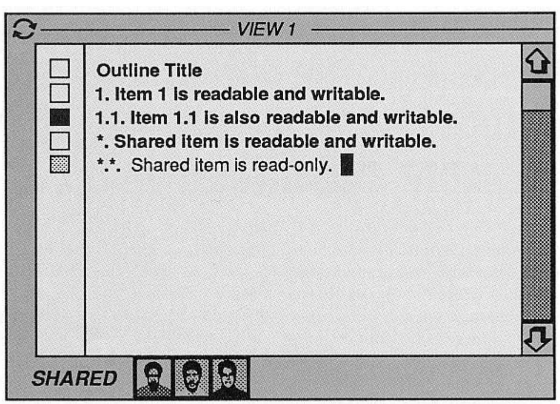
Agenda

- Optimistic replication
  - CVS, Subversion
  - Duplicated databases
  - Collaborative editing requirements
  - Operational transformation: properties for convergence, transformation functions, algorithms

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

From users to community of users: new practices

WIKIPEDIA
The Free Encyclopedia

Not logged in Talk Contributions Create account Log in

Article Talk | Read Edit View history | Search Wikipedia

Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute
Help
Learn to edit
Community portal
Recent changes
Upload file

Tools
What links here
Related changes
Atom
Special pages
Page information
Wikidata item

Languages

United States Capitol: Revision history

Help

View logs for this page (view filter log)

Filter revisions

External tools: Find addition/removal (Alternate) · Find edits by user · Page statistics · Pageviews · Fix dead links

For any version listed below, click on its date to view it. For more help, see Help:Page history and Help:Edit summary. (cur) = difference from current version, (prev) = difference from preceding version,
m = minor edit, → = section edit, ← = automatic edit summary

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

Compare selected revisions

- (cur | prev) 22:56, 6 January 2021 Sleyece (talk | contribs) . . (83,551 bytes) (+96) . . *(Be Concise)* (undo)
- (cur | prev) 22:53, 6 January 2021 Thanoscar21 (talk | contribs) m . . (83,455 bytes) (−39) . . *(Rollback edit(s) by 49.195.0.222 (talk): per WP:NPOV (RW 16))* (undo) *(Tag: Rollback)*
- (cur | prev) 22:52, 6 January 2021 49.195.0.222 (talk) . . (83,494 bytes) (+39) . . (undo) *(Tag: Reverted)*
- (cur | prev) 22:52, 6 January 2021 Sleyece (talk | contribs) m . . (83,455 bytes) (−3) . . *(Grammar)* (undo)
- (cur | prev) 22:52, 6 January 2021 2605:a601:ada1:2a00:ad83:9ec8:20bc:44c5 (talk) . . (83,458 bytes) (+4) . . *(→Major events: Fixed punctuation)* (undo) *(Tags: Mobile edit, Mobile web edit)*
- (cur | prev) 22:51, 6 January 2021 Sleyece (talk | contribs) . . (83,454 bytes) (+423) . . *(Bot Reverted too Much)* (undo)
- (cur | prev) 22:49, 6 January 2021 2605:a601:ada1:2a00:ad83:9ec8:20bc:44c5 (talk) . . (83,031 bytes) (+15) . . *(→Major events: Cleaned up wording on Trump recorded message)* (undo) *(Tags: Mobile edit, Mobile web edit)*
- (cur | prev) 22:47, 6 January 2021 2605:a601:ada1:2a00:ad83:9ec8:20bc:44c5 (talk) . . (83,016 bytes) (−1) . . *(→Major events: Replaced one other instance of "rioters" with "people")* (undo) *(Tags: Mobile edit, Mobile web edit)*
- (cur | prev) 22:45, 6 January 2021 2605:a601:ada1:2a00:ad83:9ec8:20bc:44c5 (talk) . . (83,017 bytes) (−8) . . *(→Major events: Typo)* (undo) *(Tags: Mobile edit, Mobile web edit)*
- (cur | prev) 22:45, 6 January 2021 2600:1700:2b9c:5010:a145:6992:7d5a:8714 (talk) . . (83,025 bytes) (+81) . . *(→Major events)* (undo)
- (cur | prev) 22:44, 6 January 2021 2605:a601:ada1:2a00:ad83:9ec8:20bc:44c5 (talk) . . (82,944 bytes) (+52) . . *(→Major events: Rephrased "rioters")* (undo) *(Tags: Mobile edit, Mobile web edit)*
- (cur | prev) 22:39, 6 January 2021 Jeswinj (talk | contribs) . . (82,892 bytes) (−351) . . (undo) *(Tag: Visual edit)*
- (cur | prev) 22:39, 6 January 2021 ClueBot NG (talk | contribs) m . . (83,243 bytes) (−47) . . *(Reverting possible vandalism by 2603:9001:4903:D811:9052:EFE:155C:2134 to version by 81.167.188.42. Report False Positive? Thanks, ClueBot NG. (3860441) (Bot))* (undo) *(Tag: Rollback)*
- (cur | prev) 22:39, 6 January 2021 2603:9001:4903:d811:9052:efe:155c:2134 (talk) . . (83,290 bytes) (+47) . . (undo) *(Tags: Mobile edit, Mobile web edit, Reverted)*
- (cur | prev) 22:35, 6 January 2021 81.167.188.42 (talk) . . (83,243 bytes) (+40) . . (undo)
- (cur | prev) 22:34, 6 January 2021 72.217.56.18 (talk) . . (83,203 bytes) (−58) . . *(person shot has not been reported dead as of yet!)* (undo)
- (cur | prev) 22:33, 6 January 2021 2600:1700:1c60:41d0:e5a9:3248:c03e:eb02 (talk) . . (83,261 bytes) (+73) . . *(Adding the President's reaction to the event.)* (undo) *(Tag: extraneous markup)*
- (cur | prev) 22:31, 6 January 2021 2603:8000:8e42:cb00:55ec:35dc:94e4:c9be (talk) . . (83,188 bytes) (+1) . . (undo) *(Tag: Possible vandalism)*
- (cur | prev) 22:29, 6 January 2021 92.21.72.53 (talk) . . (83,187 bytes) (+40) . . *(Adding info)* (undo) *(Tags: Mobile edit, Mobile web edit, Visual edit, Possible vandalism)*
- (cur | prev) 22:28, 6 January 2021 176.23.9.200 (talk) . . (83,147 bytes) (+10) . . (undo)

Compare selected revisions

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

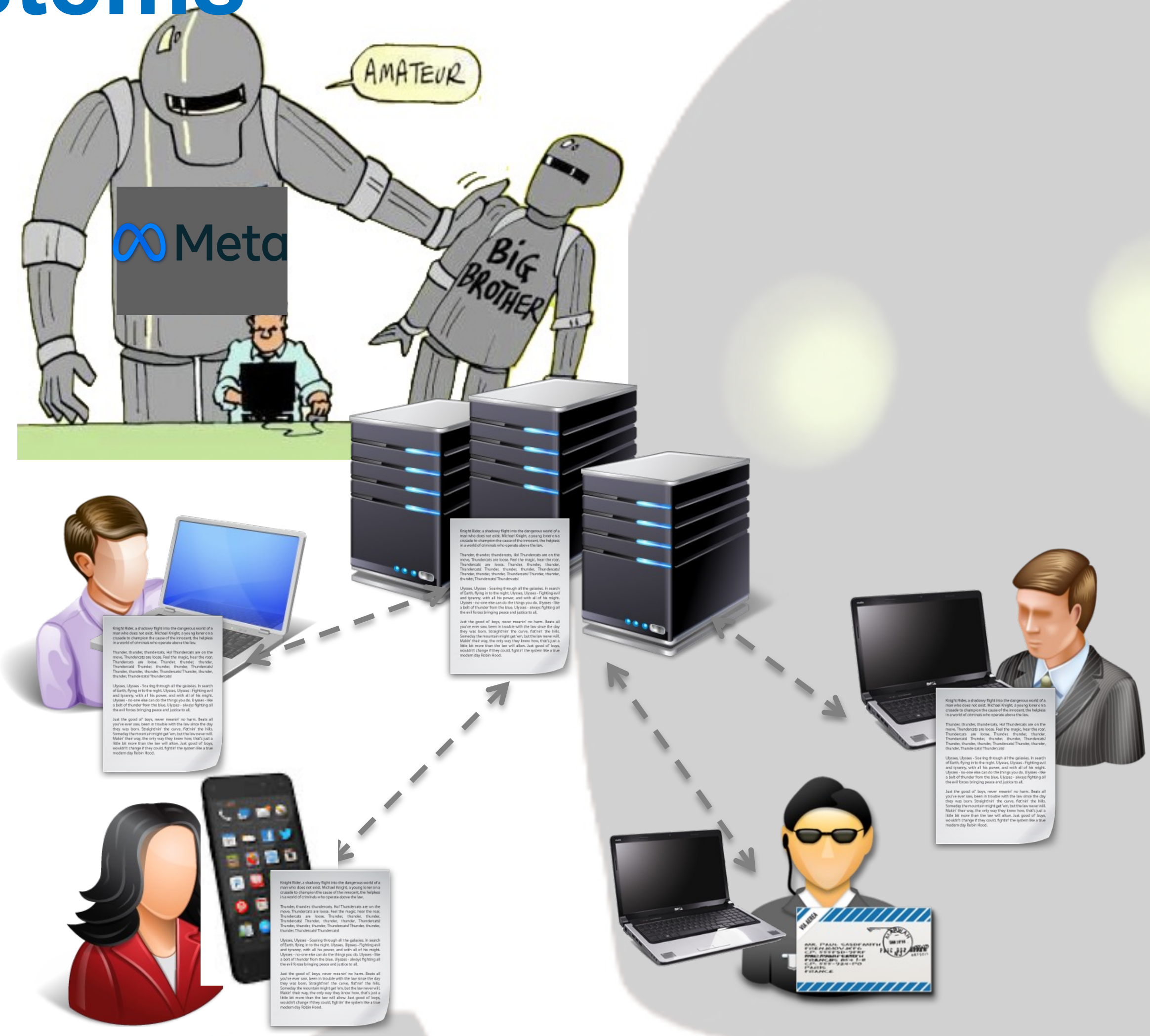
From users to community of users: new practices

Example: Intergovernmental Panel Climate Change (IPCC) assessment report editing

- Expert contributions: hundreds of scientists from different countries
- Review stages by experts and governments
- Approval sessions: government delegates and lead authors
- Editing teams: review editors

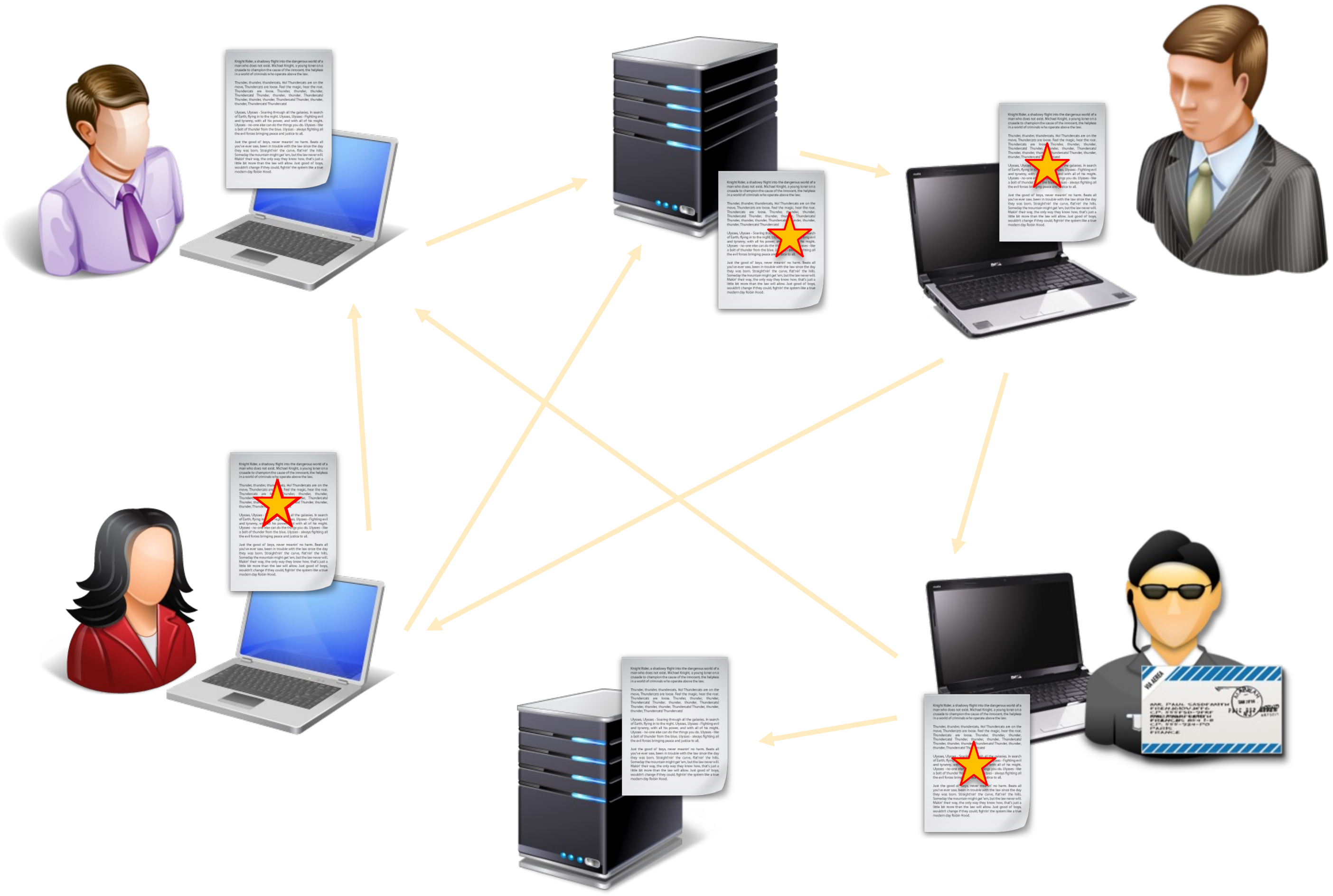
Limitations of Central Authority Systems

SCALABILITY



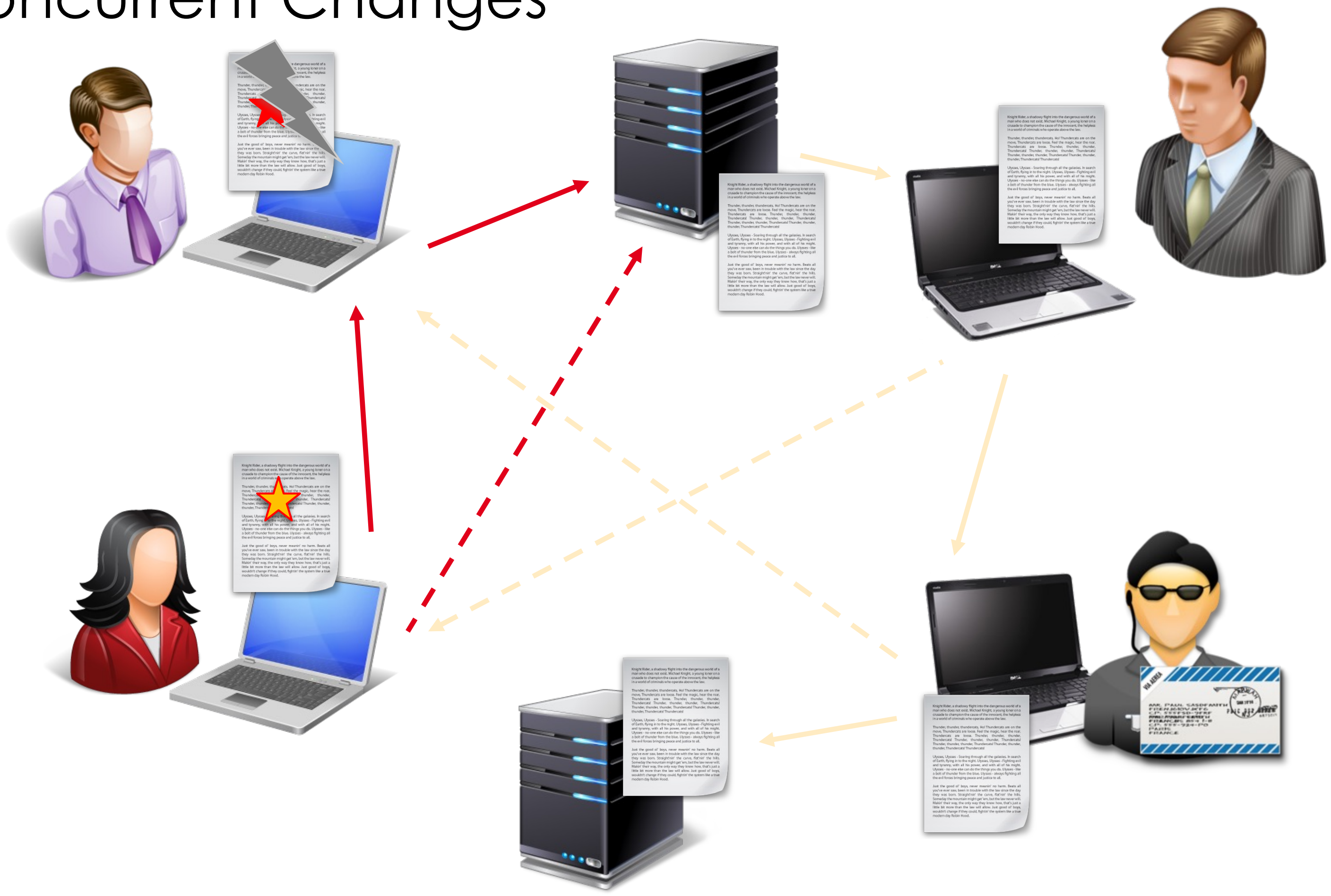
PRIVACY

Peer-to-Peer Collaborative Systems

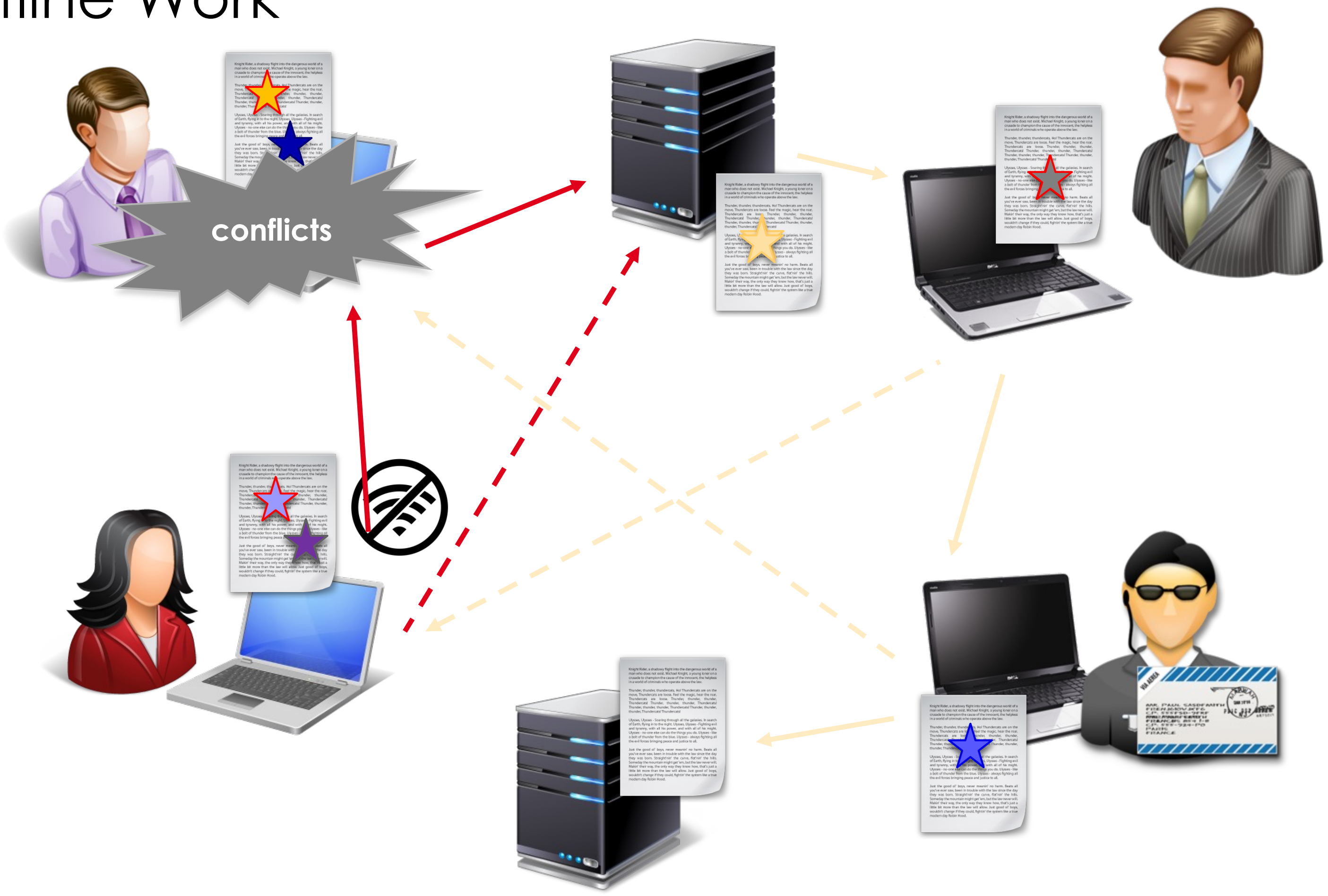


Collaboration Modes

Concurrent Changes



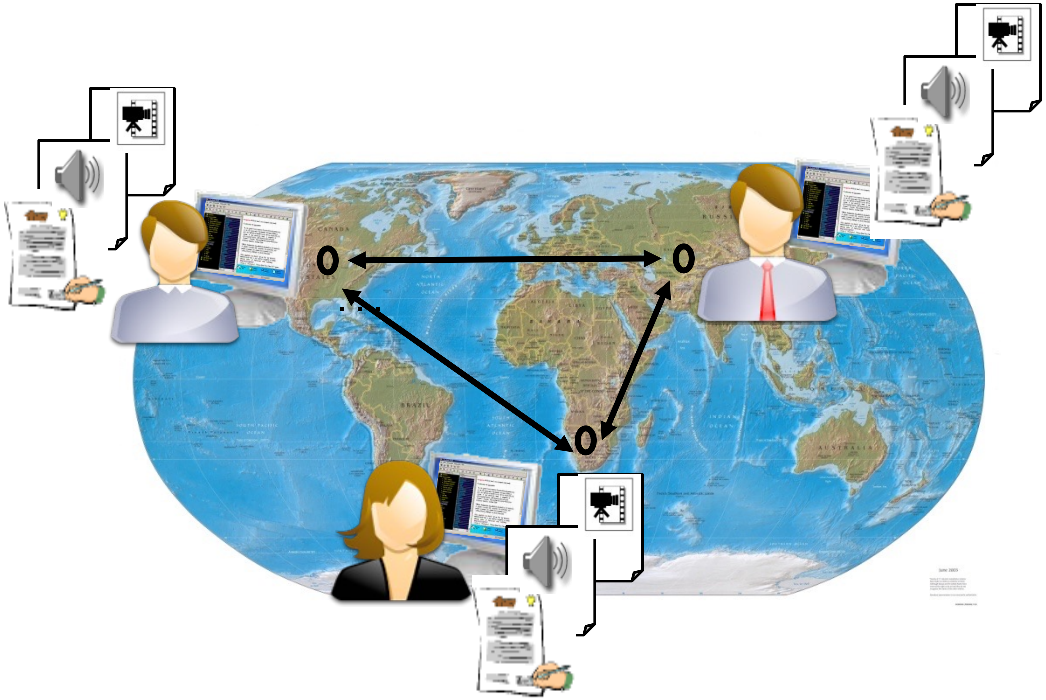
Collaboration Modes
Offline Work



Collaboration Modes

Ad-hoc Collaboration

Operational transformation



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

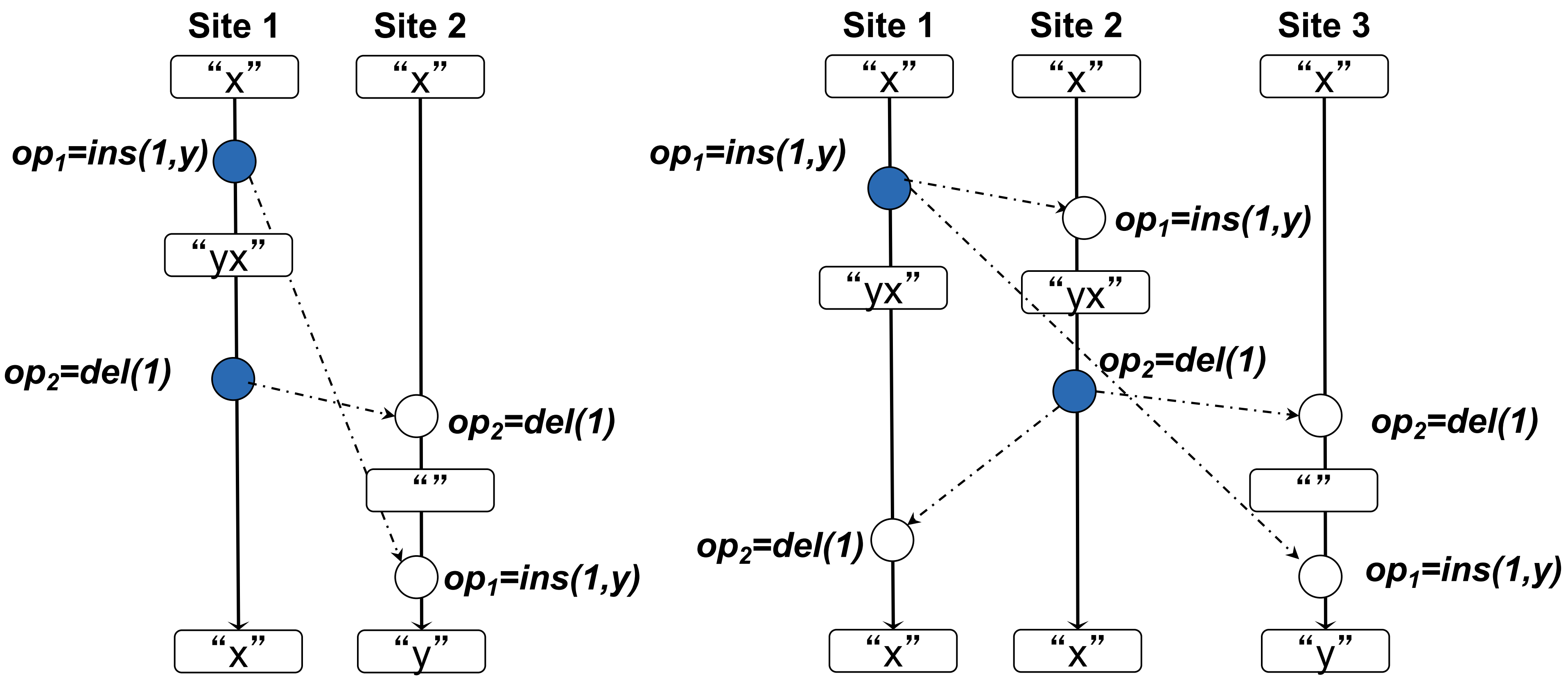
Operational transformation

- 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

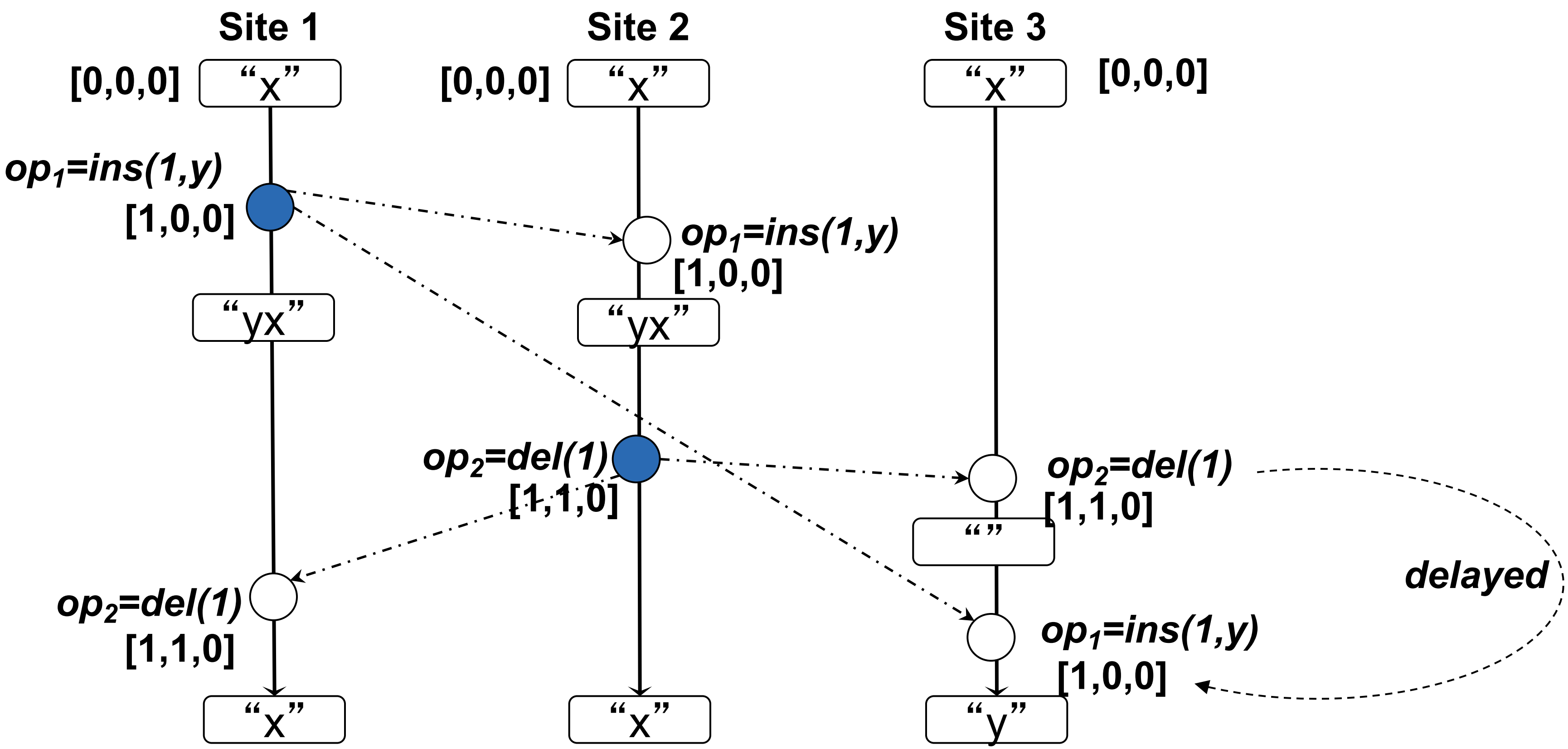
Operational transformation

- 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
"x"
"x"
op₁=ins(1,y)
"yx"
op₂=del(1)
op₂=del(1)
""
op₁=ins(1,y)
"x"
"y"
Site 1
Site 2
Site 3
"x"
"x"
"x"
op₁=ins(1,y)
op₁=ins(1,y)
"yx"
"yx"
op₂=del(1)
op₂=del(1)
""
op₂=del(1)
op₁=ins(1,y)
"x"
"x"
"y"

Causality


Site 1
Site 2
Site 3
[0,0,0]
"x"
[0,0,0]
"x"
"x"
[0,0,0]
op1=ins(1,y)
[1,0,0]
op1=ins(1,y)
[1,0,0]
"yx"
"yx"
op2=del(1)
[1,1,0]
op2=del(1)
[1,1,0]
" "
delayed
op2=del(1)
[1,1,0]
op1=ins(1,y)
[1,0,0]
"x"
"x"
"y"

Intention

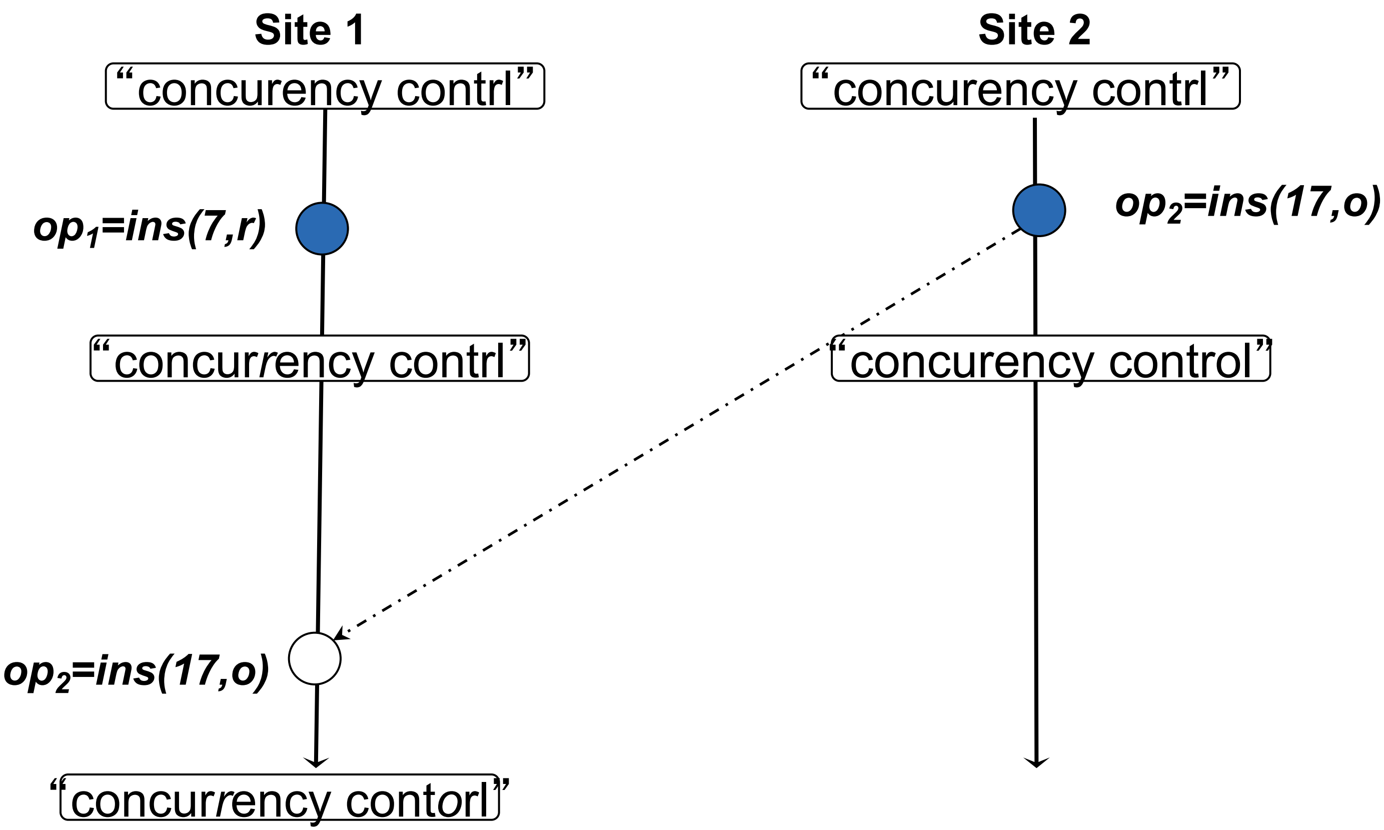
- Intention of an operation is the observed effect as result of its execution on its generation state
- Passing from initial state “ab” to final state “aXb” we can observe:
  - ins(2,X)
  - ins(a<X<b)
  - ins(a<X)
  - ins(X<b)

Preserving user intention (*)

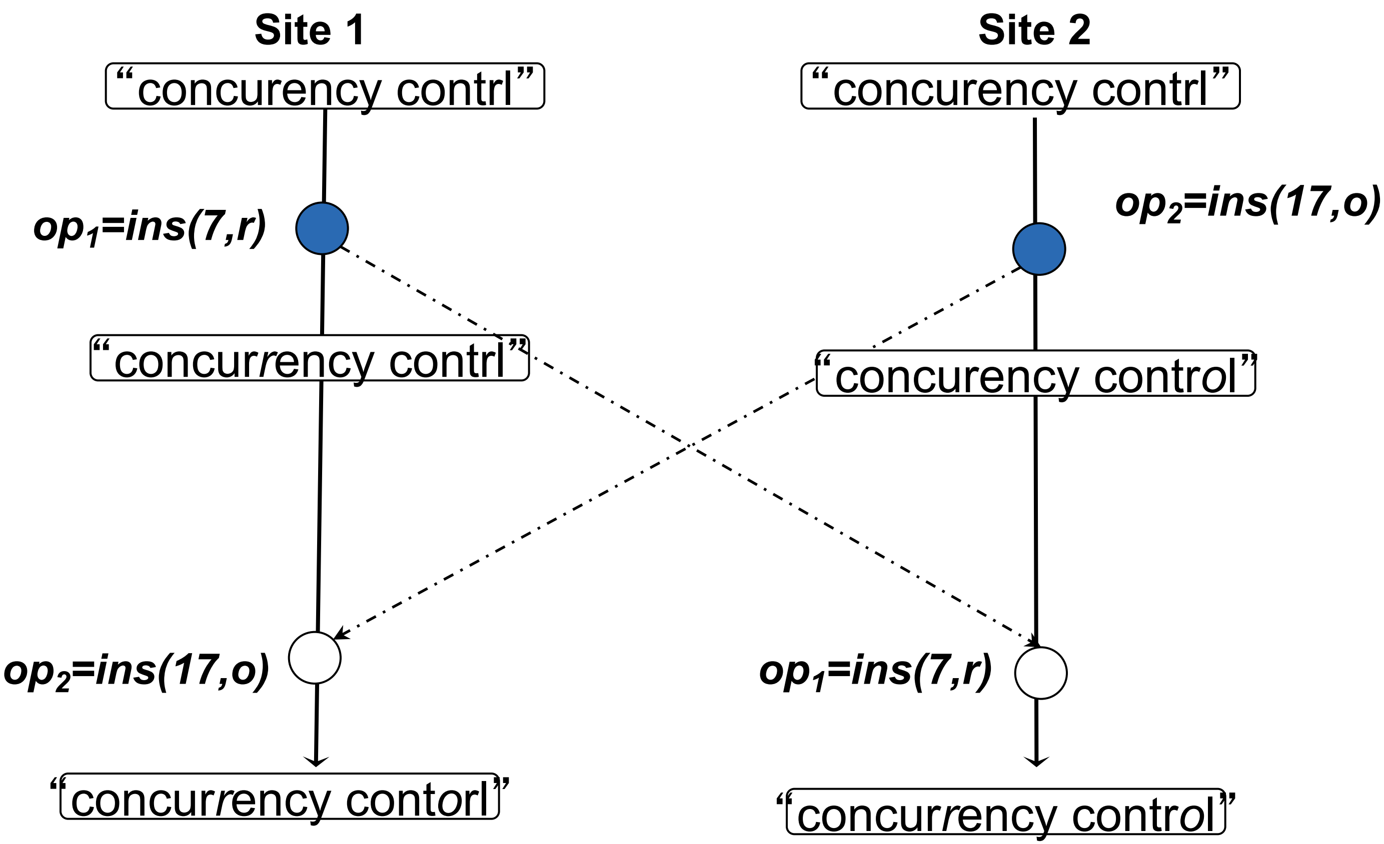
- For any operation op, the effects of executing op at all sites should be the same as the intention of op
- The effect of executing *O* does not change the effects of independent operations.

(*) Chengzheng Sun, Xiaohua Jia, Yanchun Zhang, Yun Yang, and David Chen. Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *ACM Transactions on Computer-Human Interaction*, 5(1):63−108, March 1998.

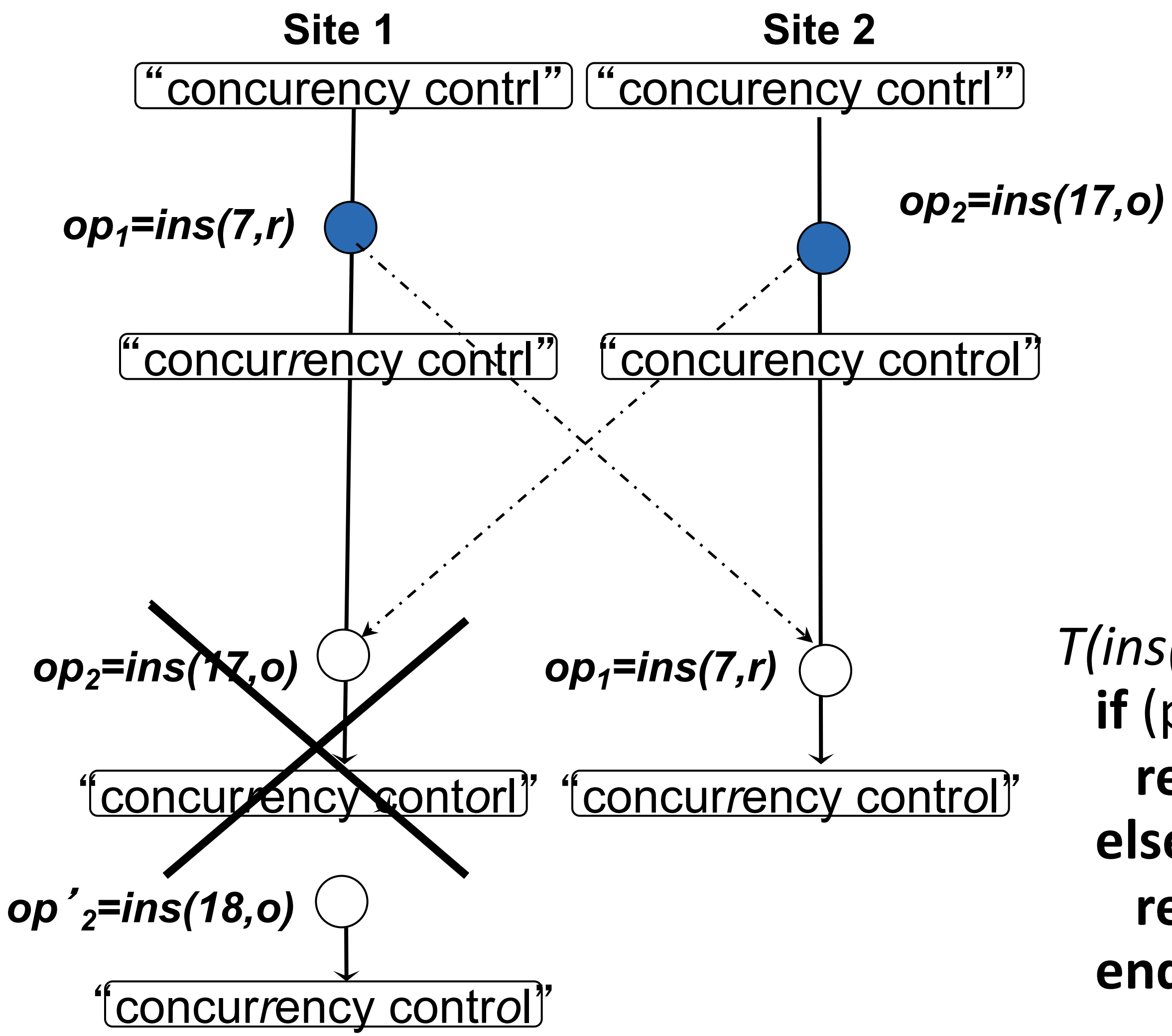
Intention violation



Intention violation + divergence



Intention preservation



$T(ins(p_1,c_1), ins(p_2,c_2))$:-

if ($p_1<p_2$)

  return $ins(p_1,c_1)$

else

  return $ins(p_1+1,c_1)$

endif

Example transformation functions

$T(ins(p_1,c_1), ins(p_2,c_2))$:-

if ($p_1 < p_2$) **return** $ins(p_1,c_1)$
else return $ins(p_1+1,c_1)$

$T(ins(p_1,c_1), del(p_2))$:-

if ($p_1 \leq p_2$) **return** $ins(p_1,c_1)$
else return $ins(p_1-1,c_1)$
endif

$T(del(p_1), ins(p_2,c_2))$:-

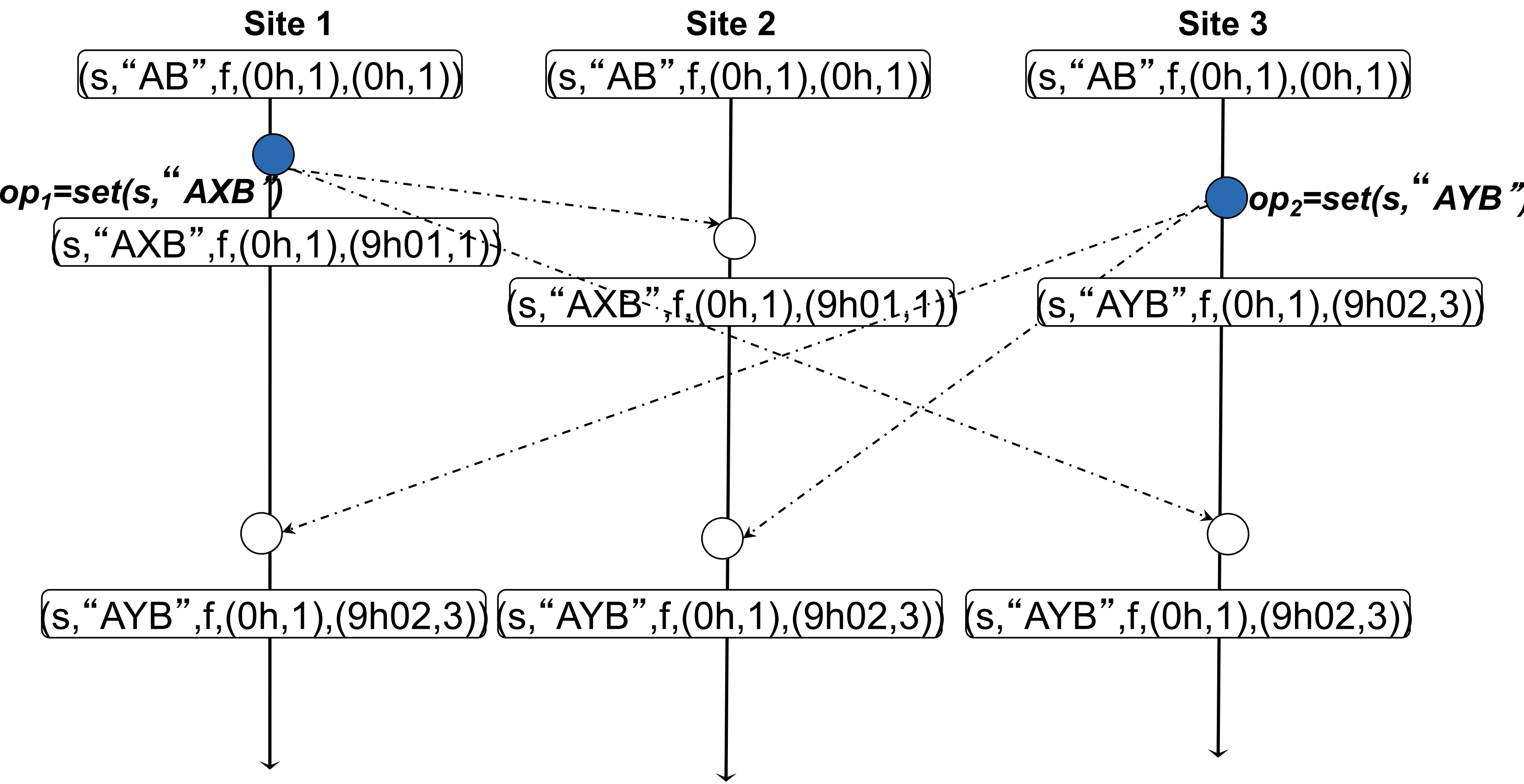
if ($p_1 < p_2$) **return** $del(p_1)$
else return $del(p_1+1)$

$T(del(p_1), del(p_2))$:-

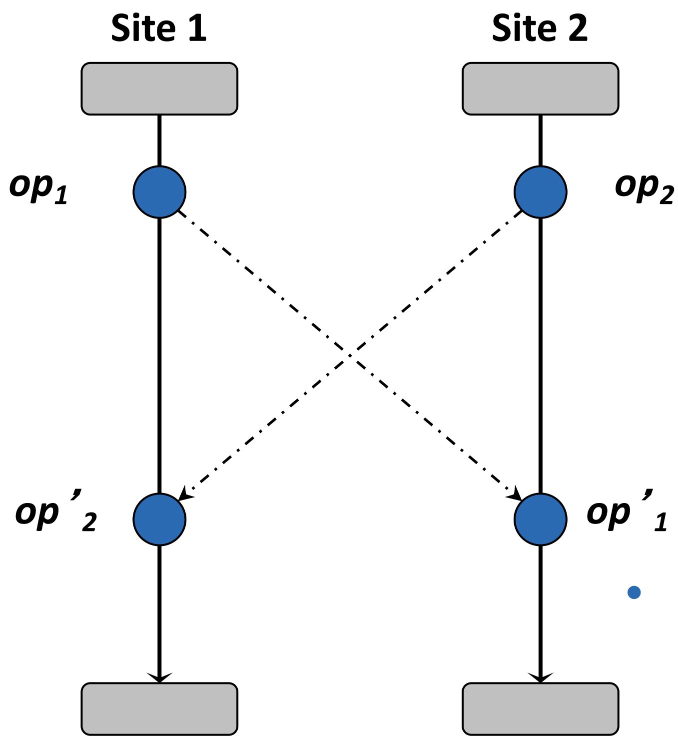
if ($p_1 < p_2$) **return** $del(p_1)$
else if ($p_1 > p_2$) **return** $del(p_1-1)$
else return *id()*

Convergence but no intention preservation

Thomas Write Rule



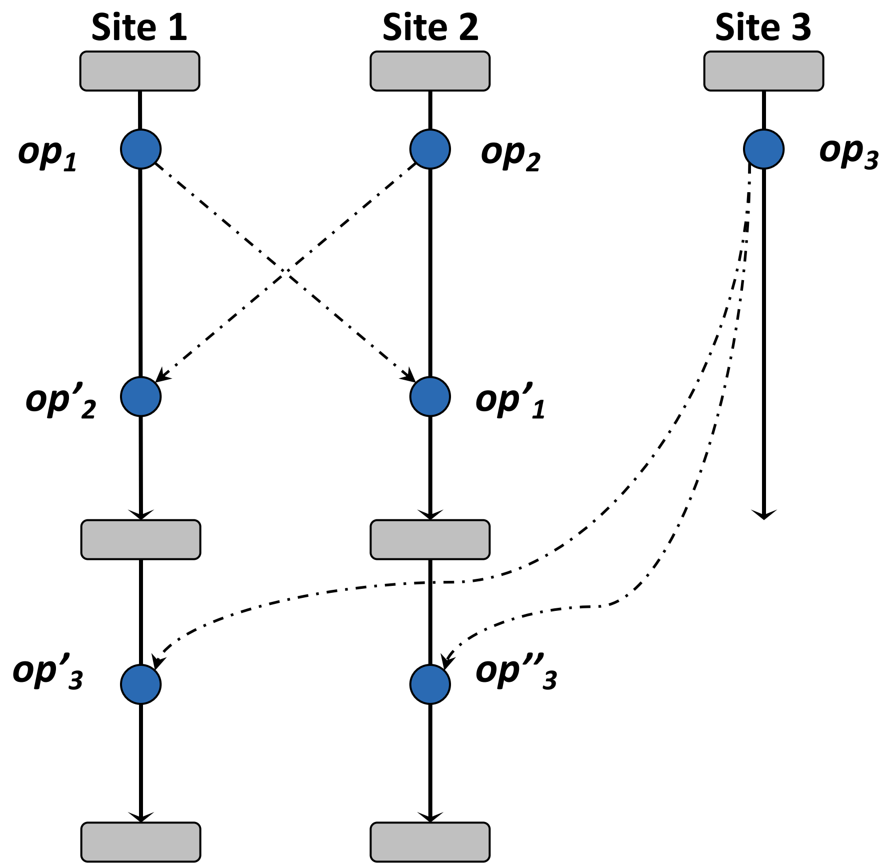
Convergence – TP1 property



- T(op_2: operation, op_1: operation) = op'_2
  - op_1 and op_2 concurrent, defined on a state S
  - op'_2 same effects as op_2, defined on $S.op_1$

[TP1] $op_1 \circ T(op_2, op_1) \equiv op_2 \circ T(op_1, op_2)$

Convergence – TP2 property

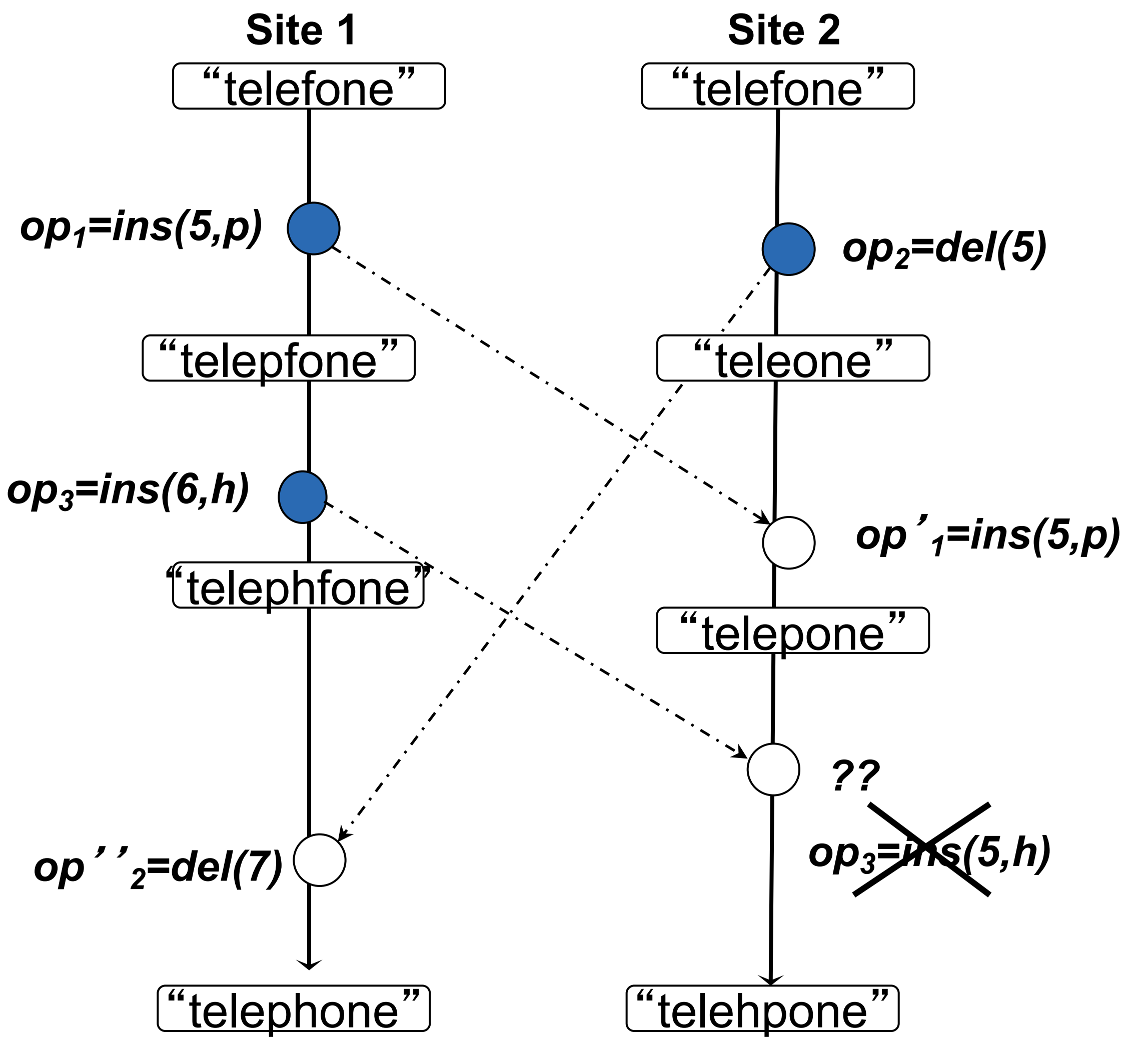


[TP2] $T(op_3, op_1 \circ T(op_2, op_1)) = T(op_3, op_2 \circ T(op_1, op_2))$

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



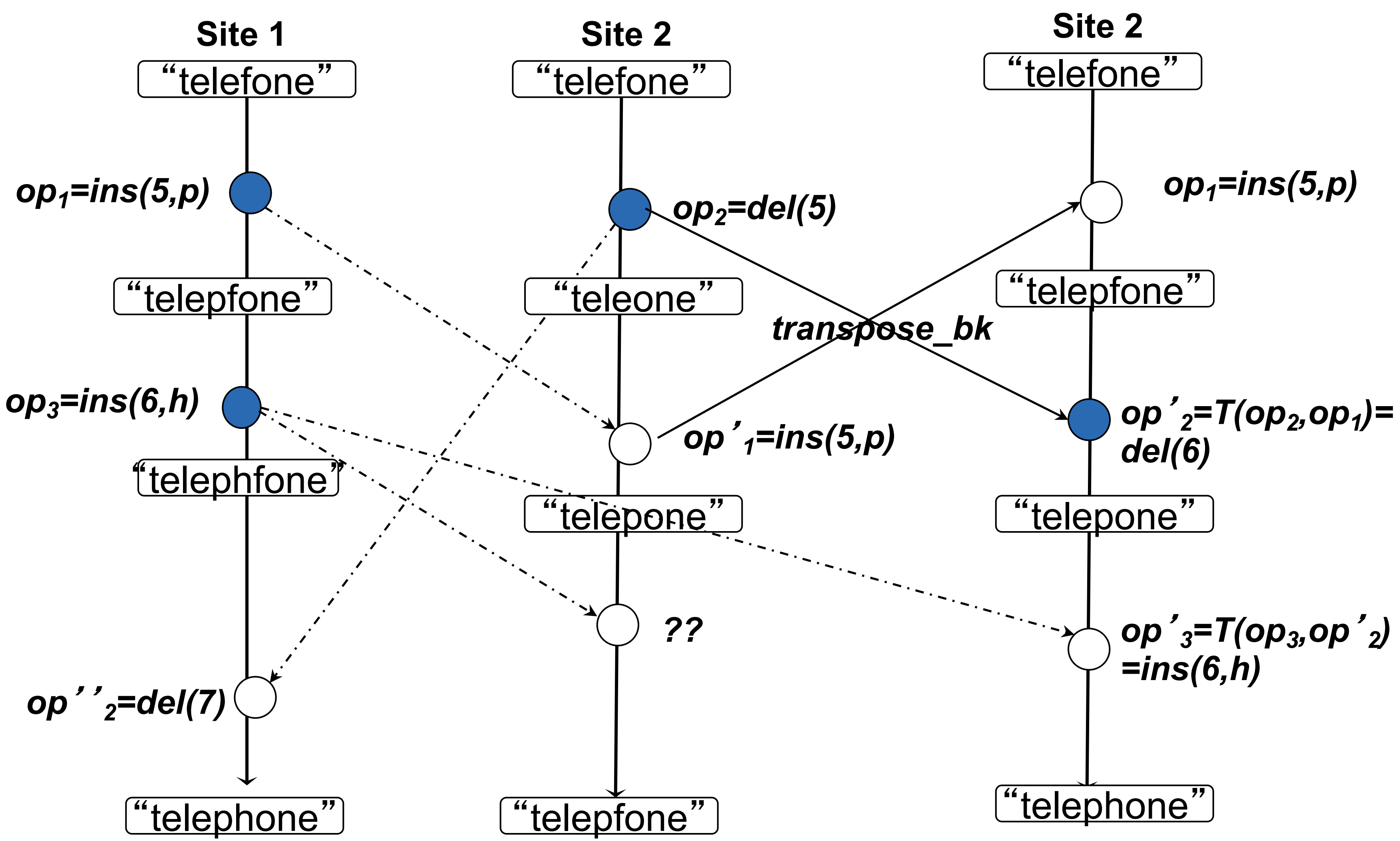
$op'_2 = T(op_2, op_1) = del(6)$

$op''_2 = T(op'_2, op_3) = del(7)$

$op'_1 = T(op_1, op_2) = ins(5)$

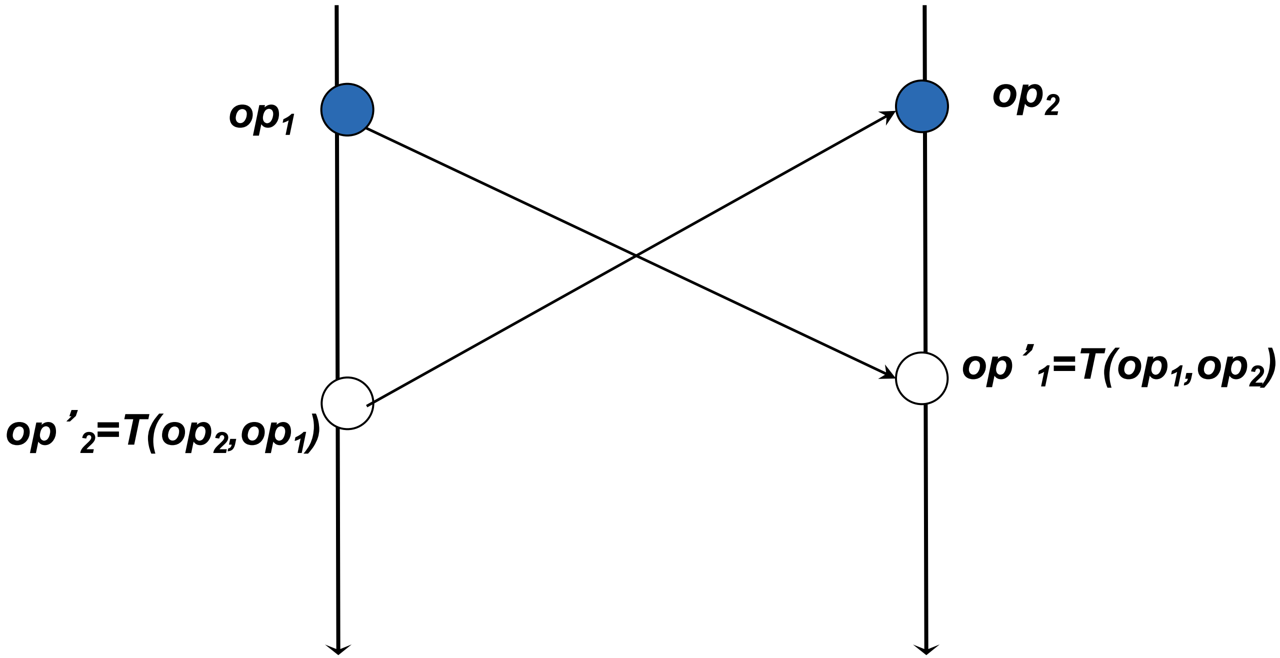
$T(op_3, op_2)$ not allowed to be performed !!!

Partial concurrency



Partial concurrency

- transpose_bk(op_1,op'_2)=(op_2,op'_1)
  - $op'_2=T(op_2,op_1)$
    Therefore $op_2=T^{-1}(op'_2,op_1)$
  - $op'_1=T(op_1,op_2)$



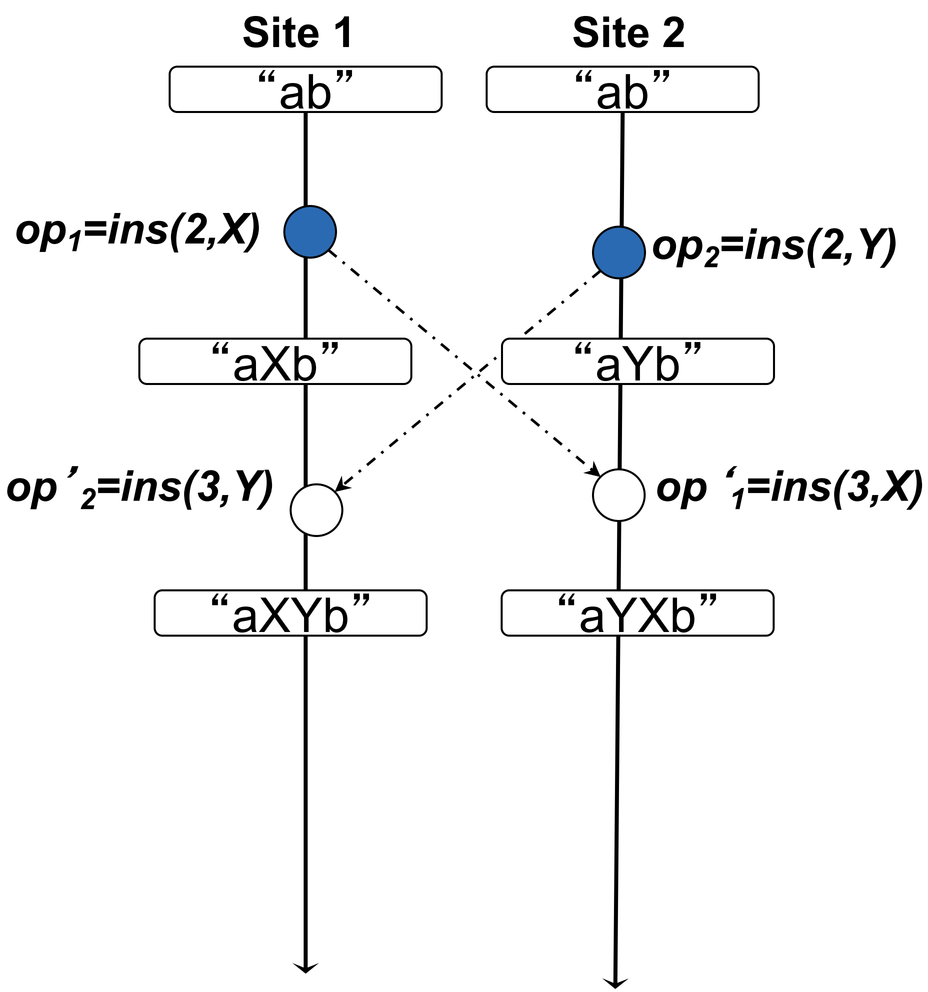
Example transformation functions

T(ins(p_1,c_1), ins(p_2,c_2)) :-
if (p1<p2) **return** *ins(p_1,c_1)*
else return *ins(p_1+1,c_1)*

T(ins(p_1,c_1), del(p_2)) :-
if ($p_1 \leq p_2$) **return** *ins(p_1,c_1)*
else return *ins(p_1-1,c_1)*
endif

T(del(p_1), ins(p_2,c_2)) :-
if ($p_1 < p_2$) **return** *del(p_1)*
else return *del(p_1+1)*

T(del(p_1), del(p_2)) :-
if ($p_1 < p_2$) **return** *del(p_1)*
else if ($p_1 > p_2$) **return** *del(p_1-1)*
else return *id()*



TP1 not respected !

Ressel transformation functions (*)

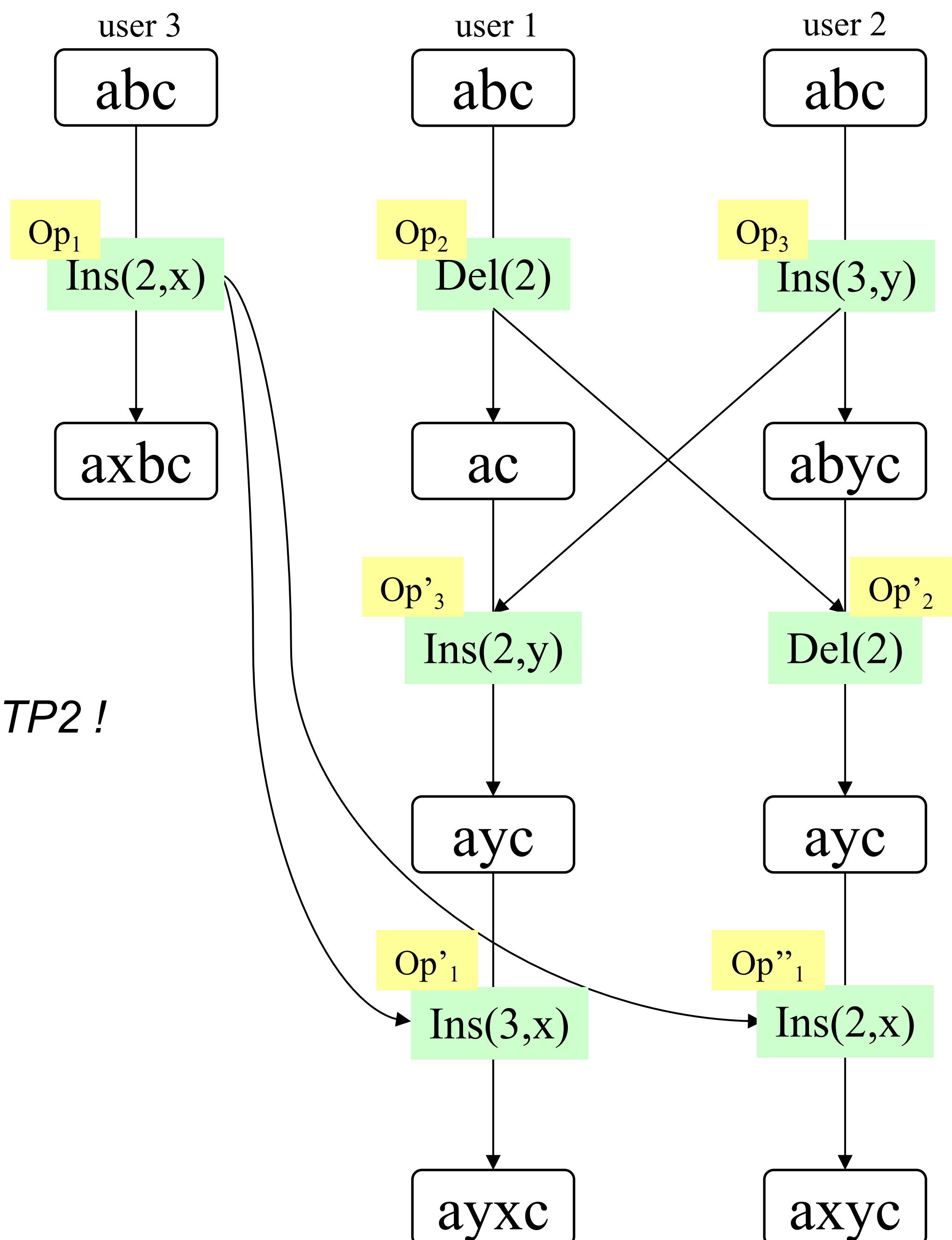
$T(ins(p_1,c_1,u_1), ins(p_2,c_2,u_2))$:-
  if $((p_1<p_2)$ or $(p_1=p_2$ and $u_1<u_2))$ **return** $ins(p_1,c_1,u_1)$
  else return $ins(p_1+1,c_1,u_1)$

$T(ins(p_1,c_1,u_1), del(p_2,u_2))$:-
  if $(p_1 \leq p_2)$ **return** $ins(p_1,c_1,u_1)$
  else return $ins(p_1-1,c_1,u_1)$
  endif

$T(del(p_1,u_1), ins(p_2,c_2,u_2))$:-
  if $(p_1<p_2)$ **return** $del(p_1,u_1)$
  else return $del(p_1+1,u_1)$

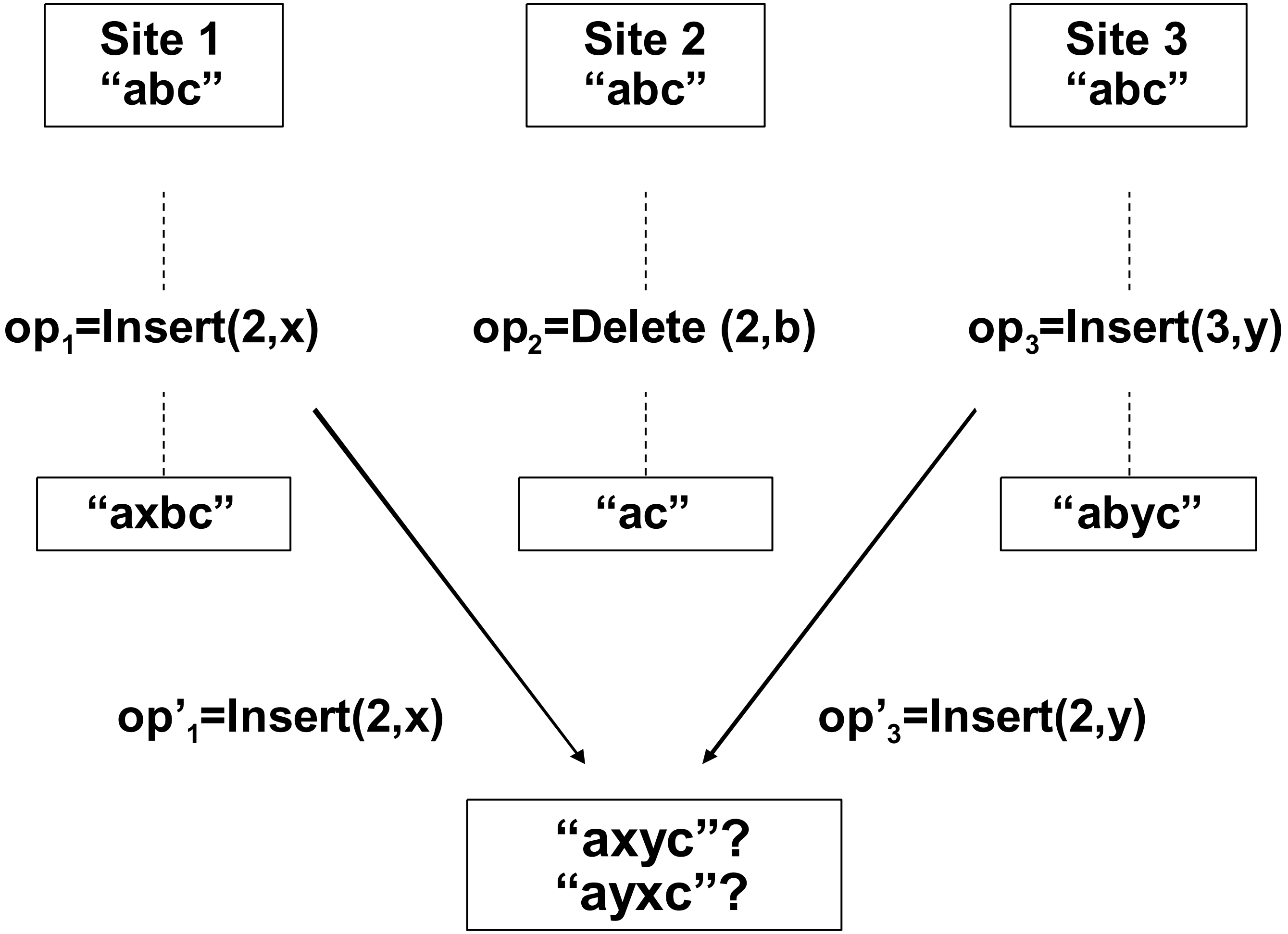
$T(del(p_1,u_1), del(p_2,u_2))$:-
  if $(p_1<p_2)$ **return** $del(p_1,u_1)$
  else if $(p_1>p_2)$ **return** $del(p_1-1,u_1)$
  else return *id()*

(*) Ressel, M., Nitsche-Ruhland, D. & Gunzenhauser, R. (1996), An integrating, transformation oriented approach to concurrency control and undo in group editors, Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW’ 96), Boston, Massachusetts, USA, pp. 288–297.



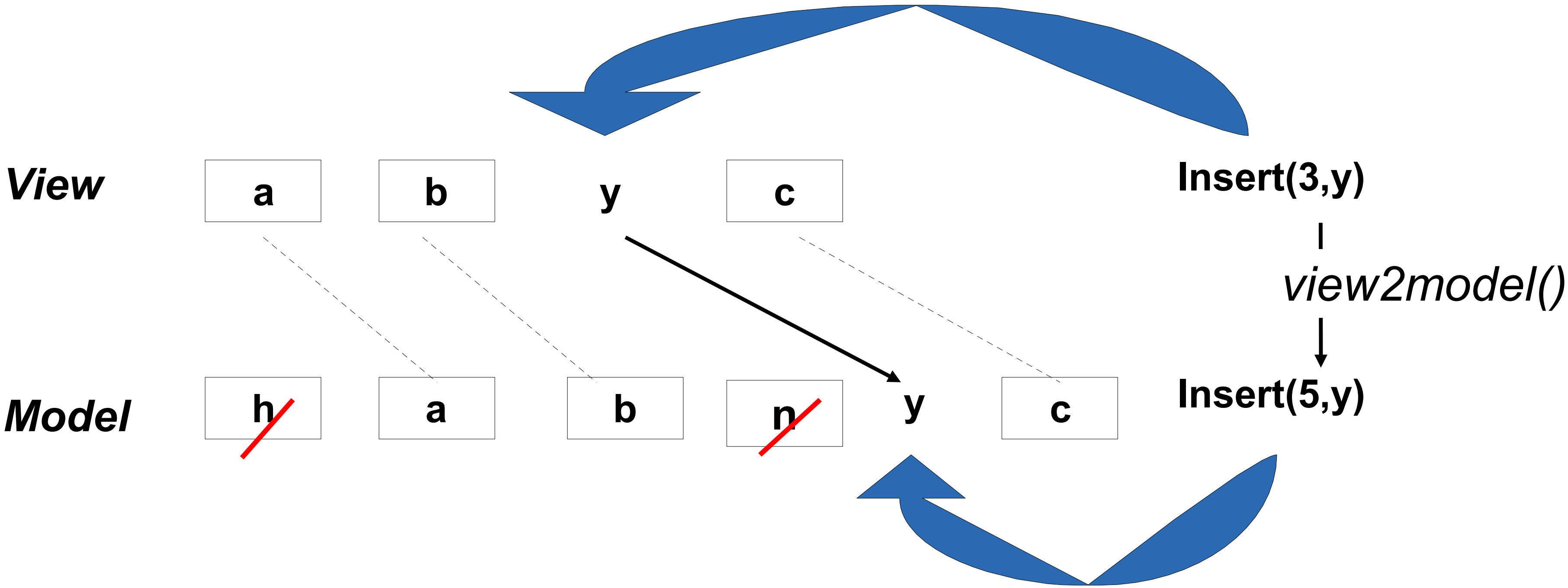
TP1 ok, but not TP2 !

False-tie problem



TTF (Tombstone Transformation Functions) Approach (*)

- Keep "tombstones" of deleted elements



(*) G. Oster, P. Urso, P. Molli, and A. Imine. Tombstone transformation functions for ensuring consistency in collaborative editing systems. In The Second International Conference on Collaborative Computing : Networking,Applications and Worksharing (CollaborateCom 2006), Atlanta, Georgia,USA, November 2006. IEEE Press.

Tombstone Transformation Functions

- $T(insert(p_1,el_1,sid_1), insert(p_2,el_2,sid_2))\{$

  $if(p_1<p_2)$ return $insert(p_1,el_1,sid_1)$

  else $if(p_1=p_2$ and $sid_1<sid_2)$ return $insert(p_1,el_1,sid_1)$
  else return $insert(p_1+1,el_1,sid_1)$
  }
- $T(insert(p_1,el_1,sid_1), delete(p_2,el_2,sid_2))\{$

  return $insert(p_1,el_1,sid_1)$

  }
- $T(delete(p_1,sid_1), insert(p_2,sid_2))\{$

  $if(p_1<p_2)$ return $delete(p_1,sid_1)$

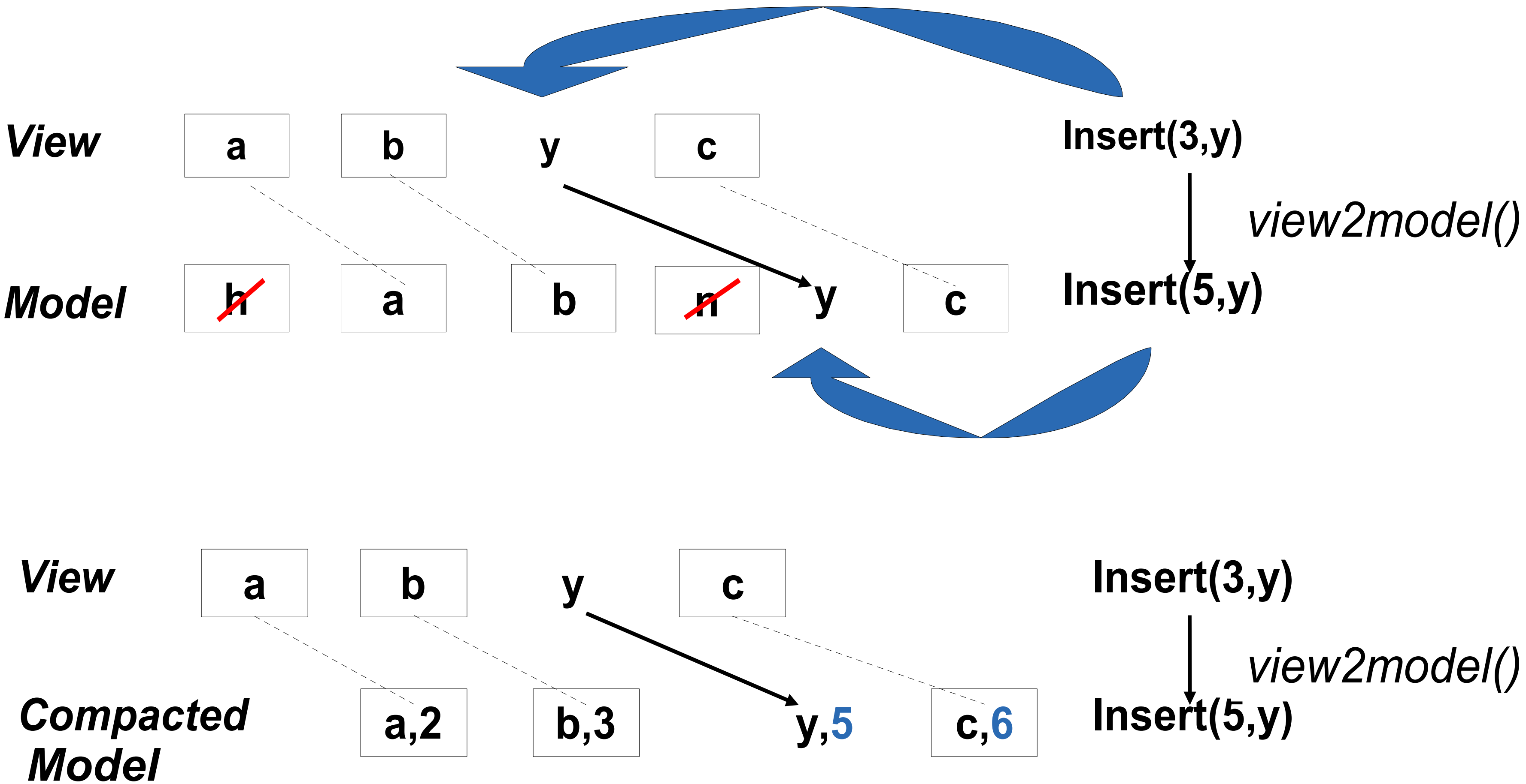
  else return $delete(p_1+1,sid_1)$

  }
- $T(delete(p_1,sid_1), delete(p_2,sid_2))\{$

  return $delete(p_1,sid_1)$

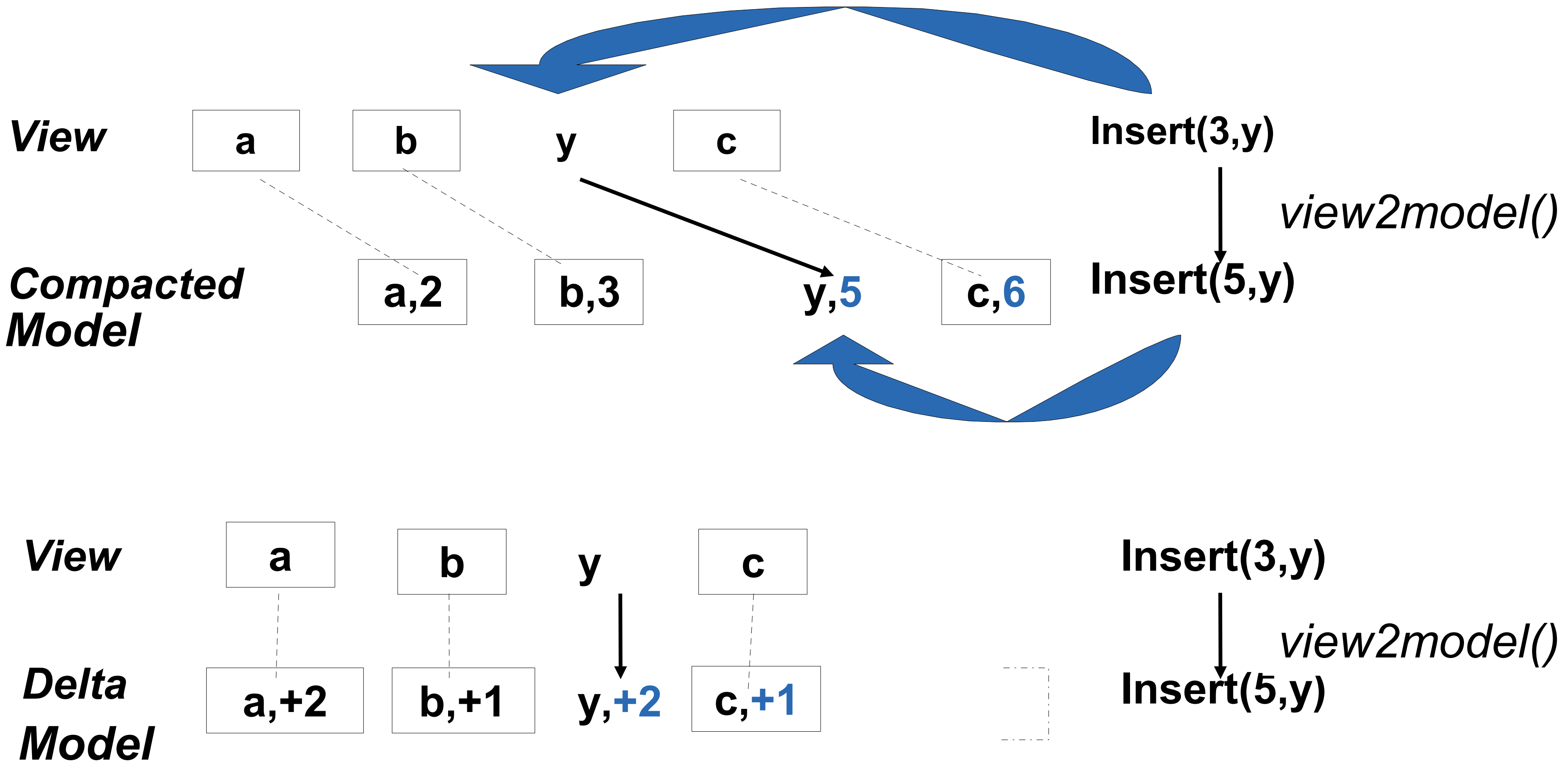
  }

Compacted storage model



- Compacted model = sequence of (character, abs_pos)

Delta storage model

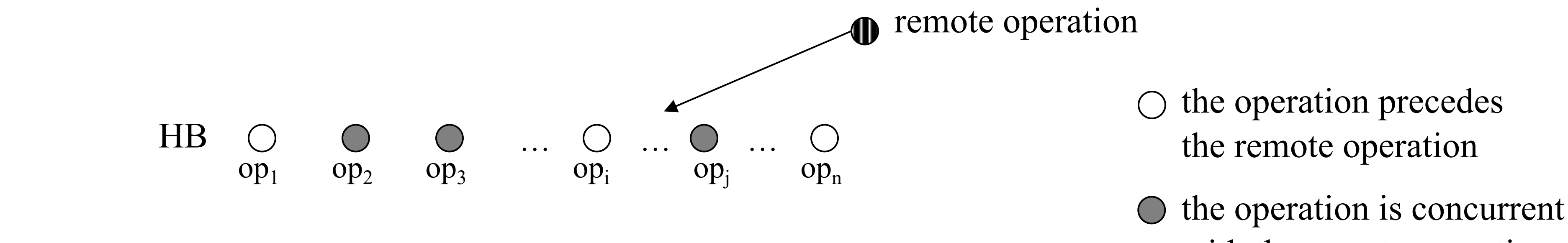


- Delta model = sequence of (character, offset)

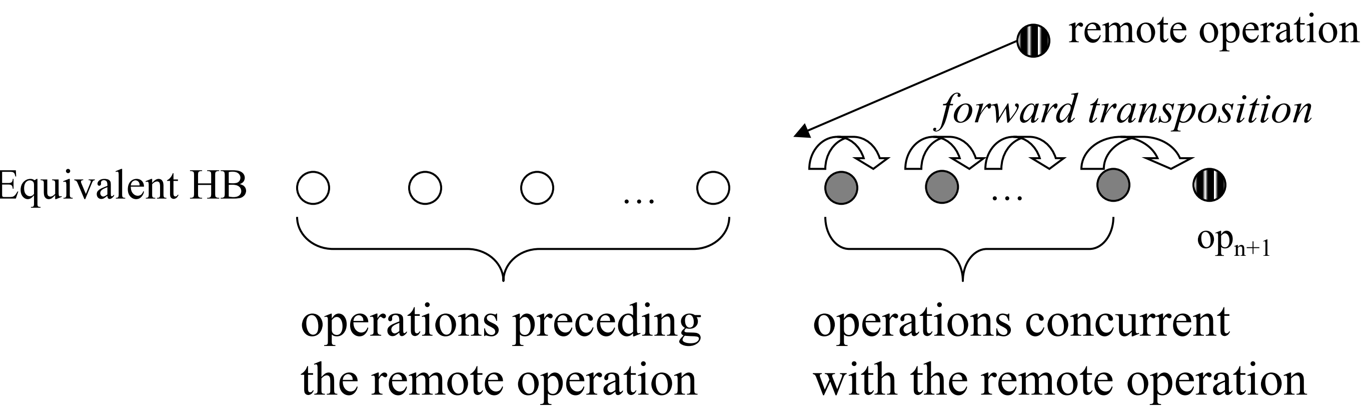
Models comparison

- 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

SOCT2 algorithm(*)
General control algorithm



a) The initial history buffer



b) Principle of integration

(*) M. Suleiman, M. Cart, and J. Ferrié. Serialization of concurrent operations in a distributed collaborative environment. In Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work : (GROUP'97), pages 435.445, Phoenix, Arizona, United States, November 1997.

GOT algorithm(*)

- Does not need to satisfy TP1 and TP2
- Requires a global serialisation order
  - Sum of state vector components
  - If equality, then priority on sites
- Requires undo/redo mechanism
- Undo/redo very costly

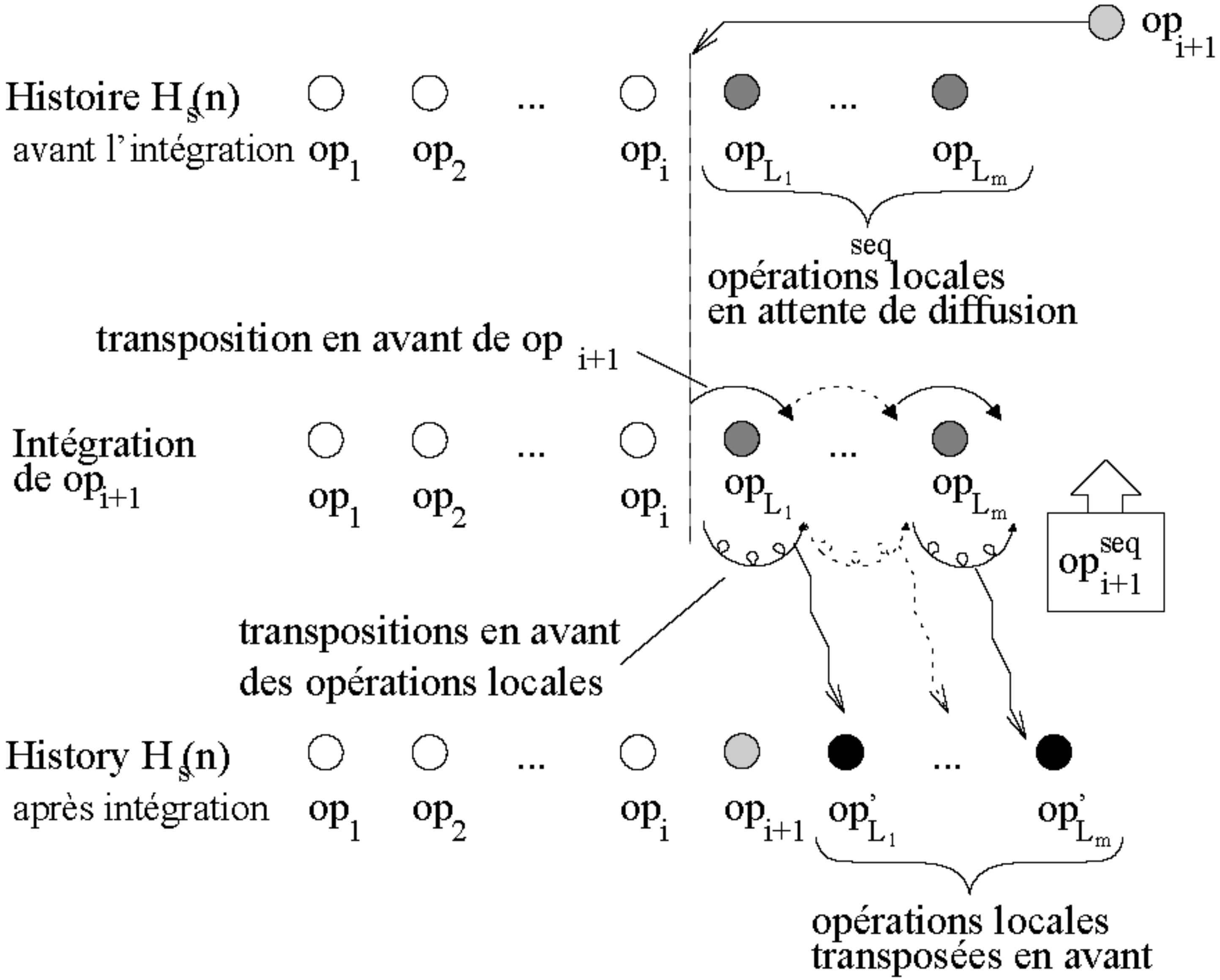
(*) Chengzheng Sun, Xiaohua Jia, Yanchun Zhang, Yun Yang, and David Chen. Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *ACM Transactions on Computer-Human Interaction*, 5(1):63−108, March 1998.

SOCT4 algorithm(*)

- Does not use undo/redo mechanism
- Eliminates TP2, but requires TP1
- Does not need state vectors
- Global order of operations according to timestamps generated by a sequencer
- Local operations executed immediately
- Assigns a timestamp to the operation and transmits it to the other sites
- Defers broadcast until all preceding operations were executed
- Transformations performed by each site

(*) Nicolas Vidot, Michèle Cart, Jean Ferrié, and Maher Suleiman. Copies convergence in a distributed real-time collaborative environment. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW'00)*, page 171−180, Philadelphia, Pennsylvania,USA, December 2000.

SOCT4 algorithm

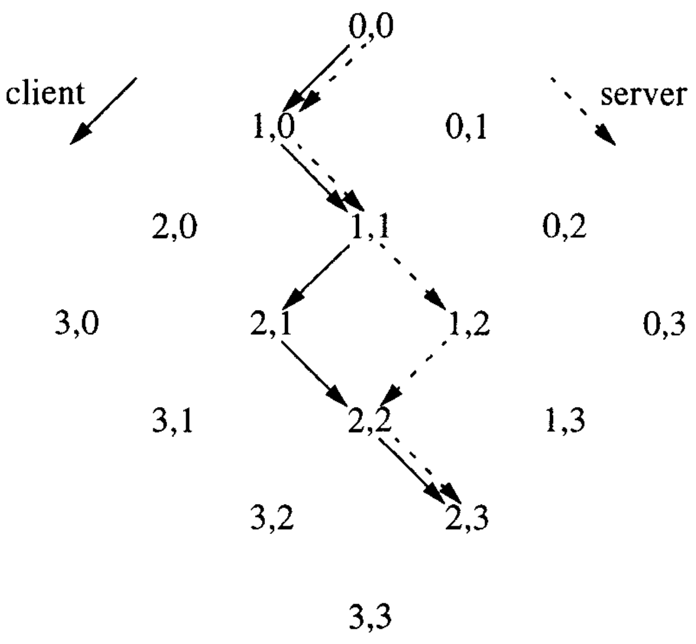


Jupiter algorithm(*)

- Used in Google Drive
- Requires a central server
- Eliminates TP2, but requires TP1
- Does not need state vectors
- Transformations done on the server + client side

(*) David A. Nichols, Pavel Curtis, Michael Dixon, and John Lamping. High-latency, low-bandwidth windowing in the jupiter collaboration system. In *Proceedings of the 8th annual ACM symposium on User interface and software technology (UIST ’95)*, page 111−120, Pittsburgh, Pennsylvania, USA, 1995.

Jupiter algorithm



- xform(c,s)={c’ ,s’ }
- xform(del x, del y)=

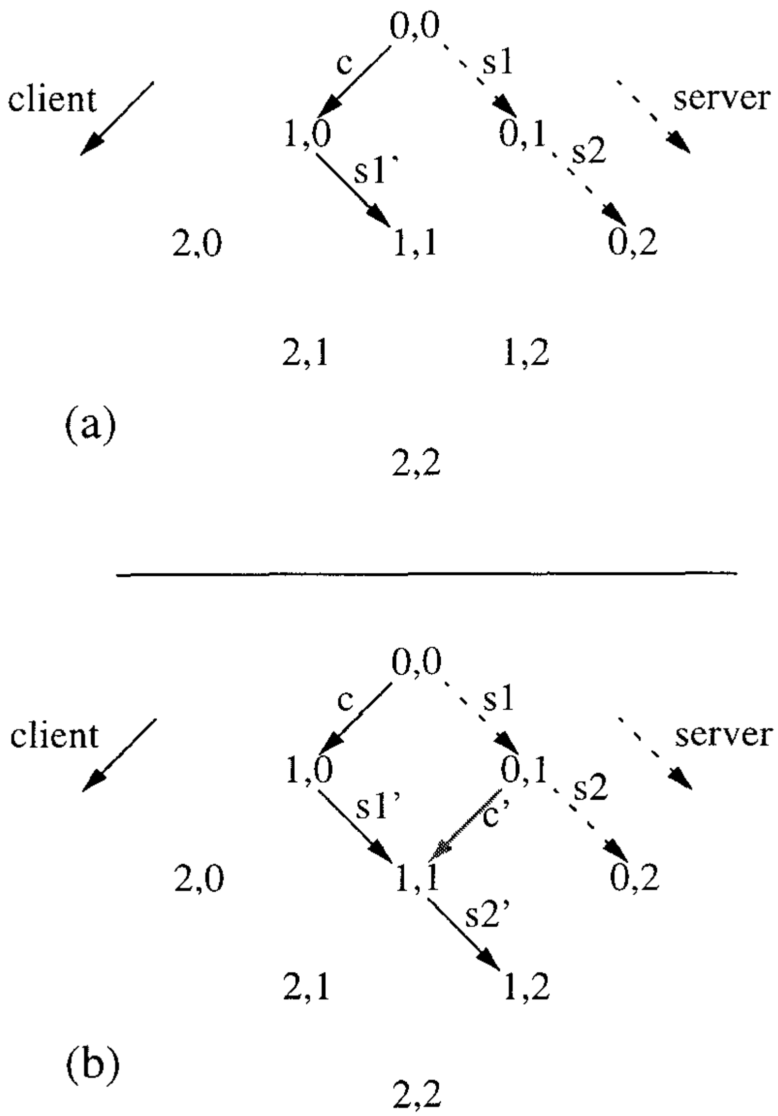
  {del x-1, del y} if x>y

  {del x, del y-1} if x<y

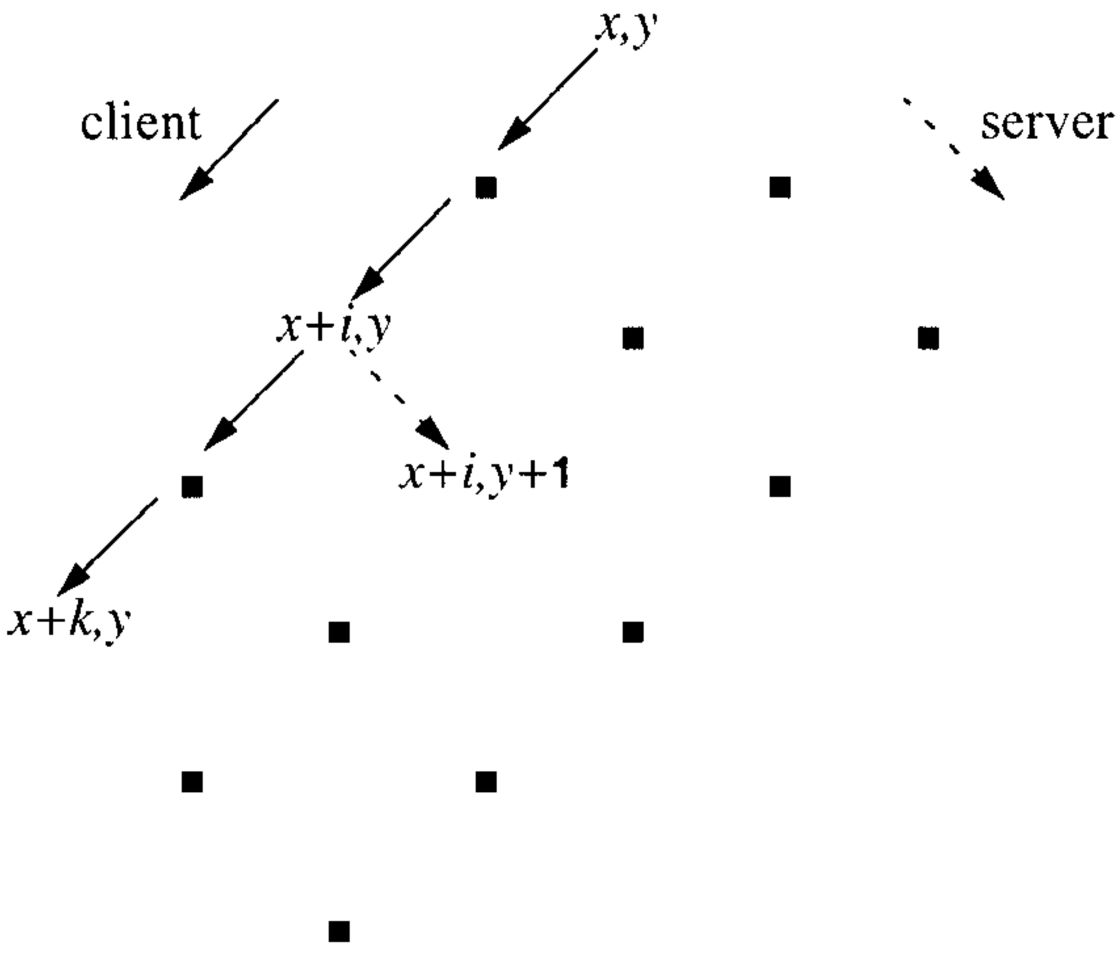
  {no-op, no-op} if x=y

…

Jupiter algorithm

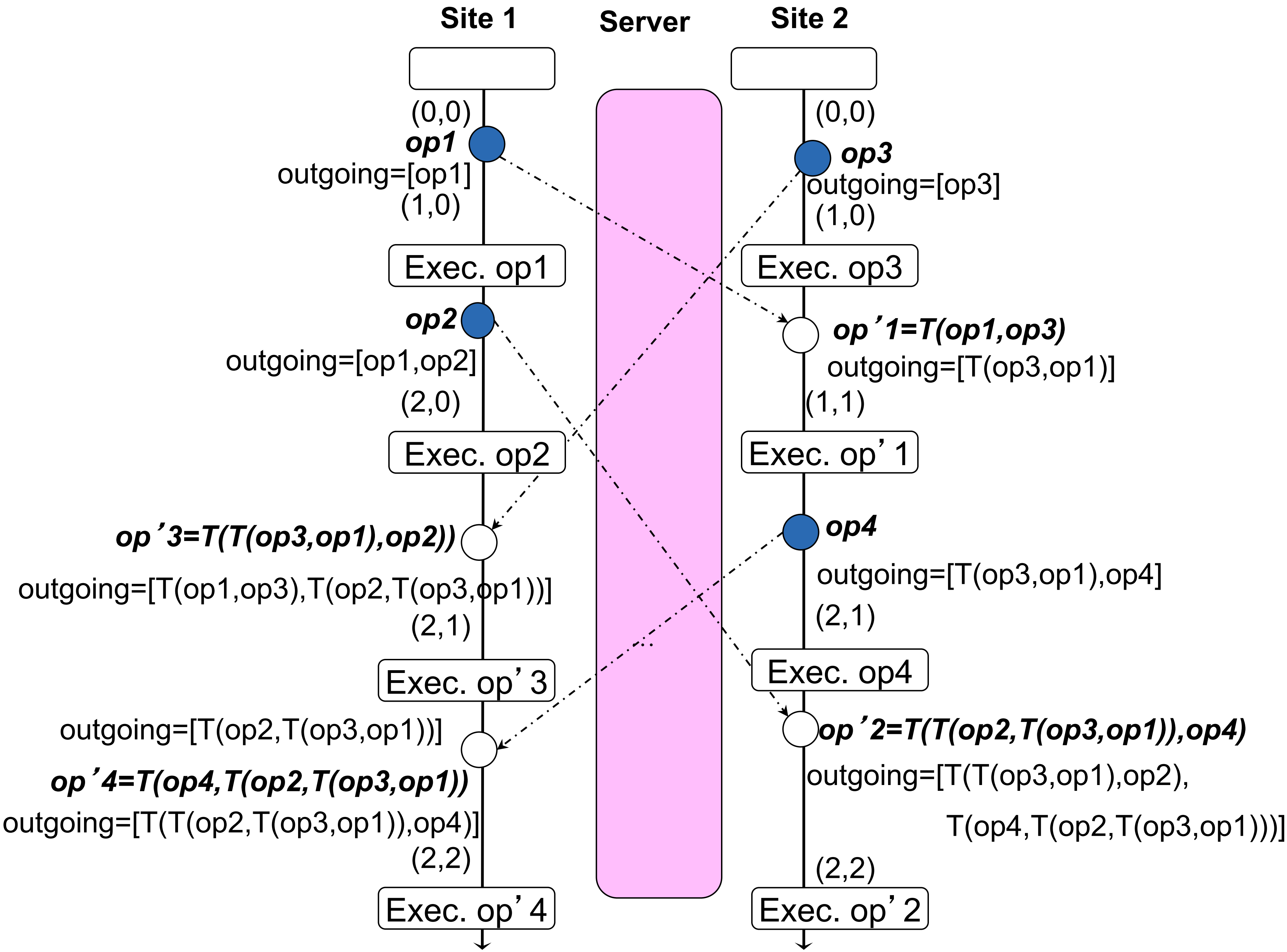


Jupiter algorithm



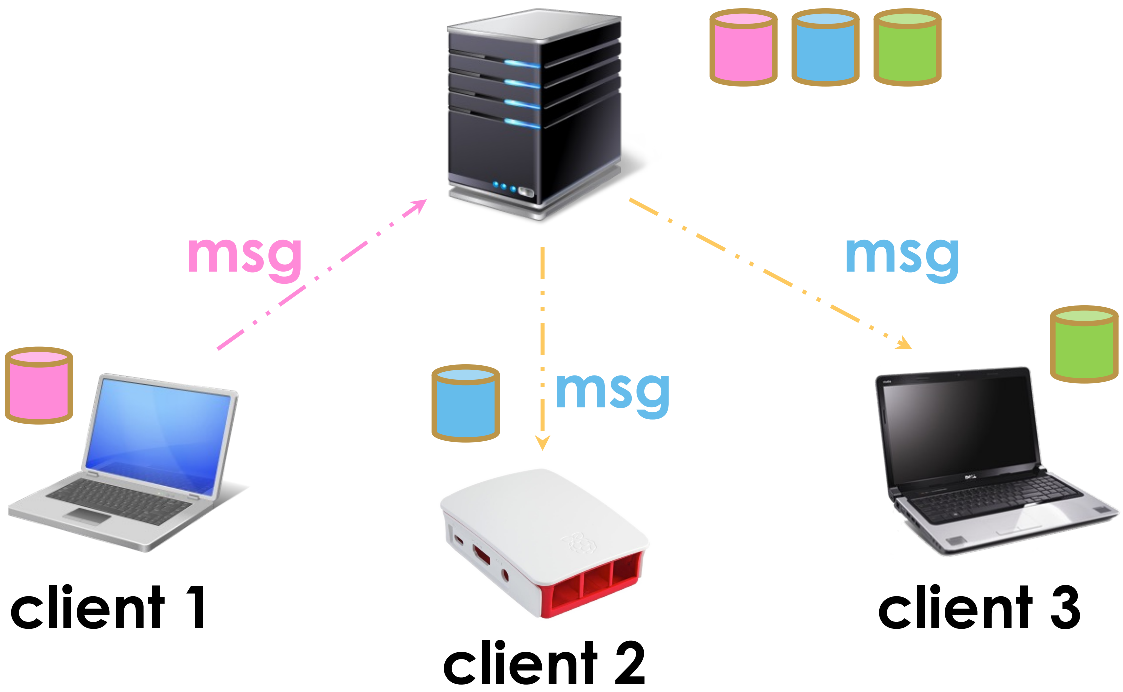
Jupiter algorithm 2 sites

```
int myMsgs = 0;  /* number of messages generated */
int otherMsgs = 0;  /* number of messages received */
queue outgoing = {};

Generate(op) {
   apply op locally;
   send(op, myMsgs, otherMsgs);
   add (op, myMsgs) to outgoing;
   myMsgs = myMsgs + 1;
}

Receive(msg) {
   /* Discard acknowledged messages. */
   for m in (outgoing) {
      if (m.myMsgs < msg.otherMsgs)
         remove m from outgoing
   }
   /* ASSERT msg.myMsgs == otherMsgs. */
   for i in [1..length(outgoing)] {
      /* Transform new message and the ones in
         the queue. */
      {msg, outgoing[i]} = xform(msg, outgoing[i]);
   }
   apply msg.op locally;
   otherMsgs = otherMsgs + 1;
}
```



Site 1
Server
Site 2
(0,0)
op1
outgoing=[op1]
(1,0)
Exec. op1
op2
outgoing=[op1,op2]
(2,0)
Exec. op2
op'3=T(T(op3,op1),op2))
outgoing=[T(op1,op3),T(op2,T(op3,op1))]
(2,1)
Exec. op'3
outgoing=[T(op2,T(op3,op1))]
op'4=T(op4,T(op2,T(op3,op1))
outgoing=[T(T(op2,T(op3,op1)),op4)]
(2,2)
Exec. op'4
(0,0)
op3
outgoing=[op3]
(1,0)
Exec. op3
op'1=T(op1,op3)
outgoing=[T(op3,op1)]
(1,1)
Exec. op'1
op4
outgoing=[T(op3,op1),op4]
(2,1)
Exec. op4
op'2=T(T(op2,T(op3,op1)),op4)
outgoing=[T(T(op3,op1),op2),
T(op4,T(op2,T(op3,op1)))]
(2,2)
Exec. op'2

Jupiter algorithm – generalisation n Clients



apply msg.op locally;

Algorithm changes at server side

apply msg.op locally;
for (c in client list) {
  if (c != client)
    send(c, msg);
}

Jupiter algorithm

- Requires a server that performs transformations
- Not suitable for P2P environments
- False tie scenario gives different results according to integration order