目录

[I 介绍](#_Toc15512_WPSOffice_Level1) [2](#_Toc15512_WPSOffice_Level1)

[II 相关工作](#_Toc29540_WPSOffice_Level1) [4](#_Toc29540_WPSOffice_Level1)

[III. 背景](#_Toc3242_WPSOffice_Level1) [6](#_Toc3242_WPSOffice_Level1)

[IV 问题公式化](#_Toc3179_WPSOffice_Level1) [10](#_Toc3179_WPSOffice_Level1)

[V、Connor 重建](#_Toc29984_WPSOffice_Level1) [13](#_Toc29984_WPSOffice_Level1)

[VI TREE-BASED CIPHERTEXTS COMPARISON APPROACH](#_Toc25355_WPSOffice_Level1) [18](#_Toc25355_WPSOffice_Level1)

[VII、复杂性和安全性分析](#_Toc16730_WPSOffice_Level1) [21](#_Toc16730_WPSOffice_Level1)

[VIII 性能评估](#_Toc3497_WPSOffice_Level1) [24](#_Toc3497_WPSOffice_Level1)

[VIIII 结论](#_Toc16684_WPSOffice_Level1) [29](#_Toc16684_WPSOffice_Level1)

## I 介绍

最近几年见证了基于图形结构数据[1]，[2]的应用程序的繁荣，如在线社交网络，道路网络，网络图[3]，生物网络和通信网络[4]，[ 5]。因此，在阿卡迪亚（例如，GraphLab [6]，Pregel [7]和TurboGraph [8]）和行业（例如，Titan，DEX和GraphBase）中都提出了许多用于管理，查询和分析海量图的系统。随着云计算的普及，图表所有者（例如，基于图形的服务的企业和初创公司）希望将他们的图形数据库外包给云服务器，这引起了对隐私的极大关注。增强数据隐私的直观方法是在将图形外包到云之前加密图形。然而，这通常以低效率为代价，因为在加密图形上执行操作非常困难。

最短距离查询是最基本的图形操作之一，其根据特定标准针对图形中的给定源对和目的地对找到最短距离。然而，在实践中，用户在执行最短距离查询时可以考虑多个标准[2]。以道路网络为例，用户可能想知道在预算内两个城市之间的总收费的最短距离。该问题可以由约束最短距离（CSD）查询来表示，该查询基于具有对其他标准的一个或多个约束的一个标准来找到最短距离。

在本文中，我们关注单约束CSD查询。 这是因为大多数实际问题可以表示为单约束CSD查询。 例如，通信网络上的这种查询可以返回从起始节点到终端节点的最小成本，具有路由延迟的阈值。 此外，多约束CSD查询通常可以分解为一组子查询，每个子查询可以被抽象为单约束CSD查询。 形式上，CSD查询1是这样的：给定原点s，目的地t和成本约束θ，找到总成本c不超过θ的s和t之间的最短距离。

该领域的现有研究大致可分为两类。第一类主要关注未加密图的CSD查询问题[2]，[9] - [12]。然而，这些方法不能容易地应用于加密图形环境中，因为在没有加密图形的特殊设计的情况下，这些方法中所需的普通图形上的许多操作（例如，添加，复制和比较）不能成功执行。第二类旨在通过加密图形[1]，[13]实现最短距离（或最短路径）查询。它们通常采用距离oracles（预言机），使得任何两个顶点之间的近似距离可以有效地计算，例如以次线性方式。这些方法的主要局限在于它们无法对基于云的加密图执行约束过滤。因此，它们不能直接应用于回答CSD查询。

受现有方案局限性的影响，本文的目标是设计一种实用的图加密方案，通过加密图形实现CSD查询。由于普通图上的CSD问题已经被证明是NP困难的[10]，现有的研究（例如，[2]）通常采用近似解，这保证了所得到的距离不超过α倍。最短距离（其中α是图所有者预定义的近似比率），受成本约束θ的影响。图形加密会使CSD问题更加复杂。因此，我们也专注于设计近似解决方案。

具体来说，本文介绍了Connor，一种新颖的图加密方案，针对加密图上的近似CSD查询。 Connor建立在一个安全的2-hop标记索引（2HCLI）上，这是一种距离预言机，可以有效地计算图中任意两个顶点之间的近似距离[1]，[2]。 安全2HCLI中的图的顶点由特定的伪随机函数（PRF）加密。 为了在允许成本过滤的同时保护图属性的实际值，我们通过订单显示加密（ORE）[14]，[15]和稍微同态加密（SWHE）来加密成本和距离（在顶点对之间） [16]，分别。 基于ORE，我们设计了一种简单但有效的基于树的密文比较协议，可以加速云端的约束过滤过程。

本文的主要贡献如下。

1）我们提出了一种新的图加密方案Connor，它可以进行近似的CSD查询。 它拥有可以在几毫秒内回答α-CSD查询的计算效率。（高性能）

2）我们设计了一个基于树的密文比较协议，这有助于我们确定两个整数之和与其密码文本之间的另一个整数与受控披露之间的关系。 该协议还可以作为其他相关应用场景中的构建块。（同态）

3）我们对Connor进行了全面的安全性分析，并证明它实现了最新的安全定义，名为CQA2-security [17]。 我们还实施了一个原型，并对现实世界的数据集进行了广泛的实验。 评估结果表明了该方案的有效性和有效性。（可证明安全）

## II 相关工作

在云计算时代，安全与隐私成为云服务用户[18]-[23]关注的焦点。本文从两个方面对相关工作进行了简要的总结。 CSD查询普通图和图的隐私保护。

A. CSD查询普通图

约束最短距离/路径在平面图上的查询问题引起了人们的广泛关注。Hansen[9]提出了一种增广Dijkstra算法，用于无索引的精确约束最短路径查询。然而，这种方法带来了巨大的计算负担。为了提高查询效率，另一个解决方案[11]侧重于近似约束最短路径查询，这些查询也是无索引的。

Storandt[12]提出了用索引查询精确约束最短路径的最新解决方案，它使用一种称为收缩层次结构的索引技术加速了查询过程。这种方法仍然会导致不切实际的高查询处理成本。Wang等人提出了一种求解大规模路网中近似约束最短路径的方法。该方法充分利用叠加图技术，在原图的基础上构造了一个尺寸比原图小得多的叠加图。因此，他们在覆盖图上建立了约束的标签索引结构，大大降低了查询成本。不幸的是，所有这些解决方案都只适合对未加密的图执行查询。

*B.图形隐私保护*

在过去的十年中，随着云计算范式的广泛采用，人们对图形隐私的关注也越来越多。Chase和Kamara[17]首先引入了图形加密的概念，他们提出了一些用于图形操作的结构，例如邻接查询和相邻查询。Cao等人利用“过滤-验证”原理，定义并解决了云计算中加密图数据的隐私保护查询问题。他们事先建立了基于特征的图索引，然后选择有效的内积进行滤波。一些方法如[13]、[25]、[26]利用差分隐私技术对图形进行私隐查询，可能会遇到这种情况

942 《IEEE信息取证与安全学报》，第13卷，第1期。2018年4月4日

弱的安全。然而，这些研究引入了高得令人望而却步的存储成本，对大型图表不实用。孟等人提出了三种计算效率较高的结构，支持使用距离神谕查询近似最短距离，并且在半诚实的云服务器上证明是安全的。

安全多方计算(SMC)技术已广泛应用于解决保护隐私的最短路径问题[27]-[30]以及其他安全计算问题[31]。Aly等人针对一般多方计算环境下传统组合图的最短路径问题，提出了两种图中最短路径的安全计算协议。Blanton等人设计了数据无关算法来安全解决单源单目标最短路径问题，在稠密图上获得了最优或接近最优的性能。Keller和Scholl[29]为SMC设计了几种无关数据结构(例如优先队列)，并利用它们计算一般图上的最短路径。Gupta等人提出了一种基于smc的方法来寻找符合策略的路径，这些路径的路由成本最低，或者满足不同网络域之间的带宽需求。然而，现有的通用SMC解决最短路径问题的方案可能会导致沉重的通信开销。

虽然对加密图的图查询已有相当多的研究，但是保护隐私的CSD查询仍然没有得到解决。本文提出了一种新颖高效的CSD查询图加密方案。

据我们所知，这是第一个能够对加密图形进行近似CSD查询的工作。 本文的其余部分安排如下。 我们总结了第二节中的相关工作，并描述了第三节中近似CSD查询的背景。 我们在第IV节中正式定义了隐私保护的近似CSD查询问题。 之后，Connor的构建在第V节中介绍，第VI部分详细描述了基于树的密文比较协议。 我们在第VII节展示了复杂性和安全性分析，通过第VIII节中的广泛实验评估了所提出的方案，并在第IX节中总结了本文。

## III. 背景

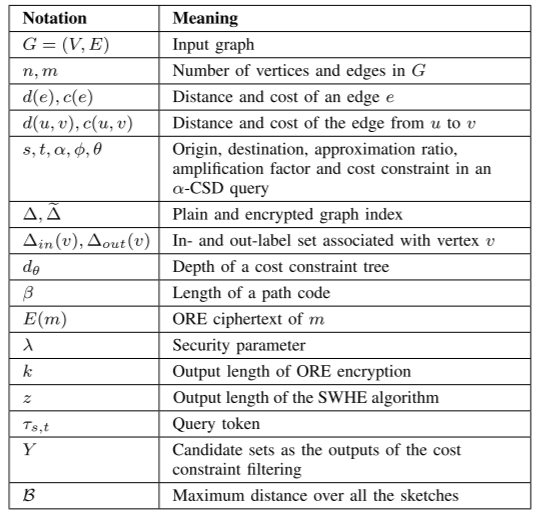
本节给出了CSD查询问题的正式定义，并介绍了用于图形查询的2HCLI结构。

A.近似CSD查询

设G = (V, E)为顶点集V和边集E的有向图（如果没有特别声明，后文提到的图即为有向图）。每条边e ∈ E对应距离d(e) ≥ 0和开销c(e) ≥ 0.

我们把开销c(e)作为约束条件。我们把连接两个顶点的一组边表示为一条路径。对于路径

P= (e1, e2,..., ek )，它的距离d(P) 定义为d(P)= ，即从起点到终点的距离。类似的，我们定义P的开销为c(P) =.本文中的符号在表I中进行了总结。

表Ⅰ 符号列表

给定一个图G，起点s ∈ V，终点t ∈ V ，且开销限制为θ，CSD查询就是找到s和t之间的最短距离d，并且总开销不超过θ. 由于CSD查询问题已被证明是NP-hard，因此我们与已有的解决方案保持一致，本文重点提出一个近似的CSD解决方案。

受未加密图上的近似最短路径查询的通用定义的启发，我们定义近似CSD查询（即α-CSD查询）如下。

定义1（α-CSD查询）：给定起点s和终点t，开销限制θ和近似度α，一个α-CSD查询返回路径P的距离d(P)，使得c(P) ≤ θ 且d(P) ≤ α · d（opt），d（opt）是一个精确CSD查询的最优解，这个精确CSD查询具有同样的起点s，终点t和限制θ.

图1显示了一个简单的图，它有五个顶点，每条边的距离和开销都在它旁边标记。设起点为a，终点为c，开销限制θ=4，精确CSD查询返回的最短距离d（opt） = 6，对应的路径是s (a, b, c). 对于近似度α = 1.5的α-CSD查询（参数不变，即起点为a，终点为c，θ=4），其中一个有效解是8，对应路径为Pα = (a, e, b, c).这是因为d(Pα) = 8 < α · d（opt） = 9 且c(Pα) = 3 < θ.

基于上述定义，对于相同的起点和终点，给定两个路径P1和P2，如果c(P1) ≤ c(P2) 且d(P1) ≤ α · d(P2)，我们称P1 α优于P2.利用这一原理，我们可以显著降低图索引的构造复杂度，因为索引中的大量冗余项可以被过滤掉。我们将在下面的小节中作进一步的说明。

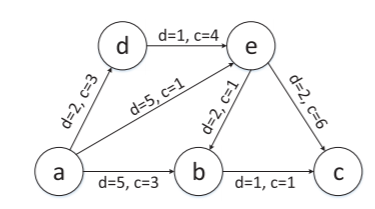


图1. α-CSD查询图示例

B.构建标签索引

本文设计的加密索引主要是基于众所周知的2HCLI构造的，2HCLI是一种特殊的数据结构，能够有效地支持最短距离查询。现在我们简要介绍2HCLI的基本思想，并说明它在构建受限标签索引中的应用。

给定图G = (V, E)，顶点集为V，边集为E，任意顶点v ∈ V都跟一个入标签集△in(v)和出标签集△out(v)相关联。

△in(v)中的每个实体对应顶点u到v的最短路径，u ∈ V.这意味着从u到v有一条或多条通路，但v不一定是u的邻居或两跳邻居。类似的，△out(v)中的每个实体代表从v到V中另一顶点u的最短路径。为了回答从起点s到终点t的最短距离查询，我们首先在标签集△out(s)和△in(t)中找出公共点，然后找出s到t的最短距离。注意△in(v)和△out(v)中的实体必须认真选取，以确保任意两个顶点s和t之间的距离都可以通过△out(s)和△in(t)计算出来。

考虑图1中的图，如果我们忽略边的开销限制，以a为起点，c为终点，基本的无限制最短距离查询就可以通过2HCLI的帮助来解决，详见图2。给定标签集△out(a)和△in(c)，就容易找出公共点的集合，里面包含点b和e.这个基础最短距离查询的最终答案是5，因为d(a, e) + d(e, c) = 5 < d(a, b) + d(b, c) = 6.

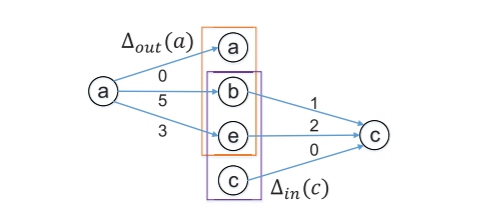


图2. 一个基于2HCLI的基础最短距离查询示例。箭头旁边每个实体d表示从起点到终点的最短距离，如a到e的最短距离为3。

虽然为只有距离条件的图构造2HCLI是简单而直接的，但是为CSD查询构造基于2HCLI的标签索引要复杂得多。这是因为在CSD查询中，边的限制条件有两种，因此在△in(v)和△out(v)标签集中，任意两点之间都可能有多种距离和开销的组合。为了便于说明，我们还以图1中的图和CSD查询为例。对应的2HCLI如图3所示，其中每个箭头旁边的二元组表示从起点到终点的距离和开销。注意，在图2的最短距离查询中，从a到c的最短距离是唯一的，就是经过e的那一条。然而，在图3所示的CSD查询设置中，按照不同的开销，通过e从a到c有四种可能的距离。由于开销条件的存在，在大尺度图中，每对顶点之间距离的数目可能显著增加，这使得构建2HCLI和解决CSD查询问题变得更加复杂。

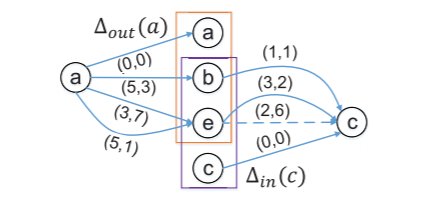


图3. 精确CSD查询的2HCLI示例。箭头旁边的每个实体(距离，开销)分别表示距离和开销。开销限制θ = 4 时，从a到e的最短距离是5.

为了提高查询效率，我们采用了离线过滤和在线过滤相结合的方法。

离线过滤的目的是降低2HCLI的结构复杂度，并尽可能地减少入标签集和出标签集中的实体数量。我们采用了[2]中提出的方法。2HCLI中的实体是精心挑选的，使得对于任何从u到v且满足限制条件θ的CSD查询，都可以只用2HCLI就可以正确解答。由于在特定的CSD查询中，2HCLI的结构应该独立于开销限制，我们可以使用“α优于”的定义去过滤入和出标签集中的冗余实体。

以图3为例，α = 1.5，从e到c的实体有两个，路径P1ec = (e, b, c)，距离-开销二元组为（3，2），路径P2ec = (e, c)，距离-开销二元组为（2，6），P1ec α优于P2ec. 因此，P2ec对应的实体可以被过滤掉（如虚线箭头所示），这有助于减少△in(c)中的实体数量。得到的2HCLI如图4所示。我们请读者参考[2]以获得更多的构造细节。

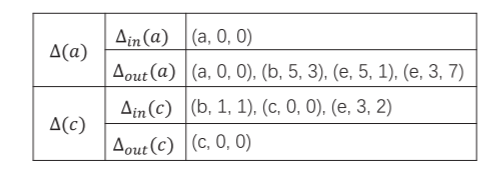


图4. 对图3中原始2HCLI进行离线过滤后得到的2HCLI.实体(u, d, c)中各项分别表示顶点标识，距离和开销。这个近似CSD查询（起点为a，终点为c，α = 1.5, θ = 4）的解是6，恰好是精确CSD查询的解。

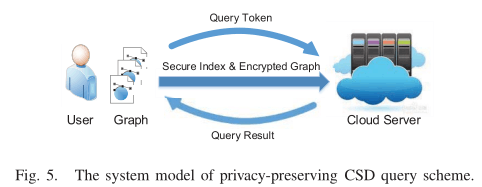
在线过滤的目的是为给定的CSD查询选择可能有效的答案，并且仅基于2HCLI。例如，给定从a到c的α-CSD查询，限制条件θ = 4，我们可以先找到△out(a)和△in(c)的公共点集V’。对于每个v ∈V’，返回c(a,v)+c(v, c) ≤ θ条件下d(a,v)+d(v, c)的最小值。由于上述比较需要与相应的密文一起执行，因此在第六节将设计一种有效的在线过滤方法。

## IV 问题公式化

本节介绍了系统模型和安全性保护隐私的模型α-CSD查询,以及图加密方案的初步研究。

A 系统模型

我们采用通用系统模型在文献[1],[17]的保护隐私α-CSD查询,如图5中所示,主要包括两种类型的实体,即一个用户和云服务器。



用户为图构造安全的可搜索索引，并将加密的索引和加密的图一起发送到云服务器。当用户执行一个α-CSD查询加密图,她首先生成一个查询令牌,然后提交到云服务器。接收到用户的查询令牌后，云服务器执行预先设计的查询算法，将安全索引中的条目与令牌匹配。最后,云服务器将答案回复用户的α-CSD查询。

图形加密方案的正式定义如下。

定义2(图加密):图加密方案Π = (KeyGen、Setup、Query)由三个部分组成多项式时间算法的工作如下:

(K, pk, sk)←KeyGen(λ):是一个概率的密钥生成算法，它将输入一个安全参数λ和输出一个密钥K和公共/秘密密钥对(pk, sk)。

△←设置(αK pk、sk、φ,G):是一种图像加密算法,它将输入一个近似比α,一个密钥K,一个密钥对(pk, sk),一个放大系数φ和一个图G,并输出一个安全索引△。

(dist q，⊥) ← Query((K, pk, sk, Φ, q)，△):是持有密钥K、密钥对(pk, sk)和查询q的用户与持有加密图索引△的云服务器之间的两方协议。执行这个协议后,用户接收远程dist q作为查询结果，云服务器接收一个终止符⊥。

B 安全模型

图加密是对称搜索可加密(SSE)[34] -[38]的一种推广。因此在我们的图形加密方案中，我们采用了安全性定义的SSE设置。这个安全定义与最新的建议是一致的安全定义在[17]，[35]和[39]中也是已知的CQA2-安全(即选择查询攻击安全性)。现在我们给出了CQA2安全的正式定义如下。

定义3 (CQA2-安全模型): Let Π = (KeyGen,Setup, Query)是一种图形加密方案，考虑下面的概率实验，其中A是半诚实的对手，S是模拟器，￡ Setup和￡ Query是(有状态的)泄漏函数。

1. Real (Π, A (λ)):

* 对手A输出一个图G,近似比α和放大系数φ。
* 挑战者通过运行Gen(1λ)来生成一个密钥K和公共/秘密密匙对(pk,sk)。通过设置(α，K ,pk、sk、φ,G),然后计算加密后的索引△，挑战者把加密索引△发送对手对手A。
* 对手A进行多项式次的自适应查询，对于每个查询q, 对手A和挑战者执行查询((K, pk, sk, φ，q), △)。
* 对手A计算一比特b∈{0,1}作为实验的输出。

2）Ideal (Π,A,S(λ)):

* 对手A输出一个图G,近似比例α和一个放大系数φ。
* 给定泄漏函数￡Setup (G)， S模拟一个安全的图索引△\*，然后将它发送到对手A。
* 对手A进行多项式次的自适应查询。对于每个查询q，给出S的泄漏函数￡query(G, Q)， 对手A和S模拟查询，其中对手A扮演云服务器的角色，S扮演用户的角色。
* 对手A计算一比特b∈{0,1}作为实验的输出。

我们说的图加密方案Π = (KeyGen,Setup, Query)是(￡ Setup, ￡Query) -安全的，不受自适应选择查询攻击，如果所有对手A都存在一个PPT模拟器S那么:

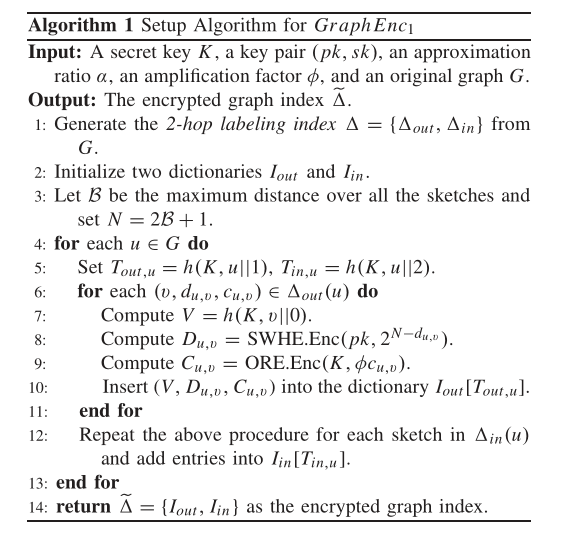
|Pr[Real Π, A (λ) = 1] − Pr[Ideal Π, A , S (λ) = 1]| ≤ negl(λ),其中negl(λ)是可忽略的值。

C 初步探究

现在，我们简要介绍一种在我们的设计中使用的加密技术，即Order-revealing 加密。

Order-revealing 加密(ORE)是order-preserving加密(OPE)方案的推广，但提供了更强的安全保证。正如Naveed等人指出的[40]，OPE-encrypted数据库极易受到推理攻击。为了解决这一局限性，提出了ORE方案 [14]，[15]，该算法是一个三元组Π= (ORE.Setup, ORE.Encrypt, ORE.compare) 描述如下:

* ORE.Setup(1λ) → sk: 输入一个安全参数λ，然后输出密钥sk。
* ORE. Encrypt(sk,m) → ct: 输入密钥sk和消息m，输出密文ct。
* ORE.Compute → z: 输入两个密文ct 1和ct 2，输出r∈{0,1}，表示对应的明文m1和m2之间的大于或小于关系。



该算法：

输入是密钥K，密钥对(pk,sk)，近似值比α,一个放大系数φ,原始图G。

输出是图加密索引。

主要流程是：

1. 从图G中生成2-hop标签索引：△out，△in。
2. 初始化这两个索引字典Iout,Iin。
3. 定义B为图的最大距离，并设置N = 2B+1;
4. 循环图中所有的顶点u，并做如下操作：

设置Tout,u = h(K,u || 1)， Tin,u = h(K,u || 2)

循环集合△out(u)中的二元组(△out(u)表示从u到V中另一顶点v的最短路径)，并做如下操作：

计算V = h(K,v || 0)

计算 距离Du,v = SWHE.Enc(pk,2 N−du,v )

计算 花费Cu,v = ORE.Enc(K,φcu,v )

把计算结果三元组插入到字典Iout[Tout,u]

按照上面步骤计算在△in（u）每个缩略图,并插入到字典Iout[Tin,u]

5.返回最终的图加密索引。

## V、Connor 重建

在本节中，我们将介绍用于隐私保护α-CSD查询的图加密方案Connor。

1. **重建简述**

重建过程基于两个特定的伪随机函数h和g，以及SWHE方案。在本文中，我们采用文献[16]中提到的SWHE方法。 关于h和g的参数信息，参考公式（1），

h：{0,1}λ×{0,1} \*→{0,1}λ（1a）

g：{0,1}λ×{0,1} \*→{0,1}λ+ z + k（1b）

其中λ是加密参数，k和z分别表示通过ORE和SWHE加密后的输出长度。

我们从简单的图 GraphEnc1 =（KeyGen，Setup，Query）开始，如下所示，包括：

•KeyGen：给定安全参数λ，用户随机生成用于SWHE的密钥K和一对公钥和私钥（pk，sk）。

•Setup：给定原始图G，近似比α和放大系数φ，user通过算法1可以计算得到加密的图索引. 其中，图G的2HCLI 可以通过第III-B节中描述的方法生成。

假设B是所有边的最大距离，并且N = 2B + 1。受到文献[1]的启发，每个距离  可以通过SWHE算法加密为，以保护其真实值（第8行）。考虑到2x + 2y受2max(x,y)-1的限制，由SWHE加密的距离可以在一定数量的距离对上获得最小值。

对每个路径开销，乘以放大系数φ，由ORE算法加密（第9行）。φ一个很大的整数，应仔细选择，用以扩大的空间。在实际应用中，φ和所有连接边上的最大成本值的乘积应该足够大（例如，至少280），其用于为输入提供足够的随机性选择。由于φ对用户来说是私有的，因此云服务器无法学习的实际值。

•Query: 要执行具有原点s，目的地t和花销约束θ的α-CSD查询，用户生成查询标记和，并将它们发送到云服务器。云服务器从索引获得和。对于出现在和中的每个加密顶点标识符v，云服务器会采用一个花销约束策略（将在第VI节中详细描述），并将每对满足成本约束φθ的节点对，添加到候选集Y中。需要注意的是，成本约束已经放大了φ倍，因为我们加密了，而不是。

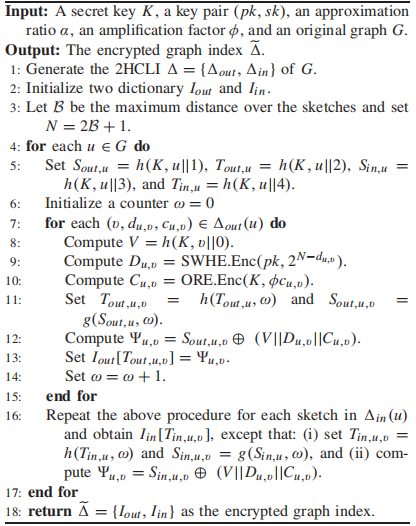
然后，云服务器直接获得，其中对于Y中的每对，。上述计算的正确性遵循SWHE的特性。可以参考文献[1]中的详细解释。

最后，云服务器将d返回给用户，用户使用密钥sk来解密d，以获得α-CSD查询的答案。

请注意，这种简单的方法不仅可以正确回答加密图形上的α-CSD查询，还可以保护节点身份信息、距离和开销信息。

然而，对于算法1获得的已加密的图索引，在不执行任何查询的情况下，仍然会导致信息泄露。一方面，它揭示了每个加密图结构的长度，即和，以及所有图中ORE加密花销的信息。另一方面，它还公开了和之间的公共节点的数量，它表示将u连接到v的顶点的数量。特别是，如果云服务器知道没有公共顶点在和之间，它得知你无法达到v。

**图2的算法2：设置算法**



**B.保护隐私的α-CSD查询**

为了增强对敏感信息的保护，我们构建了一个隐私保护的α-CSD查询方案Graph Enc2 =（KeyGen，Setup，Query），其中密钥生成过程与Graph Enc1相同，改进的索引构造和CSD查询过程，分别在算法2和3中进行了展示。

Graph Enc2的Setup工作原理如下。用户首先构建图G的2HCLI△，然后加密与u∈G相关的图（即，和），如第2-17行所述。

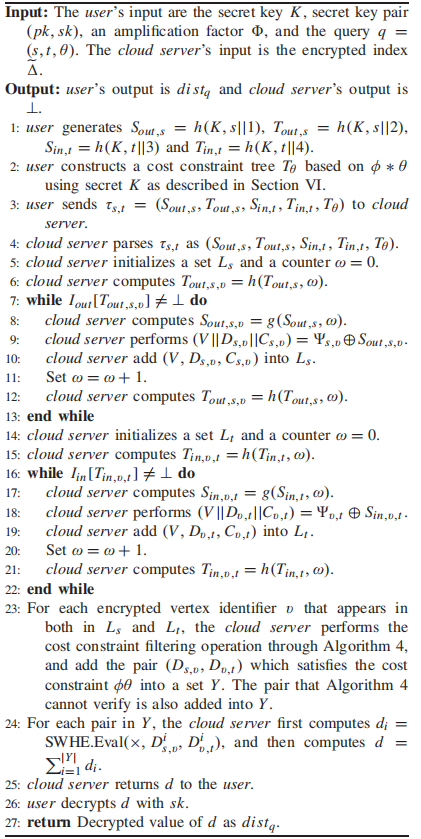
请注意，为了防止出现先前提出的方法中关于图大小泄漏的问题，我们将每个加密图和分开，并确保它们分别存储在字典中，大小为1。更确切地说，我们利用计数器ω并为中的每个个体生成唯一的和（第11行）。同样，每个个体的唯一和也可以生成（第16行）。 （或）表示该个体将存储在（或）中的位置，这确保字典（或）中的每个位置仅具有一个个体。

（或）用于与进行XOR运算。由于（或）相比于每个图是不同的，因此XOR运算得到的无法区分，这保证了静态加密图索引既不泄露和之间的公共顶点数，也不泄露成本的花销信息。

算法3中的查询实现如下。假设用户想计算s和t之间的最短距离，其总成本不超过θ。她首先生成查询标记并将其发送到云服务器（第1-3行）。在接收到令牌时，云服务器在索引中搜索并获取和（第5-22行）。也就是说，云服务器迭代地判断字典是否包含密钥。如果存在，则将相应的个体添加到集合中。

一旦获得和，云服务器就执行花销约束过滤算法（第23行），并计算d（第24行），这与直接方法中描述方法相同。最后，用户利用秘钥sk可以从云服务器解密d后对应的答复。

**图Enc2的算法3查询算法**



## VI TREE-BASED CIPHERTEXTS COMPARISON APPROACH

基于树的密文比较方法

1. Scenarios(情景)

假设有一个用户（U）和一个服务器（R）。U有许多整数，这些整数先用一种加密算法加密，然后传给R。 U向R请求来获得一个和不超过θ的整数对（x，y）。注意，除了大于、相等或小于关系之外，x、y和θ的明文不能向R公开。一种简单的方法是下载所有整数，在本地计算求和，并选择满足约束的整数对。然而，如果想要将这种计算方法加载到云端，这个方法就没有意义了。因此，应该找一个实际的办法来解决这一问题。

请注意，这种情形和众所周知的SMC方案是不同的。在SMC的设置当中，一组（两个或两个以上）具有私人投入的缔约方希望根据其投入来进行函数计算，同时除了函数的结果外什么都不显示。该函数用于许多实际应用，如外汇市场。SMC是一个协作计算问题，解决了一组互相不信任的参与者之间的隐私保护问题。

所有整数对（x,y）和约束条件（θ）的密文都上传到云服务器，云服务器来负责不等式测试。此外，我们还可以将两个密文的和与另一个密文之间的关系告诉服务器，这在文献[17]中被称为受控披露。

似乎我们可以利用同态加密技术，因为它支持计算x+y时的加操作，不过由于同态加密是概率加密，因此我们无法通过x+y和θ的密文来确定他们之间的关系。

1. Main Idea(主要思想)

基于树的密文比较协议的主要思想是用ORE原语对整数进行编码。根据我们目前的知识来看，没有方法能同时支持ORE和同态性质。因此，我们设计了一种新的方法来解决这个问题，这是由下列事实推动的：

如果我们想比较x+y和θ，我们可以比较x和θ/2以及y和θ/2。通过这两种对应关系的组合我们可以得到四种结果：

如果x>θ/2（x≤θ/2）和y>θ/2（y≤θ/2），我们可以知道x+y>θ（x+y≤θ）。在剩下的两个案例中，即x>θ/2和y<θ/2，或x≤θ/2和y≥θ/2，我们无法获获得确定性结果。这时，我们可以进一步将θ/2分为θ/4。然后我们可以比较x，y与θ/4和3θ/4。

通过迭代执行这样的操作，我们可以越来越确定确定x+y和θ之间的关系。由于ORE性质，上述密文操作很容易便可以执行。接下来，我们将展示如何通过使用树结构有效地实现这个想法。C. Details of Protocol（协议细节）

为了实现通过密文来比较x+y和θ，我们构造了一个成本约束树，它的节点表示与θ相关的特定值。为了清楚起见，我们定义E（m）为m的ORE密文。

树结构的示例如图6所示。

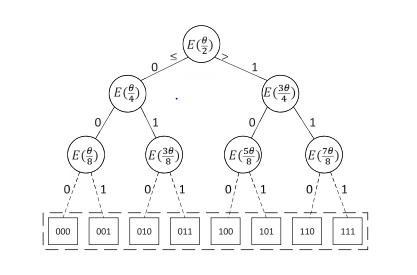
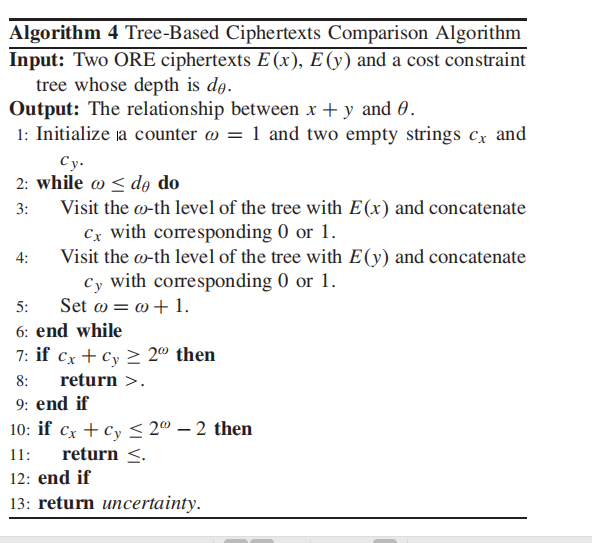


Fig. 6.

对于每个节点，我们将它的左子树定义为0，右子树定义为1。如果一个整数不大于该节点的值，则与左子路径进行进一步比较；否则，与右子树。因此，自根节点到叶节点的任一路径，我们可以得到一个路径代码，有效的显示了比较过程。例如，传入一个大小为5θ/16的整数，它将经过节点E（θ/2）、E（θ/4）和E（3θ/8），从而以代码010结束。我们定义了路径代码的长度（位数）为β。β实际上等于以dθ表示的树的深度。

现在我们可以通过如下过程来确定x+y和θ之间的关系。我们首先得到x和y的ORE密文，以及它们的路径代码cx和cy（通过遍历树得到）。当计算cx+cy时，如果发生溢出（即cx+cy≥2β），我们知道x+y>θ。如果cx+cy≤2β−2，我们也知道x+y≤θ否则，我们无法确定关系，最终以不确定性结束。

我们总结一下算法4中的这个过程

****

## VII、复杂性和安全性分析

本节介绍了提出的图加密模型Connor的复杂性和安全性分析。

## 复杂性分析（个人理解：图加密模型Connor组成：建立+查询。 各过程复杂度主要影响因素分析：建立算法时间和空间复杂度：O (nμ)，n-顶点数，μ-总草图；

## 查询算法过程：查询令牌生成+云服务端CSD查询。时间复杂度：O(ηdθ)，空间负责度： O ()。 -G每个顶点关联最大值，dθ-我不知道）

Connor主要由建立和查询算法组成，如算法2和3所述。

决定建立算法复杂度的主要因素是对图G生成的2HCLI进行加密 .让μ成为G中所有顶点的总草图,然后时间和空间复杂度都是O (nμ),其中n是G的顶点数。

查询算法包括用户端的a)查询令牌生成过程和b)云服务器端的CSD查询过程。让η为与G中的每个顶点关联的最大值。a)查询令牌生成过程的复杂度主要是由构建成本约束树决定，时间复杂度和空间复杂度都是O ()。b)对于CSD查询过程,获取Ls和Lt、执行成本约束过滤和进行距离计算的时间复杂度分别是O(η)，O(ηdθ)和O(η)；上面三个组件的空间复杂度分别是O(η)，O(η+ 2 dθ)和和O(η)。因此,CSD查询过程总的时间复杂度和空间复杂度分别是O(ηdθ)和O(η+ 2 dθ)。

## B .安全分析

我们现在介绍Connor的安全分析。为了清晰起见，我们首先讨论泄漏函数，然后证明了Connor在CQA2 安全模型下是安全的。

### 1)设置泄漏

我们构造的泄漏函数LSetup揭示了从图G安全2HCLI中可以推导出的信息，包括图n中顶点总数，所有示意图上的最大距离

B = maxu∈vmax {(V, du, V,Cu、V)∈△out(V, du, V,Cu、V)∈△in}du, V,

和的大小。更准确地说，的大小由和中草图实体的总数组成，分别用和表示。因此，泄漏函数LSetup = (n,B, , )。

注意，成对成本的顺序关系和成本与成本约束之间的顺序关系不包括在其中LSetup。因为每个实体的草图，加密后使用一个唯一的整数值进行XOR操作,这使每个实体在草图是无法区分的。

### 2)查询泄漏（个人理解：泄露函数组成：查询模式泄漏、草图模式泄漏、成本模式，分别将其形式化进行分析，模拟器S构造伪加密查询列表，敌手对于输出结果无法区分真假。）

我们构建的泄漏函数LQuery由查询模式泄漏、草图模式泄漏、成本模式泄漏组成。直观地说，查询模式泄漏揭示了以前是否出现过查询，草图模式泄漏揭示了与查询顶点关联的草图、两个不同草图之间的公共顶点以及查询顶点的草图的大小。成本模式泄漏揭示了1)成本之间的顺序关系，2)查询过程中成本与成本约束之间的顺序关系。我们将这些泄漏函数形式化如下。

定义4(查询模式泄漏):设q = (q1, q2，…， qm)是一个非空的查询序列。每个查询qi指定一个元组(ui、vi、θi)。

对于任何两个查询qi和q j,定义Sim(qi,qj) = (ui = u j, vi = v j,θi =θj), i.e., 即对于每个元素qj= (ui、vi、θi)是否成立。 然后，查询模式泄漏函数LQP (q)返回一个m×m(对称)矩阵，其中每个条目(i, j)等于Sim(qi, q j)。注意，LQP (q)不会泄漏查询顶点的标识。

定义5(草图模式泄漏):给定一个图G的安全2 HCLI 和查询q = (u, v,θ),素描泄漏模式函数LSP（，q)被定义为(∑,)。∑是一个列表,其中的每个元素都是与顶点相关联的草图,ϒ是一对(X, Z),其中X = h (v): (v,d,c)∈Iout和Z = h (v): (v d c)∈Iin是多重集合，h:{0,1}λ×{0,1}∗→{0,1}λ是一个特定的伪随机函数。

定义6(成本模式泄漏):成本约束查询中θ基本上可以由一定数量的均匀间隔构成。让dθ成本约束树Tθ(由第六节)的深度。与θ相关联的间隔((i−1)θ/ 2 dθiθ/ 2 dθ],当1≤≤2 dθ。分配每个区间μ列表。,第i个间隔和μi有关。 它存储这个区间的所有成本值。泄露间隔信息形式的Arr数组,其中第i个间隔是μi 元素(即。,Arr[i]=μi)。此外，假设z是被查询顶点的草图中的条目总数。对于每一对cost ci和cj，其大于、相等和小于的顺序关系分别可以表示为1、0和- 1。泄露的订单信息的成本是一个z×z(对称)矩阵∇每个条目(i, j)是1,0,或者−1。

因此,成本模式泄漏函数 Lcp (, q) = (Arr∇)。 因此，LQuery = (LQP(q)，LS P(， q)，LC P(, q))。

泄漏函数是在2HCLI上定义的，而不是在原始图上定义的。事实上，原始图的信息泄漏仅限于被查询的源-目标顶点的最小路径数。它可以定义为一个n×n(对称)矩阵，其中n是图中顶点的个数。其中的每个元素为NULL、0或正整数，表示不确定状态(拓扑结构得到很好保护)、断开连接或两个查询顶点的最小路径数。

对于2HCLI成本值，我们引入一个使用者放大系数φ来扩大明文和密文空间。因此，服务器不能仅从泄漏函数LC P(，q)所显示的订单信息来推断真实的成本值。对于2HCLI中的距离值，我们使用SWHE加密来保护它们的真实值不受服务器的影响。

定理1:如果加密原语g、h、ORE和SWHE是安全的，那么所提议的图形加密方案= (K eyGen、Setup、Query)是(LSetup、LQuery)-在自适应选择查询攻击情形下是安全的。

证明:关键思想是构造一个模拟器S。给定泄漏函数LSetup和LQuery, S构造一个伪加密的2HCLI结构△∗= {I\*out, I\*in}和一个查询\*列表。如果对于所有的PPT敌手A来说，他们无法区分这两款游戏的真假，我们可以说我们的图形加密方案(LSetup,LQuery)是安全的，可以抵抗自适应的选择查询攻击。

### 3) 仿真 （个人理解：对于敌手攻击，先从已有查询结果查询是否存在，有就用以前的生成，没有就根据定理公式生成）

S处理每一个顶点的ui(1≤i≤n)，生成一个基于泄漏函数LSetup在2HCLI中假的I out。S随机选择wi ui n 1 wi =,和样品Ii←{0,1}λ和ηi←{0,1}λ均匀无重复。对于所有的0≤i< wi, S会采取以下步骤来模拟每个素描:S计算lwi = h (li wi)和ηwi = h(ηi wi), h是一个特定的伪随机函数。然后，它通过计算v∗= h(K∗，v||0)来加密ui草图中的每个顶点v，其中K∗是一个伪密匙。它随机生成两个整数d和c，并使用SWHE和ORE方案对2N - d (N = 2B + 1)和c进行加密，得到密文D\*和C\*。

让=ηwi⊕(V∗| | D∗| | C∗)。S将W\*存在Iout\*索引中，I\*out显示在索引外。也就是说，I\*out [lwi] =。类似地，S生成一个伪I∗in，最后得到伪2HCLI∗= {I∗out, I∗in}。

模拟q∗：给定泄漏函数LQuery = (LQP(q)，LS P(△,q)，LCP(q))， S模拟查询令牌如下。S首先检查被查询的顶点S和t是否出现在之前的任何查询中。如果s之前出现过，s就会将S∗out s和T∗out 设置为之前使用的值。否则,它会T∗out =li和S∗out,s =ηi一些以前未使用的li和ηi。然后记得ηi之间的协会,李。查询顶点t，S会采取相同的步骤:设置S∗in,t和T∗in,t,类似地,将t与所选ηi和li联系起来。

模拟一个假成本约束树T∗θ,S首先检查如果θ出现在查询之前任何查询。如果θ出现，此前,S设置T∗θ以前使用的值。否则,构造一个完整的基于二叉树的θ和带有随机生成密钥的ORE模型加密每一个树节点。S返回这个加密树作为T∗θ。

S模拟查询过程，如下所示。给定查询令牌(S∗out,T∗out, s, S∗in,t, T∗in,t, T∗θ), S首先检查先前查询查询结果是否存在。如果是，S返回以前用作查询结果的值。否则，S检查被查询的顶点S(或t)之前是否被查询过。如果查询顶点s在之前的查询中出现，s将Ls\*设置为以前使用过的Lsp(f, q)的∑值。否则，对于新出现的顶点s, s将执行以下步骤:用s生成草图,首先初始化一组Ls∗年代和计数器ω∗= 0。(细节我也不太懂)

然后,它迭代计算T∗out,s,v = h (T∗out,s,w∗)和S∗out,s,v = g (S∗out,w\*),并添加元组(V∗, D\*s,d, C∗s,d)到L∗s,直到I∗out(T∗out, s, d)不存在,在那里( V∗s,D\*s,v, C∗s,v) =I∗[T∗out, s, d]⊕S∗out,s,v。类似地，S获取顶点t的集合 Lt。在获得L\*s和L∗t, S执行成本约束过滤操作基T∗θ得到候选集Y∗。该定理由SWHE的CPA-security得到，也就是说，S基于Y\*执行SWHE计算并返回查询结果。

由于加密原语g、h、ORE和SWHE是安全的，所以伪的2HCLI结构\*和查询序列q与真实的无法区分。因此，对于所有的PPT敌手A来说，他们无法区分这两款游戏是真实的还是理想的。因此,我们有

|Pr(Real ii, A(λ)= 1)−Pr(Ideal,A, S(λ)= 1]|≤negl(λ), 其中negl(λ)是一个可忽略的函数。

## VIII 性能评估

本节通过对真实的数据集进行实验，从而对图形加密方案进行评估。

## A.步骤

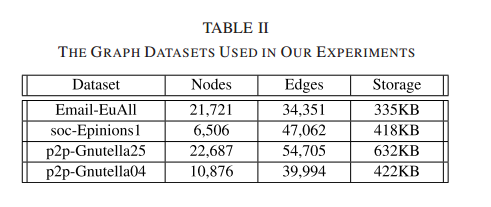
### 1.试验

我们实现了在 [2] 中介绍的构建 2HCLI 的方法。实现中的 ORE 和 SWHE 分别遵循 [15] 和 [16] 中描述的方法，此外，我们采用GMP库进行大整数运算。我们设置了安全参数 λ= 128 并且所有基本密码基元依赖OpenSSL库做支撑。我们实验中的所有算法都是用c++实现的。

硬件配置：Intel Xeon处理器，2.6 GHz, 8gb内存。

### 2.Graph 集合

我们实验中使用的数据集见 表2。所有这些数据集都是公开的来自斯坦福快照网站e3，并按照指导建模图表。对于socket - epinions1和Email-EuAll数据集，我们在有限的计算资源下随机选择它们的子集来构造可用索引。因为这些图是未加权的，我们为每条边生成一个长度和一个权重，其值服从均匀分布在1到100之间。采用成本准则作为约束条件。



### 3.方法比较

由于这是第一项解决加密图上CSD查询问题的工作，所以我们将我们的这个方法与未加密图上的方法进行了比较。我们按照 [2] 中引入的最先进的纯文本图方法实现了这种方法。唯一的区别是我们在原始图上构造2HCLI，而不是覆盖图。因此，我们的这个方法的实现具有更高的查询效率，但与此同时，索引构造的复杂性会更高。

### 4.查询集

我们针对每个数据集随机生成 200 个查询。每个查询中的源 s 和目标 t 也是随机选择的。另外，每一对 (s, t) 的成本约束 θ 的设置如下所述：我们用cmin表示从s 到t的所有路径的最小成本的下界,用cmax表示从s到t的最短路径的的最小花费的上界。**如果成本约束 θ< cmin,那么将会没有有效的查询结果;如果成本约束 θ> cmax,那么最短的距离（shortest distance）将会是一直是有效的查询结果**，为了减轻 θ 对性能的影响,我们为每个查询随机选择 50 个 θ 值,这些值得区间为[cmin, cmax]。

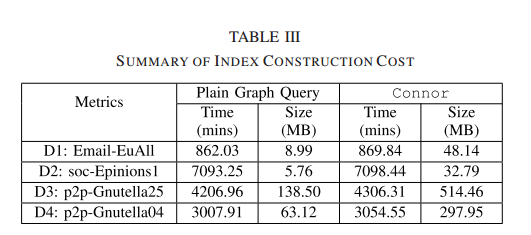
**另一个重要的参数是 α ,这决定了近似保证 α-CSD查询。由于 α 对于所有查询是一个恒定常量,所以我们把它看作一个系统参数而不是特定查询的一部分。为了实现查询精度和系统效率之间的平衡,我们为所有查询设置了近似系数 α= 1.5**

## B.安全2HCLI和查询令牌的评估

### 1.索引大小和构造时间

图的索引构造是一次性的并且是离线计算的，这个过程包括两个步骤:

一、是构建无格式**2 hcli**,也就是与原始无格式**CSD**的索引构造相同,另一个是加密无格式**2 hcli**（这是本文的重点）,因此, ,我们考虑将第一步的输出作为未加密图的索引。索引大小和构建时间如表三所示。需要注意的是，不同数据集的索引大小和构建时间有很大的差异，这主要是由于图的拓扑的差异造成的。**与原来最短距离查询中任意两个顶点之间只有一条最短路径不同，在CSD查询问题中，任意两个顶点之间通常存在多个约束最短路径。从直观上看，密集图可能比稀疏图具有更高的索引构建成本。**

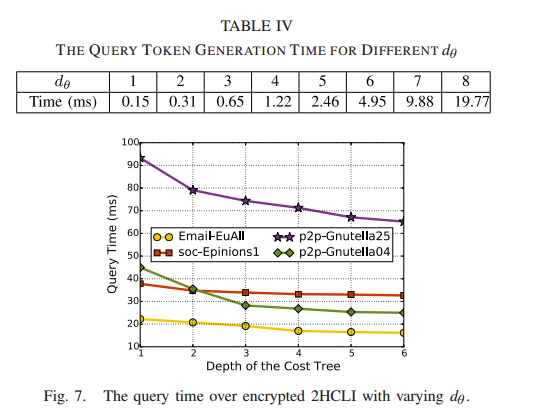


通常情况下，**每个加密索引的大小大约比对应的普通索引大6倍**。最重要的是索引的构建时间的比较：加密图略高于未加密图。因此，提高加密图上索引构建效率的关键是加快构建该图的无格式 2HCLI的过程。我们把这些尝试留到以后再做。

### 2.查询令牌生成

查询令牌的构造独立于特定的图形，我们现在分析查询令牌的大小和生成时间。查询令牌主要由5个元素,也就是： Sout,s, Tout,s,  
Sin,t , Tin,t , 和 Tθ. 前4个元素的长度都是16字节。**因为每个ORE密文的大小是16字节,一个深度为 dθ 成本树 Tθ 的大小为**

**字节。因此,查询令牌的总大小是。由于 dθ 是一个相对较小的值，所以查询令牌的大小通常是小于1 KB。不同dθ的查询令牌生成时间如表4所示**。虽然查询令牌生成时间在 dθ 的影响下显著增加，但是一般情况下时间花费适中(例如,当dθ小于等于6)。



**C.查询效率和准确性的评估**

1)查询效率:评估查询效率,对于每个θ,我们生成成本约束*dθ*树有不同的深度。查询时间定义为从提交查询令牌到接收查询结果的时间间隔。我们计算了200个查询的平均查询时间。

平均查询时间与不同*dθ*加密2 HC LI是描绘在图7中,dθ从1增加到6。我们可以看到，**对于不同的图形数据集，查询时间变化很大**。对于每个数据集,提高*dθ*可以减少查询时间。这是因为大*dθ*可以过滤掉更多的距离对超过成本约束从而减少了在时间消耗上占主导地位的SWHE算法的候选距离计算量。

SHEN *et al.*: 基于云的加密图的近似CSD查询

图8所示、查询时间在平原2 HCLI加密2HCLI (dθ= 6)

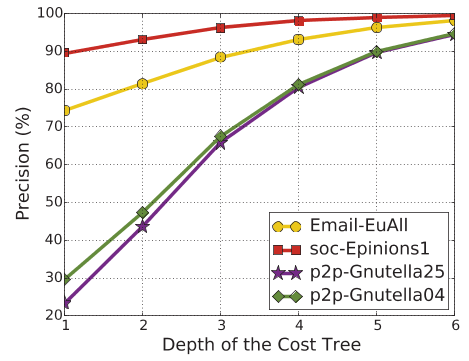


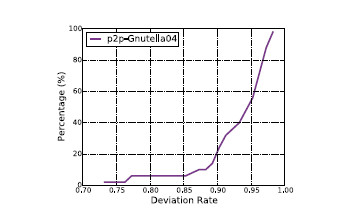
图9所示、查询精度的不同深度dθ成本树

图8给出了不同数据集的普通和加密场景下的查询时间。**加密后的2HCLI的查询时间要比普通2HCLI的查询时间长，这是因为加密后的2HCLI对密文的操作非常耗时(如成本过滤和距离计算)**。此外，这些操作的时间复杂度与表III中所列的图形索引的大小密切相关，这导致了图8中四个数据集的差异。

2)**查询精度:在Connor中，影响查询精度的因素有两个，即基于树的密文比较和距离计算**。前者可能会在候选集合Y中保留一些不满足成本约束的距离对，而后者利用SWHE的特性，基于Y中的所有候选，获得一个近似但不精确的最短距离。

我们使用众所周知的度量精度(P)来评估成本约束滤波过程的精度。*P* = ，其中*Tp*和*Fp*分别表示成本真正满足或超过成本约束的Y中距离对的个数。我们使用与上面介绍的相同的查询，并为每个查询计算P。最后,我们可以获得的平均精度P̄所有的查询。

图9给出了P̄查询精度和深度之间的关系的成本约束树*dθ*在不同的数据集。我们可以看到所有的数据集,**P̄随*dθ*,因为成本约束树更大深度*dθ*帮助我们检测约束违反有更高的概率**,在第六部分讨论。特别是, 当*dθ*= 6时，P̄超过94%以上所有的数据集。

图10所示、CDF实验组的偏差率不同的查询(*dθ*= 6)

为了评估最终查询结果的准确性，我们提出了一个名为偏差率的度量。设re和rp分别为Connor返回的查询结果和算法对对应的平面图返回的查询结果。然后,我们定义偏差率ξ= re / rp,这表明多远偏离rp。显然，接近1的偏差率描述了更准确的查询结果。

图10为数据集p2p-Gnutella04上偏差率的累积分布函数(CDFs)。我们可以看到,**ξ大于0.90查询结果的80%以上,并大于0.73是最糟糕情况**。因此，Connor能够在计算复杂度适中的情况下达到较高的精度。

## VIIII 结论

在本文中，我们介绍了Connor，第一种支持基于云的近似CSD查询的图形加密方案。特别地，我们提出了一种基于树的密码文本比较协议，用于控制披露的成本约束过滤。安全分析表明，**Connor可以实现CQA2安全**。我们实现了一个原型，并使用实际的图形数据集评估了性能。评价结果验证了Connor的有效性。在未来的工作中，我们计划设计支持动态索引更新的技术。