Distributed Transaction Processing in the Escada Protocol

Alfrânio T. Correia Júnior

alfranio@lsd.di.uminho.pt.

Grupo de Sistemas Distribuídos
Departamento de Informática
Universidade do Minho
Scenario

- Dependable Databases are core components of our Information Society.
- Software based replication is an attractive solution to assure dependability.
- Problems arise when attempting to preserve the following:
  - Strong consistency (1SR).
  - Updates are carried through any replica.
Scenario (Problems)

- Traditional database replication protocols do not scale up well due to:
  - The high number of messages exchanged among the replicas.
  - The deadlock rate proportional to $n^3$, where $n$ is the number of replicas, which is impractical.
Scenario (Earlier Solutions)

- Lazy Replication relaxes the consistency criteria.
- Master/Slave chooses the replica that can receive the updates.
Escada Approach

- Pattern on the DBSM
- Replication based on group communication
- Update-everywhere and deferred updates
- Atomic broadcast to propagate the transaction’s processing data
- Total order is combined with a conflict detection process to assure 1SR.
Protocol Overview

01 - Local Execution
02 - Atomic Multicast (RS, WS, WV)
03 - Certification

Termination Protocol
Protocol Overview

T1 = { A, B, C }  
T2 = { C, D, E }  

A, B, C, D, E  
A, B, C, D, E  
A, B, C, D, E  

Distributed Transaction Processing in the Escada Protocol – p.7/33
Escada Approach

• Target:
  • Large scale distributed systems.
  • Provides partial replication (Partial DBSM)
  • Reduces resource consumption exploiting application’s locality.
How the Escada can be augmented to provide partial replication?
Main Contributions

- Distributed Transaction Processing
- Semantic Caching
- Extensions to the PostgreSQL
- Evaluation Process (TPC-C and TPC-W)
Contribution

Distributed Transaction Processing
Partial DBSM

- **Execution:**
  - A site that handles a transaction may not be able to locally complete its execution.
  - It is possible that no single site can do it.

- **Termination Protocol:**
  - The distributed execution fragments the knowledge about conflicts.
  - How to decide if a transaction can commit or not?
Execution

- Rewrites the queries mapping the original relations to the actual fragments.
- Sites are contacted according to the replicated fragments.
Termination Protocol

- The updated information is propagated to the pertinent replicas.
- What can we say about the read and write sets?
  - They could be sent to all the replicas allowing a deterministic certification (non-voting).
  - They could be sent just to the pertinent replicas requiring a voting certification.
Non-Voting

T1 = \{ A, B, C \}
T2 = \{ C, D, E \}

Distributed Transaction Processing in the Escada Protocol – p.15/33
Voting

T1 = \{ A, B, C \}

T2 = \{ C, D, E \}
Experiments

Latency

- DBSM - WAN
- PDBSM - WAN
- PDBSMRAC - WAN
- DDE - WAN
- Ideal Locking

Clients

Latency (ms)

0 50 100 150 200 250 300 350 400

250 500 750 1000

Distributed Transaction Processing in the Escada Protocol – p.17/33
Contribution

Semantic Caching
Semantic Caching

- Aims at reducing communication among distributed replicas.
- Exploits the broadcast to build the refresh mechanisms.
- Reduces the management overhead when compared to tuple and page-based solutions.
- Roughly, it caches the result sets and identifies them based on predicates.
Semantic Caching

- The satisfiability background is behind this approach.
- Two predicates $S$ and $F$ are satisfiable if $S \land F$ is not a contradiction.
- The main idea: "upon receiving a query is possible to use a previous request?"
Semantic Caching

- We use an important class of queries, called SPJ queries.
- However, other queries could be cached.
Experiments

Profits

![Graph showing profits with different numbers of clients (5, 10, 15, and 20) over time elapsed. The x-axis represents cache entries (x100), and the y-axis represents time elapsed (ms).]
Contribution

PostgreSQL
PostgreSQL

- We identified that the 1SR correctness criteria suggested by the DBSM is not achievable.
- We present an informal algorithm about how to exactly extract the read sets.
- Finally, we show how to extend the "rule mechanisms" of the PostgreSQL.
Contribution

Evaluation Process
Evaluation Process

- We use a simulation tool that allows the combination of real and simulated code.

- The replication protocols, our main goal, are real implementations.

- This approach permits us to focus on our goal while at the same allows:
  - Variations on the network architecture, the workload, concurrency control policies.
Evaluation Process

- TPC-W mimics an Internet commerce application.
- TPC-C mimics a wholesale supplier (OLTP).
Latency DBSM & Centralized

- DBSM - LAN
- 1 CPU
- 3 CPUs

Clients vs. Latency (ms)
Partial DBSM

Latency

Latency (ms)

Clients

DBSM - WAN
PDBSM - WAN
PDBSMRAC - WAN
DDE - WAN
Ideal Locking

Distributed Transaction Processing in the Escada Protocol – p.29/33
Partial DBSM

Network Bandwidth

Kbytes/s

Clients

DBSM - WAN
PDBSM - WAN
Conclusion

• Database replication based on group communication is a feasible solution.
• The non-voting solution scales up better.
• The semantic caching reduces communication.
• Partial replication reduces resource consumption.
Future Work

- Improve our simulation tool: user friendly.
- Develop a version of our semantic caching to mobile elements.
- Revisit the Epsilon Serializability to Improve DBSM performance.
The End

Unfortunately, I don’t know how to the things that I need to do. However, as a good start point I certainly know how I must not. Sometimes, this knowledge avoids worthless efforts. Sometimes, we simply need to go into the wrong direction to choose another path.