Class Exercises

Module: Distributed Systems

Chapter 5: Replication and Consistency (1/2)

Question 1.

Two primary reasons for replication: reliability and performance.

- Increasing reliability:
  - If a replica crashes, system can continue working by switching to other replicas.
  - Avoid corrupted data: can protect against a single, failing write operation.
- Improving performance:
  - Important for distributed systems over large geographical areas.
  - Divide the work over a number of servers.
  - Place data in the proximity of clients

Question 2.

P1	*W(x)a*						
P2			*W(x)b*				
P3		*R(x)a*	*W(x)c*				
P4					*R(x)a*	*R(x)b*	*R(x)c*
P5				*R(x)b*	*R(x)a*		*R(x)c*

a) As we can see, the operation W2(x)b is potentially dependent of process W1(x)a since the value of b may be calculated from the value read by R3(x)a. Because of that, the write processes of a and b are potentially causally related, hence all processes must view them in the same order. However, in P5, R5(x)b comes before R5(x)a. Thus, this model does not satisfy the causal consistency model.

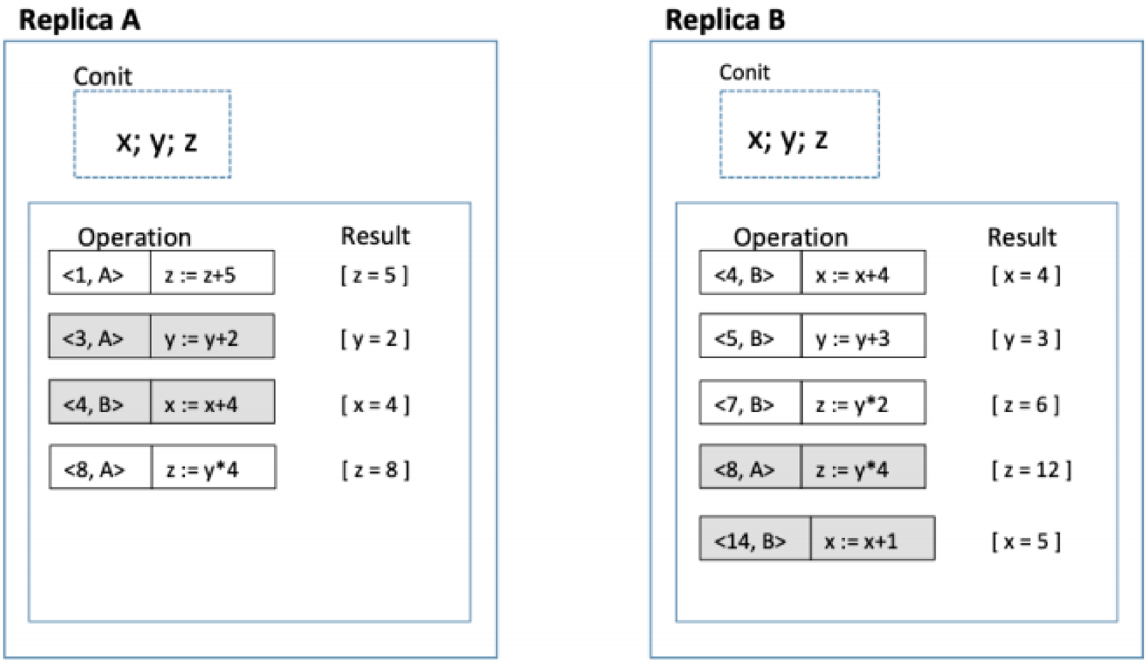
b) In order for this model to be sequentially consistent, any execution is the same as if all read/write ops were executed in some global ordering, and the operation of each client process appear in the order specified by its program. However, as can be seen, the W(x)b event happens after W(x)a on a global ordering scale but in P5, R(x)b happens before R(x)a. Because of that, this model is not sequentially consistent.

Question 3.

A conit or consistency specifies the data unit over which consistency is to be measured.

- **Large conit** (represents a lot of data) may bring replicas sooner in an inconsistent state.
  - Example: two replicas may differ in no more than one outstanding update
    - (a) Two updates lead to update propagation
    - (b) No update propagation is needed (yet)
  - This problem is particularly important when the data items contained in a conit are used completely independently, in which case they are said to falsely share the conit.
- **Small conit** may lead to excess overhead in managing conits since it will create a substantial amount of conit need to be handled.

Question 4.



Replica A

Conit

x; y; z

Operation		Result
<1, A>	z := z+5	[z = 5]
<3, A>	y := y+2	[y = 2]
<4, B>	x := x+4	[x = 4]
<8, A>	z := y*4	[z = 8]

Replica B

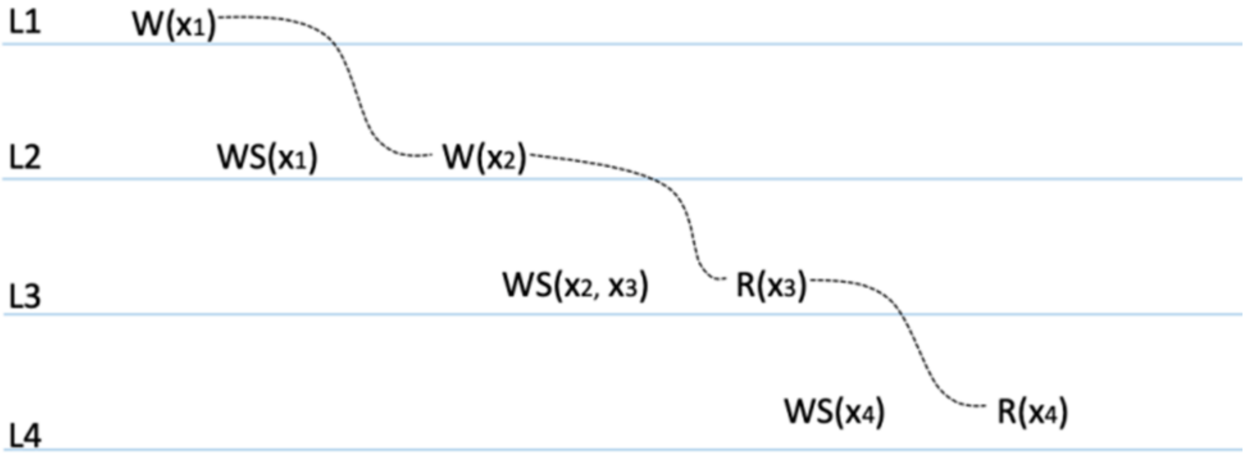
Conit

x; y; z

Operation		Result
<4, B>	x := x+4	[x = 4]
<5, B>	y := y+3	[y = 3]
<7, B>	z := y*2	[z = 6]
<8, A>	z := y*4	[z = 12]
<14, B>	x := x+1	[x = 5]

Vector clock A = (9, 4) Order deviation = 3 Numerical deviation = (3, 12)	Vector clock B = (8, 15) Order deviation = 4 Numerical deviation = (2, 12)

Question 5.



Regarding the monotonic-read consistency, the client performs R(x3) in L3 follow by the read operation R(x4) at L4. However, in L4 only the write operation WS(x4) have been performed. Because there are no guarantees that this set also contains all operations contained in WS(x2, x3). Hence, this system violates the monotonic-read consistency model.

Question 6.

a) For DNS service, the solution for updating data is server-based protocols (push-based) with multicasting, since this service require the server to be able to keep track of all client updates along with their name as well as IP address.

b) For WWW service, the solution for updating data is client-based protocols (pull-based) with unicasting, since the client in this service ask the web server to check and update the cached data items, and the server does not need to take care of the client.

Question 7.

Remote-write protocol	**Local-write protocol**
In both protocol, there are multiple copies of an item x (replication)	
Primary-based, all read and write operation are forwarded to a fixed server	Primary-based, a single copy is migrated between processes

Writes go through a primary copy and are forwarded to backups, reads may work locally (also well-suited for sequential consistency, since primary serializes updates)	Writes are performed after migrating primary to local server, updates are forwarded to other backups locations. Reads may proceed locally

Question 8.

a)

- Downloading game: Data-centric consistency
- Updating game: Data-centric consistency
- Chatting: Client-centric consistency

b)

- Regarding the read-write conflict:

  $9800 + 3000 = 12800 > 10000 \rightarrow N_W + N_R > N$

  So this system successfully avoided the read-write conflict
- Regarding the write-write conflict:

  $9800 > \frac{10000}{2} \rightarrow N_W > \frac{N}{2}$

  So this system also manage to avoid the write-write conflict