Serializability with Snapshot Isolation under the Hood

Mihaela Bornea [1], S. Elnikety [2], O. Hodson [2], A Fekete [3]

[1]IBM Research [2]Microsoft Research [3]University of Sydney

Motivation

Concurrency Control

Replication Model

Readset Certification

Evaluation

Conclusions

Transaction Processing in Replicated Databases

- Database Replication:
  - Higher availability & better performance
  - Maintaining consistency is challenging
- State of the Art:
  - GSI Replicated Databases.
  - Each replica uses Snapshot Isolation (SI).
- Goal:
  - Global One Copy Serializability.
  - Overall Isolation level stronger than the one of individual components.
  - The replicated system keeps its performance.

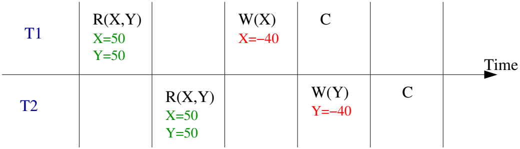
Transaction Isolation

- Isolation is a correctness criterion.
- Concurency in the system.
- Multiple levels of isolation:
    - Snapshot Isolation.
    - Serializability.

Snapshot Isolation

- Multi-version concurency control technique.
- Important
  - Used by Oracle, SQL Server, Postgres.
  - Sometimes the strongest isolation level available.
- Attractive performance
  - Read-only transactions never block or abort.
  - Read-only transactions do not block update transactions.
  - Updates might abort. Certification needed.
    - checks for ww conflicts.

Anomaly under SI



T1	R(X,Y) X=50 Y=50		W(X) X=−40	C		
T2		R(X,Y) X=50 Y=50		W(Y) Y=−40	C	

- X,Y balance of two bank accounts.
- T_1 and T_2 withdraw 90E from X and Y
- Logic: $X + Y > 0$

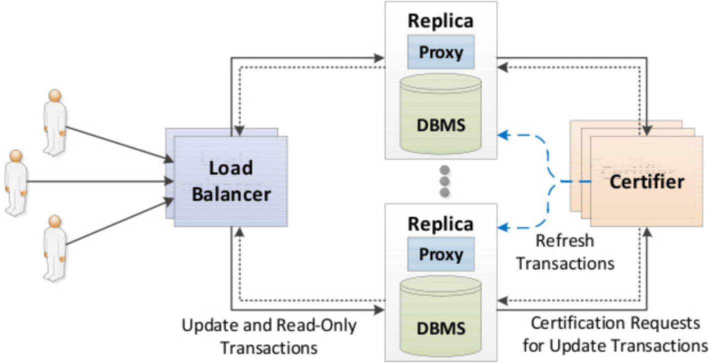
Serializability

- The strongest DB isolation level.
- Illusion that transactions execute serially.
- Programmers want it:
  - As if there is no concurrency.
- Commonly implemented with 2PL.
  - expensive to achieve.

Serializability under SI

- Centralized Database
  - Modify database engine, SSI.
  - Use Fekete's work [SIGMOD 2008, best paper]
- Replicated Databases
  - Open question.
  - No modification of the database engine.

GSI Replicated Database



SQL Transaction Model

A. SELECT $expr\_list$ FROM R_i WHERE $pred(R_i)$

B. INSERT INTO R_i VALUES $(values)$

C. UPDATE R_i SET $attr\_values$ WHERE $pred(R_i)$

D. DELETE FROM R_i WHERE $pred(R_i)$

E. SELECT $agg(attr)$ FROM R_i WHERE $pred(R_i)$
GROUP BY $group\_attr$
HAVING $pred(agg(attr))$

F. SELECT $attr\_list$
FROM $R_1 \ldots R_i \ldots R_n$
WHERE $pred(R_1)$ LOP ...LOP $pred(R_i)$ LOP ... LOP $pred(R_n)$
LOP $pred(attr_{i,j}, attr_{i,j})$

G. SELECT $attr\_list$
FROM $R_1 \ldots R_i \ldots R_n, SQ$
WHERE $pred(R_1)$ LOP ...LOP $pred(R_i)$ LOP ... LOP $pred(R_n)$
LOP $pred(SQ)$

H. SELECT $attr\_list$
FROM $R_1 \ldots R_i \ldots R_n$
WHERE $pred(R_1)$ LOP ...LOP $pred(R_i)$ LOP ... LOP $pred(R_n)$
LOP $pred(attr_i, SQ)$

1SR Needs Readsets

- Snapshot Isolation (SI) → Generalized Snapshot Isolation (GSI)
  - Certify Writeset
- Serializability → One Copy Serializability (1SR)
  - Certify Writeset
  - Certify Readset
- Yes, we have a proof :) !

Writesets

- The Writeset contains modified tuples
- Introduced by UPDATE, INSERT and DELETE
- Includes both new and old tuple values
- All Writesets are managed at the Certifier.
- Writeset certification is required by both GSI and 1SR
  - checks if concurrent transactions modify the same item.
- It is well knows how to manage the Writesets

Readsets

- The Readset contains read tuples.
- Introduced by SELECT, UPDATE, INSERT and DELETE.
- Readsets certification is required by 1SR.
  - checks if a transaction reads data modified by concurrent transactions.
- Readset identification is challenging:
  - never done in replicated setting.

So far ...

- We introduced SI.
- Sometimes SI is not enough !
- Serializability needed:
    - Keep the nice properties of SI.
    - Open Problem for replicated databases:
        - Readset management is difficult!

Main Contribution - Readset Management

- Framework to manage the Readsets
- Observation: each SQL statement has a predicate.
  - The Readset is a list of predicates.
  - Readset certification requires predicate evaluation.

Certifier Design

- The Certifier manages:
    - persistent log.
    - main memory database, CertDB.
- The log is used for durability.
- CertDB is used to certify update transactions.
- CertDB maintains the Writeset of recently committed transactions.
- CertDB schema:
    - the replicated schema.
    - commit version attribute.

Readset Certification

- Intuition:
    - Ensures that if the transaction executes on the latest version it would read the same values.
- Implementation:
    - Replica identifies the Readset:
        - Extracts the predicate of each SQL statement.
    - Replica expresses the readset as certification queries.
    - The certification queries are evaluated on CertDB
    - Empty conflict set indicates serializable execution

Concurrent Transactions

- Snapshot versions at originating replicas.
- Commit version of a transaction.
- CertDB contains the writesets and committed version.
- Consider a transaction T:
  - $version > snapshot(T)$

Readset for SELECT Statements

Transaction Queries

A. SELECT $\mathit{expr\_list}$ FROM R_i WHERE $\mathit{pred}(R_i)$

Certification Queries

A. SELECT $\star$ FROM R_i WHERE $\mathit{pred}(R_i)$ AND
$\mathit{version} > \mathit{snapshot}(T)$

Readset for UPDATE Statements

Transaction Queries

B. INSERT INTO R_i VALUES (*values*)
C. UPDATE R_i SET *attr_values* WHERE $pred(R_i)$
D. DELETE FROM R_i WHERE $pred(R_i)$

Certification Queries

B. SELECT $\star$ FROM R_i WHERE $pk = @pk$ AND $version > snapshot(T)$
C. SELECT $\star$ FROM R_i WHERE $pred(R_i)$ AND $version > snapshot(T)$
D. SELECT $\star$ FROM R_i WHERE $pred(R_i)$ AND $version > snapshot(T)$

Certifying the Readset also detects ww conflicts.

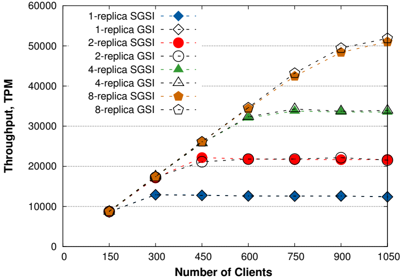
Experimental Study

- Impact of providing 1SR vs. GSI:
  - Lower throughput and higher response time
  - Higher abort rate
- Replicated system with 8 replicas
- TPC-W

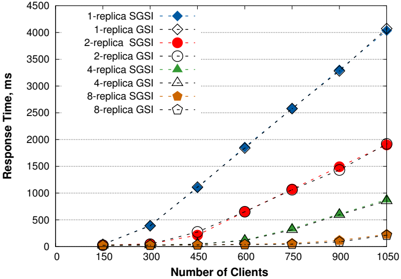
Workload

- TPC-W benchmark:
  - Web application (online book store).
  - Database schema consists of 10 tables.
  - Database size: 800 MB.
  - 13 transaction templates.
  - Ordering Mix(50% updates).
  - Browsing Mix (5% updates).
- Metrics:
  - Transactions per minute (TPM).
  - Response time.
  - Abort rate.

Scaling of SGSI with Replication Degree

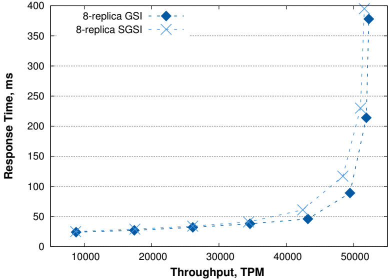


Throughput of TPC-W Shopping Mix (20% updates)

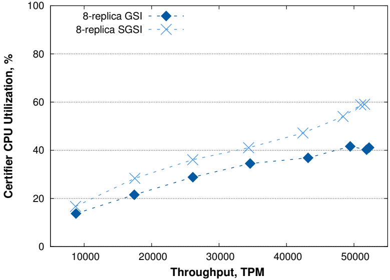


Resp. Time of TPC-W Shopping Mix (20% updates)

Comparing SGSI to GSI

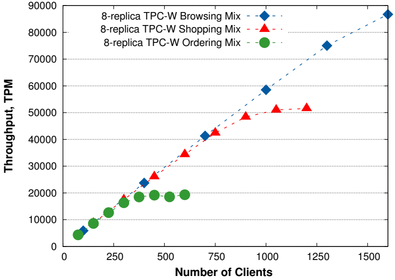


Scalability of TPC-W Shopping Mix (20% updates)

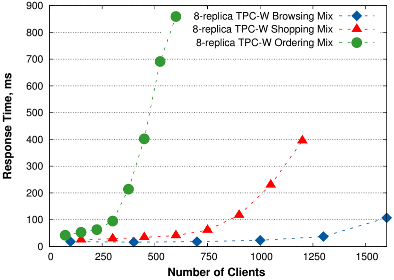


Certifier CPU Utilization TPC-W Shopping Mix (20% updates)

Sensitivity to Update Transaction Ratio

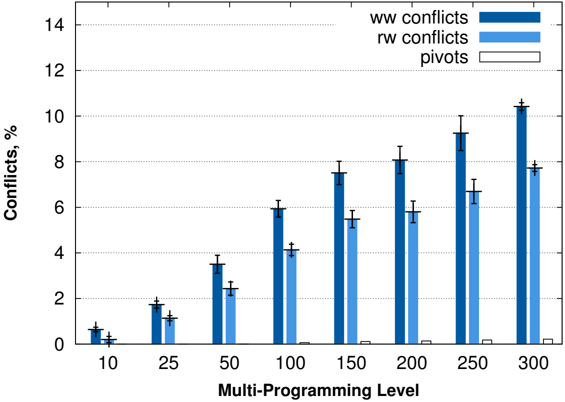


SGSI Throughput of TPC-W Mixes.



SGSI Response Time of TPC-W Mixes.

Abort Analysis via SmallBank



Conclusions

- We introduced SGSI:
  - 1SR in replicated databases.
- Built a replicated system prototype.
- Evaluated SGSI performance:
  - SGSI is practical.
  - Moderated cost for small degree of replication.
  - Performance and scaling is comparable with GSI.

Readset for Joins

Transaction Queries

F. SELECT *attr_list*
FROM $R_1 \ldots R_i \ldots R_n$
WHERE $pred(R_1)$ LOP ...LOP $pred(R_i)$ LOP ... LOP $pred(R_n)$
LOP $pred(attr_{i,j}, attr_{i,j})$

Certification Queries

for each relation R_i
F. SELECT $\star$ FROM R_i WHERE $version > snapshot(T)$

- An upper-set of the Readset is certified.
- False aborts.

Data Managed at Certifier

- Accuracy depends the data maintained at the Certifier.
- False aborts:
  - not enough information to evaluate the Readset
- Solution:
  - manage a copy of relations at the Certifier.
  - physical design tuning problem.

Extended CertDB

- Each data item has several instances.
- New instace: UPDATE,INSERT.
- Expired: UPDATE, DELETE.
- Each copy relation is augmented with V_{Start} and V_{End}.
- V_{Start} and V_{End} determine:
  - update predicate: $upd(R_i)$.
  - visibility predicate: $vis(R_i)$.

Extended Certification

Transaction Queries

F. SELECT $attr\_list$
FROM $R_1 \ldots R_i \ldots R_n$
WHERE $pred(R_1)$ LOP ...LOP $pred(R_i)$ LOP ... LOP $pred(R_n)$
LOP $pred(attr_{i,j}, attr_{i,j})$

Certification Queries

SELECT * FROM $R_{1_C} \ldots R_{i_C} \ldots R_{n_C}$
WHERE $(query\_pred)$
AND $(upd(R_{1_C})$ ...OR $upd(R_{i_C})$ ... OR $upd(R_{n_C}))$
AND $(vis(R_{1_C})$ ...AND $vis(R_{i_C})$ ... AND $vis(R_{n_C}))$