# 第 17 讲:时间戳排序并发控制

**15-445/645 数据库系统(2022 年秋季)**

https://15445.courses.cs.cmu.edu/fall2022/

卡内基梅隆大学安迪·帕夫洛

## 1 时间戳排序并发控制

时间戳排序(T/O)是一类乐观的并发控制协议，其中 DBMS 假定事务冲突很少。DBMS 不要求事务在被允许读/写数据库对象之前获得锁，而是使用时间戳来确定事务的可序列化顺序。

每个事务 Ti 被分配一个唯一的固定时间戳 TS(Ti)，该时间戳是单调递增的。不同的方案在交易过程中的不同时间分配时间戳。有些高级方案甚至会为每笔交易分配多个时间戳。

如果 TS(Ti)< TS(Tj)，那么 DBMS 必须确保执行调度与 Ti 出现在 Tj 之前的串行调度是等价的。

有多种时间戳分配的实现策略。DBMS 可以使用系统时钟作为时间戳，但是在夏令时等极端情况下会出现问题。另一种选择是使用逻辑计数器。但是，这有溢出和跨多台机器的分布式系统维护计数器的问题。也有混合方法使用这两种方法的组合。

## 基本时间戳排序(Basic T/O)

基本时间戳排序协议(Basic T/O)允许在不使用锁的情况下对数据库对象进行读写。相反，每个数据库对象 X 都被标记为成功地对该对象执行读(表示为 R-TS(X))或写(表示为 W-TS(X))的最后事务的时间戳。然后，DBMS 为每个操作检查这些时间戳。如果一个事务试图以一种违反时间戳顺序的方式访问一个对象，那么该事务将被中止并重新启动。潜在的假设是，违规将是罕见的，因此这些重启也将是罕见的。

### 读操作

对于读操作，如果 TS(Ti)< W-TS(X)，这违反了 Ti 相对于 X 的前写入者的时间戳顺序(不想读取“未来”写入的内容)。因此，Ti 被中止，并使用新的时间戳重新启动。否则，读取是有效的，并且允许 Ti 读取 X。然后 DBMS 将 R-TS(X)更新为 R-TS(X)和 TS(Ti)的最大值。它还必须在私有工作区中创建 X 的本地副本，以确保 Ti 的可重复读取。

### 写操作

对于写操作，如果 TS(Ti)< R-TS(X)或 TS(Ti)< W-TS(X)，则 Ti 必须重新启动(不希望覆盖“未来”更改)。否则，DBMS 允许 Ti 写入 X 并更新 W-TS(X)。同样，它需要生成 X 的本地副本，以确保 Ti 的可重复读取。

**优化:托马斯写规则**

对写操作的优化是，如果 TS(Ti) < W-TS(X)， DBMS 可以忽略写操作并允许事务继续，而不是中止并重新启动它。这就是所谓的托马斯*写规则*。注意，这违反了 Ti 的时间戳顺序，但这没有关系，因为没有其他事务将读取 Ti 对对象 X 的写入。

如果基本 T/O 协议不使用托马斯写规则，它会生成一个可冲突序列化的调度。它不能有死锁，因为没有事务在等待。然而，如果短事务持续产生冲突，则存在长事务“饥饿”的可能性。

它还允许不可恢复的调度。如果事务只在它们读取的所有事务的更改提交之后提交，那么调度是*可恢复*的。否则，DBMS 不能保证事务读取将在从崩溃中恢复后恢复的数据。

**潜在的问题:**

- 从复制数据到事务工作区和更新时间戳的高开销。
- 长时间运行的事务可能会饿死。交易从新交易中读取信息的可能性增加。
- 在高并发系统上受到时间戳分配瓶颈的影响。

## 3 乐观并发控制 (OCC)

乐观并发控制(OCC)是另一种乐观并发控制协议，它也使用时间戳来验证事务。OCC 在冲突数量较低时效果最好。这是当所有的事务都是只读的，或者当事务访问不相交的数据子集时。如果数据库很大，并且工作负载没有倾斜，那么冲突的可能性就很低，因此 OCC 是一个很好的选择。

在 OCC 中，DBMS 为每个事务创建一个*私有工作区*。事务的所有修改都应用到这个工作区中。任何读取的对象被复制到工作区中，任何写入的对象也被复制到工作区中并在那里进行修改。其他事务不能读取另一个事务在其私有工作区中所做的更改。

当事务提交时，DBMS 比较事务的工作空间*写集*，以查看它是否与其他事务冲突。如果没有冲突，写集就被安装到“全局”数据库中。

OCC 包括三个阶段:

1. **读阶段:**在这里，DBMS 跟踪事务的读/写集，并将它们的写操作存储在一个私有工作区中。
2. **验证阶段:**当事务提交时，DBMS 检查它是否与其他事务冲突。
3. **写阶段:**如果验证成功，DBMS 将私有工作空间更改应用到数据库。否则，它中止并重新启动事务。

**验证阶段**

DBMS 在事务进入验证阶段时为其分配时间戳。为了确保只允许可序列化的调度，DBMS 检查 Ti 与其他事务的 RW 和 WW 冲突，并确保所有冲突都是单向的。

**方法 1:**向后验证(从年轻的事务到旧的事务)

**方法 2:**前向验证(从较早的交易到较早的交易)

这里我们描述了前向验证的工作原理。DBMS 检查提交事务与所有其他正在运行的事务的时间戳顺序。尚未进入验证阶段的事务被分配一个∞时间戳。

若 TS(Ti)< TS(Tj)，则必须满足以下三个条件之一:

1.在 Tj 开始执行之前，Ti 完成了所有三个阶段(串行排序)。

2.Ti 在 Tj 开始写阶段之前完成，并且 Ti 不写入 Tj 读取的任何对象。

- WriteSet(Ti)∩ReadSet(Tj) =∅。

3.Ti 在 Tj 完成其 Read 阶段之前完成其 Read 阶段，并且 Ti 不会对 Tj 正在读取或写入的任何对象进行写入。

- WriteSet(Ti)∩ReadSet(Tj) =∅，WriteSet(Ti)∩WriteSet(Tj) =∅。

**潜在的问题:**

- 将本地数据复制到交易的私有工作区的高开销。
- 验证/写入阶段的瓶颈。
- 与其他协议相比，终止可能更浪费，因为它们只发生在事务已经执行之后。
- 遭受时间戳分配瓶颈。

## 4 个隔离级别

可串行化是有用的，因为它允许程序员忽略并发问题，但强制执行它可能会允许太少的并行性并限制性能。我们可能希望使用较弱的一致性水平来提高可伸缩性。

隔离级别控制一个事务暴露给其他并发事务操作的程度。

**异常现象:**

**脏读:读取**未提交的数据。

不可重复**读取(Unrepeatable read):**重复读取会得到不同的结果。

**幻影读:**对于相同的范围扫描查询，插入或删除会导致不同的结果。

**隔离级别(最强到最弱):**

1. SERIALIZABLE:没有幻影，所有读取都是可重复的，没有脏读取。
2. 可重复读取:可能会出现幻影。
3. READ-COMMITTED:可能会发生幻影和不可重复的读取。
4. 未提交读:所有异常都可能发生。

作为 SQL-92 标准一部分定义的隔离级别只关注可能在基于 2pl 的 DBMS 中发生的异常。还有两个额外的隔离级别:

1. 游标稳定性
   - 在可重复读取和读取提交之间
   - 防止丢失更新异常。
   - **IBM DB2** 中的默认隔离级别。
2. 快照隔离
   - 保证在事务中进行的所有读取都看到事务开始时存在的数据库的一致快照。
   - 只有当事务的写入不与快照之后的并发更新冲突时，事务才会提交。

   易受写偏斜异常影响。