Having Your Cake and Eating it Too: Combining Strong and Eventual Consistency

Konrad Siek and Paweł T. Wojciechowski
Poznań University of Technology
{konrad.siek,pawel.t.wojciechowski}@cs.put.edu.pl

13 IV 2013

http://dsg.cs.put.poznan.pl
def thread:
    lock_a.acquire()
    lock_b.acquire()
    a = b
    lock_a.release()
    b = b + 1
    lock_b.release()
def thread:
    lock_a.acquire()
    lock_b.acquire()
    a = b
    lock_a.release()
    b = b + 1
    lock_b.release()

def thread:
    transaction.start()
    a = b
    b = b + 1
    transaction.commit()
def thread:
    lock_a.acquire()
    lock_b.acquire()
    a = b
    lock_a.release()
    b = b + 1
    lock_b.release()

def thread:
    transaction.start()
    a = b
    b = b + 1
    transaction.commit()

Advantages:

- ease of use on top
- efficient concurrency control under the hood
Transaction Abstraction

Transaction:

\[ T_i \left[ \begin{array}{c} \text{op}_1, \text{op}_2, \ldots, \text{op}_n \end{array} \right] \]

where \( \text{op} = \{ r(x)v, w(x)v, \ldots \} \)

and \( x \) is some shared object

Commitment:

\[ \{ x = 1 \} \quad T_i \left[ \begin{array}{c} w(x)2 \end{array} \right] \quad \{ x = 2 \} \]

Rollback:

\[ \{ x = 1 \} \quad T_i \left[ \begin{array}{c} w(x)2 \end{array} \right] \quad \{ x = 1 \} \]

\[ \{ x = 1 \} \quad T_i \left[ \begin{array}{c} w(x)2 \end{array} \right] \quad \rightarrow \quad T_i' \left[ \begin{array}{c} w(x)2 \end{array} \right] \quad \{ x = 2 \} \]
Distributed Transactional Memory

Replicated TM

$T_1$
Distributed Transactional Memory

Distributed Transactions
Distributed Transactional Memory

\[
\begin{array}{c}
\text{x} \\
\text{y} \\
\text{z}
\end{array}
\quad \text{T}_1
\quad \text{T}_2
\\quad \begin{array}{c}
\text{a} \\
\text{b} \\
\text{c}
\end{array}
\]

Distributed Transactions
Supremum Versioning Algorithm

Pessimistic approach

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1 \left[ r(x)_1, w(x)_2 \right] \\
\left| T_2 \right[ & \quad \leftarrow r(x)_2, w(x)_3 \right] \quad \{x = 3, y = 2\}
\end{align*}
\]
Supremum Versioning Algorithm

Pessimistic approach

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1 \left[ \begin{array}{c} r(x)1, w(x)2 \end{array} \right] \\
| T_2 \left[ \begin{array}{c} r(x)2, w(x)3 \end{array} \right] & \quad \{x = 3, y = 2\}
\end{align*}
\]

- Defer execution to prevent conflicts
Supremum Versioning Algorithm

Pessimistic approach

\[
\{ x = 1, y = 1 \} \quad T_1 \quad r(x)1, w(x)2 \quad \{ x = 3, y = 2 \}
\]

| T_2 | r(x)2, w(x)3 |

- Defer execution to prevent conflicts (tolerate high contention)
Supremum Versioning Algorithm

Pessimistic approach

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1 \left[ \begin{array}{c}
    r(x)1, w(x)2 \\
\end{array} \right] \\
| T_2 \left[ \begin{array}{c}
    r(x)2, w(x)3 \\
\end{array} \right] & \quad \{x = 3, y = 2\}
\end{align*}
\]

- Defer execution to prevent conflicts (tolerate high contention)
- Avoid (most) forced aborts
Supremum Versioning Algorithm

Pessimistic approach

\[
\{ x = 1, y = 1 \} \quad T_1 \left[ r(x)1, w(x)2 \right] \\
\quad | \quad T_2 \left[ \right] \quad r(x)2, w(x)3 \quad \{ x = 3, y = 2 \}
\]

- Defer execution to prevent conflicts (tolerate high contention)
- Avoid (most) forced aborts (safe irrevocable operations)
Supremum Versioning Algorithm

Pessimistic approach

\[ \{x = 1, y = 1\} \quad T_1 \quad \{r(x)1, w(x)2\} \]
\[ \quad \| \quad T_2 \quad \Rightarrow r(x)2, w(x)3 \quad \{x = 3, y = 2\} \]

- Defer execution to prevent conflicts (tolerate high contention)
- Avoid (most) forced aborts (safe irrevocable operations)

Early release on last use

\[ \{x = 1, y = 1\} \quad T_1 \quad \{r(x)1, w(x)2, r(y)1, w(y)2\} \]
\[ \quad \| \quad T_2 \quad \Rightarrow r(x)2, w(x)3 \quad \{x = 3, y = 2\} \]
Supremum Versioning Algorithm

Pessimistic approach

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1 \quad \begin{array}{c} r(x)1, w(x)2 \\ \end{array} \\
| \quad T_2 & \quad \begin{array}{c} r(x)2, w(x)3 \\ \end{array} \quad \{x = 3, y = 2\}
\end{align*}
\]

- Defer execution to prevent conflicts (tolerate high contention)
- Avoid (most) forced aborts (safe irrevocable operations)

Early release on last use

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1 \quad \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2 \\ \end{array} \\
| \quad T_2 & \quad \begin{array}{c} r(x)2, w(x)3 \\ \end{array} \quad \{x = 3, y = 2\}
\end{align*}
\]
Supremum Versioning Algorithm

Pessimistic approach

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1 \left[ r(x)1, w(x)2 \right] \\
| T_2 \left[ & \Rightarrow r(x)2, w(x)3 \right] & \quad \{x = 3, y = 2\}
\end{align*}
\]

- Defer execution to prevent conflicts (tolerate high contention)
- Avoid (most) forced aborts (safe irrevocable operations)

Early release on last use

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1 \left[ r(x)1, w(x)2, r(y)1, w(y)2 \right] \\
| T_2 \left[ & \Rightarrow r(x)2, w(x)3 \right] & \quad \{x = 3, y = 2\}
\end{align*}
\]

Completely distributed (no leader, dispatcher, etc.)
Supremum Versioning Algorithm

Pessimistic approach

\[ \{ x = 1, y = 1 \} \quad T_1 \quad [ r(x)1, w(x)2 ] \]
\[ \quad \mid T_2 \quad [ \quad \overset{r(x)2, w(x)3}{\longrightarrow} \quad ] \quad \{ x = 3, y = 2 \} \]

- Defer execution to prevent conflicts (tolerate high contention)
- Avoid (most) forced aborts (safe irrevocable operations)

Early release on last use

\[ \{ x = 1, y = 1 \} \quad T_1 \quad [ r(x)1, w(x)2, r(y)1, w(y)2 ] \]
\[ \quad \mid T_2 \quad [ \quad \overset{r(x)2, w(x)3}{\longrightarrow} \quad ] \quad \{ x = 3, y = 2 \} \]

Completely distributed (no leader, dispatcher, etc.)

Strong consistency
Bank Application
def deposit:
    account.deposit(sum)
def deposit:
    account.deposit(sum)

def withdraw:
    account.withdraw(sum)
def deposit:
    account.deposit(sum)

def withdraw:
    account.withdraw(sum)

def balance:
    print account.getBalance()
def deposit:
    account.deposit(sum)

def withdraw:
    account.withdraw(sum)

def balance:
    print account.getBalance()

def transfer:
    account1.withdraw(sum)
    account2.deposit(sum)
def deposit:
    account.deposit(sum)

def withdraw:
    account.withdraw(sum)

def balance:
    print account.getBalance()

def transfer:
    account1.withdraw(sum)
    account2.deposit(sum)

def audit:
    for a in accounts:
        sum += a.getBalance()
    value = bank.getCapital()
    bank.setCapital(sum)
    print "Accumulated capital", sum - value
def deposit:
    transaction.start()
    account.deposit(sum)
    transaction.commit()

def withdraw:
    transaction.start()
    account.withdraw(sum)
    transaction.commit()

def balance:
    transaction.start()
    print account.getBalance()
    transaction.commit()

def transfer:
    transaction.start()
    account1.withdraw(sum)
    account2.deposit(sum)
    transaction.commit()

def audit:
    transaction.start()
    for a in accounts:
        sum += a.getBalance()
    value = bank.getCapital()
    bank.setCapital(sum)
    print "Accumulated capital", sum - value
    transaction.commit()
```python
def deposit:
    transaction.start()
    account.deposit(sum)
    transaction.commit()

def withdraw:
    transaction.start()
    account.withdraw(sum)
    transaction.commit()

def balance:
    transaction.start()
    print account.getBalance()
    transaction.commit()

def transfer:
    transaction.start()
    account1.withdraw(sum)
    account2.deposit(sum)
    transaction.commit()

def audit:
    transaction.start()
    for a in accounts:
        sum += a.getBalance()
    value = bank.getCapital()
    bank.setCapital(sum)
    print "Accumulated capital", sum - value
    transaction.commit()
```

weaken consistency → improve efficiency
Eventually Consistent Extension
Eventually Consistent Extension
What we require from weak transactions

- do not wait for variables
What we require from weak transactions

- do not wait for variables
- do not block other transactions
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency
- do not disturb consistent transactions
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency
- do not disturb consistent transactions
- converge
Transaction Modes

Transaction $T_1$

$T_1 \left[ r(x)v_c, w(x)u_c \right]$
Transaction Modes

Transaction $T_1$

$$T_1 \left[ r(x) v_c, w(x) u_c \right]$$

Consistent mode

$$T_1^c \left[ r(x) v_c, w(x) u_c \right]$$
Transaction Modes

Transaction $T_1$

\[ T_1[ r(x)v_c, w(x)u_c ] \]

Consistent mode

\[ T_1^c[ r(x)v_c, w(x)u_c ] \]

Eventually consistent mode

\[ T_1^{ec}[ r(x)v_{ec}, w(x)u_{ec} ] \]
Transaction Modes

Transaction $T_1$

$T_1 \left[ r(x)v_c, w(x)u_c \right]$

**Consistent mode**

$T_1^c \left[ r(x)v_c, w(x)u_c \right]$

**Eventually consistent mode**

$T_1^{ec} \left[ r(x)v_{ec}, w(x)u_{ec} \right]$

Execute consistent and inconsistent modes simultaneously
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency
- do not disturb consistent transactions
- converge
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency
- do not disturb consistent transactions
- converge ✔
Variable Modification Versions

\[ \{ x = 1, y = 1 \} \quad T_1 \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2 \\ r(x)2, w(x)3 \end{array} \right] \quad T_2 \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2 \\ r(x)2, w(x)3 \end{array} \right] \quad \{ x = 3, y = 2 \} \]
Variable Modification Versions

\[ \{ x = 1, y = 1 \} \quad T^e_1 \quad \left[ \begin{array}{c} r(x)_1, w(x)_2, r(y)_1, w(y)_2 \end{array} \right] \quad T_2 \quad \left[ \begin{array}{c} r(x)_2, w(x)_3 \end{array} \right] \quad \{ x = 3, y = 2 \} \]
Variable Modification Versions

\[ \{ x = 1, y = 1 \} \quad T^{ec}_1 \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2 \\ r(x)1, w(x)2, r(y)1, w(y)2 \end{array} \right] \]

\[ \left[ \begin{array}{c} r(x)2, w(x)3 \\ r(x)2, w(x)3 \end{array} \right] \quad \{ x = 3, y = 2 \} \]

Snapshot Read Consistency

\[ \begin{array}{c}
T_1 \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2, w(y)3 \\ r(x)1, w(x)2, r(y)1, w(y)2, w(y)3 \end{array} \right] \\
T_2 \left[ \begin{array}{c} r(x)2, w(x)3 \\ r(x)2, w(x)3 \end{array} \right] \\
T_3 \left[ \begin{array}{c} r(x)2, w(x)3, r(y)3, w(y)4 \\ r(x)2, w(x)3, r(y)3, w(y)4 \end{array} \right] \\
T_4 \left[ \begin{array}{c} r(x)2, w(x)3, r(y)3, w(y)4 \\ r(x)2, w(x)3, r(y)3, w(y)4 \end{array} \right]
\end{array} \]
Variable Modification Versions

\[
\begin{align*}
\{x = 1, y = 1\} & \quad T_1^{ec} \quad r(0)x1, w(1)x2, r(0)y1, w(1)y2 \\
\mid T_2 & \quad r(1)x2, w(2)x3 \\
\{x = 3, y = 2\}
\end{align*}
\]

Snapshot Read Consistency

\[
\begin{align*}
T_1 & \quad r(0)x1, w(1)x2, r(0)y1, w(1)y2, w(2)y3 \\
\mid T_2 & \quad r(1)x2, w(2)x3 \\
\mid T_3 & \quad r(2)x3, w(3)x4, r(2)y3, w(3)y4 \\
\mid T_4 & \quad r(3)x4, w(4)x5 \\
\{x, y\}
\end{align*}
\]
Variable Modification Versions

\[
\{ x = 1, y = 1 \} \quad T_1^\text{ec} \left[ r(x)_1, w(x)_2, r(y)_1, w(y)_2 \right] \\
| T_2 \left[ \right] \\
| r(x)_2, w(x)_3 \right] \{ x = 3, y = 2 \}
\]

Snapshot Read Consistency

\[
T_1 \left[ r(x)_1, w(x)_2, r(y)_1, w(y)_2, w(y)_3 \right] \\
| T_2 \left[ \right] \left[ \right] \left[ \right] \left[ \right] \left[ \right] \\
| r(x)_3, w(x)_4, r(y)_3, w(y)_4 \right] \\
| r(x)_4, w(x)_5 \right] \\
\{ x, y \}, \{ x, y \}
\]
Variable Modification Versions

\[ \{x = 1, y = 1\} \quad T_1^{ec} \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2 \\ r(x)2, w(x)3 \end{array} \right] \]

\[ \{x = 3, y = 2\} \]

Snapshot Read Consistency

\[
\begin{align*}
T_1 & \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2, w(y)3 \\ r(x)2, w(x)3 \end{array} \right] \\
T_2 & \left[ \begin{array}{c} r(x)2, w(x)3 \end{array} \right] \\
T_3 & \left[ \begin{array}{c} r(x)3, w(x)4, r(y)3, w(y)4 \\ r(x)4, w(x)5 \end{array} \right]
\end{align*}
\]

\[ \{1, 2\}, \{2, 2\}, \{3, 3\}, \{4, 3\} \]
Variable Modification Versions

\[ \{ x = 1, y = 1 \} \quad T_1^{ec} \quad \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2 \\ r(x)2, w(x)3 \end{array} \right] \quad \{ x = 3, y = 2 \} \]

Snapshot Read Consistency

\[ T_1 \quad \left[ \begin{array}{c} r(x)1, w(x)2, r(y)1, w(y)2, w(y)3 \end{array} \right] \quad \{ x, y \}, \{ x, y \}, \{ x, y \}, \{ x, y \}, \{ x, y \} \]
Variable Modification Versions

\[
\begin{align*}
\{ x = 1, y = 1 \} & \quad T_1^{ec} \left[ r(0)^1, w(0)^2, r(0)^1, w(0)^2 \right] \\
& \quad | T_2 \left[ \quad r(1)^1, w(2)^3 \right] \quad \{ x = 3, y = 2 \}
\end{align*}
\]

Snapshot Read Consistency

\[
\begin{align*}
T_1 \left[ r(0)^1, w(1)^2, r(0)^1, w(1)^2, w(2)^3 \right] \\
& \quad | T_2 \left[ \quad r(1)^1, w(2)^3 \right] \\
& \quad | T_3 \left[ \quad r(2)^3, w(3)^4, r(2)^3, w(3)^4 \right] \\
& \quad | T_4 \left[ \quad r(3)^4, w(4)^5 \right]
\end{align*}
\]

\[
\begin{align*}
\{ \frac{1}{x}, \frac{2}{y} \}, & \quad \{ \frac{2}{x}, \frac{2}{y} \}, \quad \{ \frac{3}{x}, \frac{3}{y} \}, \quad \{ \frac{4}{x}, \frac{3}{y} \}, \quad \{ \frac{3}{x}, \frac{2}{y} \}, \quad \{ \frac{1}{x}, \frac{1}{y} \}
\end{align*}
\]
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency
- do not disturb consistent transactions
- converge ✓
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency ✔
- do not disturb consistent transactions
- converge ✔
Transactions:
- record the latest committed version of variable

Consistent Snapshot in Practice
Transactions:
- record the latest committed version of variable
- record the latest released version of variable (early release)
Consistent Snapshot in Practice

Transactions:
- record the latest committed version of variable
- record the latest released version of variable (early release)
- when releasing a variable early: record variables that were not released early
What we require from weak transactions

- do not wait for variables
- do not block other transactions
- internal consistency ✓
- do not disturb consistent transactions
- converge ✓
What we require from weak transactions

- do not wait for variables ✔
- do not block other transactions ✔
- internal consistency ✔
- do not disturb consistent transactions
- converge ✔
Write Bufferring

\[
\{ x = 1, y = 1 \} T_1 \left[ r(x)^1, w(x)^2, r(y)^1, w(y)^2, w(y)^3 \right] \{ x = 2, y = 3 \}
\]
Write Bufferring

\[
\begin{align*}
\{x = 1, y = 1\} &\quad T_1 \quad [ r(x)^1, w(x)^2, r(y)^1, w(y)^2, w(y)^3 \} \\
\{x = 1, y = 1\} &\quad \{x = 2, y = 3\}
\end{align*}
\]
Write Bufferring

\[ \{ x = 1, y = 1 \} \ T_1 \ [ \ r(x)1, w(x)2, r(y)1, w(y)2, w(y)3 \] \ \{ x = 1, y = 1 \} \ \{ x = 2, y = 3 \} \]

Consistent mode either:

- applies the buffered writes
Write Bufferring

\[ \{ x = 1, y = 1 \} \ T_1 \ [ \ r(x)^0, w(x)^2, r(y)^1, w(y)^2, w(y)^3 \} \ \{ x = 1, y = 1 \} \ \{ x = 2, y = 3 \} \]

Consistent mode either:
- applies the buffered writes (if consistency condition allows)
Write Bufferring

\[
\{ \overline{x} = 1, \overline{y} = 1 \} \ T_1 \ [ \overline{r(x)1}, \overline{w(x)2}, \overline{r(y)1}, \overline{w(y)2}, \overline{w(y)3} ] \ \{ \overline{x} = 1, \overline{y} = 1 \} \\
\{ \overline{x} = 2, \overline{y} = 3 \}
\]

Consistent mode either:

- applies the buffered writes (if consistency condition allows)
- re-executes from scratch
What we require from weak transactions

- do not wait for variables ✓
- do not block other transactions ✓
- internal consistency ✓
- do not disturb consistent transactions
- converge ✓
What we require from weak transactions

- do not wait for variables ✓
- do not block other transactions ✓
- internal consistency ✓
- do not disturb consistent transactions ✓
- converge ✓
Eventually Consistent SVA

\[
\{ x = 1, y = 1 \} \quad T_1 \left[ \begin{array}{c} r(x)^0, w(x)^1, r(y)^0, w(y)^1 \end{array} \right] \\
\mid T_2^c \left[ \begin{array}{c} r(x)^1, w(x)^2, r(y)^1, w(y)^2 \end{array} \right] \\
\mid T_2^{ec} \left[ \begin{array}{c} r(x)^0, w(x)^3 \end{array} \right] \quad \{ x = 3, y = 2 \}
\]
Eventually Consistent SVA

$$\begin{align*}
\{x = 1, y = 1\} & \quad T_1 [ \ r(x)^0, w(x)^1, r(y)^0, w(y)^1 \ ] \\
| \quad T_2^{c} [ \ r(x)^1, w(x)^2 \ ] \\
| \quad T_2^{ec} [ \ r(x)^0, w(x)^2 \ ] \\
| \quad T_3 \quad T_3 [ \ r(x)^2, w(x)^3 \ ] \quad \{x = 3, y = 2\} \\
\end{align*}$$
Summary

- eventual consistency extension for pessimistic distributed TM
- minimal extra cost
- eventually consistent transactions read consistent snapshots
- strongly consistent transactions are unaffected
Related Papers:

