一种基于GPU调度加速和关联事务融合的全同态数据库中间件

姓名：关荣鑫

www.huawei.com

HUAWEI

专利申请合规Checklist

检查项类型	检查提示项	确认结果 [不涉及，是，否]
是否直接引用外来非公开信息	（1）Idea/交底书/专利申请文档不涉及在外部交流、技术合作、产品合作、采购等场景中从他人获取到的外来非公开信息，包括但不限于他人资料、仿真数据、测试结果、模型图、原理图、器件结构图、软件算法和其他非公开技术方案等；尤其注意不能引入带有他人标志或保密标识/水印的图片或资料；	否
是否以外非信息为基础进行改进	（2）Idea/交底书/专利申请文档不存在以外非信息的技术方案或技术构思为基础或主要设计参考而提出的改进方案，尤其注意不能在背景技术中引用外非	否
是否涉及外网引入	（3）Idea/交底书/专利申请文件不涉及从外部网站下载的有保密义务和/或使用限制的信息，如在线签署保密协议或者最终用户协议中限制分发或商业使用等	否
	（4）Idea/交底书/专利申请文件不涉及利用个人注册账号等途径从外部网站下载的非公开信息	否
职务成果归属	（5）Idea/交底书/专利申请文档所涉及的发明方案100%属于发明人在华为的职务成果，不涉及前雇主/前研究机构/学校的职务成果	是
其他	（6）Idea/交底书/专利申请文档等使用的技术材料中，不涉及来自于其他部门/其他产品线或关联公司，不包括来源不明的或无法确定是否属于外非信息的图片及资料	不涉及
	（7）Idea/交底书/专利申请文档不涉及其他公司的私有协议、私有接口、私有标准等。	不涉及
	（8）Idea/交底书/专利申请文档不涉及通过对未上市产品或软件通过拆解、反编译、反汇编等技术手段获取到的信息	不涉及
是否涉及出口管制	（9）如果Idea含有海外发明人，请确认技术是否达到出口管制规格。如是，是否已获得相应的出口许可?	否
发明人署名确认	（10）确保所有发明人都做出了实质性贡献。（包括：a. idea的初始创造者，关键技术路线设计者； b.《技术交底书》主笔人，技术实施例完整设计者；c.对idea的至少一个创新点的新颖性或创造性提供了有帮助的技术特征者。）	

发明背景 - 云端数据库广泛应用于各行各业

- 在数据规模爆炸增长的时代，越来越多政府、企业，与个人使用云端数据库存储海量私有数据，享受云端数据库例如弹性扩展等诸多优点。

案例一：政务数据管理

政府存储海量政务数据（例如财政信息和公民身份信息）于云端。政府查询与更新财政信息和公民身份信息。

图1：陕西省财政厅采用华为云 ServiceStage 重构数字财政，实现政府财政信息的统一化存储与管理。

案例二：金融数据分析

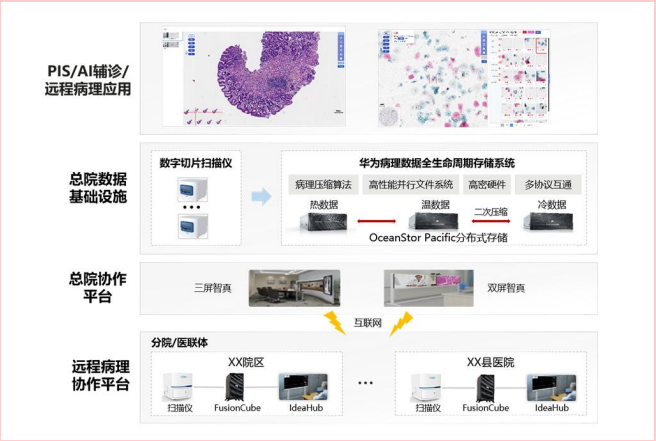
银行存储海量金融数据于云端。银行查询云端数据库进行融合数据分析和异构数据整合管理。

图2：工商银行采用华为 GaussDB (DWS) + Fusioninsight MRS 实现湖仓一体的融合数据分析解决方案。

案例三：医疗数据诊断

医院存储海量医疗数据于云端。医院查询病理数据以进行高效的阅片诊断。

图3：上海瑞金医院采用华为 OceanStor Pacific 分布式存储，实现数字化病理高性能调阅和数据缩减



发明背景 - 云端密态数据库极具商业价值

- 两大商业动机推动云端密态数据库发展
  - **数据机密性是数据库核心需求**：用户私有数据具有高度机密性。当部署于云端数据库时，一旦数据明文泄漏，恶意攻击者便可攫取机密数据进行牟利。因此，必须保证云端数据库的数据机密性才能释放其商业价值。
  - **法律法规强制保护数据机密性**：政府立法保护机密数据，例如中华人民共和国数据安全法。因此，必须保证云端数据库的数据机密性才能推动其合法商业落地。
- 工业界亟须云端数据库保障数据机密性，投入海量资源研发“密态数据库”。
  - 案例1：华为 GaussDB 迫切需要密态数据库解决方案，在云存储，金融，电子商务等领域实现数据的可算不可见（摘录自 GaussDB Full Encryption [VLDB’21]）。
  - 案例2：微软 Azure SQL AE 依靠 TEE 实现了基于 AES256 算法的数据透明加解密，但具有极大局限性，例如不支持密文层级直接运算，依赖 TEE 硬件环境（摘录自 Azure SQL Database Always Encrypted [SIGMOD’20]）。
  - 案例3：MySQL Enterprise Encryption 提供基于非对称加密算法的列级数据加密能力，但面临和 Azure SQL 数据库类似困境。
- 学术界发表众多关于密态数据库的顶级国际会议论文
  - CryptDB [SOSP’11], Senate [Security’21], Titanium [Security’22], HE3DB (CCS’23) ...
  - 然而，现有研究存在三大痛点：
    - ◆性能低下：低吞吐、高时延，为了完成特定的密态数据库 CRUD 任务须要承受极高的密码学代价。
    - ◆安全易损：执行密文事务需进行密文解密、明文运算与再加密的过程，容易遭受数据泄露。
    - ◆架构单一：绑定单一特定数据库架构（例如 SQL 数据库），无法推广到其他多种数据库架构。

发明背景 - 理想的云端密态数据库应实现的目标

安全性

- 目标一：**数据机密性**
  - 在数据于**不可信**云端数据库传输、存储和计算的全生命周期中，数据明文不被泄露。
- 目标二：**数据完整性**
  - 在数据传输、存储和计算过程中，数据不被恶意篡改或丢弃。

功能性

- 目标三：**高效性**
  - 提供超越现有主流密态数据库的事务执行高吞吐量和低时延，尽可能逼近非密态数据库的性能。
- 目标四：**异构支持**
  - 支持 SQL 和 NoSQL 等多种数据库架构。

发明背景 - 全同态加密

- 全同态加密（Fully Homomorphic Encryption，FHE）是一类强安全的新型非对称加密算法。
- 全同态加密**可抵抗量子攻击**。它基于容错学习问题(LWE/RLWE)或格密码学最短向量问题(CVP)，对比当前主流的非量子安全的非对称加密算法（例如 RSA）更具安全性。
- 可以直接对全同态加密的密文进行**加法**和**乘法**运算，而无需经历传统加密算法的不安全且开销高的运算流程（即先解密，明文运算，再加密）。

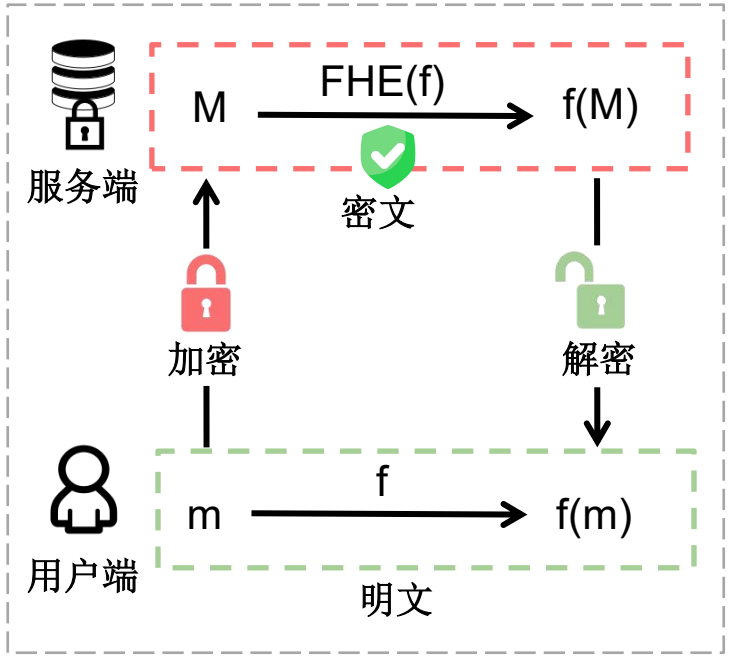


图4：全同态加密算法的运算流程

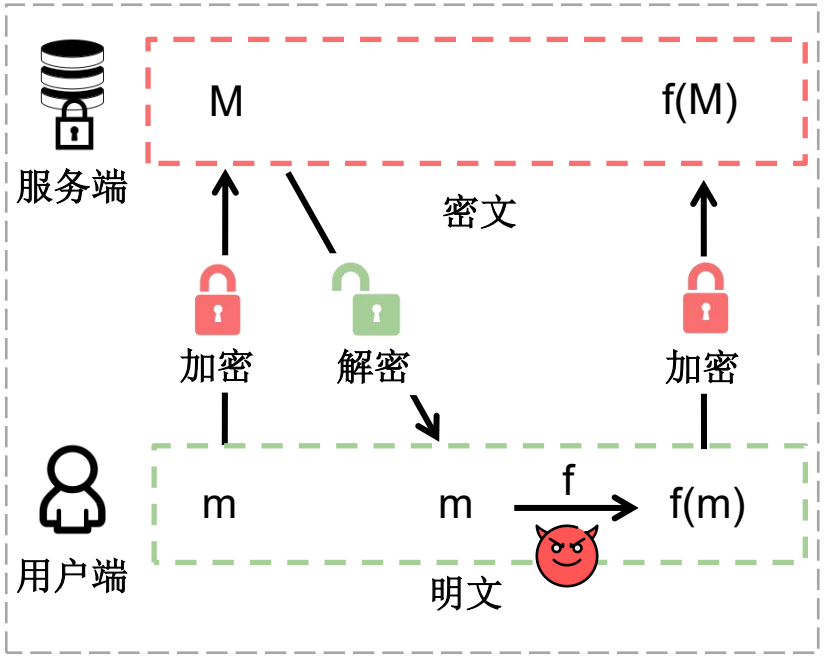


图5：传统加密算法的运算流程

发明背景 - 全同态加密

- 全同态加密发展迅猛，目前已迭代到第四代：
  - 第一代：基于格密码学，证明全同态加密的理论可行性，但性能不可用。代表作：[Gentry 2009]，[DGHV 2010]。
  - 第二代：基于LWE/RLWE，引入leveling schemes，支持整数算术运算，性能可用。代表作：[BFV 2012]，[BGV 2011]。
  - 第三代：基于第二代引入多种性能优化策略，例如避免relinearization和优化bootstrapping。代表作：[GSW 2013]，[FHEW 2014]。
  - 第四代：引入高效的密文取整操作，降低噪声增速，支持浮点数算术运算或布尔表达式运算。代表作：[CKKS 2017]，[TFHE 2016]。
- 本发明考虑纯软件的（不使用第三方可信硬件的）基于全同态加密的密态数据库场景，支持基于整数或浮点数算术运算的全同态加密算法。

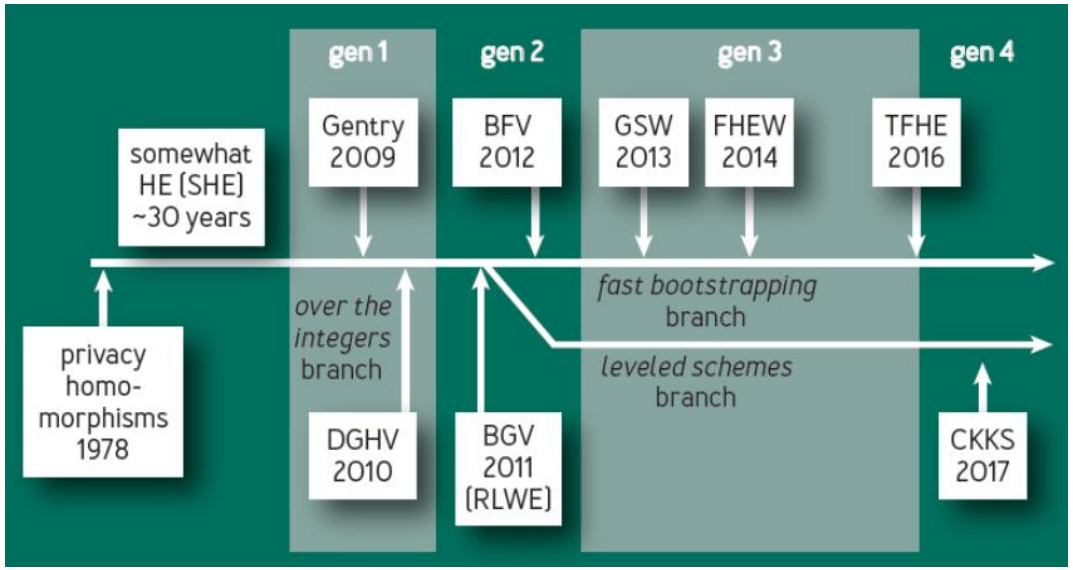


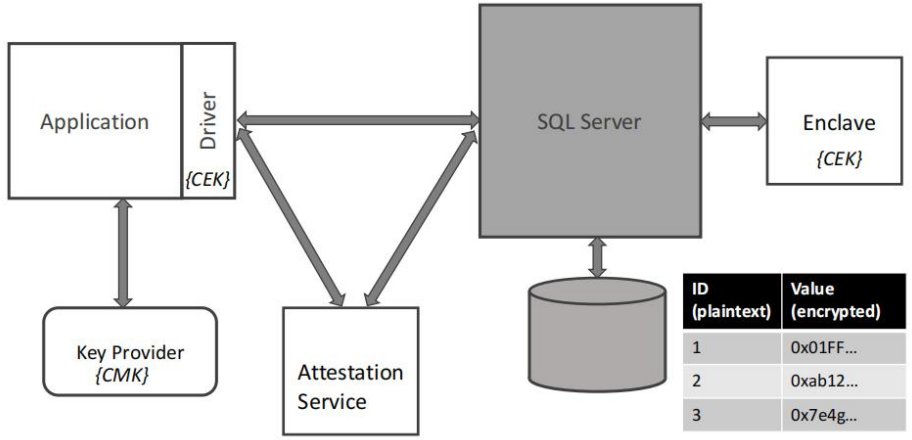
图6：全同态加密算法发展路线

现有工作 - 密态数据库

基于传统密码学的密态数据库

- 使用传统密码学算法加密关键数据
- 优点：
  - 增删查改性能优秀
- 缺点：
  - **不支持密文运算**，或需要在 TEE 中解密数据并对明文进行运算

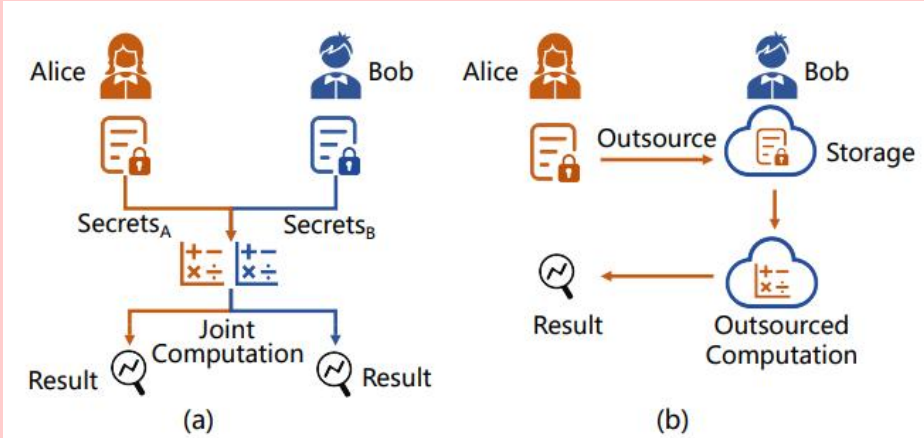
图7： Azure SQL AE 支持加密列级数据，并在 TEE 中提供明文比较和字符串匹配等复杂功能。



基于同态加密的密态数据库

- 使用同态密码学算法加密全表数据
- 优点：
  - 支持密文查询和聚合运算
- 缺点：
  - **性能极差**，查询和修改为线性复杂度
  - **不支持多种异构数据库**

图8： HE3DB 使用 TFHE 完成查询的谓词比较操作，使用 CKKS 完成聚合运算。

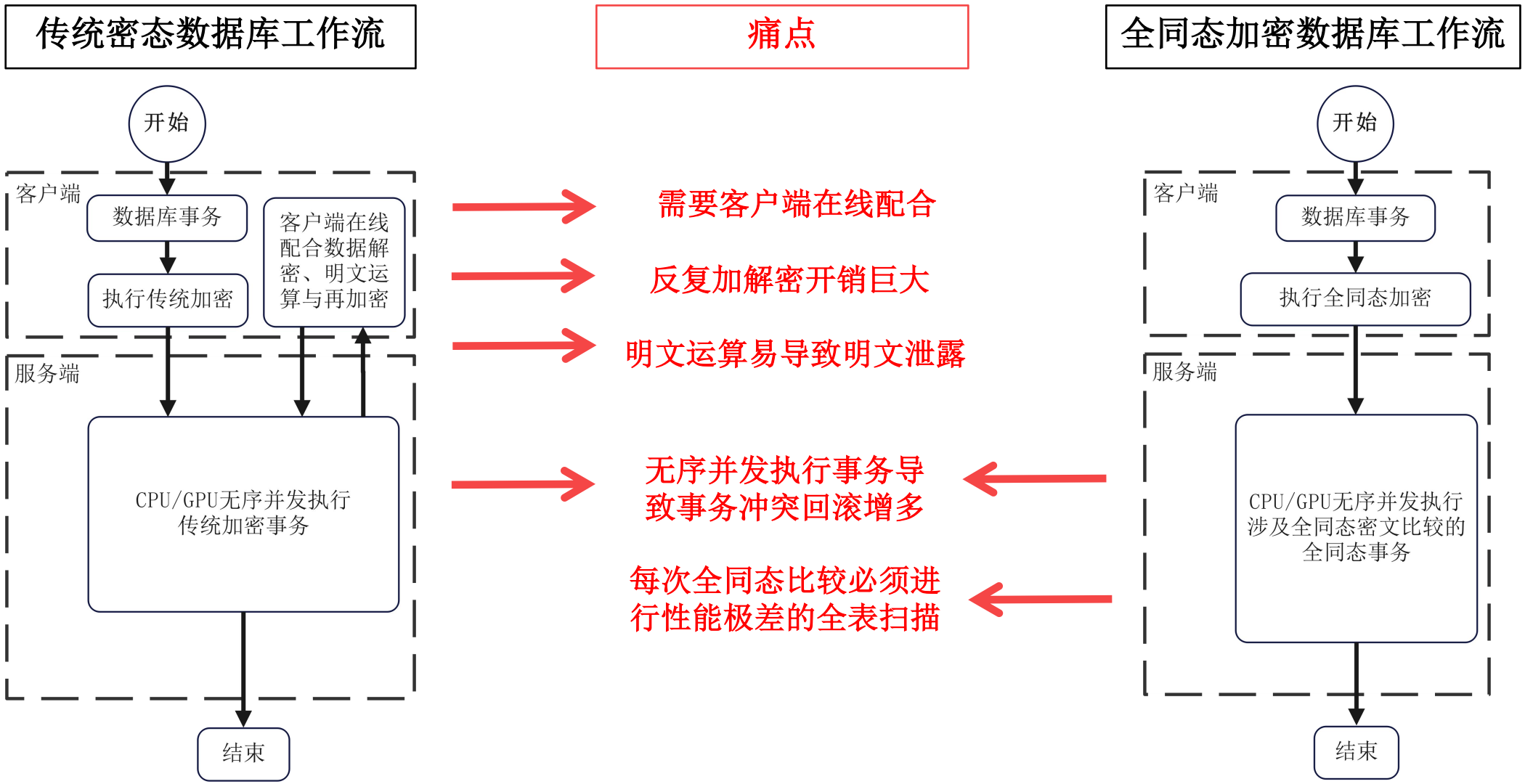


发明目标 - 安全、高效、支持异构的全同态数据库中间件

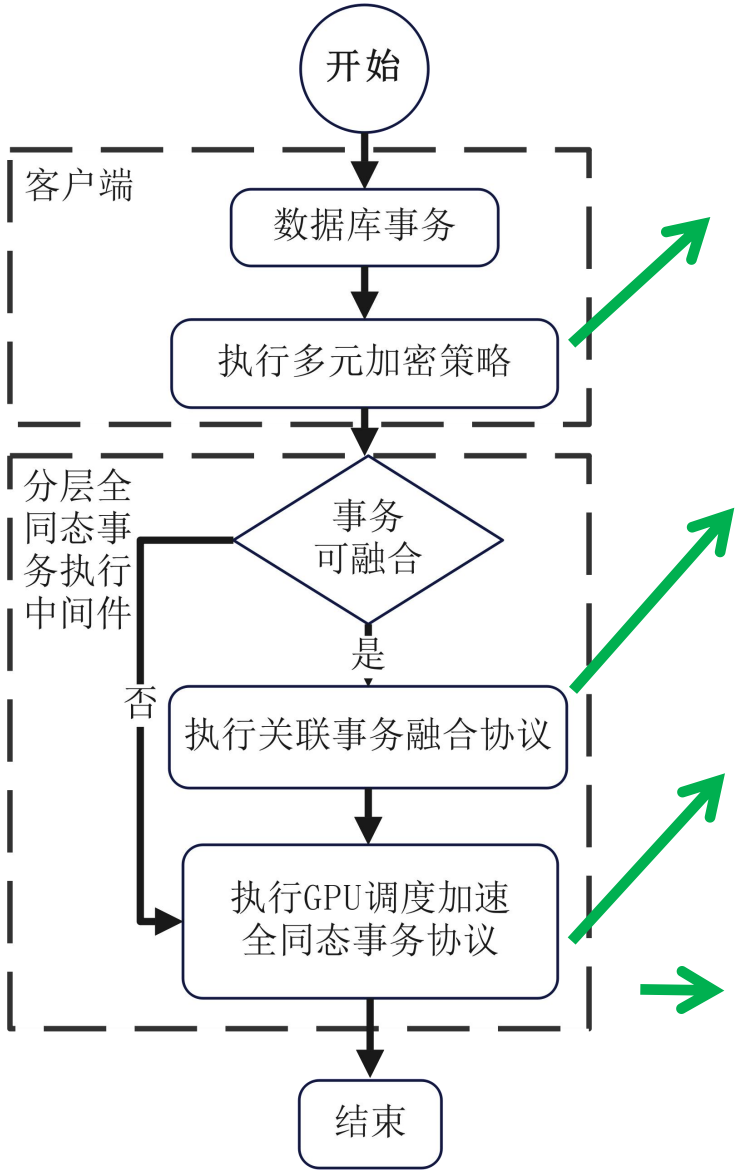
- 本发明是首个使用GPU加速和关联事务融合且支持异构数据库的全同态数据库中间件。
  - 安全性：使用全同态密文加密关键数据，可直接在全同态密文层级进行运算，保护关键数据不被泄露、篡改与删除。
  - 高效性：保持云端数据库操作原有复杂度，并使用GPU加速和关联事务融合技术加速全同态事务执行。
  - 异构支持：引入数据库驱动层对异构数据库算子进行统一抽象，为异构云端数据库（例如 SQL 和 NoSQL）提供支持。

系统	数据机密性	密文可算性	高效性	异构支持
MySQL	✗	✗	✓	✗
Azure SQL AE [SIGMOD’20]	✓	✗	✓	✗
GaussDB [VLDB’21]	✓	✗	✓	✗
CryptDB [SOSP’11]	✓	✓	✗	✗
HE3DB [CCS’23]	✓	✓	✗	✗
本发明GACM	✓	✓	✓	✓

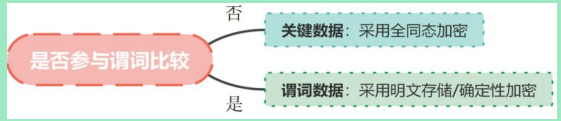
现有传统/全同态密态数据库工作流程与痛点



本发明工作流程与实现目标总结

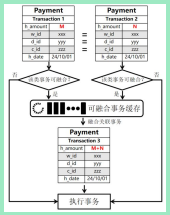


- **多元加密策略**



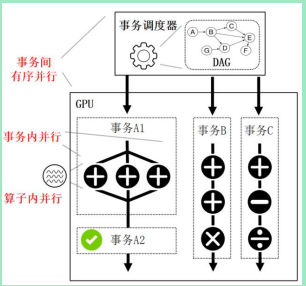
实现目标：(**数据机密性**)使用全同态加密保护关键数据，确保关键数据全生命周期明文不可见；(**数据完整性**)确保明文与密文数据不被篡改；(**高效性**)使用明文/确定性加密存储参与谓词比较的数据，避免全同态密文比较

- **保护点1：关联事务融合协议**



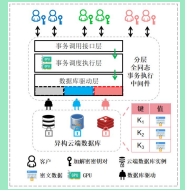
实现目标：(**高效性**)减少数据库端的全同态事务**实际执行数量**，进而提高端到端事务处理吞吐量并降低平均处理时延。

- **保护点2：GPU调度加速全同态事务协议**



实现目标：(**高效性**)将存在数据依赖关系的事务分配到相同GPU计算流串行执行，将无数据依赖的事务交由多条GPU计算流并行执行，**消除事务并发冲突导致的事务回滚与资源浪费**

- **分层全同态事务执行中间件**



实现目标：(**异构支持**)采用分层中间件架构，通过提供特定于数据库的可插拔驱动实现对异构云端数据库的支持。

技术挑战1：全同态密文比较操作损害数据库可伸缩性

- **技术挑战**：全同态密文比较操作采用如图y所示的迭代算法，时间复杂度为O(MN)，其中M为参与比较密文差值的精度，N为比较的密文总数。这使得每个含全同态密文谓词比较的数据库操作都会引发**全表线性扫描**，使各种数据库索引数据结构（例如B+树）失效，严重损害数据库可伸缩性。
- 解决方案：**多元加密策略**。
  - 核心思想：避免开销极大的全同态密文比较，同时最大化保护数据机密性。
  - 根据数据在事务中是否参与谓词比较，将数据分为以下两类并采取相应的(加密)存储策略：
    - 对于不参与谓词比较的关键数据，使用全同态加密
    - 对于参与谓词比较的数据，使用明文/确定性加密存储
- 有益效果见有益效果 – 本发明可高效伸缩

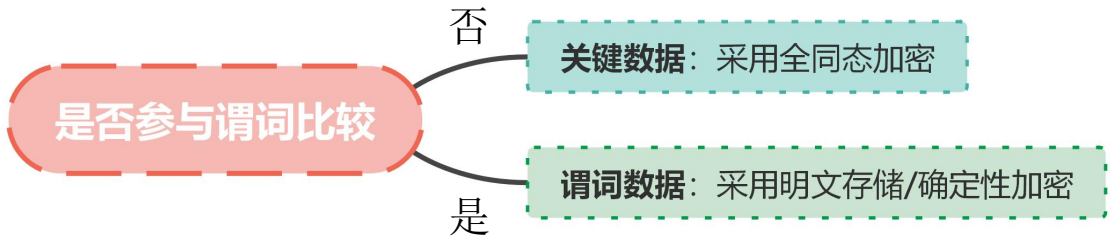


图9：多元加密策略

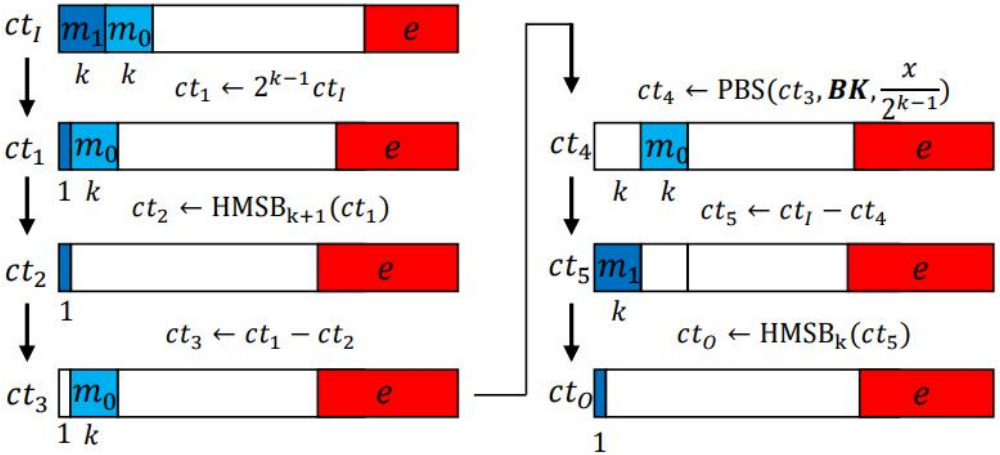


图10：全同态密文比较迭代算法

技术挑战2：在数据库引擎层面实现本专利技术保护点面临两大技术挑战

- 在数据库引擎层面实现本专利技术保护点面临两大技术挑战：
  1. **性能衰减**：本专利的计算开销(例如全同态运算)挤占数据库主机宝贵的运行资源，不利于释放本专利的最佳性能。
  2. **不支持异构数据库**：需要针对各种数据库分别实现该专利技术方案，无法做到一套专利实现在多种数据库上运行。
- 解决方案：适配异构数据库引擎的分层全同态事务执行中间件
  1. **事务调用接口层**：面向客户提供事务统一实现和调用接口，使客户可以无感知地在异构云端数据库上实现与执行事务。
  2. **事务调度执行层**：运行全同态数据库GPU二重加速协议与关联事务融合协议。
  3. **数据库驱动层**：通过数据库驱动连接中间件与异构数据库，每个驱动只需提供针对于特定数据库的操作原语(例如查询与更新)。

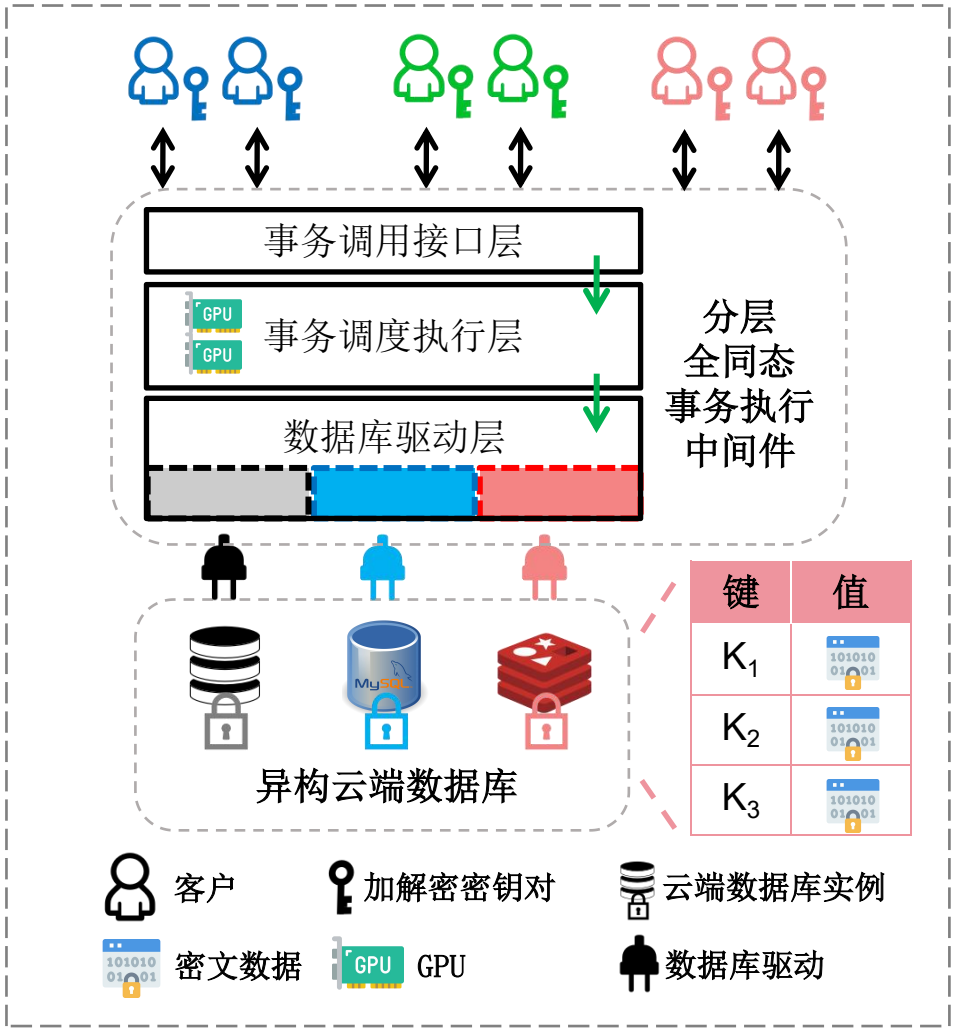


图11：可插拔全同态事务执行中间件使用场景

保护点1 – 关联事务融合协议

- **关联事务**是指业务逻辑相同但传入参数不同的多个数据库事务。
- **现有数据库不对关联事务进行识别与融合操作。**
- **观察：**在各类数据库应用中，关联事务**广泛存在且高频执行**。例如TPC-C的Payment事务和SmallBank的WriteCheck事务
  - 通过识别关联事务并融合它们的传入参数，从而生成并执行单一等价事务，可以显著降低全同态密文运算次数与降低关联事务并发冲突。
- **本专利解决方案：关联事务融合协议**
  1. 事务融合元信息声明：在事务静态编码阶段，开发者提供事务的融合元信息，例如(1)事务可融合性；(2)两个关联事务可融合的判定谓词；(3)关联事务融合指令，等等。
  2. 关联事务在线融合：在事务动态执行阶段，根据融合元信息，该发明对事务进行识别和分流处理：

     ◆对于不可融合事务，直接执行。

     ◆对于可融合事务，先将一定时间窗口内（例如100ms）接收的可融合事务放入缓存区，然后融合它们的传入参数生成单一等价事务，最终执行等价事务。
- **本保护点与现有方法的差异**
  - 进行关联事务融合执行，提高事务执行吞吐量并降低平均时延
  - 减少事务实际执行数量，降低计算资源占用率

保护点1 - 关联事务融合协议

- 示例：可融合的 TPC-C Payment 事务

```
def payment(
  h_amount: hadd,
  w_id: equal,
  d_id: equal,
  c_id: equal,
  h_date: nil):

  # 0. 开始事务
  tx = DB.startTx()
  # 1. 获取仓库、区域、顾客信息
  ...
  # 2. 更新顾客余额和顾客今年累计支付金额
  tx.execute(update_customer_sql, (
      hsub(customer.balance, h_amount),
      hadd(customer.ytd_payment, h_amount),
      c_id))
  # 3. 更新仓库金额和区域金额，插入支付历史
  ...
  # 4. 提交事务
  tx.commit()
```

Payment关联事务融合指令是对h_amount执行同态加法操作

Payment关联事务可融合的判定谓词是w_id, d_id, c_id各自满足等价关系

融合所得的新Payment事务可取任意原事务的h_date作为新事务的h_date入参

融合元信息

图12：带融合元信息的TPC-C Payment事务部分实现代码

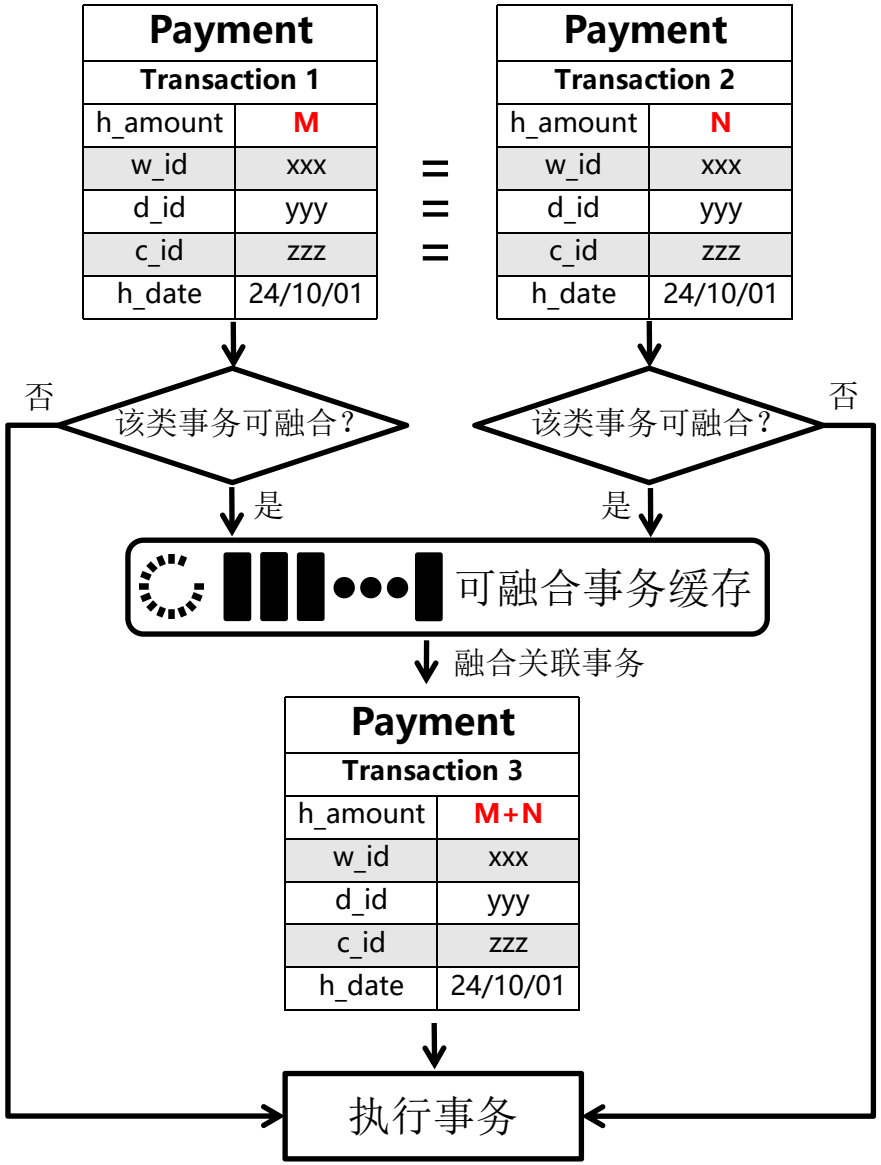


图13：关联事务在线融合

保护点1 – 关联事务融合协议

- 有益效果
  - 图14展示使用TPC-C测试SQL数据库的端到端吞吐，其中item表含10^6行记录。图15展示使用SmallBank测试NoSQL数据库的端到端性能，其中account表含10^6行记录。**随机生成所有事务请求**且**设置10%事务可融合**。GACM-M是保护点1的端到端性能结果。
  - 实验表明，保护点1使得本发明吞吐量超越最新系统1.2-2.3倍，平均时延下降最高30.9%。
  - 更全面的数据参考有益效果 – 本发明实现更高吞吐量和更低平均时延

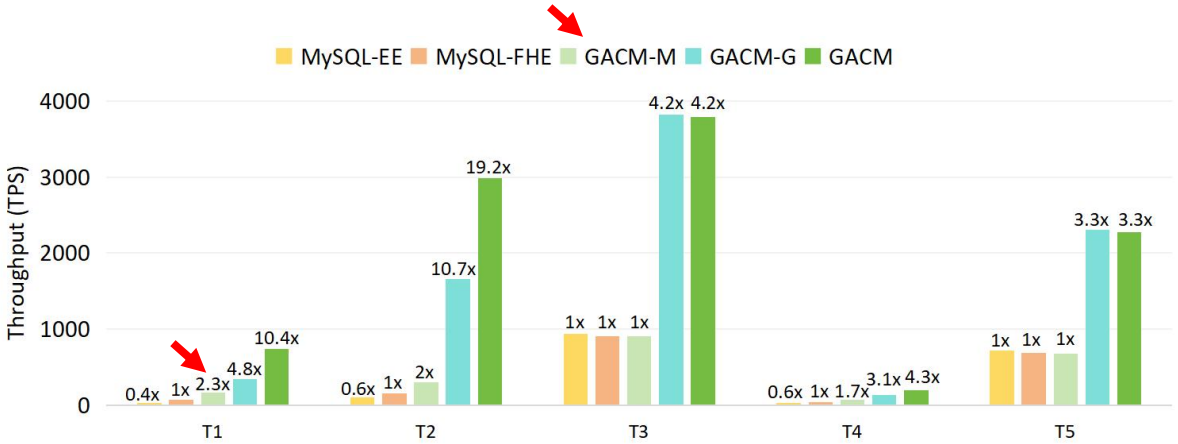


图14：基于TPC-C数据集的SQL密态数据库吞吐量对比

SmallBank	Redis-FHE		GACM-M		GACM-G		GACM	
	tput	latency	tput	latency	tput	latency	tput	latency
Q1: Balance	612	230	736	181	3107	182	3340	144
Q2: Deposit Checking	570	253	794	195	3033	216	4179	161
Q3: Transaction Saving	589	249	817	208	2945	207	4247	152
Q4: Amalgamate	547	305	541	282	2812	274	2794	249
Q5: Write Check	549	329	766	284	2989	301	4119	271

图15：基于Smallbank数据集的NoSQL密态数据库端到端性能

保护点1 – 关联事务融合协议

- **传统方案特征**
  - ■ 实际事务执行数量始终等于客户事务请求数量
  - ■ 同类事务不论可融合性均具有一致的吞吐和平均时延
- **本方案外显特征与侵权取证方法**
  - ➢ **外显特征1：第三方系统实际事务执行数量小于客户事务请求数量**
    - 构造并提交一组可融合事务执行请求
    - 使用数据库管理工具(例如MySQL Enterprise Monitor)监控实际事务执行数量是否小于客户请求数量
  - ➢ **外显特征2：第三方系统可融合事务比不可融合事务具有更高吞吐和更低平均时延**
    - 分别构造并提交一组可融合事务与一组不可融合事务执行请求，这两组事务复用相同的事务逻辑
    - 计算两组事务的端到端吞吐和平均时延，判断可融合事务是否具有显著更高吞吐和更低平均时延

保护点2 - GPU调度加速全同态事务协议

- GPU 加速全同态事务的现有方法
  - 方法一：GPU 并行加速全同态算子内部运算，但同一事务不同算子仍然串行执行
  - 方法二：GPU 无序并行执行多个全同态事务，但由于并发度提高导致事务冲突回滚增加，平均时延不降反增
- 观察
  - 在各类数据库应用中，普遍存在事务内部语句无数据依赖关系，这类语句不必串行执行。例如 TPC-C 的 Delivery 事务分别对十个 district 执行无数据依赖的查询与更新语句。
  - 在各类数据库应用中，普遍存在多个事务具有数据依赖关系，并发执行这类事务将导致事务回滚与重新执行。例如 SmallBank 的 Amalgamate 事务，如果并发执行两个事务请求，参数分别为 (Alice, Bob) 和 (Bob, John)，则后提交的事务将发生回滚并重新执行。

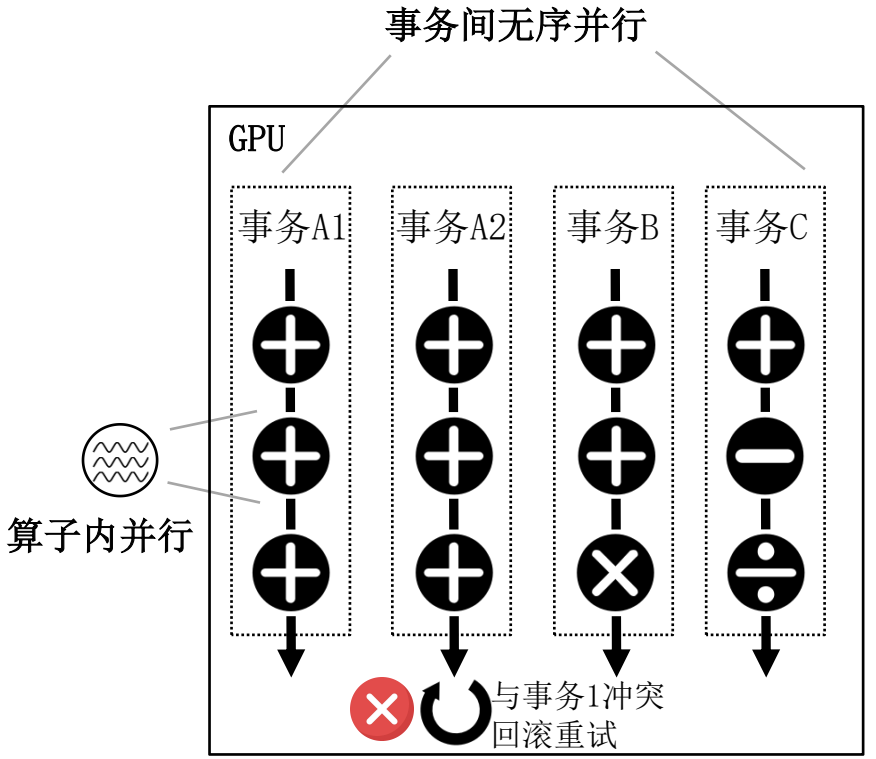


图16： GPU加速全同态事务现有方法示意图

保护点2 - GPU调度加速全同态事务协议

- **本专利解决方案：使用GPU加速和事务调度器从三个层级加速全同态事务执行**
  1. 算子内并行：GPU多线程并行执行单个全同态算子内部运算
  2. 事务内并行：GPU多线程并行执行单个事务内部多个无数据依赖关系的语句
  3. 事务间有序并行：使用事务调度器分析所有事务的数据依赖关系并记录在有向无环图(DAG)中
     1. 将存在数据依赖关系的事务调度到相同GPU计算流中串行执行
     2. 将没有数据依赖关系的事务调度到不同GPU计算流中并行执行。
- **本保护点与现有方法的差异**
  - 引入事务内并行，提高并行度，降低平均时延
  - 通过数据依赖分析和调度，实现事务间有序并行，提高吞吐和资源有效利用率

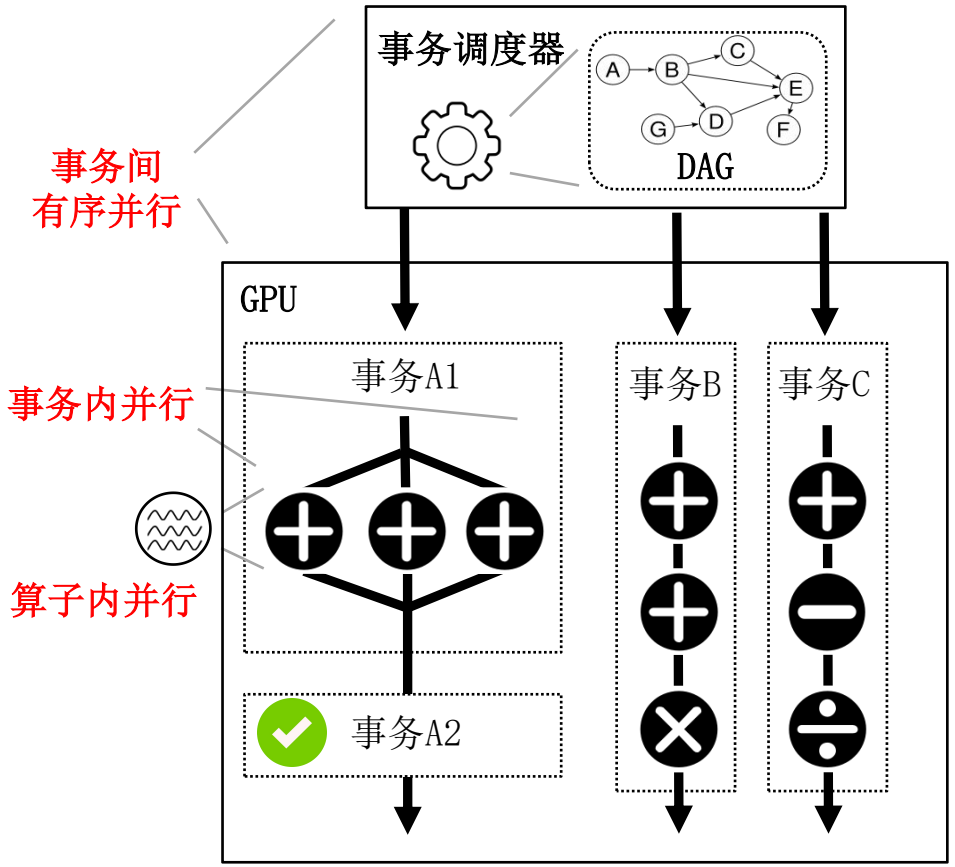


图17： GPU调度加速全同态事务协议

保护点2 – GPU调度加速全同态事务协议

- 有益效果
  - 图18展示使用TPC-C测试SQL数据库的端到端吞吐，其中item表含10^6行记录。图19展示使用SmallBank测试NoSQL数据库的端到端性能，其中account表含10^6行记录。**随机生成所有事务请求**且设置**25%事务存在数据依赖关系**。GACM-G是保护点2的端到端性能结果
  - 实验表明，保护点1使得本发明吞吐量超越最新系统3-10.7倍，平均时延下降最高8.9%
  - 更全面的数据参考有益效果 – 本发明实现更高吞吐量和更低平均时延

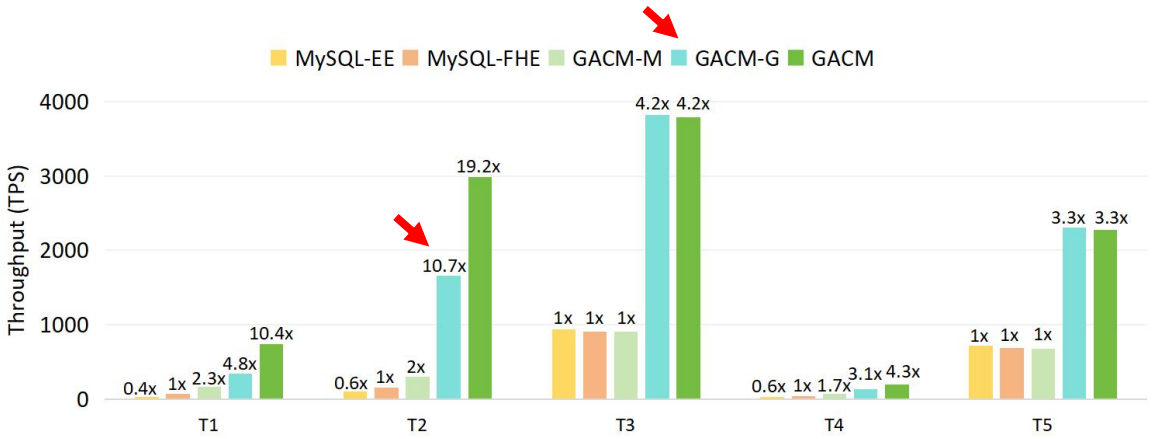


图18：基于TPC-C的SQL密态数据库吞吐量对比

	Redis-FHE		GACM-M		GACM-G		GACM	
SmallBank	tput	latency	tput	latency	tput	latency	tput	latency
Q1: Balance	612	230	736	181	3107	182	3340	144
Q2: Deposit Checking	570	253	794	195	3033	216	4179	161
Q3: Transaction Saving	589	249	817	208	2945	207	4247	152
Q4: Amalgamate	547	305	541	282	2812	274	2794	249
Q5: Write Check	549	329	766	284	2989	301	4119	271

图19：基于Smallbank的NoSQL密态数据库端到端性能

保护点2 - GPU调度加速全同态事务协议

- **传统方案特征**
  - 无论事务之间是否存在数据依赖关系，总使用多个SM并行执行多个事务
  - 执行存在数据依赖关系的多个事务时，发生多次回滚与重试
- **本方案外显特征与侵权取证方法**
  - **外显特征1：第三方系统使用GPU加速，仅使用单个SM(流多处理器)执行具有数据依赖关系的多个事务时，但使用多个SM执行无数据依赖关系的多个事务**
    - 分别提交一组存在数据依赖关系的事务和不存在数据依赖关系的事务，这两组事务复用相同的事务逻辑
    - 使用nvidia-smi等工具监控事务执行期间，存在数据依赖关系的多个事务是否只占用一个SM，无数据依赖关系的多个事务是否占用多个SM
  - **外显特征2：第三方系统对于具有数据依赖关系的多个事务采用串行执行且不发生回滚**
    - 分别提交一组存在数据依赖关系的事务和不存在数据依赖关系的事务，这两组事务复用相同的事务逻辑
    - 使用数据库管理工具(例如MySQL Enterprise Monitor)监控事务执行顺序和事务重试信息，判断存在数据依赖关系的事务是否采用串行执行，无数据依赖关系的事务是否采用并行执行

本发明技术方案实施案例（对传统异构非密态数据库进行加密与加速）

- 实施例：使用本专利加速传统异构非密态云端数据库(例如MySQL和Redis)运行关键数据加密的银行转账业务
  - 银行账户数据库可以简化为三元组(账户ID，账户余额，最近操作时间)，其中账户余额是关键数据，使用全同态加密进行保护；其余数据例如账户ID与最近操作时间是非关键数据，使用明文存储或确定性加密。
  - 客户执行本专利编译完成的客户端程序，生成或获取用户密钥对，根据银行账户数据库的多元加密策略对银行转账事务传入参数进行加密，随后将全同态银行转账事务执行请求发送至中间件程序。
  - 中间件程序在云端部署，负责对银行转账关联事务的融合与使用GPU加速事务执行。
  - 中间件程序通过数据库驱动与异构云端数据库(例如MySQL和Redis)进行数据查询、插入、更新与删除等操作。

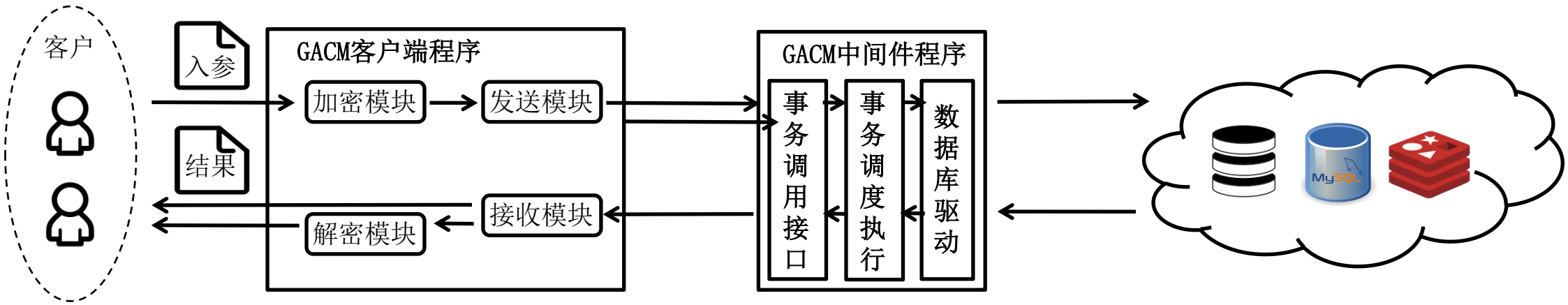


图20：本发明技术方案实施案例

本发明防侵权检测方法

- **防侵权核心方法**
  - 多次构造事务并采样事务执行端到端吞吐和时延指标，获取外显特征（保护点1,2）
  - 监控第三方系统硬件资源占用信息（保护点2）

- **本发明具有以下外显特征：**
  - 外显特征1：第三方系统实际事务执行数量小于客户事务请求数量 → 保护点1
  - 外显特征2：第三方系统可融合事务比不可融合事务具有更高吞吐和更低平均时延 → 保护点1
  - 外显特征3：第三方系统使用GPU加速，使用单个SM(流多处理器)执行具有数据依赖关系的多个事务时，使用多个SM执行无数据依赖关系的多个事务 → 保护点2
  - 外显特征4：第三方系统对于具有数据依赖关系的多个事务采用串行执行且不发生回滚 → 保护点2

- **若第三方系统声称满足本发明所有的发明目标，且同时满足上述所有外显特征，则该第三方系统极有可能侵权**

本发明防侵权检测方法

- **外显特征1：第三方系统实际事务执行数量小于客户事务请求数量** → 保护点1
  - 构造并提交一组可融合事务执行请求
  - 使用数据库管理工具(例如MySQL Enterprise Monitor)监控实际事务执行数量是否小于客户请求数量
- **外显特征2：第三方系统可融合事务比不可融合事务具有更高吞吐和更低平均时延** → 保护点1
  - 分别构造并提交一组可融合事务与一组不可融合事务执行请求，这两组事务复用相同的事务逻辑
  - 计算两组事务的端到端吞吐和平均时延，判断可融合事务是否具有显著更高吞吐和更低平均时延
- **外显特征3：第三方系统使用GPU加速，使用单个SM(流多处理器)执行具有数据依赖关系的多个事务时，使用多个SM执行无数据依赖关系的多个事务** → 保护点2
  - 分别提交一组存在数据依赖关系的事务和不存在数据依赖关系的事务，这两组事务复用相同的事务逻辑
  - 使用nvidia-smi等工具监控事务执行期间，存在数据依赖关系的多个事务是否只占用一个SM，无数据依赖关系的多个事务是否占用多个SM
- **外显特征4：第三方系统对于具有数据依赖关系的多个事务采用串行执行且不发生回滚** → 保护点2
  - 分别提交一组存在数据依赖关系的事务和不存在数据依赖关系的事务，这两组事务复用相同的事务逻辑
  - 使用数据库管理工具(例如MySQL Enterprise Monitor)监控事务执行顺序和事务重试信息，判断存在数据依赖关系的事务是否采用串行执行，无数据依赖关系的事务是否采用并行执行

有益效果

- **原型系统**：GACM是本发明的原型系统，GACM-M只加入关联事务融合能力，GACM-G只加入GPU加速能力。
- **基线系统**：本发明对比了MySQL EE，MySQL FHE 和 Redis FHE
  - MySQL-EE (MySQL Enterprise Encryption)是一个广泛商用的SQL传统密态数据库，采用传统加密算法（例如 RSA）。
  - MySQL-FHE是一个基于MySQL和全同态加密的基线SQL数据库，不具备关联事务融合和GPU加速能力。
  - Redis-FHE是一个基于Redis和全同态加密的基线NoSQL数据库，不具备关联事务融合和GPU加速能力。
- **测试基准**：本发明使用TPC-C测试SQL数据库场景下各系统的性能（MySQL-EE，MySQL-FHE，GACM-M，GACM-G和GACM），使用Smallbank测试NoSQL数据库场景下各系统的性能（Redis-FHE，GACM-M，GACM-G和GACM）。
- **实验问题**
  - 高效性：对比现有系统，本发明是否取得更低平均时延与更高吞吐量？
  - 可伸缩性：本发明能否高效扩展到更大数据集？
- **天然满足的目标**
  - 数据机密性：在传输、存储和计算过程中，关键数据使用全同态加密，保障明文不泄露与不被篡改。
  - 异构支持：采用分层全同态事务执行中间件架构，通过提供数据库驱动即可适配异构云端数据库。

有益效果 – 本发明实现更高吞吐量和更低平均时延

- 图21和图23使用TPC-C对比了SQL数据库场景下各系统性能，其中item表的记录数为10^6。图22和图24使用SmallBank对比了NoSQL数据库场景下各系统性能，其中account表的记录数为10^6。所有事务请求为随机生成，且设置10%事务可融合和25%事务存在数据依赖关系。
- 可以看出：对比基线全同态数据库MySQL-FHE和Redis-FHE，本发明带来约3.3-19.2倍吞吐提升和最高39%平均时延降低
  - （保护点1）关联事务融合协议可带来约1.2-2.3倍吞吐提升和最高30.9%平均时延降低
  - （保护点2）GPU调度加速全同态事务协议可带来约3-10.7倍吞吐提升和最高8.9%平均时延降低

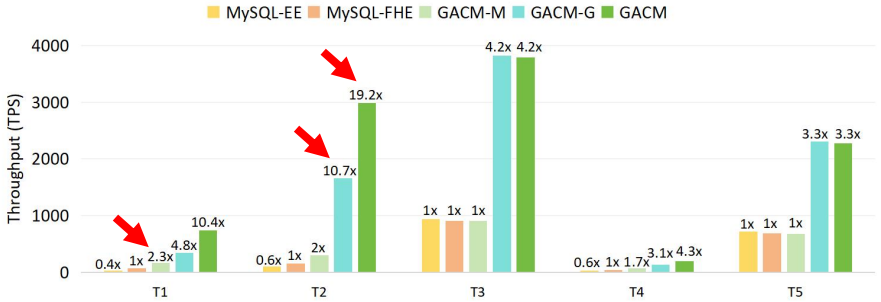


图21：基于TPC-C数据集的SQL密态数据库吞吐量对比

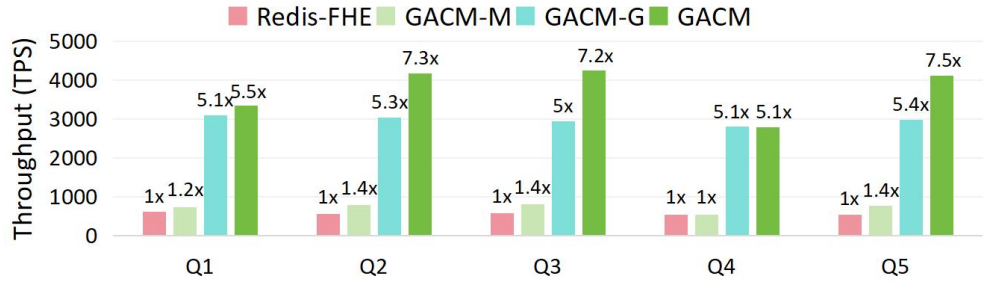


图22：基于Smallbank数据集的NoSQL密态数据库吞吐量对比

	MySQL-EE		MySQL-FHE		GACM-M		GACM-G		GACM	
TPC-C	tput	latency	tput	latency	tput	latency	tput	latency	tput	latency
T1: New Order	28	2130	71	960	165	886	342	933	737	855
T2: Payment	99	398	155	248	302	239	1663	255	2988	240
T3: Order Status	940	73	912	95	903	86	3817	99	3792	88
T4: Delivery	29	1887	45	1840	76	1574	138	1690	195	1361
T5: Stock Level	720	179	694	213	683	176	2304	219	2271	180

图23：基于TPC-C数据集的SQL密态数据库端到端性能

	Redis-FHE		GACM-M		GACM-G		GACM	
SmallBank	tput	latency	tput	latency	tput	latency	tput	latency
Q1: Balance	612	230	736	181	3107	182	3340	144
Q2: Deposit Checking	570	253	794	195	3033	216	4179	161
Q3: Transaction Saving	589	249	817	208	2945	207	4247	152
Q4: Amalgamate	547	305	541	282	2812	274	2794	249
Q5: Write Check	549	329	766	284	2989	301	4119	271

图24：基于Smallbank数据集的NoSQL密态数据库端到端性能

有益效果 – 本发明可高效伸缩

- 图25和图26分别展示了 GACM 在 TPC-C 和 SmallBank 测试基准不同数据量下的端到端吞吐量变化。
- 可以看出：
  - 本发明**支持高效横向伸缩**：GACM 可在**数据量**显著增加的场景下，维持事务执行吞吐量基本稳定。
  - 本发明**支持高效纵向伸缩**：GACM 可在**数据库表结构**显著复杂的场景下，维持事务执行吞吐量基本稳定。

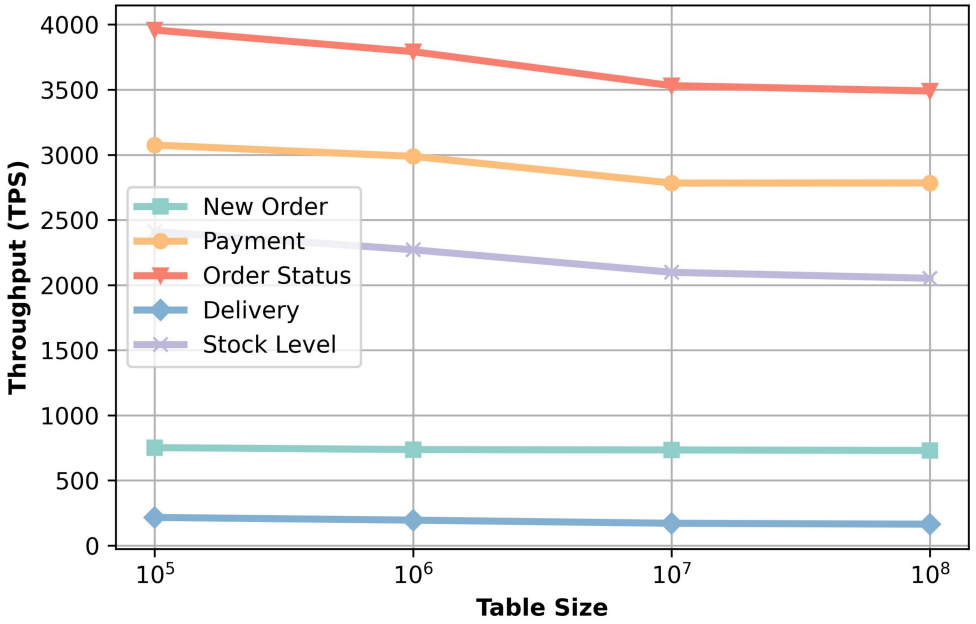


图25：GACM 在 TPC-C 不同数据量下的端到端吞吐量

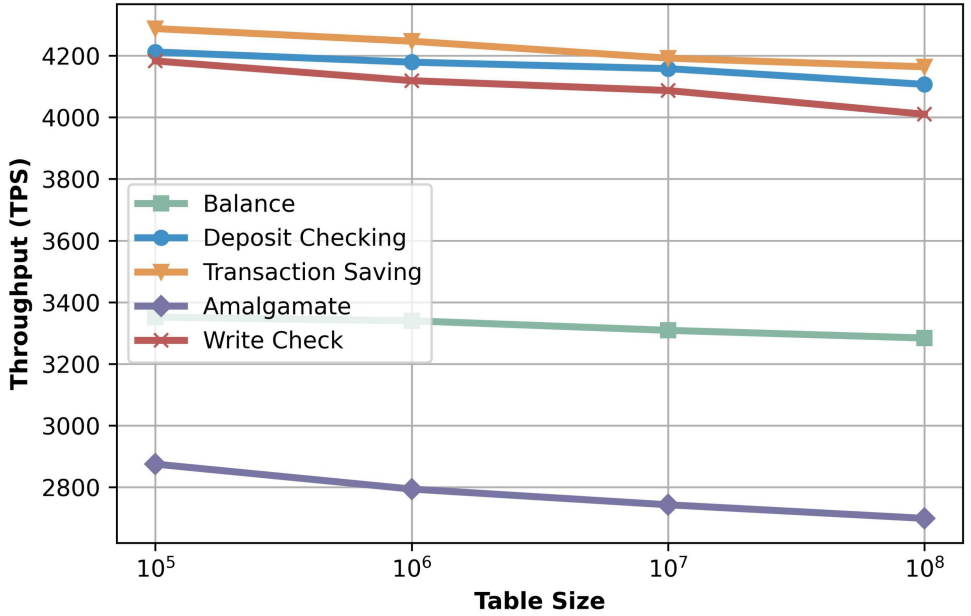


图26：GACM 在 SmallBank 不同数据量下的端到端吞吐量

结论：申请者认为本发明有重大专利价值

- 云端密态数据库目前广泛应用于金融、政务、电商等多个领域，蕴藏极高的**商业价值**和**落地潜力**。

- 现有系统**无法同时满足**事务执行**高效性**、**数据机密性**和**异构数据库支持**这三个核心目标，极大阻碍了来自不同行业的客户（例如政府或企业）采纳云端密态数据库。

- 本发明利用全同态加密数据库的**密文可算性**的特点，基于对全同态运算特征和事务数据依赖性的观察，通过使用GPU和**关联事务融合**加速数据库全同态事务执行，并引入**分层事务中间件**设计，实现了全同态数据库的**高效执行和异构支持**。

- 本发明有益于华为云、GaussDB、数据存储产品线等部门，**作为现有数据库的有力补充**，守护核心机密数据（例如技术方案实施案例）。

检索情况

国家/地区	检索网站	检索公式 (关键词及组合方式)	文献数量(指依据检索关键字直接获得的文献数，不需要列出具体的文献信息)
CN/US/EP	http://www.incopat.com （推荐使用。账号获取：直接通过主页“IP登录”进入） 使用教程及密码： http://3ms.huawei.com/hi/group/2033427/wiki_5014231.html		
其它网站	https://scholar.google.com（适用于使用**专利号**直接检索） 标准网站等其它网站，请自行增加。	GPU-accelerated fully homomorphic encryption database transaction Dependency-based scheduling for fully homomorphic encryption database transaction	262 21

Thank You

www.huawei.com

www.huawei.com

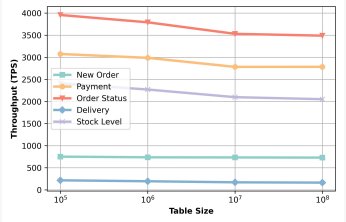
HUAWEI

一种基于GPU调度加速和关联事务融合的全同态数据库中间件

- **现有方案一：现有全同态数据库使用全同态加密（例如CKKS）**
- ☒ 缺点1：**密文运算性能极差**。全同态算术运算开销高昂，并且涉及全同态密文比较的数据查询需要进行**复杂度高达O(MN)的全表扫描**
- ☒ 缺点2：**不支持异构数据库**。系统设计**无法迁移到不同数据库架构**

- **现有方案二：传统密态数据库使用传统加密算法（例如RSA）**
- ☒ 缺点1：**性能低下**。密文事务执行牵涉**先解密、后加密**的繁琐流程
- ☒ 缺点2：**明文易泄**。密文事务执行期间会导致数据**明文暴露**。

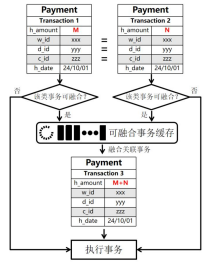
- **有益效果：分别使用TPC-C和Smallbank进行端到端性能测试**
- ☑ GACM带来端到端约**3-30倍吞吐量提升**和**最高60.9%平均时延降低**
  - ☑ 多元加密策略可带来**约1.5-3倍**吞吐量提升（贡献1）
  - ☑ 全同态数据库GPU二重加速协议可带来**约5倍**的吞吐量提升（贡献2）
  - ☑ 关联事务融合协议可带来**约2倍**吞吐量提升（贡献3）
- ☑ GACM具备**横向(数据量)**和**纵向(数据库表结构)**的**高可伸缩性**（贡献1）



本发明解决方案

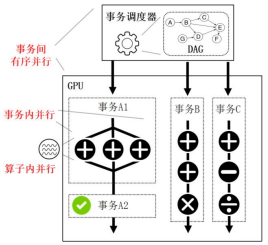
- **贡献1：关联事务融合协议**

实现目标：通过在静态编码阶段**声明事务融合元信息**和在动态执行阶段**融合关联事务**，**显著降低全同态事务实际执行数量**，提高端到端吞吐和显著降低时延。



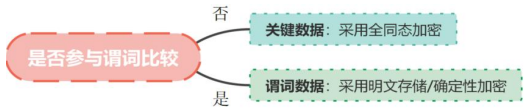
- **贡献2：GPU调度加速全同态事务协议**

实现目标：使用GPU和事务调度器实现三个层级并行加速（**算子内并行**，**事务内并行**，**事务间有序并行**），**显著提高吞吐量和降低时延**。



- **多元加密策略**

实现目标：使用**全同态加密关键数据**与使用**明文/确定性加密谓词数据**，避免全同态比较，兼顾**查询高效性**与**数据机密性**



- **分层全同态事务执行中间件**

实现目标：通过将GACM设计为三层架构（即统一的**事务调用接口层**和**事务调度执行层**，与针对于特定数据库的**数据库驱动层**实现了**对异构数据库引擎的适配**