Agenda

• Optimistic replication
  ▪ CVS, Subversion
  ▪ Duplicated databases
  ▪ Collaborative editing requirements
  ▪ Operational transformation: properties for convergence, transformation functions
Example: CVS, Subversion
Lock-modify-unlock solution
Copy-modify-merge solution
Copy-modify-merge solution

Harry compares the latest version to his own

A new merged version is created

The merged version is published

Now both users have each others’ changes
Duplicated databases (Thomas Write Rule 1975) (*)

• Model
  - A set of independent DBMPs
  - Each DBMP has its own copy of the database
  - DBMPs communicate via messages
  - Communications are subject to failures
  - Messages between two sites are delivered in the same order they were sent
  - No use of global timestamps

• The system is correct if it eventually converges

Duplicated databases (Thomas Write Rule 1975)

DBMP₁  DBMP₂

<table>
<thead>
<tr>
<th>op₁</th>
<th>op₂</th>
</tr>
</thead>
</table>

Not possible

DBMP₁  DBMP₂  DBMP₃

<table>
<thead>
<tr>
<th>op₁</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>op₂</th>
</tr>
</thead>
</table>

Possible
Duplicated databases (Thomas Write Rule 1975)

- The database = collection of (selector, value) pairs
- Operations:
  - Selection:
    - get(selector) returns the current associated value
  - Assignment:
    - set(selector, new_value) replaces associated value with new_value
  - Creation:
    - new(selector, initial_value) adds (selector, initial_value) entry
  - Deletion:
    - delete(selector, value) deletes existing (selector, value) pair
Duplicated databases (Thomas Write Rule 1975)

• How to guarantee that copies are consistent?

```
dbmp1
new(x,5)
set(x,8)
set(x,9)

dbmp2

Database
(x,9) (x,8)
```
Thomas Timestamps

• In the face of concurrent modifications to an entry, how to select the « most recent » change?

• Thomas timestamps before Lamport timestamps !

• A timestamp is a pair (T,D)
  ▪ T is a network time standard (time-of-day)
  ▪ D is a DBMP identifier

• Timestamps comparison
  ▪ (T1,D1)>(T2,D2) iff (T1>T2) or (T1=T2 and D1>D2)

• If D1=D2 and T1=T2, then the same operation
Database entry

- \( E ::= (S, V, T) \)
  - \( S \) is the selector
  - \( V \) is the value
  - \( T \) is the timestamp = (Time, DBMP id) of the last change to the entry
Thomas write rule = last writer wins

Database
new(x,5)
set(x,8)
set(x,9)

DBMP\(_1\)  DBMP\(_2\)

\((x,5,(10h,1)))\)
\((x,8,(10h02,1)))\)
\((x,9,(10h03,2)))\)

\((10h02,1)<(10h03,2)\)

\((x,5,(10h,1)))\)
set(x,9)

\((x,9,(10h03,2)))\)
\((x,9,(10h03,2)))\)

\((10h03,2)>(10h02,1)\)
Creation/update

- Assume the creation will arrive and create the entry right away
- Creation operation ignored at arrival
Creation/update

DBMP_1

new(x,2)

set(x,3)

DBMP_2

(x,2,"

DBMP_3

(x,3,"

ignored
Deletion

- Solution: never remove an entry, mark « deleted » flag
Tombstones

- $E ::= (S, V, F, T)$
  - $S$ is the selector
  - $V$ is the value
  - $F$ is the deleted/not-deleted flag
  - $T$ is the timestamp = $(\text{Time}, \text{DBMP id})$ of the last change to the entry

- $F = t$ if deleted
- $F = f$ if not-deleted
Tombstones

DBMP_1
(x,3,f,(10h,2))
delete(x)
(x,3,t,(10h01,1))

DBMP_2
(x,3,f,(10h,2))
set(x,5)
(x,5,f,(10h02,2))

DBMP_3
(x,3,f,(10h,2))
(x,5,f,(10h02,2))

DBMP_4
(x,3,f,(10h,2))
(x,3,t,(10h01,1))
(x,3,t,(10h01,1))
Tombstones prevent recreation
Tombstones

- DBMP1 cannot distinguish in which of the two cases DBMP2 is

\[
\begin{align*}
\text{DBMP}_1 & \rightarrow (x,3,f,(10h,2))
\end{align*}
\]

\[
\begin{align*}
delete(x) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,f,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
debale(x) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
delete(x) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
delete(x) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
delete(x) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
delete(x) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h03,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h02,2))
\end{align*}
\]

\[
\begin{align*}
set(x,5) \rightarrow (x,3,t,(10h01,1))
\end{align*}
\]

• Solution: Associate to an entry the creation timestamp
Tombstones

- $E ::= (S, V, F, CT, T)$
  - $S$ is the selector
  - $V$ is the value
  - $F$ is the deleted/not-deleted flag
  - $CT$ is the timestamp for creation
  - $T$ is the timestamp = (Time, DBMP id) of the last change to the entry

- If $F=f$ and $CT=T$, then creation
- If $F=f$ and $CT<T$, then assignment
- If $F=t$, then deletion
Tombstones

DBMP\(_1\)

- \((x,3,f,(10h,2),(10h,2))\)
- delete\((x)\)
- \((x,3,t,(10h,2),(10h01,1))\)

DBMP\(_2\)

- \((x,3,f,(10h,2),(10h,2))\)
- set\((x,5)\)
- \((x,5,f,(10h,2),(10h02,2))\)

Same creation time =>

- delete

- Same creation time =>
  - delete

Claudia Ignat, ignatcla@loria.fr
Tombstones

**DBMP\(_1\)**

\[(x,3,f,(10h,2),(10h,2))\]

**delete(x)**

\[(x,3,t,(10h,2),(10h01,1))\]

\[(x,5,f,(10h03,2),(10h03,2))\]

Different creation time => recreate

**DBMP\(_2\)**

\[(x,3,f,(10h,2),(10h,2))\]

**set(x,5)**

\[(x,3,t,(10h,2),(10h01,1))\]

\[(x,5,f,(10h03,2),(10h03,2))\]
Garbage collection

- Make sure of no reception of assignments with same S and the same or older CT
- Remember assumption: Modifications of a DBMP delivered in sequential order
- Each DBMP maintains two « timestamp vectors »
  - Last modifications from all DBMPs
    - LM[i] last timestamp from DBMP i
    - Modified each time an operation is received
  - Oldest timestamps received by each DBMP
    - OT[i] oldest timestamp received by DBMP i
    - Sent upon reception of a delete
- Can do garbage collection if timestamp of delete ≤ timestamp of min(OT)
Garbage collection

$DBMP_1$

$LM=[[]$
$OT=[]$

$DBMP_2$

$LM=[(1h,1)]$
$OT=[]$

$DBMP_3$

$LM=[(1h,1)]$
$OT=[]$

$new(x)$

$(x,1,f,(1h,1),(1h,1))$

$LM=[(2h,2)]$
$OT=[]$

$LM=[(2h,2),(3h,3)]$
$OT=[]$

$LM=[(1h,1),(3h,3)]$
$OT=[]$

$(x,1,f,(1h,1),(1h,1))$

$LM=[[(1h,1)]]$
$OT=[]$

$LM=[(1h,1)]$
$OT=[]$

$LM=[(1h,1),(2h,2)]$
$OT=[]$

$LM=[[(1h,1)]$
$OT=[]$

$LM=[(1h,1)]$
$OT=[]$

$LM=[[(1h,1),(2h,2)]]$
$OT=[]$

$LM=[[(1h,1),(2h,2),(2h,2)]]$

$LM=[[(1h,1),(3h,3),(3h,3)]]$

$LM=[[(1h,1),(2h,2),(2h,2)]]$

$LM=[[(2h,2),(3h,3),(3h,3)]]$

$LM=[((1h,1),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$

$LM=[((1h,1),(3h,3),(3h,3))]]$
Garbage collection

DBMP\_1
LM=[(2h,2),(3h,3)]
OT=[]
delete(z)
(z,3,t,(3h,3),(4h,1))
LM=[(2h,2),(3h,3)]
OT=[(3h,2)]
LM=[(2h,2),(3h,3)]
OT=[(3h,2),(2h,3)]

DBMP\_2
LM=[(1h,1),(3h,3)]
OT=[]
LM=[(4h,1),(3h,3)]
OT=[]
LM=[(4h,1),(3h,3)]
OT=[]
LM=[(4h,1),(3h,3)]
OT=[(2h,3)]

DBMP\_3
LM=[(1h,1),(2h,2)]
OT=[]
LM=[(4h,1),(2h,2)]
OT=[]
LM=[(4h,1),(2h,2)]
OT=[(3h,2)]
LM=[(4h,1),(2h,2)]
OT=[(3h,2)]
Garbage collection

DBMP₁
LM=[(2h,2),(3h,3)]
OT=[(3h,2),(2h,3)]

LM=[(5h,2),(3h,3)]
OT=[(3h,2),(2h,3)]

DBMP₂
LM=[(4h,1),(3h,3)]
OT=[(2h,3)]

LM=[(4h,1),(3h,3)]
OT=[(2h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]

LM=[(4h,1),(5h,2)]
OT=[(3h,1),(3h,2)]
Garbage collection

• z can be garbaged
Collaborative editing: from users to community of users

“Isn’t it chaotic to all edit in the same document, even the same paragraph, at the same time?”

“Why would a group ever want to edit in the same line of text at the same time?”

GROVE, 1989
Collaborative editing: from users to community of users

2013: MOOC “Fundamentals of Online Education: Planning and Applications” with 40,000 participants
2016: Nuit debout, more than 70 people edit a pad
2018: online CSCW PC meeting with 120 members
Collaborative editing: from users to community of users

Real-time Wikipedia

Ponte Morandi: Revision history

View logs for this page (view filter log)

Search for revisions

For any version listed below, click on its date to view it. For more help, see Help:Page history and Help:Edit summary.

External tools: Revision history statistics • Revision revision search • Edit by user • Number of watchers • Page view statistics • Fix dead links

(cur) = difference from current version, (prev) = difference from preceding version, m = minor edit, s = section edit, a = automatic edit summary

(newest | oldest) View (newer 24 | older 24) (20 | 50 | 100 | 250 | 500)

Compare selected revisions

• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)
• (cur) 12:22, 14 August 2018 Pigsheathing (talk | contribs) m . (4,619 bytes) (-1) . (→top: ce) (undo)

Compare selected revisions

(newest | oldest) View (newer 24 | older 24) (20 | 50 | 100 | 250 | 500)
Limitations of Central Authority Systems

SCALABILITY

PRIVACY
Peer-to-Peer Collaborative Systems
Collaboration Modes

Concurrent Changes
Collaboration Modes

Offline Work

conflicts
Collaboration Modes
Ad-hoc Collaboration
Operational transformation

- Domain of application: collaborative editing
- Document replication
  - Disconnected work
  - Better response time for real-time collaboration
Operational transformation

• Optimistic replication model
  ▪ An operation is:
    ○ Locally executed,
    ○ Sent to other sites,
    ○ Received by a site,
    ○ Transformed according to concurrent operations,
    ○ Executed on local copy

• 2 components:
  ▪ An integration algorithm: diffusion, integration
  ▪ Some transformation functions
Operational transformation

• Textual documents seen as a sequence of characters

• Operations
  ▪ ins(p, c)
  ▪ del(p)

• Three main issues
  ▪ Causality preservation
  ▪ Intention preservation
  ▪ Convergence
Causality

Site 1

Site 2

Site 3

Site 1

Site 2

Site 3

Site 1

Site 2

Site 3

\( \text{op}_1 = \text{ins}(1, y) \)

\( \text{op}_2 = \text{del}(1) \)

\( \text{op}_2 = \text{del}(1) \)

\( \text{op}_1 = \text{ins}(1, y) \)

\( \text{op}_2 = \text{del}(1) \)

\( \text{op}_2 = \text{del}(1) \)

\( \text{op}_1 = \text{ins}(1, y) \)

\( \text{op}_1 = \text{ins}(1, y) \)

\( \text{op}_1 = \text{ins}(1, y) \)

\( \text{op}_2 = \text{del}(1) \)

\( \text{op}_1 = \text{ins}(1, y) \)

\( \text{op}_1 = \text{ins}(1, y) \)
Causality

Site 1

\[ [0,0,0] \]

\[ "x" \]

\[ op_1 = \text{ins}(1,y) \]

\[ [1,0,0] \]

\[ "yx" \]

\[ op_2 = \text{del}(1) \]

\[ [1,1,0] \]

\[ "x" \]

Site 2

\[ [0,0,0] \]

\[ "x" \]

\[ op_1 = \text{ins}(1,y) \]

\[ [1,0,0] \]

\[ "yx" \]

\[ op_2 = \text{del}(1) \]

\[ [1,1,0] \]

\[ "x" \]

Site 3

\[ [0,0,0] \]

\[ "x" \]

\[ op_1 = \text{ins}(1,y) \]

\[ [1,0,0] \]

\[ "y" \]

\[ "yx" \]

delayed
Intention

- Intention of an operation is the observed effect as result of its execution on its generation state
- Passing from initial state “ab” to final state “aXb” we can observe:
  - ins(2,X)
  - ins(a<X<b)
  - ins(a<X)
  - ins(X<b)
Preserving user intention (*)

• For any operation op, the effects of executing op at all sites should be the same as the intention of op.

• The effect of executing O does not change the effects of independent operations.

Intention violation

Site 1
“concurrency control”

$op_1 = \text{ins}(7, r)$

“concurrency control”

$op_2 = \text{ins}(17, o)$

Site 2
“concurrency control”

“concurrency control”
Intention violation + divergence

Site 1

```
"concurrency control"
```

```
op_1 = ins(7, r)
```

```
"concurrency control"
```

Site 2

```
"concurrency control"
```

```
op_2 = ins(17, o)
```

```
"concurrency control"
```

```
op_1 = ins(7, r)
```

```
op_2 = ins(17, o)
```

```
"concurrency control"
```
Intention preservation

\[ T(\text{ins}(p_1, c_1), \text{ins}(p_2, c_2)) :\]
\[ \text{if } (p_1 < p_2) \]
\[ \quad \text{return } \text{ins}(p_1, c_1) \]
\[ \text{else} \]
\[ \quad \text{return } \text{ins}(p_1+1, c_1) \]
\[ \text{endif} \]
Example transformation functions

\[ T(\text{ins}(p_1, c_1), \text{ins}(p_2, c_2)) : - \]
\[ \text{if } (p_1 < p_2) \text{ return } \text{ins}(p_1, c_1) \]
\[ \text{else return } \text{ins}(p_1+1, c_1) \]

\[ T(\text{ins}(p_1, c_1), \text{del}(p_2)) : - \]
\[ \text{if } (p_1 \leq p_2) \text{ return } \text{ins}(p_1, c_1) \]
\[ \text{else return } \text{ins}(p_1-1, c_1) \]
\[ \text{endif} \]

\[ T(\text{del}(p_1), \text{ins}(p_2, c_2)) : - \]
\[ \text{if } (p_1 < p_2) \text{ return } \text{del}(p_1) \]
\[ \text{else return } \text{del}(p_1+1) \]

\[ T(\text{del}(p_1), \text{del}(p_2)) : - \]
\[ \text{if } (p_1 < p_2) \text{ return } \text{del}(p_1) \]
\[ \text{else if } (p_1 > p_2) \text{ return } \text{del}(p_1-1) \]
\[ \text{else return } \text{id}() \]
Convergence but no intention preservation

Thomas Write Rule

Site 1

\(\text{op}_1=\text{set}(s, "AXB")\)

\(\{s, "AXB", f, (0h,1), (9h01, \dagger)\}\)

Site 2

\(\{s, "AXB", f, (0h,1), (9h01, \dagger)\}\)

Site 3

\(\text{op}_2=\text{set}(s, "AYB")\)

\(\{s, "AYB", f, (0h,1), (9h02, 3)\}\)

\(\{s, "AYB", f, (0h,1), (9h02, 3)\}\)

\(\{s, "AYB", f, (0h,1), (9h02, 3)\}\)
Convergence – TP1 property

- $T(o_2: \text{operation}, o_1: \text{operation}) = o'_2$
  - $o_1$ and $o_2$ concurrent, defined on a state $S$
  - $o'_2$ same effects as $o_2$, defined on $S.o_1$

$[TP1] \quad o_1 \circ T(o_2, o_1) \equiv o_2 \circ T(o_1, o_2)$
Convergence – TP2 property

\[ TP2 \] \( T(op_3, \, op_1 \circ T(op_2, \, op_1)) = T(op_3, \, op_2 \circ T(op_1, \, op_2)) \)
OT Problems

- Design and verify Transformation functions $T$
- $T$ also known as transpose_fd
- Verification of conditions TP1 and TP2
  - Combinatorial explosion ($>100$ cases for a string)
  - Iterative process
  - Repetitive and error prone task
Partial concurrency

Site 1

“telefone”

\( \text{op}_1 = \text{ins}(5, p) \)

“telefone”

\( \text{op}_3 = \text{ins}(6, h) \)

“telefone”

\( \text{op''}_2 = \text{del}(7) \)

“telefone”

Site 2

“telefone”

\( \text{op}_2 = \text{del}(5) \)

“telefone”

\( \text{op'}_1 = \text{ins}(5, p) \)

“telefone”

\( \text{op'}_2 = \text{T}(\text{op}_2, \text{op}_1) = \text{del}(6) \)

“telefone”

\( \text{op'}'_2 = \text{T}(\text{op'}_2, \text{op}_3) = \text{del}(7) \)

“telefone”

\( \text{op'}_1 = \text{T}(\text{op}_1, \text{op}_2) = \text{ins}(5) \)

“telefone”

T(\text{op}_3, \text{op}_2) \text{ not allowed to be performed !!!}

\( \text{op}_3 = \text{ins}(5, h) \)
Partial concurrency

Site 1
- "telefone"
  - $op_1 = \text{ins}(5, p)$
  - $op_2 = \text{del}(7)$

Site 2
- "telefone"
  - $op_2 = \text{del}(5)$
  - $op' = \text{ins}(5, p)$

 transpose_bk

Site 2
- "telefone"
  - $op' = \text{ins}(6, h)$
  - $op' = \text{T}(op_2, op_1) = \text{del}(6)$

$op' = \text{T}(op_3, op'_2) = \text{ins}(6, h)$
Partial concurrency

- \( \text{transpose}_\text{bk}(\text{op}_1, \text{op}'_2) = (\text{op}_2, \text{op}'_1) \)
  - \( \text{op}'_2 = T(\text{op}_2, \text{op}_1) \)
    Therefore \( \text{op}_2 = T^{-1}(\text{op}'_2, \text{op}_1) \)
  - \( \text{op}'_1 = T(\text{op}_1, \text{op}_2) \)

\[ \text{op}'_2 = T(\text{op}_2, \text{op}_1) \]
\[ \text{op}'_1 = T(\text{op}_1, \text{op}_2) \]
Example transformation functions

\[ T(\text{ins}(p_1, c_1), \text{ins}(p_2, c_2)) :\]
\[
\text{if } (p_1 < p_2) \text{ return } \text{ins}(p_1, c_1) \\
\text{else return } \text{ins}(p_1+1, c_1) 
\]

\[ T(\text{ins}(p_1, c_1), \text{del}(p_2)) :\]
\[
\text{if } (p_1 \leq p_2) \text{ return } \text{ins}(p_1, c_1) \\
\text{else return } \text{ins}(p_1-1, c_1) \\
\text{endif}
\]

\[ T(\text{del}(p_1), \text{ins}(p_2, c_2)) :\]
\[
\text{if } (p_1 < p_2) \text{ return } \text{del}(p_1) \\
\text{else return } \text{del}(p_1+1) 
\]

\[ T(\text{del}(p_1), \text{del}(p_2)) :\]
\[
\text{if } (p_1 < p_2) \text{ return } \text{del}(p_1) \\
\text{else if } (p_1 > p_2) \text{ return } \text{del}(p_1-1) \\
\text{else return } \text{id}() 
\]

TP1 not respected!
Ressel transformation functions (*)

\[ T(\text{ins}(p_1, c_1, u_1), \text{ins}(p_2, c_2, u_2)) :\]
\[ \text{if } ((p_1 < p_2) \text{ or } (p_1 = p_2 \text{ and } u_1 < u_2)) \text{ return } \text{ins}(p_1, c_1, u_1) \]
\[ \text{else return } \text{ins}(p_1 + 1, c_1, u_1) \]

\[ T(\text{ins}(p_1, c_1, u_1), \text{del}(p_2, u_2)) :\]
\[ \text{if } (p_1 \leq p_2) \text{ return } \text{ins}(p_1, c_1, u_1) \]
\[ \text{else return } \text{ins}(p_1 - 1, c_1, u_1) \]
\[ \text{endif} \]

\[ T(\text{del}(p_1, u_1), \text{ins}(p_2, c_2, u_2)) :\]
\[ \text{if } (p_1 < p_2) \text{ return } \text{del}(p_1, u_1) \]
\[ \text{else return } \text{del}(p_1 + 1, u_1) \]

\[ T(\text{del}(p_1, u_1), \text{del}(p_2, u_2)) :\]
\[ \text{if } (p_1 < p_2) \text{ return } \text{del}(p_1, u_1) \]
\[ \text{else if } (p_1 > p_2) \text{ return } \text{del}(p_1 - 1, u_1) \]
\[ \text{else return } id() \]

TP1 ok, but not TP2 !
False-tie problem

Site 1
“abc”

op₁=Insert(2,x)

“axbc”

Site 2
“abc”

op₂=Delete (2,b)

“ac”

Site 3
“abc”

op₃=Insert(3,y)

“abyc”

op’₁=Insert(2,x)

op’₃=Insert(2,y)

“axyc”?

“ayxc”?
TTF (Tombstone Transformation Functions) Approach (*)

- Keep “tombstones” of deleted elements

Tombstone Transformation Functions

- $T(\text{insert}(p_1, e_1, s_1), \text{insert}(p_2, e_2, s_2))$
  
  ```
  if(p_1 < p_2) return \text{insert}(p_1, e_1, s_1)
  else if(p_1 = p_2 \text{ and } s_1 < s_2) return \text{insert}(p_1, e_1, s_1)
  else return \text{insert}(p_1+1, e_1, s_1)
  ```

- $T(\text{insert}(p_1, e_1, s_1), \text{delete}(p_2, e_2, s_2))$
  
  ```
  return \text{insert}(p_1, e_1, s_1)
  ```

- $T(\text{delete}(p_1, s_1), \text{insert}(p_2, s_2))$
  
  ```
  if(p_1 < p_2) return \text{delete}(p_1, s_1)
  else return \text{delete}(p_1+1, s_1)
  ```

- $T(\text{delete}(p_1, s_1), \text{delete}(p_2, s_2))$
  
  ```
  return \text{delete}(p_1, s_1)
  ```
Compacted storage model

- Compacted model = sequence of (character, abs_pos)

```
Model
h a b n y c

View
a b y c

Compacted Model
a,2 b,3 y,5 c,6

Insert(3,y)
Insert(5,y)

view2model()
```
Delta storage model

- Delta model = sequence of (character, offset)
Models comparison

- Basic Model
  - Deleted characters are kept
  - Size of the model is growing infinitely

- Compacted Model
  - Update absolute position of all characters located after the effect position

- Delta Model
  - Update the offset of next character

- Our observations
  - View2model can be optimised (caret position)
  - Overhead of view2model is not significant