国外的 Mark[14] 提出了一种部分 RDF 加密的方案 PRE(Partial RDF Encryption),其中 RDF–graph 中的敏感数据针对一组接收者进行加密,而所有非敏感数据仍然公开可读,实现了 RDF 数据集的加密,保障了 RDF 数据的安全性。Kasten[15] 提出了一种方法,可以在加密数据上执行用户定义的 SPARQL(SPARQL Protocol and RDF Query Language)查询。RDF 图数据仅部分显示给那些授权执行查询的用户,实现了 RDF 数据集的加密,保障了 RDF 数据的安全性。Javier[16] 提出的压缩加密方案,使用 AES[17](Advanced Encryption Standard)对称加密来加密压缩后的数据,将用户的 ID 作为对称密钥的唯一标识符,只有对应的用户才能解密密文信息,实现了敏感数据的有效保护。

上述的 RDF 压缩方案可以有效压缩 RDF 数据集,但无法保证 RDF 数据的安全;RDF 加密方案提高了 RDF 数据的安全性,但是,查询加密数据集需要解密所有的数据,然后再对解密后的数据进行查询操作,解密全部数据,相对比较耗时。

鉴于此,该文提出一种综合 RDF 压缩和 RDF 加密的方案,即基于字典的压缩加密查询方案(Dictionary Encrypted Query,DEQ),DEQ 通过建立字典将原始 RDF 数据集中地转化成短的 ID,将原始三元组转化成 ID 三元组,形成密文,并将数据的核心部分字典存放在可信区域,把密文 ID 三元组存放在不可信区域。DEQ 不仅可以压缩原始 RDF 数据集,降低 RDF 数据集占用的空间,还能防止隐私数据被不可信的第三方破解,而且支持对密文数据的查询。最后通过实验验证了该方案的有效性与可行性。

1　相关理论

1.1　RDF

RDF 图 G 是来自($(I \cup B) \times I \times (I \cup B \cup L)$ 的一组有限三元组(subject,predicate,object)[18],其中 I 表示 URI,B 表示空节点、L 表示 RDF 文字。RDF 图可以组合在一起管理,组成一个 RDF 数据集,即一组 RDF 图[19]。RDF 数据集是一个由默认图 G 和命名图 g_i 组成的集合 DS:DS = $\{G, (g_1, G_1), \cdots, (g_n, G_n)\}$,其中 $g_i \in I$,g_i 是图的名字。RDF 数据集图示例如图 1 所示,RDF 数据集三元组结构示例如图 2 所示。

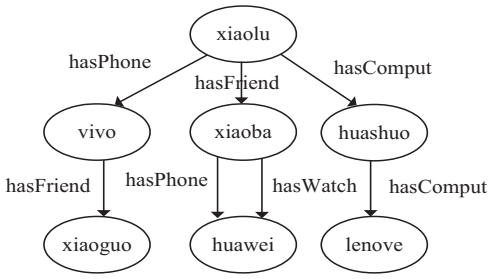

图 1　RDF 数据集图示例

```
<http://example.com/xiaolu> <http://example.com/hasFriend>  <http://example.com/xiaoba>.
<http://example.com/xiaolu> <http://example.com/hasPhone>   <http://example.com/vivo>.
<http://example.com/xiaolu> <http://example.com/hasComput>  <http://example.com/huashuo>.
<http://example.com/xiaoba> <http://example.com/hasFriend>  <http://example.com/xiaoguo>.
<http://example.com/xiaoba> <http://example.com/hasPhone>   <http://example.com/huawei>.
<http://example.com/xiaoba> <http://example.com/hasComput>  <http://example.com/lenove>.
<http://example.com/xiaoba> <http://example.com/hasWatch>   <http://example.com/huawei>.
```

图 2　RDF 数据集三元组示例

1.2　HDT

HDT 是一种二进制序列化格式,其核心思想是通过字典将 RDF 数据集中的所有数据换成短的 ID,将三元组转化成 ID 三元组,从而减少结构冗余,来降低 RDF 数据集占用的空间。HDT 通过三个逻辑组件来管理 RDF 数据集。三个组件分别是:

(1)Header 组件:主要包括描述 RDF 数据集的逻辑和物理元数据,它作为数据集中信息的入口点;

(2)Dictionary:组件提供了数据集中使用的术语

集合,并将它们映射到唯一的整数 ID。它使术语可以被其相应的 ID 替换,实现数据的有效压缩;

(3) Triples 组件:表示 ID 替换后底层图的纯结构。

1.3 邻接表

邻接表是一种用于表示图的数据结构。在邻接表中,图中的每个节点都与一个列表相对应,该列表包含与该节点直接相连的所有其他节点。这个列表可以是一个数组、链表或者其他数据结构,其中存储的元素为该节点所连接的节点的标识符或者对象。邻接表是一种紧凑的数据结构,可以有效地表示具有大量节点和边的图。并且在某些图算法中,邻接表也比其他数据结构更有效率。

图 1 对应的邻接表结构如图 3(a)所示,其中每个节点中取的是 URI 的缩写。在 HDT 方案中,将原始三元组转换为 ID 三元组后,通过邻接表形式存储 ID 值,形式如图 3(b)所示,图中的 ID 是原始 RDF 三元组经过字典转换得到的唯一 ID 值。

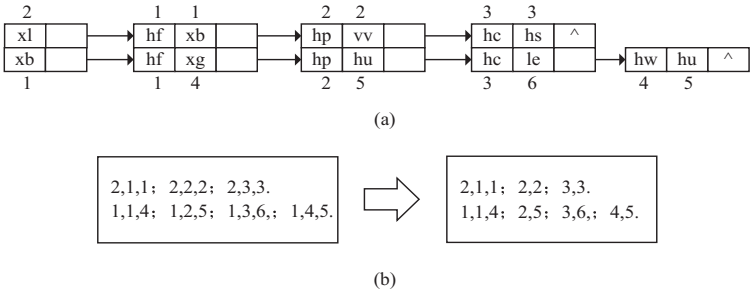

图 3 邻接表与 ID 邻接表结构

2 模型总体架构与技术实现

压缩加密查询模型架构如图 4 所示。架构包括压缩加密模块、可信区域、不可信区域三个部分。压缩加密模块包括数据压缩(构建字典)和数据加密(生成密文三元组)。字典通过原始 RDF 数据集中的三元组中的元素生成,字典主要是降低 RDF 数据集中的 URI 冗余,并且通过字典建立起 URI 与短 ID 的一一映射,降低三元组的大小。密文 ID 三元组是原始 RDF 数据集中的三元组通过构建的字典,将原始三元组中的主题、谓词、对象替换成 ID 值,最终生成 ID 三元组集,即是密文 ID 三元组。可信区域包括查询转换器和结果翻译器。查询转换器主要是重写用户的查询语句。查询翻译器主要是将密文结果转化成原始的三元组。不可信区域主要是缓解可信区域的存储压力,存放密文数据,并对密文进行查询。

2.1 构建字典

构建字典主要是为了减少 RDF 数据集中 URI 的冗余,同时为每个不同的 URI,不同的 RDF 文字,空节点生成一个唯一的 ID,并在生成密文 ID 三元组时用 ID 替换对应的 URI(或空节点或 RDF 文字)。构建字典应该遵循以下规则:

(1)URI 是以“<”开头,并以“>”结尾。若当前元素为 URI,则获取“<”和“>”之间的内容保存在字典中,并生成对应的 ID。例如:“<http://example. com/xiaoming>”是一个统一资源定位符(Uniform Resource Locator,URL),将“ http://example. com/xiaoming”保存在字典中,并生成对应的 ID 值 1;

(2)空节点是以“_:”开头的。如当前元素为空节点,则直接添加到字典中;例如:“_:xiaoyu”是一个空节点,直接将“_:xiaoyu”添加到字典中,并生成对应的 ID 值 2;

(3)RDF 文字是通过双引号包起来的数据。如当前元素是 RDF 文字,则直接将 RDF 文字添加到字典中,并生成对应的 ID。例如:“xiaohua”是一个 RDF 文字,直接将“xiaohua”添加到字典中,并生成对应的 ID 值 3。

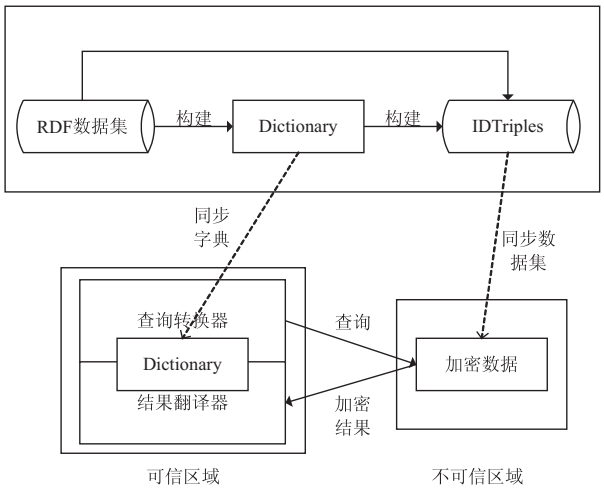

图 4 总体架构

生成字典 ID 也应该遵循以下规则:

(1)SO 字典:对于既出现在主题中,又出现在对象中的 URI 和空节点,将它们添加到 SO 字典,并且生成的 ID 值的范围为[1,|SO|];

(2)S 字典:对于只出现在主题中的 URI 和空节点,将它们添加到 S 字典,并且生成的 ID 值的范围为

[|SO|+1,|S|];

(3)O 字典:对于只出现在对象中的 URI、空节点和 RDF 文字,将它们添加到 O 字典,并且生成的 ID 值的范围为[|SO|+1,|O|];

(4)P 字典:对于出现在谓词中的 URI,将它们添加到 P 字典,并且生成的 ID 值的范围为[1,|P|]。

图 5 所示的是通过 RDF 数据集构建的字典。

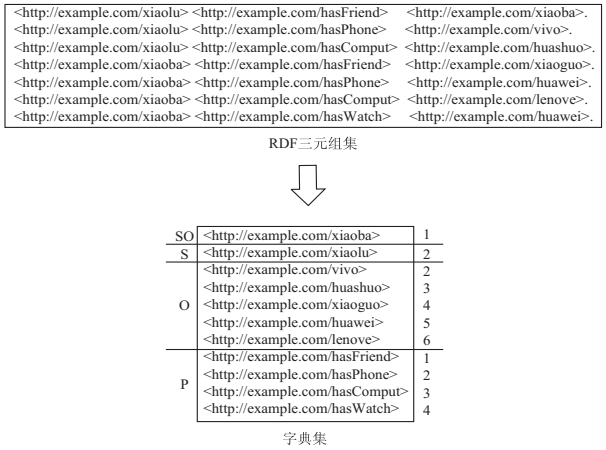

图 5　构建字典过程

2.2　生成密文 ID 三元组

生成密文 ID 三元组主要结合上一步生成的字典,将 RDF 数据集中每个原始的三元组转换成密文 ID 三元组。图 1 中的 RDF 数据集生成密文 ID 三元组的过程如图 6 所示。

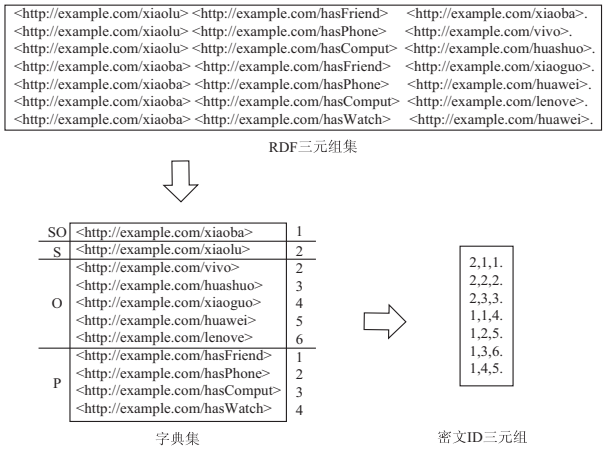

图 6　生成密文 ID 三元组过程

2.3　重写用户查询语句

查询转换器主要负责接收并重写用户的查询语句,并将重写后的查询转发给不可信区域。例如,查询语句“http://example.com/xiaolu http://example.com/hasPhone ? o.”经过查询转换器转换后得到查询语句“2 2 ? o.”。

查询转换算法实现如表 1 所示。

表 1　查询转换算法

输入:查询模式(s,p,o)、SO 字典、S 字典、O 字典、P 字典
输出:查询转换结果(ids,idp,ido)

```
1. begin
2.   ids,idp,ido ← s,p,o
3. //处理 s
4.     if ! s.startWith("?")
5.       ids ← SO.stringToId(s)
```

续表 1

```
6.     if ids = null
7.       ids ← S.stringToId(s)
8. //处理 o
9.     if ! o.startWith("?")
10.      ido ← SO.stringToId(o)
11.    if ido = null
12.      ido ← O.stringToId(o)
13. //处理 p
14.   if ! p.startWith("?")
15.      idp ← P.stringToId(p)
16.  return (ids,idp,ido)
17. end
```

2.4　解密解压数据

结果翻译器主要负责将不可信区域返回的密文数据进行转换,转换成真实 RDF 数据返回给用户。例

如,查询返回的密文 ID 三元组结果为“2 2 2”,经过结果翻译器翻译后得到“http://e-xample. com/xiaolu http://example. com/hasPhone http://exa-mple. com/vivo”。

解密解压算法实现如表 2 所示。

表 2 解密解压算法

输入:密文结果集 ET、SO 字典、S 字典、O 字典、P 字典
输出: 明文结果集 Triples

```
1. begin
2.   Triples ←Ø,ds,dp,do ← null
3.   for t ∈ ET
4. //处理 t. s
5.     if  t. s < SO. len( )
6.       ds ← SO. idToString( t. s)
7.     else
8.       ds ← S. idToString( t. s)
9. //处理 t. o
10.    if  t. o ∈ SO
11.      do ← SO. idToString( t. o)
12.    else
13.      do ← O. idToString( t. o)
14. //处理 t. p
15.    dp ← P. idToString( t. p)
16.    Triples. add( ds,dp,do)
17.  return Triples
18. end
```

2.5 查询加密数据

不可信区域主要存储加密的数据,缓解可信区域的内存空间,并在密文 ID 三元组中执行查询操作,并将查询的密文结果返回给可信区域中的结果翻译器。例如在不可信区域执行的查询为:“2 2 ? o.”,查询得到的密文 ID 三元组为“2 2 2”。

密文查询算法实现如表 3 所示。

表 3 密文查询算法

输入:查询转换结果(ids,idp,ido)、谓词序列 Sp、对象序列 So、位序列 Bp、Bo
输出:密文结果集 ET

```
1. begin
2.   ET ←Ø,ids,idp,ido ← null
3.   //查找谓词集
4.   start_index ← select1( Bp, ids-1) + 1
5.   end_index ← select1( Bp, ids)
6.   P_size ← end_index - start_index + 1
7.   P_list←retrieve( Sp,start_index,P_size)
8.   //查找谓词
9.   p_index ← binary_search( P_list, idp)
10.  p_index ← start_index + p_index - 1
11. //查询对象集
12.  start_index←select1( Bo,p_index-1)+1
13.  end_index← select1( Bo,p_index) + 1
14.  O_size ← end_index - start_index + 1
15.  O_list← retrieve( So,start_index,O_size)
```

续表 3

```
16.  //查找对象
17.  o_index ← binary_search( O_list, ido)
18.  ET. add( s,P_list,O_list)
19.  return ET
20. end
```

3 实验结果

3.1 实验数据集

在真实数据集和生成的数据集上进行对比实验,来验证提出的压缩加密查询方案。并且为了与 Fernández 提出的压缩加密方案进行对比,选择相似的数据集。真实数据集是 DBpedia,并且使用 Lehigh University Benchmark (LUBM)数据生成器从 100 所大学(LUBM-100,包括 0.13 亿三元组)到 400 所大学(LUBM-400,包括 0.53 亿三元组)获得增量大小的合成数据集。LUBM 以大学领域的本体为特征,其中大学的所有实体,如学生、教授和课程,都以三元组格式描述。5 个数据集按照数据集的大小从低到高排列如表 4 所示,表中还包括每个数据集的一些统计信息,包括三元组数量、公共主题对象数量、主题数量、对象数量、谓词数量(分别用|Triples|,|SO|,|S|,|O|,|P|表示)。在 5 个不同数据集上进行了对比实验,主要从压缩加密时间、压缩加密结果、解密解压时间、查询时间四个方面将 DEQ 压缩加密查询方案与 HDTcrypt 压缩加密方案进行对比。并从理论上对每个实验做了分析,理论分析结果与实验结果相吻合。

表 4 数据集相关信息

数据集	|Triples|	|SO|	|S|	|O|	|P|
LUBM-100	13 405 383	501 368	2 179 767	1 623 319	18
DBpedia	11 343 240	494 809	513 698	2 528 345	111
LUBM-200	26 696 580	997 279	4 341 310	3 230 829	18
LUBM-300	39 874 037	1 489 307	6 483 679	4 824 293	18
LUBM-400	53 362 118	1 991 768	8 675 118	6 455 524	18

3.2 压缩加密时间

压缩加密时间通过执行 10 次压缩加密操作,10 次加密操作,得到 10 次压缩加密时间,最后取 10 次压缩加密时间总和的平均值作为最终的压缩加密时间。平均压缩加密时间的计算结果如表 5 所示。

表 5 平均压缩加密时间 min

数据集	HDTcrypt	DEQ
LUBM-100	2.6	1.3
DBpedia	3.4	1.6
LUBM-200	5.8	2.7
LUBM-300	8.8	3.9
LUBM-400	11.8	5.1

从理论上分析, DEQ 的思想与 HDTcrypt 的思想一样,都是将原始 RDF 数据集通过字典转换成字典和 ID 三元组,所以在压缩数据集方面,两种方案的压缩时间没有很大差别。但是,HDTcrypt 需要进一步对字典与 ID 三元组加密来保证数据的安全性,所以 HDTcrypt 的压缩加密时间要比 DEQ 的多,因为对于 HDTcrypt 来说,原始 RDF 数据需要经过压缩、加密两个步骤,生成加密数据。DEQ 只需要对 RDF 数据集进行一步处理,就能达到压缩加密的效果。所以在理论上 DEQ 的压缩加密时间要比 HDTcrypt 的压缩加密时间更少。

从实验结果来看,HDTcrypt 压缩加密 RDF 数据集的时间是 DEQ 的 2 倍以上,随着数据集的增大,DEQ 的性能优势越明显。两种方案压缩加密时间都是随着数据集的增大而增大,即压缩加密时间的大小与 RDF 数据集的大小成正相关。

3.3　压缩加密结果

压缩加密结果比较是通过 HDTcrypt 方案压缩、HDTcrpty 方案加密得到的数据大小与 DEQ 方案压缩加密得到的数据大小进行比较。HDTcrypt 方案压缩、HDTcrypt 方案加密与 DEQ 方案压缩加密后的数据大小如表 6 所示。表中包括数据集的原始大小,HDTcrypt 方案压缩后的文件大小,HDTcrypt 方案加密后的文件大小,DEQ 方案压缩加密后的文件大小。

表 6　压缩加密结果　G

数据集	文件大小	HDTcrypt 压缩	HDTcrypt 加密	DEQ 压缩加密
LUBM-100	2.21	0.07	0.13	0.09
DBpedia	3.42	0.46	0.41	0.37
LUBM-200	4.43	0.15	0.26	0.18
LUBM-300	6.63	0.23	0.40	0.27
LUBM-400	8.87	0.31	0.54	0.36

从理论上分析,DEQ 的思想与 HDTcrypt 的思想一样,都是通过构建字典,将 RDF 数据集中的 URI(或空节点或 RDF 文字)转化成短的 ID 值,从而降低 URI 的冗余,进一步达到压缩数据集的目的。所以在压缩方面,两种方案的压缩结果没有很大差别,但是,HDTcrypt 需要进一步对字典与 ID 三元组通过加密算法加密数据来保证数据的安全性,而加密算法加密数据时会破坏压缩后数据的结构,所以会导致数据集变大,所以 HDTcrypt 的压缩加密结果要好于 DEQ,因为 DEQ 只需要经过字典喜欢换后就形成了密文 ID 三元组,即同时实现了压缩与加密。

从实验结果来看,DEQ 与 HDTcrypt 中的压缩数据集的性能相当,DEQ 压缩加密性能要高于 HDTcrypt。DEQ 和 HDTcrypt 的压缩结果表明,当数据集中的 URI 结构化程度越高,压缩效果也越好。总体来说,DEQ 在压缩加密性能上优于 HDTcrypt。

3.4　解密解压时间

解密解压时间通过执行 10 次解密操作,10 次解压操作,得到 10 次解密解压时间,最后取 10 次解压解密时间总和的平均值作为最终的解压解密时间。平均解压解密时间的计算结果如表 7 所示。

表 7　解密解压时间　s

数据集	HDTcrypt	DEQ
LUBM-100	38.0	23.7
DBpedia	50.2	35.3
LUBM-200	82.9	60.9
LUBM-300	120.0	88.4
LUBM-400	157.8	105.3

从理论上分析,DEQ 的密文数据解密解压只需要将密文 ID 三元组通过字典直接转换,即结合字典根据 ID 值获取对应的 URI(或空节点或 RDF 文字),得到原始的 RDF 三元组;而 HDTcrypt 加密得到的密文数据需要经过解密、解压两个步骤。即首先需要将所有的密文数据加载并解密,然后解压所有数据,得到原始的 RDF 数据集。所以理论上 DEQ 在密文数据解密解压时间上要优于 HDTcrypt。

从实验结果来看,DEQ 的解密解压时间更少,所以 DEQ 解密解压性能更好。通过实验结果可以看出,解密解压的时间会随着数据集的增大而逐渐增长,即解压解密时间的大小与 RDF 数据集的大小成正相关,这一点在 HDTcrypt 和 DEQ 都有体现。

3.5　查询时间

对 7 种查询模式(SPO, SP?, S? O, S??,? PO, ?? O,? P?)分别进行查询实验,通过执行 10 次相同的查询语句,得到 10 次查询时间,最后取 10 次查询时间总和的平均值作为最终的查询时间。DEQ 和 HDTcrypt 分别在 LUBM-200 数据集和 DBpedia 数据集上进行 7 种查询模式的对比,平均查询时间如表 8 所示。

表 8　不同数据集上不同查询模式的平均查询时间 s

查询模式	LUBM-200 数据集		DBpedia 数据集	
	HDTcrypt	DEQ	HDTcrypt	DEQ
SPO	4.05	0.15	4.45	0.14
SP?	4.04	0.21	4.44	0.14
S? O	4.05	0.16	4.45	0.15
S??	4.05	0.17	4.45	0.16
? PO	4.21	0.15	4.61	0.15
?? O	5.02	0.47	5.42	0.17
? P?	9.27	1.62	5.40	0.21

从理论上分析,对于查询操作,HDTcrypt 压缩加密后得到的密文数据必须先获取所有的密文数据,然

后解密所有的数据,其次才能执行查询操作,获取满足条件的压缩后的三元组,最后将查询得到的结果中的 ID 转换成对应的 URI(或空节点或 RDF 文字),才能得到原始的三元组。DEQ 首先加载所有的密文 ID 三元组,然后执行查询操作,获取满足添加的密文 ID 三元组,最后再将查询的结果中的 ID 转换成对应的 URI(或空节点或 RDF 文字),得到原始的三元组。所以,理论上 DEQ 在查询效率上要优于 HDTcrypt,主要体现在 DEQ 比 HDTcrypt 少了一步解密操作。

从实验结果来看,DEQ 在查询效率上要高于 HDTcrypt。在 LUBM 数据集上执行?P?查询模式时,查询时间往往是其他模式的 2 倍以上,这是由于 LUBM 数据集的谓词数量只有 18 个,所以每个谓词相关联的数据非常多,所以?P?查询模式结果转化也会花费比其他查询模式更多的时间,从而增加了?P? 查询模式的查询时间。

4 结束语

DEQ 综合了 RDF 压缩方案与 RDF 加密方案的优点,不仅压缩了原始 RDF 数据集的大小,而且支持在密文 ID 三元组上执行查询操作。将密文 ID 三元组存储在不可信区域,在不可信区域执行查询,将字典数据存储在可信区域,在可信区域执行解密操作,保障了数据的安全性。实验结果表明 DEQ 有不错的性能,不仅压缩加密后的数据小于 HDTcrypt 的数据,而且在压缩加密时间、解密解压时间、查询时间上都优于 HDTcrypt,验证了 DEQ 的可行性与有效性。

当然 DEQ 也存在不足,字典占用的空间比较大,并且密文转换需要加载全部的字典。所以未来将研究如何对字典进行分块存储,从而实现在转换密文 ID 三元组时,只加载部分字典数据,而不需要加载所有的字典数据。

参考文献:

[1] 宋 佳,温亮明,李 洋. 科学数据共享 FAIR 原则:背景,内容及实践[J]. 情报资料工作,2021,42(1):57-68.

[2] 赵锦波. 面向区块链数据隐私保护的可搜索加密研究[D]. 西安:西安电子科技大学,2019.

[3] 王 炎. 面向机密性和完整性保护的安全数据融合技术研究[D]. 衡阳:南华大学,2018.

[4] FERNÁNDEZ J D,MARTÍNEZ-PRIETO M A,GUTIERREZ C. Compact representation of large RDF data sets for publishing and exchange[C]//The semantic web - ISWC 2010:9th international semantic web conference. Shanghai:Springer,2010:193-208.

[5] MARTÍNEZ-PRIETO M A,ARIAS G M,FERNÁNDEZ J D. Exchange and consumption of huge RDF data[C]//The semantic web:research and applications:9th extended semantic web conference. Heraklion:Springer,2012:437-452.

[6] CURÉ O,BLIN G,REVUZ D,et al. Waterfowl:a compact,self-indexed and inference-enabled immutable RDF store[C]//The semantic web:trends and challenges:11th international conference. Anissaras:Springer,2014:302-316.

[7] HERNÁNDEZ-ILLERA A,MARTÍNEZ-PRIETO M A,FERNÁNDEZ J D. Serializing RDFin compressed space[C]//2015 data compression conference. Snowbird:IEEE,2015:363-372.

[8] HERNÁNDEZ-ILLERA A,MARTÍNEZ-PRIETO M A,FERNÁNDEZ J D. RDF-TR:exploiting structural redundancies to boost RDF compression[J]. Information Sciences,2020,508:234-259.

[9] ÁLVAREZ-GARCÍA S,BRISABOA N,FERNÁNDEZ J D,et al. Compressed vertical partitioning for efficient RDF management[J]. Knowledge and Information Systems,2015,44(2):439-474.

[10] MANETH S,PETERNEK F. Grammar-based graph compression[J]. Information Systems,2018,76:19-45.

[11] SULTANA T,LEE Y K. Expressive rule pattern based compression with ranking in Horn rules on RDF style kb[C]//2021 IEEE international conference on big data and smart computing (BigComp). Jeju Island:IEEE,2021:13-19.

[12] SULTANA T,LEE Y K. gRDF:an efficient compressor with reduced structural regularities that utilizes gRePair[J]. Sensors,2022,22(7):2545.

[13] 符海东,彭 燊,黄 莉,等. HDVM:基于关系矩阵的关联数据压缩查询模型[J]. 电子学报,2018,46(3):721-729.

[14] GIERETH M. On partial encryption of rdf-graphs[C]//The semantic web - ISWC 2005:4th international semantic web conference. Galway:Springer,2005:308-322.

[15] KASTEN A,SCHERP A,ARMKNECHT F,et al. Towards search on encrypted graph data[C]//CEUR workshop proceedings. Aachen:CEUR Workshop Proceedings,2014.

[16] FERNÁNDEZ J D,KIRRANE S,POLLERES A,et al. HDTcrypt:compression and encryption of RDF datasets[J]. Semantic Web,2020,11(2):337-359.

[17] DAEMEN J,RIJMEN V. The design of Rijndael:the advanced encryption standard (AES)[M]. Berlin:Springer,2020:1-8.

[18] GUTIERREZ C,HURTADO C,MENDELZON A O. Foundations of semantic web databases[C]//Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems. New York:Association for Computing Machinery,2004:95-106.

[19] SHI J,YAO Y,CHEN R,et al. Fast and concurrent RDF queries with RDMA-based distributed graph exploration[C]//Proc of the 12th USENIX symposium on operating systems design and implementation. Savannah:USENIX Association,2016:317-332.