Assignment 5 Phase 2 Report

110062229 翁語辰

110081014 程詩柔

110062171 陳彥成

Implementation比較

1.ConservativeLockTable.java

TA:

- 變數宣告
  - 除了conservative lock會用的S_Lock和X_LOCK以外，還額外保留了IS_LOCK、IX_LOCK、SIX_LOCK的implementation
  - sLockers用的是linklist
- 取得S_LOCK的實作:
  - 助教用hasSLock()判斷該txn是否已經拿到sLock了
  - 呼叫sLockable()，來判斷該txn是否能拿lock
- 取得X_LOCK實作上同理
- 釋放Lock實作
  - 由Mgr呼叫releaseLocks()將所有txn中的readset、writeset object傳入LockTable release()中判斷該object是否拿著lock
  - 拿到Lock的object會被傳入releaseLock()中釋放拿到的Lock並通知其他尚在等待的object可以拿Lock

我們

- 變數宣告
  - 實作getSLock和getXLock
  - sLockers用的是Set
- 取得S_LOCK的實作:
  - 直接呼叫lockers.slockers.contains()判斷該txn是否已經拿到sLock了
  - 將sLockable()的判斷條件直接寫在迴圈判斷條件中，節省呼叫多個function的時間
- 取得X_LOCK實作上同理
- 釋放Lock實作
  - Mgr將bookKeys(存所有writeSet、readSet的primarykey)直接傳進LockTable的releaseAll()中用迴圈判斷所有object中哪些拿著Lock
  - 拿到Lock的object會被傳入releaseLock()中釋放拿到的Lock並通知其他尚在等待的object可以拿Lock

整體比較:

由於我們只使用了getXLock和getSLock來實作conservative Lock，所有code相比助教得來的精簡。且條件判斷是直接寫迴圈條件中，並沒有額外寫function呼叫做判斷，可以節省呼叫多個functioln判斷的時間。

2.ConservativeConcurrencyMgr.java

TA:

- bookReadKeys()、bookWriteKey()將Object放進requestQueue中等待，若尚未在LockerMap中，會生成Locker與object一併放進LockerMap中。
- 多實作了 crabbing locking，讓一些不需要用的lock提早release

我們:

- 與助教作法相似，但我們是使用bookPrimaryKeys()，將所有writeSet和readSet中Object的Locker一次初始化完

比較:

- 助教的作法效能上會比我們提升蠻多，因為 crabbing locking可以讓lock提前被release而不是等到commit或rollback完，所以txn concurrency會提升。而我們的寫法則是必須等到commited或rollback才會release lock，transaction concurrency會比較低。

3.PrimaryKey.java

TA:

- 實作genHashCode產生hashcode並由 hashCode() return產生的hashcode
- implement toString()

我們:

- 直接由hashCode()產生並return hashcode

比較:

- 整體實作上大致相同

4.StoredProcedure.java

TA:

- 額外宣告Transaction scheduleTransactionSerially function並在prepare(Object... pars)中呼叫

我們:

- 我們是直接寫在prepare(Object... pars)中，並沒有額外宣告function來create txn，而是在create完txn後去呼叫bookPrimaryKeys

比較:

- 整體實作上大致相同

benchmark結果比較

Prameter : RW_TX_RATE=0.8

TA:

20240515-204530-microbench.txt

```
# of txns (including aborted) during benchmark period: 171800
MICRO_TXN - committed: 171800, aborted: 0, avg latency: 0 ms
TOTAL - committed: 171800, aborted: 0, avg latency: 1 ms
```

我們:

20240508-214945-microbench.txt

```
# of txns (including aborted) during benchmark period: 29151
MICRO_TXN - committed: 29151, aborted: 0, avg latency: 4 ms
TOTAL - committed: 29151, aborted: 0, avg latency: 4 ms
```

TA的commit數比我們多十幾萬，latancy也比我們少，推測是因為實作了crabbing locking的關係，讓lock可以提前被release因此txn wait的時間減短，commtted數自然也就增加了。