Salt
Combining ACID and BASE in a distributed database

Chao Xie, Chunzhi Su, Manos Kapritsos, Yang Wang, Navid Yaghmazadeh, Lorenzo Alvisi, Prince Mahajan

The University of Texas at Austin
TRANSACTIONS ARE GREAT

Four properties in a single abstraction

- Atomicity
- Consistency
- Isolation
- Durability (ACID)

- Ease of programming
- Easy to reason about
Concurrent control limits performance

2PC protocol is costly
THE ALTERNATIVE: BASE
THE ALTERNATIVE: BASE

- Write custom code to get better performance
THE ALTERNATIVE: BASE

- Write custom code to get better performance
- Complexity gets out of control

Application  Implement Consistency  Storage Interface

BASE Storage (e.g., put, get)
A STARK CHOICE

Ease of programming & Performance

Performance

Ease of programming

BASE

ACID
20% of the causes account for 80% of the effects

Vilfredo Pareto
NOT ALL TRANSACTIONS ARE CREATED EQUAL

• Many transactions are not run frequently
• Many transactions are lightweight

20% of the causes account for 80% of the effects
AN OPPORTUNITY

- Identify critical transactions
- BASE-ify only critical transactions
SALT

Motivation

Base Transactions & Salt Isolation

Achieving Salt Isolation

Evaluation
MORE CONCURRENCY!

Transfer

Is c ≥ $10?
c = c - $10
s = s + $10

Part 1

Is c ≥ $10?
c = c - $10

Part 2

s = s + $10
MORE CONCURRENCY!

Transfer

Is $c \geq 10$?

c = c - $10$

s = s + $10$

Transfer

Part 1

Is $c \geq 10$?

c = c - $10$

Transfer

Part 2

s = s + $10$

Transfer

Part 1

Is $c \geq 10$?

c = c - $10$

Transfer

Part 2

s = s + $10$
CORRECTNESS AT RISK

Balance
- Read c
- Read s

Transfer
- Is c ≥ $10?
  - c = c - $10
  - s = s + $10
CORRECTNESS AT RISK

Transfer

Part 1
Is $c \geq 10$?

- $c = c - 10$

Part 2

- $s = s + 10$

Exposed state

Balance
- Read $c$
- Read $s$

Read $c$
Read $s$
CORRECTNESS AT RISK

Part 1
Is $c \geq $10?  
c = c - $10

Balance
Read c
Read s

Part 2
s = s + $10

Finer Isolation for one transaction may affect all transactions!!
Performance vs Complexity

Better Performance → More Interleavings → More Complexity
Performance vs Complexity

Better Performance

More Interleavings
(only among perf-critical txns)

Other Transactions Unaffected
Behaves differently when interacting with different transactions
Balance

Read c

Read s

Time

Transfer 1

Is c ≥ $10?

c = c - $10

s = s + $10

Transfer 2

Is c ≥ $10?

c = c - $10

s = s + $10
BASE TRANSACTION

Behaves differently when interacting with different transactions
BASE INTERACT WITH BASE

Fine Isolation granularity between BASE transactions
BASE INTERACT WITH BASE

Fine Isolation granularity between BASE transactions
BASE transactions provide coarse Isolation granularity to ACID transactions
BASE INTERACT WITH ACID

BASE transactions provide coarse Isolation granularity to ACID transactions
SALT ISOLATION

BASE transactions: multiple granularities of Isolation

To BASE transactions: a sequence of small ACID transactions

To ACID transactions: a single, monolithic ACID transaction

Performance & Ease of Programming
SALT

Motivation

Base Transactions & Salt Isolation

Achieving Salt Isolation

Evaluation
ONE MECHANISM

LOCKS

Three flavors

ACID locks
Alkaline locks
Saline locks
ACID LOCKS

ACID 1
Write x

ACID 2
Read x

Execute

Lock Table

<table>
<thead>
<tr>
<th></th>
<th>AC-R</th>
<th>AC-W</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC-R</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>AC-W</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
Conflicting with ACID & alkaline locks

<table>
<thead>
<tr>
<th></th>
<th>AC-R</th>
<th>AC-W</th>
<th>alk-R</th>
<th>alk-W</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC-R</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>AC-W</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>alk-R</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>alk-W</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
SALINE LOCKS

Conflict only with ACID locks

<table>
<thead>
<tr>
<th></th>
<th>AC-R</th>
<th>AC-W</th>
<th>alk-R</th>
<th>alk-W</th>
<th>sal-R</th>
<th>sal-W</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC-R</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>AC-W</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>alk-R</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>alk-W</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>sal-R</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>sal-W</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
A SUBTLE PROBLEM

ACID reads uncommitted value of x!

Dirty Read!

ACID reads uncommitted value of x!
A subtle problem

For the solution, please read our paper
Guarantee

Salt prevents all ACID transactions from being affected by BASE transactions either directly or indirectly.
SALT

Motivation

Base Transactions & Salt Isolation

Achieving Salt Isolation

Evaluation
QUESTIONS TO ANSWER

What is the performance gain of Salt compared to ACID?

Can we get most performance gain compared to the BASE approach?
EXPERIMENTAL SETUP

Configuration

• Emulab Cluster (Dell Power Edge R710)
• 10 shards, 3-way replicated

Workloads

• TPC-C
• Fusion Ticket
• Microbenchmarks
PERFORMANCE GAIN

Fusion Ticket

Throughput (transactions/sec)

Latency (ms)

6.5X
REAP MOST PERFORMANCE OF BASE

Fusion Ticket

Throughput (transactions/sec)

Number of BASE-ified transactions

6.5X

7.2X

0

10000

8000

6000

4000

2000

0

ACID 1 2 3

Raw ops

… … …

Raw operations
RELATED WORK

Optimizing ACID Performance

- H-Store, Granola, F1, Sagas, Transaction Chain, Calvin ...

BASE with enhanced semantics (e.g., partition local transactions)

- ElasTras, G-Store, Megastore ...
Pain Point
Transactional systems do not scale

Key Abstraction
Base Transaction

Promising Results

SALT

Fusion Ticket

Throughput (transactions/sec)

Number of BASEified transactions

ACID 1 2 3 Raw ops

6.5X 7.2X