Lockless Transaction Isolation in Hyperledger Fabric

Published by IBM researcher
In 2019

Ramesh Adhikari, Graduate Research Assistant
School of Computer and Cyber Sciences, Augusta University

September, 2022

AUGUSTA UNIVERSITY

Hyperledger and Motivation for this Paper

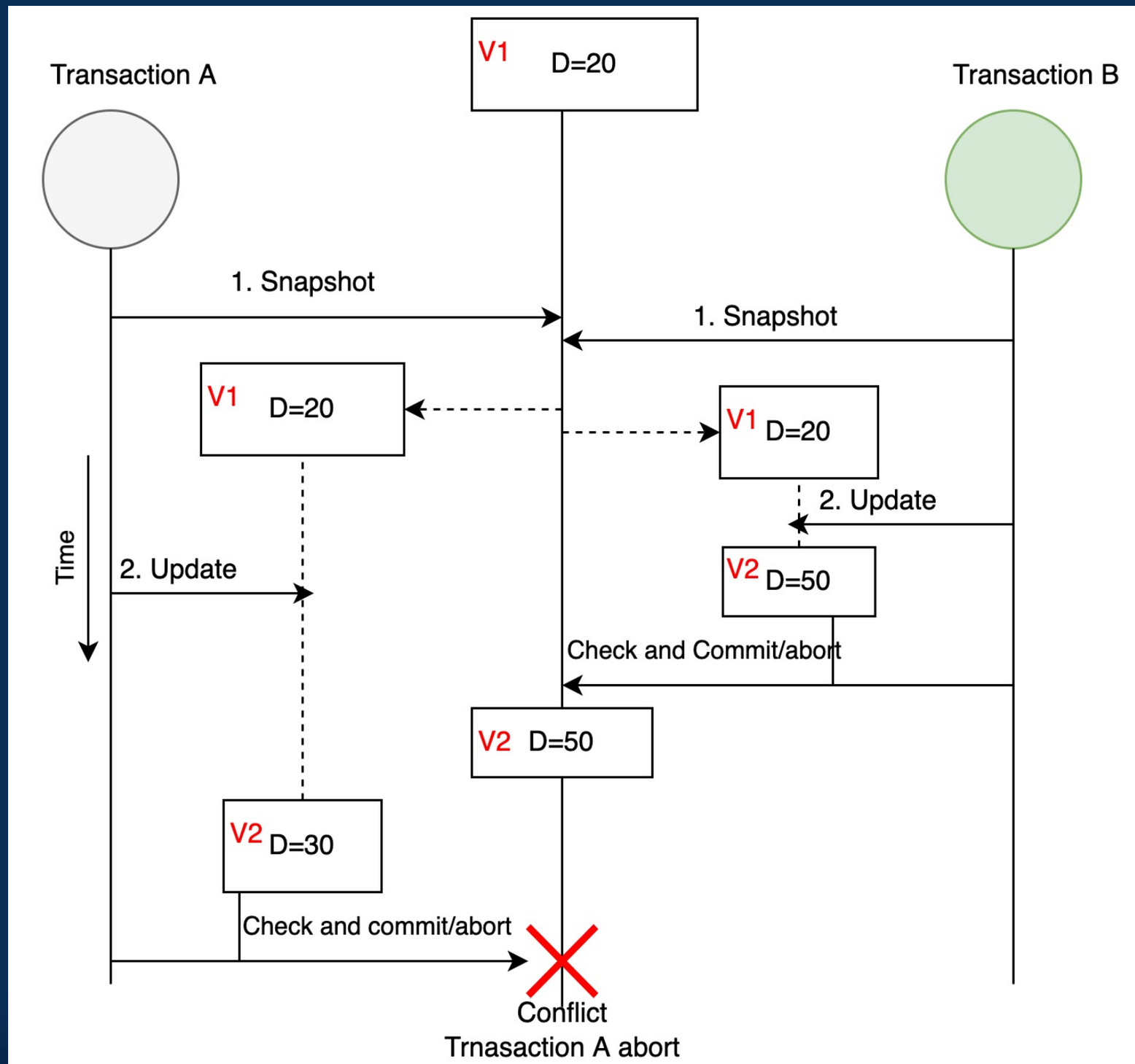
- Hyperledger Fabric platform is **an open source blockchain framework hosted by The Linux Foundation**
- In previous work, Transaction isolation is attained by locking the entire state database during simulation of transactions and database updates
- This lock is one of the major performance bottlenecks as observed by previous work

Main idea

- Provide lock-free approach for providing transaction isolation in Hyperledger Fabric
- Used the concept of multiversion concurrency control (MVCC); published in ACM; 1983
- Use savepoint as a boundary between transactions, and uses it to detect whether the simulated transaction is violating the transaction isolation
- When an isolation violation is detected, the transaction simulation is aborted

Multiversion concurrency control (MVCC)

- At any point in time, there will be multiple versions of the same data present for each concurrent process (transaction)
- But only one version is treated as the truth (let's call it the "master" data)



Proposed Algorithm

Algorithm 1 Simulation

```
1: savepoint ← ⊥

2: procedure BEGIN
3:     savepoint ← DB.GetSavePoint()

4: procedure GET(key)
5:     ⟨val,ver⟩ ← DB.GetState(key)
6:     if ⟨val,ver⟩ ≠ ⊥ then
7:         if ver > savepoint then return ERROR    ▷ abort
8:     return val
```

Simulation Example

Time	Tx	Op	Key	Val	Ver
1	tx1	read	savepoint	$\langle 100, 275 \rangle$	$\langle 100, 275 \rangle$
2	tx1	read	A	20	$\langle 100, 250 \rangle$
3	tx2	put	A	21	$\langle 101, 345 \rangle$
4	tx2	put	B	47	$\langle 101, 345 \rangle$
5	tx1	read	B	47	$\langle 101, 345 \rangle$
6	tx1	abort			

Example 1. Simulation abort

- tx1 performs a read of key A after which it performs a read of key B, while tx2 performs a put to key A and a put to key B
- read of key A returned before the commit of tx2, with an approved version, and the read of key B returned after, with a version definitely greater than the recorded savepoint
- So , in this proposed solution tx1 is aborted
- Only after the commit of the entire block, the savepoint is updated with the latest committed transaction number and block number.

Conclusion

- This work presented a new lock-free approach for providing transaction isolation in Hyperledger Fabric
- This solution outperforms the current implementation by 8.1x

Thank you!

AUGUSTA UNIVERSITY