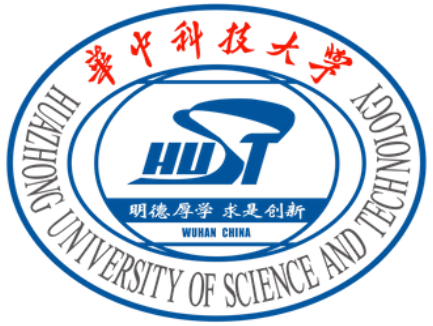


Motor: Enabling Multi-Versioning for Distributed Transactions on Disaggregated Memory

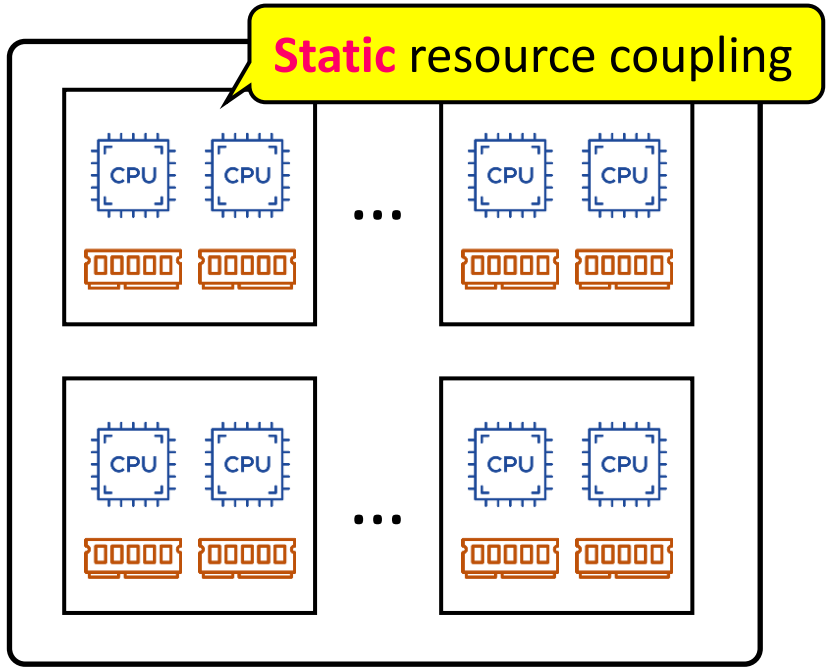
Ming Zhang, Yu Hua, Zhijun Yang

Huazhong University of Science and Technology, China

18th USENIX Symposium on Operating Systems Design and Implementation (**OSDI**), 2024

Insufficient Memory Utilization in Cloud

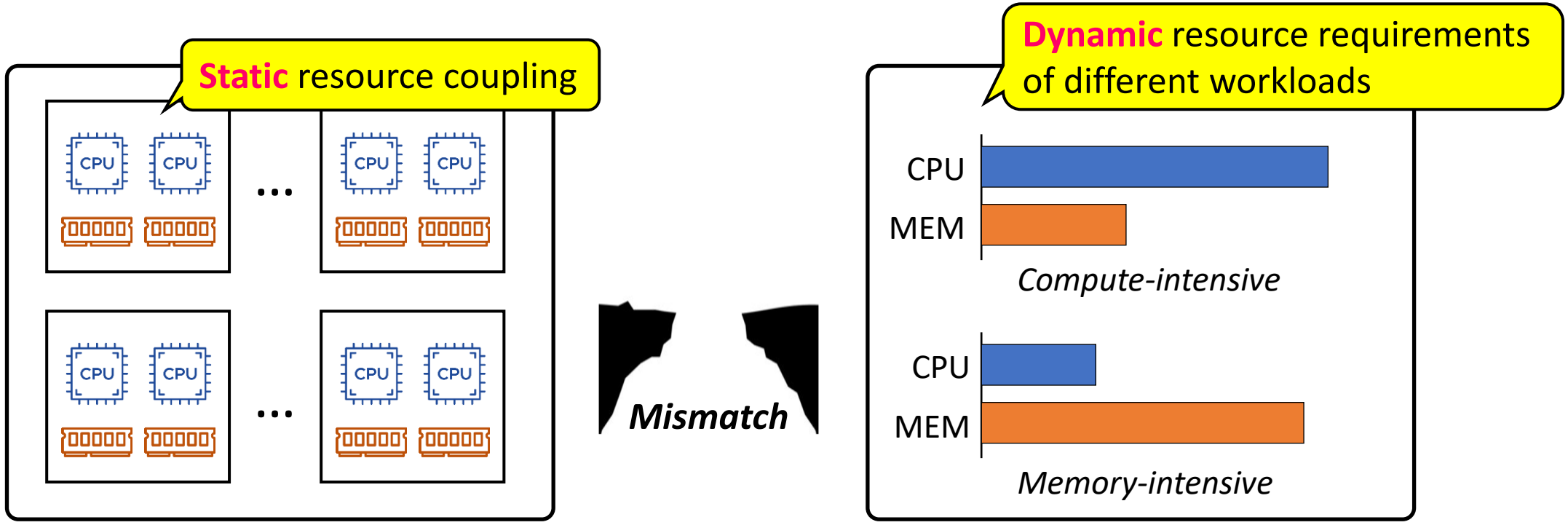
- About 20%~60% [1-4]
- One major reason: monolithic server



[1] MemTrade@SIGMETRICS'23, Borg@EuroSys'20, LegoOS@OSDI'18
[2] Google Production Cluster Trace. https://github.com/google/cluster-data
[3] Alibaba Production Cluster Trace. https://github.com/alibaba/clusterdata
[4] Snowflake Dataset. https://github.com/resource-disaggregation/snowset

Insufficient Memory Utilization in Cloud

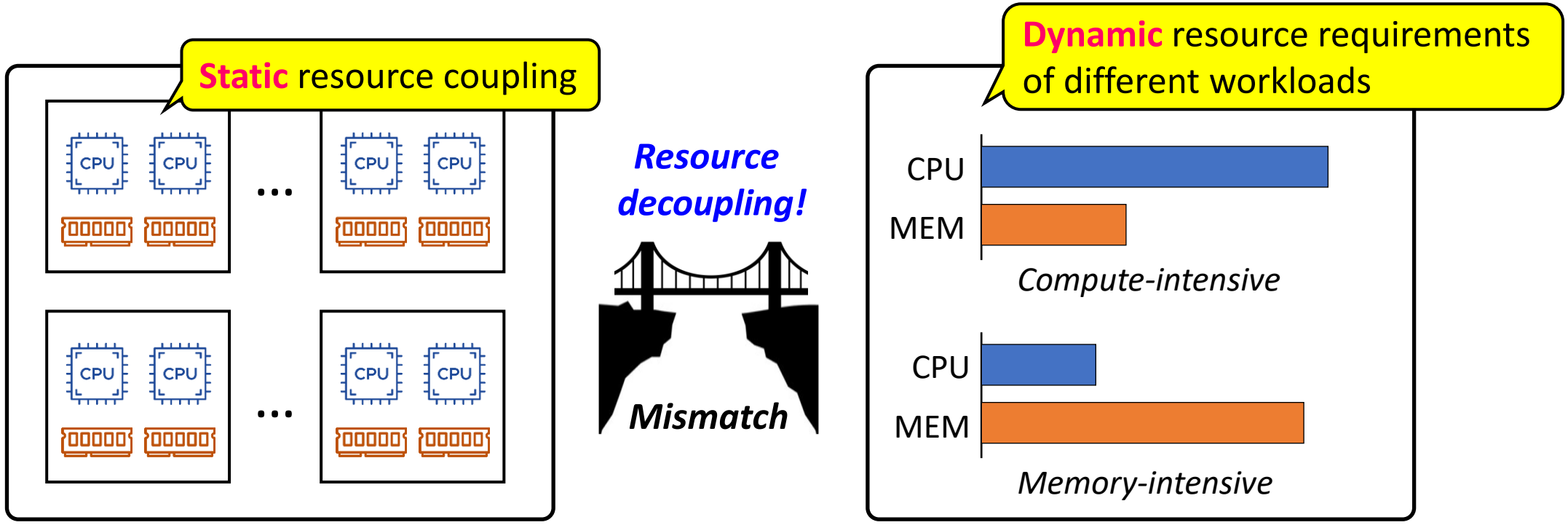
- About 20%~60% [1-4]
- One major reason: monolithic server



[1] MemTrade@SIGMETRICS'23, Borg@EuroSys'20, LegoOS@OSDI'18
[2] Google Production Cluster Trace. https://github.com/google/cluster-data
[3] Alibaba Production Cluster Trace. https://github.com/alibaba/clusterdata
[4] Snowflake Dataset. https://github.com/resource-disaggregation/snowset

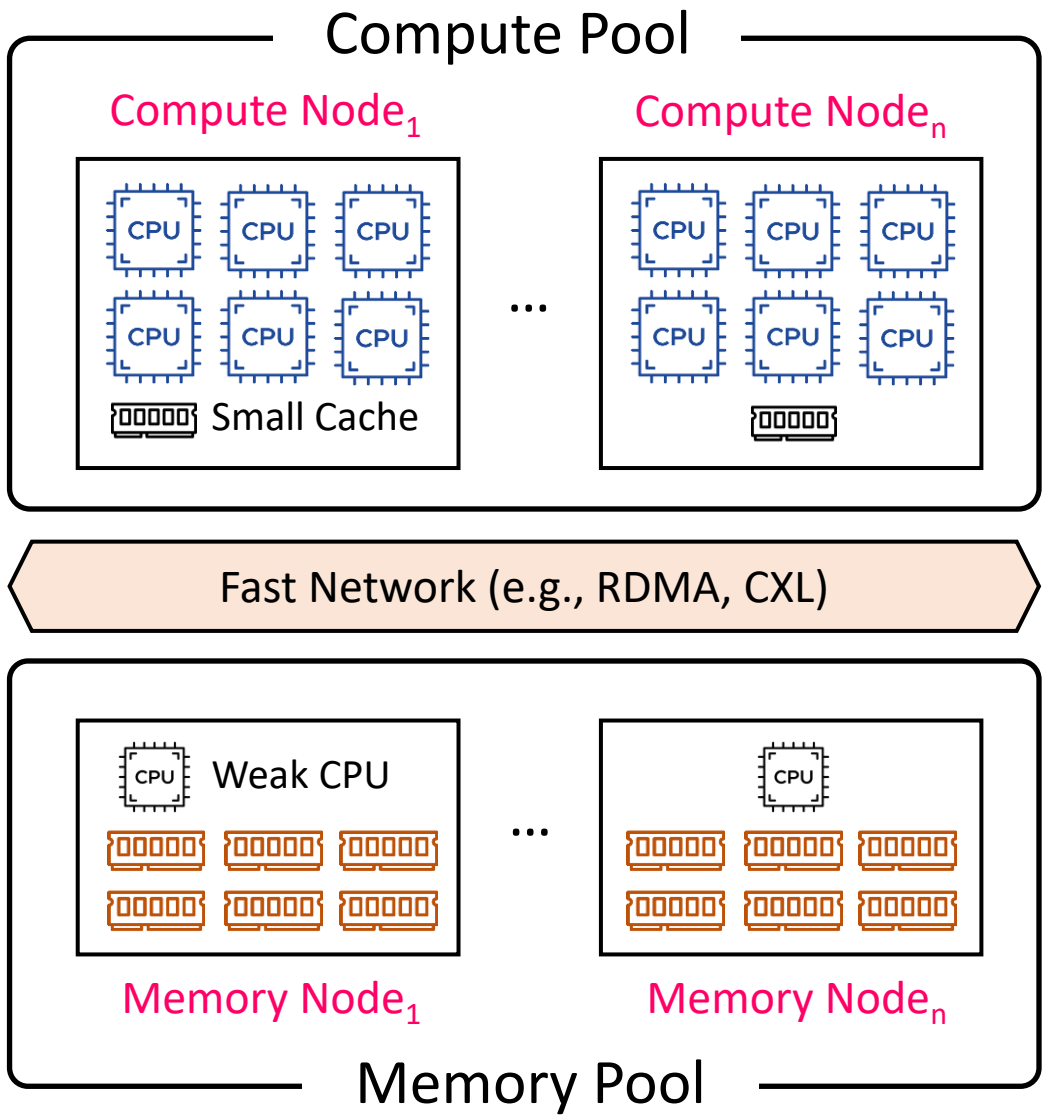
Insufficient Memory Utilization in Cloud

- About 20%~60% [1-4]
- One major reason: monolithic server

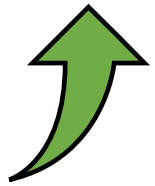


[1] MemTrade@SIGMETRICS'23, Borg@EuroSys'20, LegoOS@OSDI'18
[2] Google Production Cluster Trace. https://github.com/google/cluster-data
[3] Alibaba Production Cluster Trace. https://github.com/alibaba/clusterdata
[4] Snowflake Dataset. https://github.com/resource-disaggregation/snowset

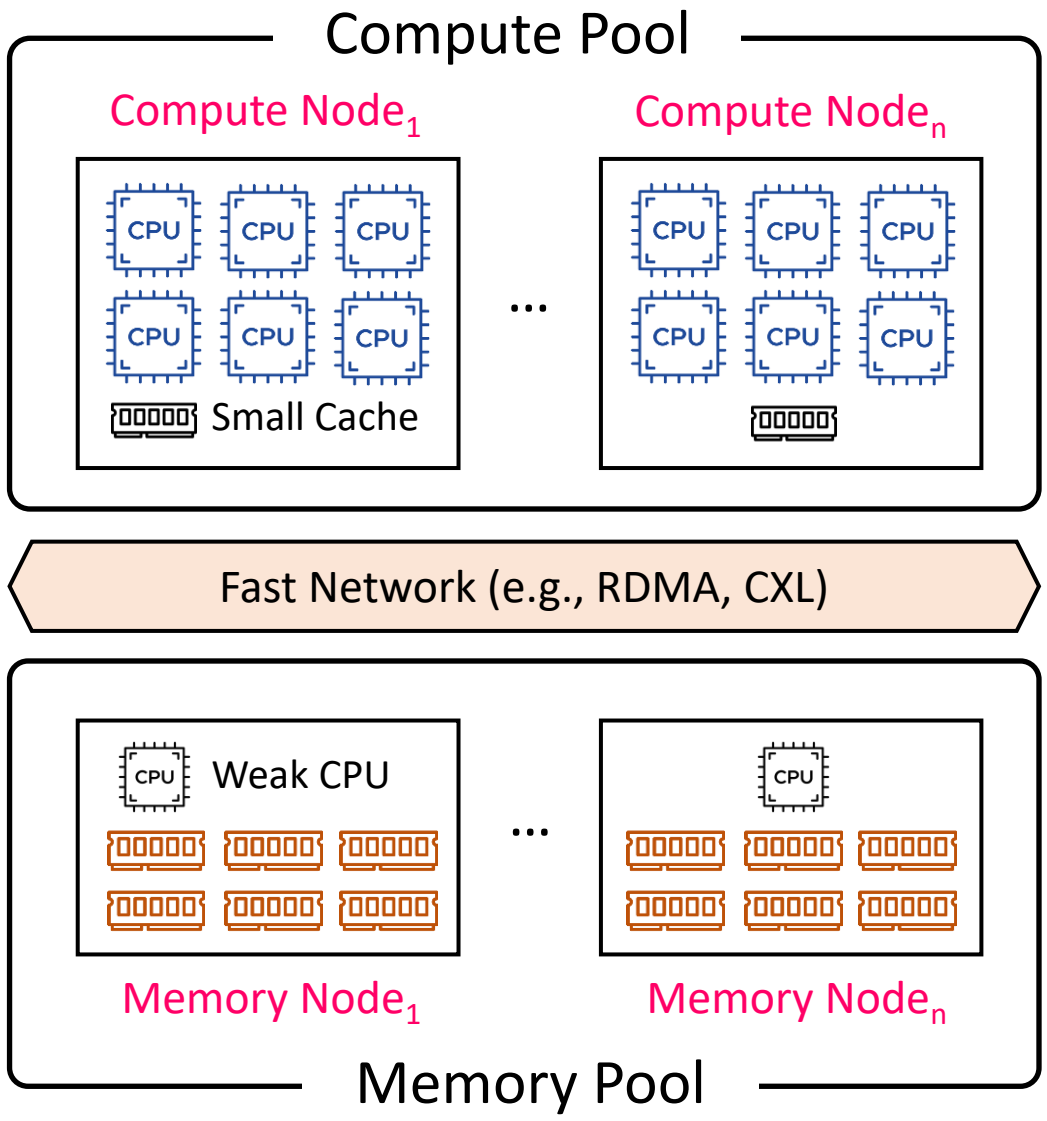
Disaggregated Memory (DM)



Resource utilization
Elasticity



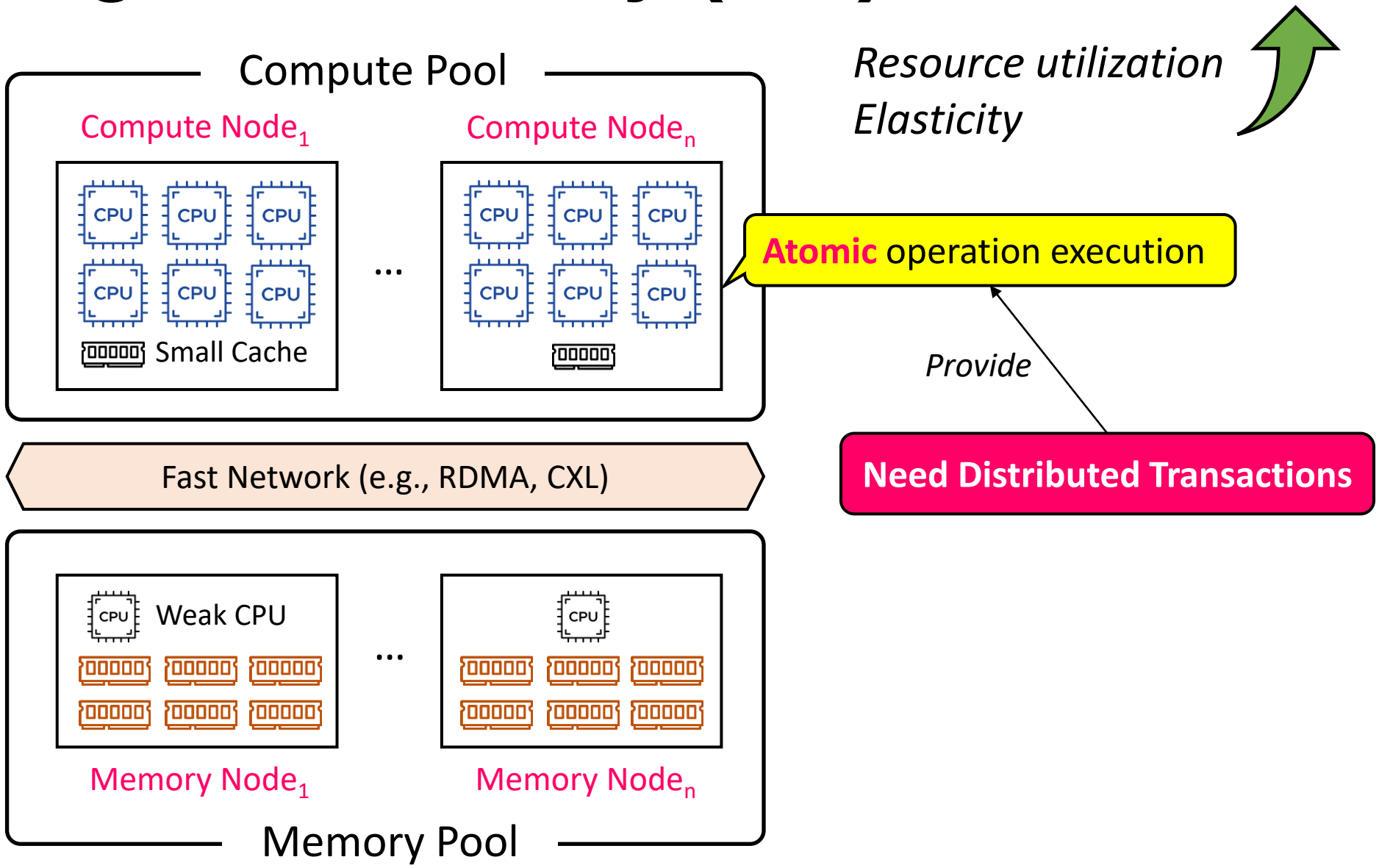
Disaggregated Memory (DM)



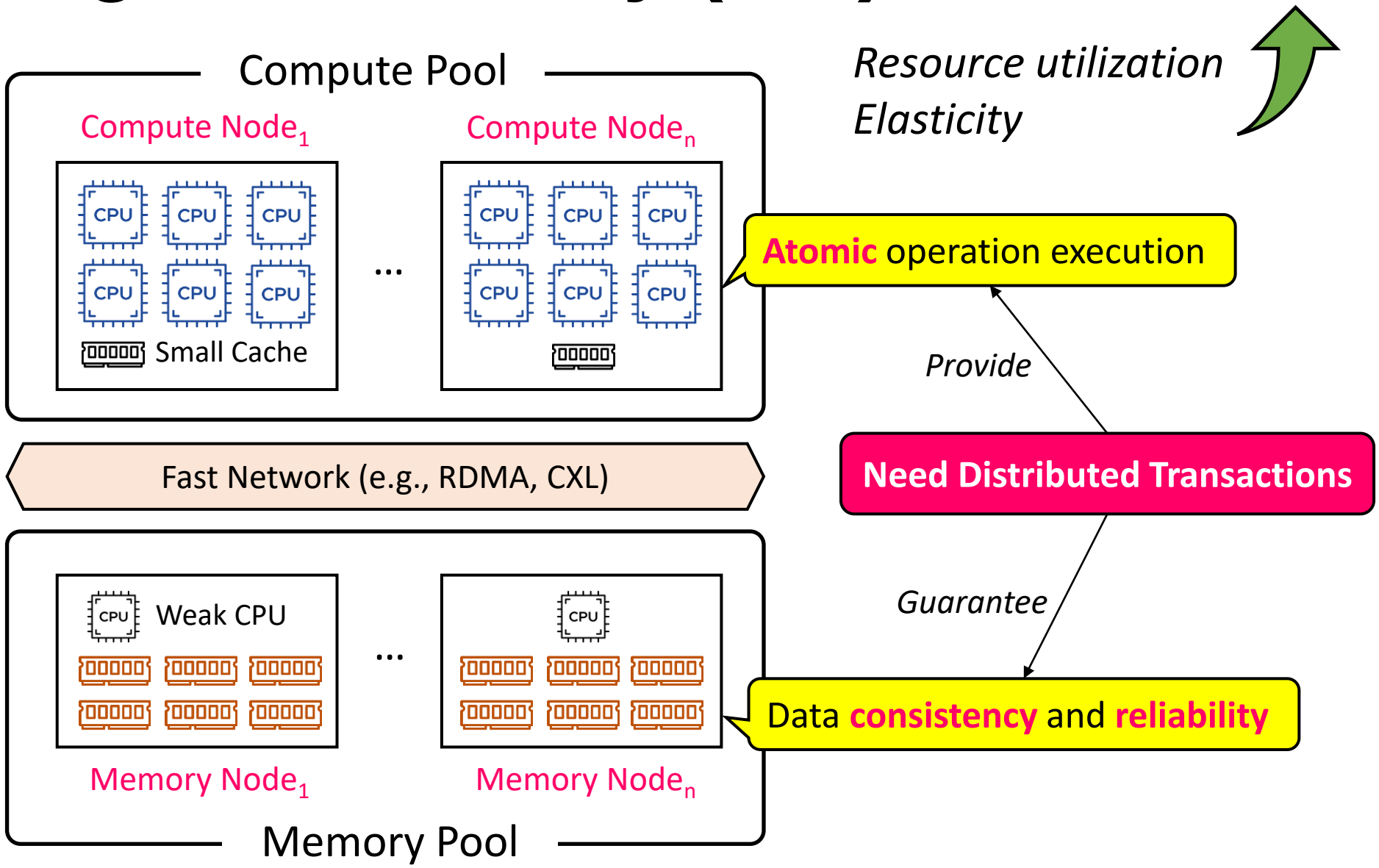
Resource utilization
Elasticity

Need Distributed Transactions

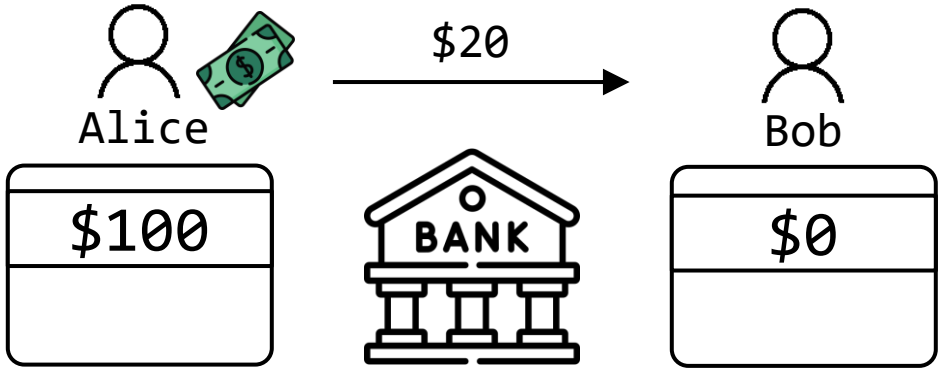
Disaggregated Memory (DM)



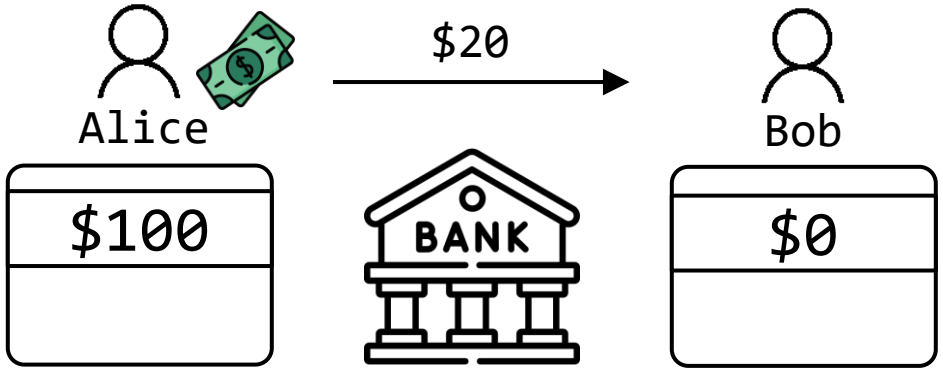
Disaggregated Memory (DM)



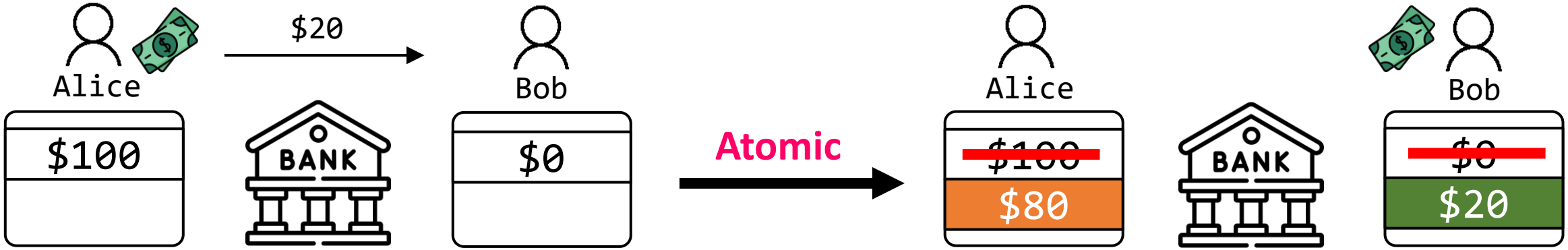
Transaction



Transaction



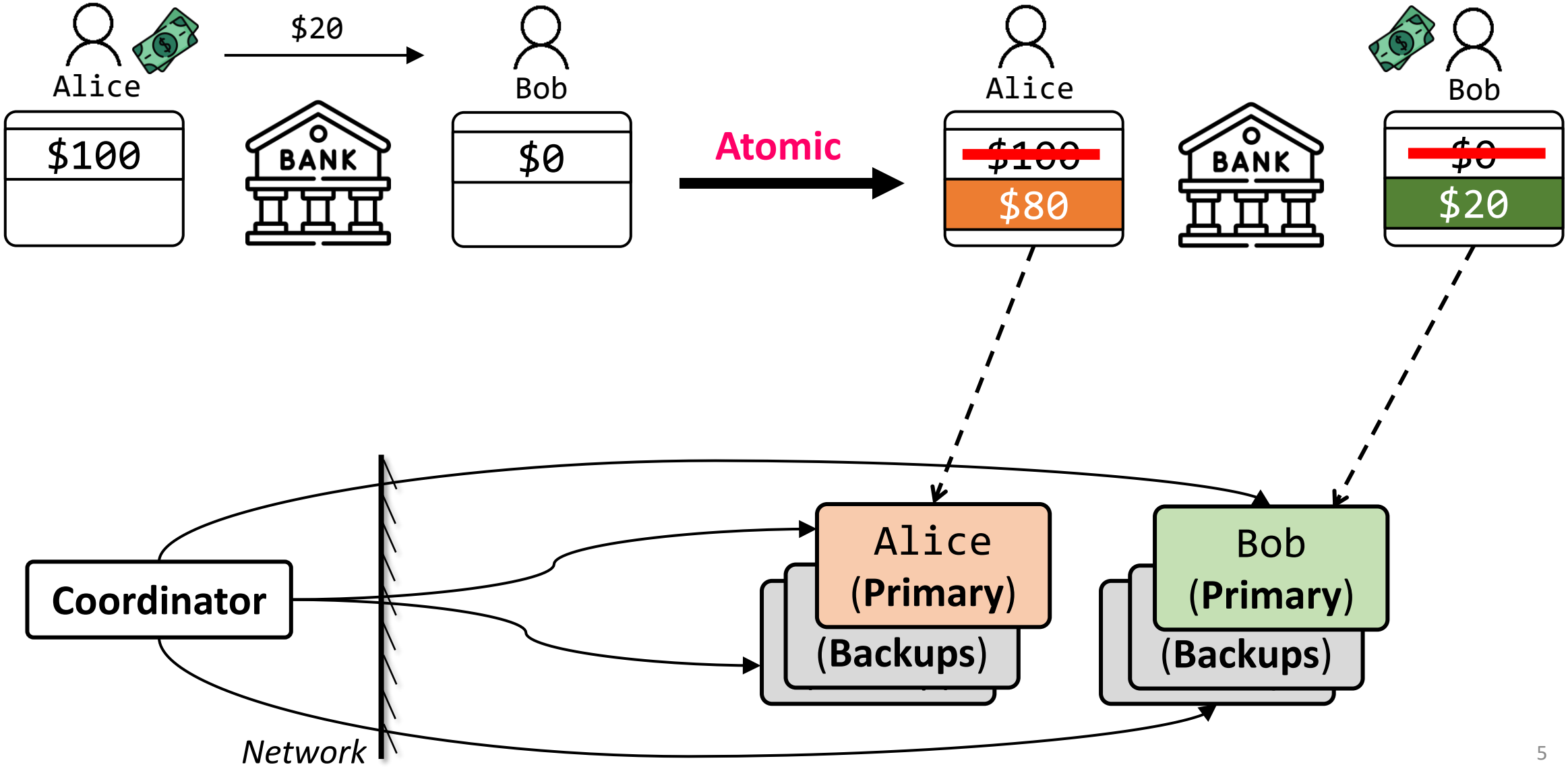
Transaction



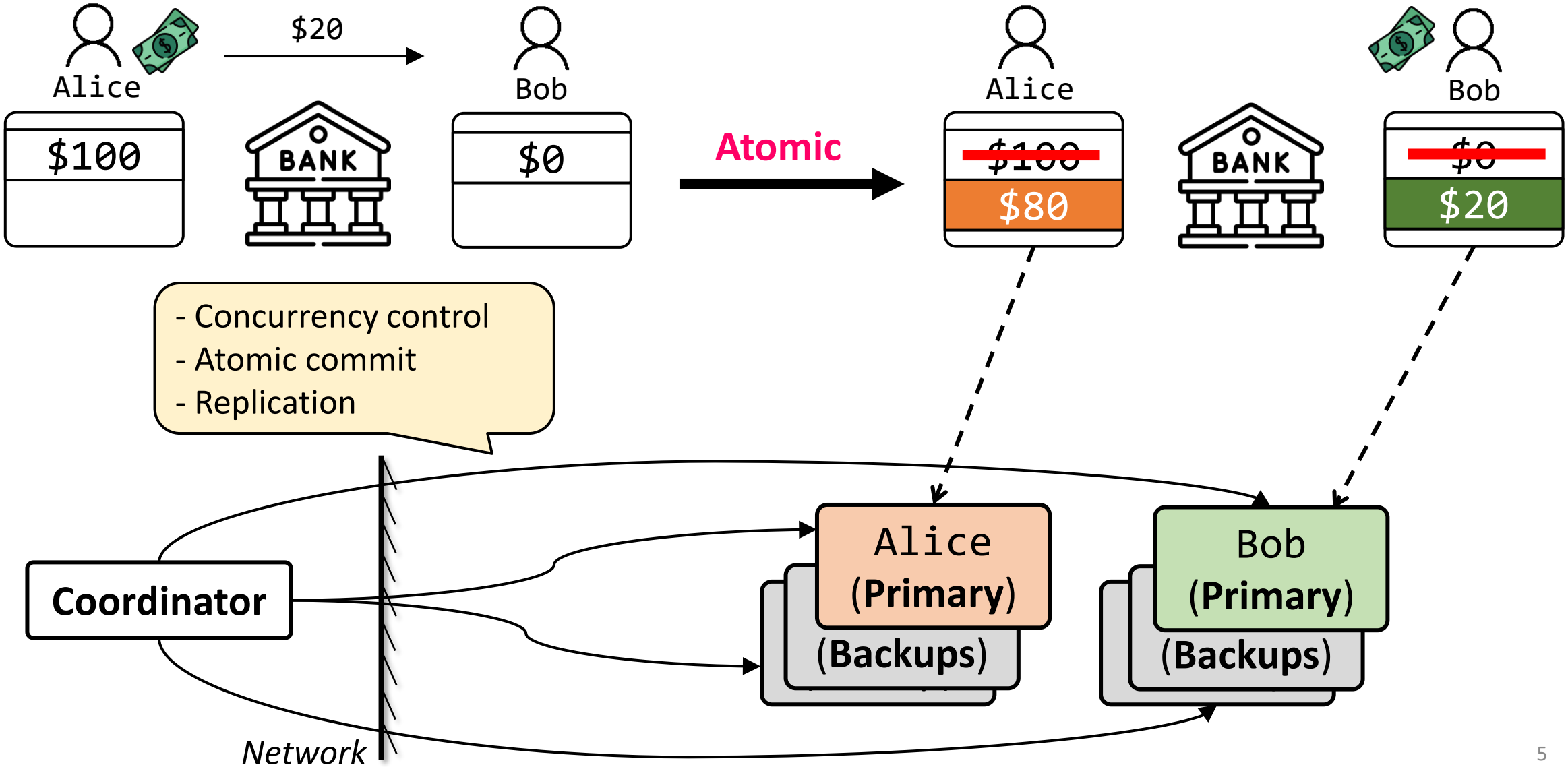

```
Txn begin
  Alice: $100 → $80
    Bob:   $0 → $20
Txn end
```

Transaction

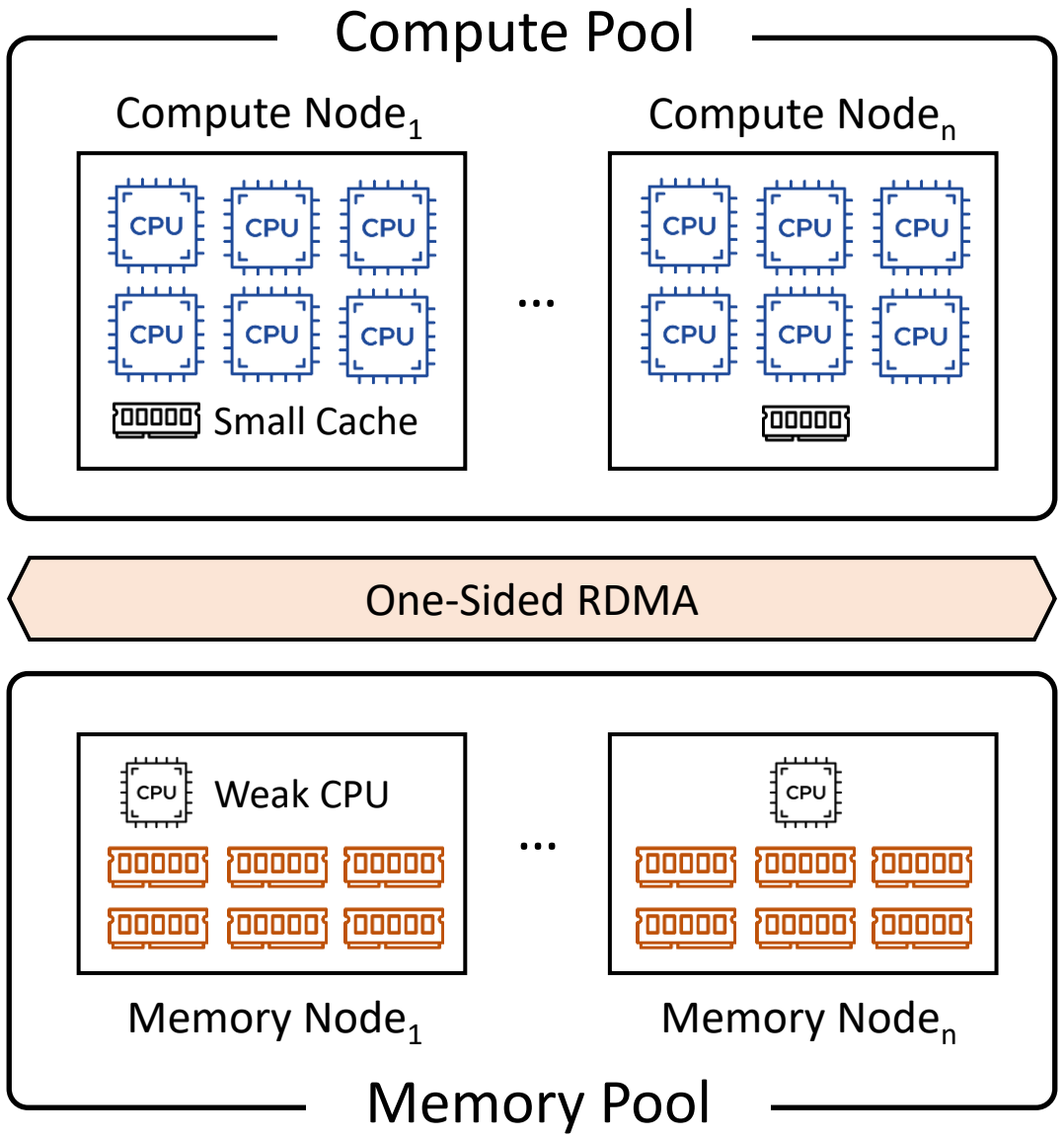
Distributed Transaction



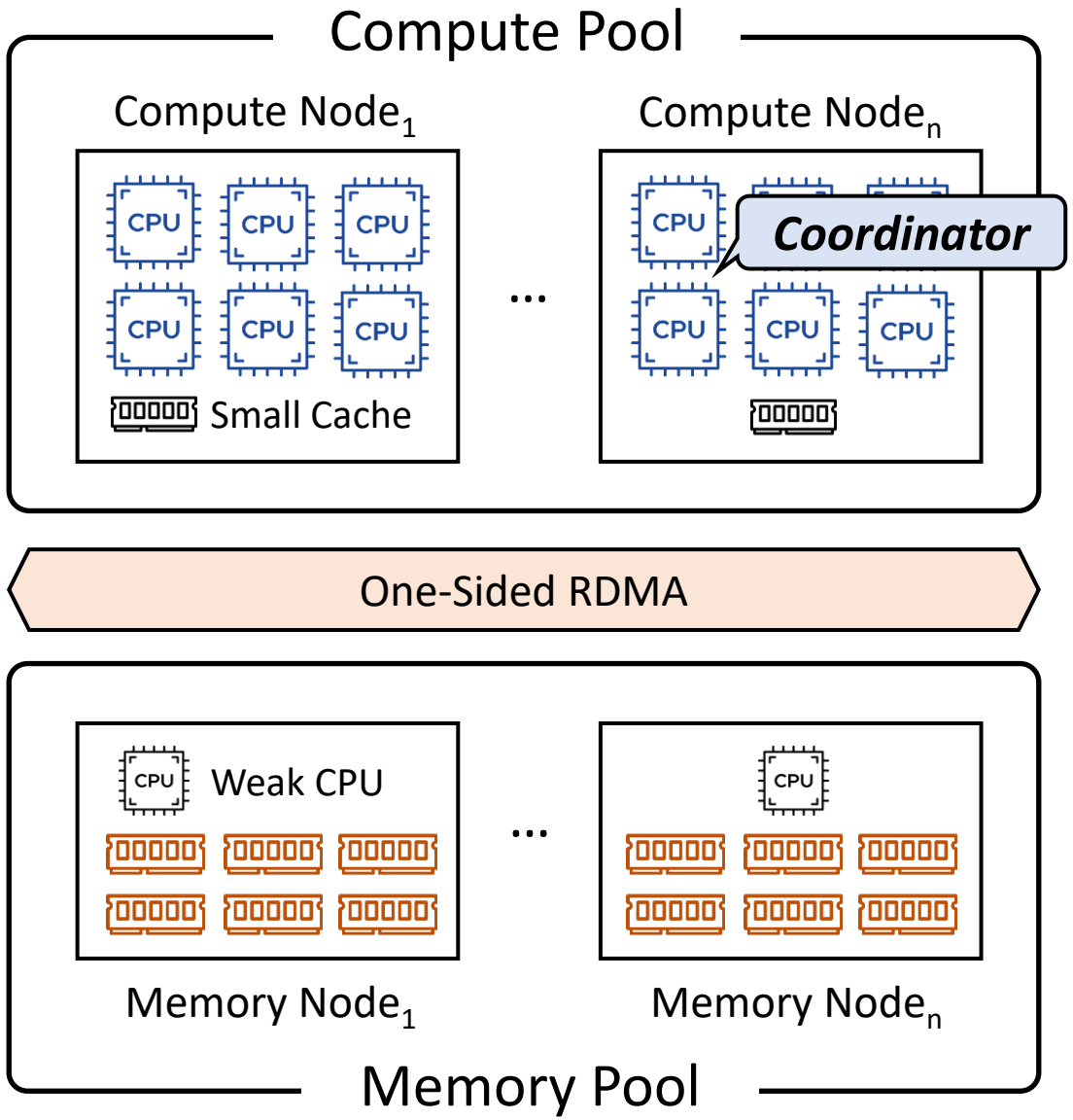
Distributed Transaction



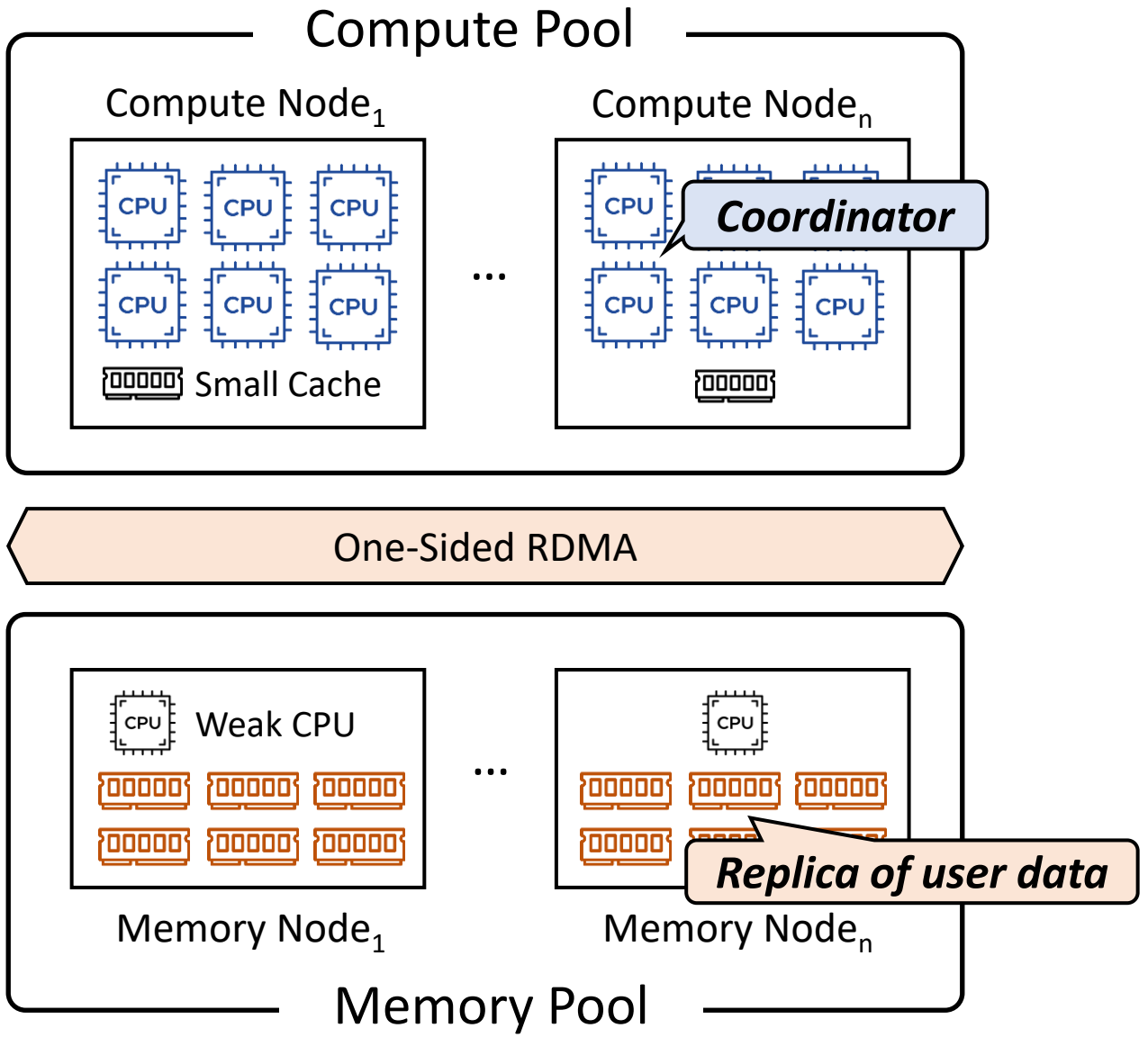
Distributed Transaction on DM



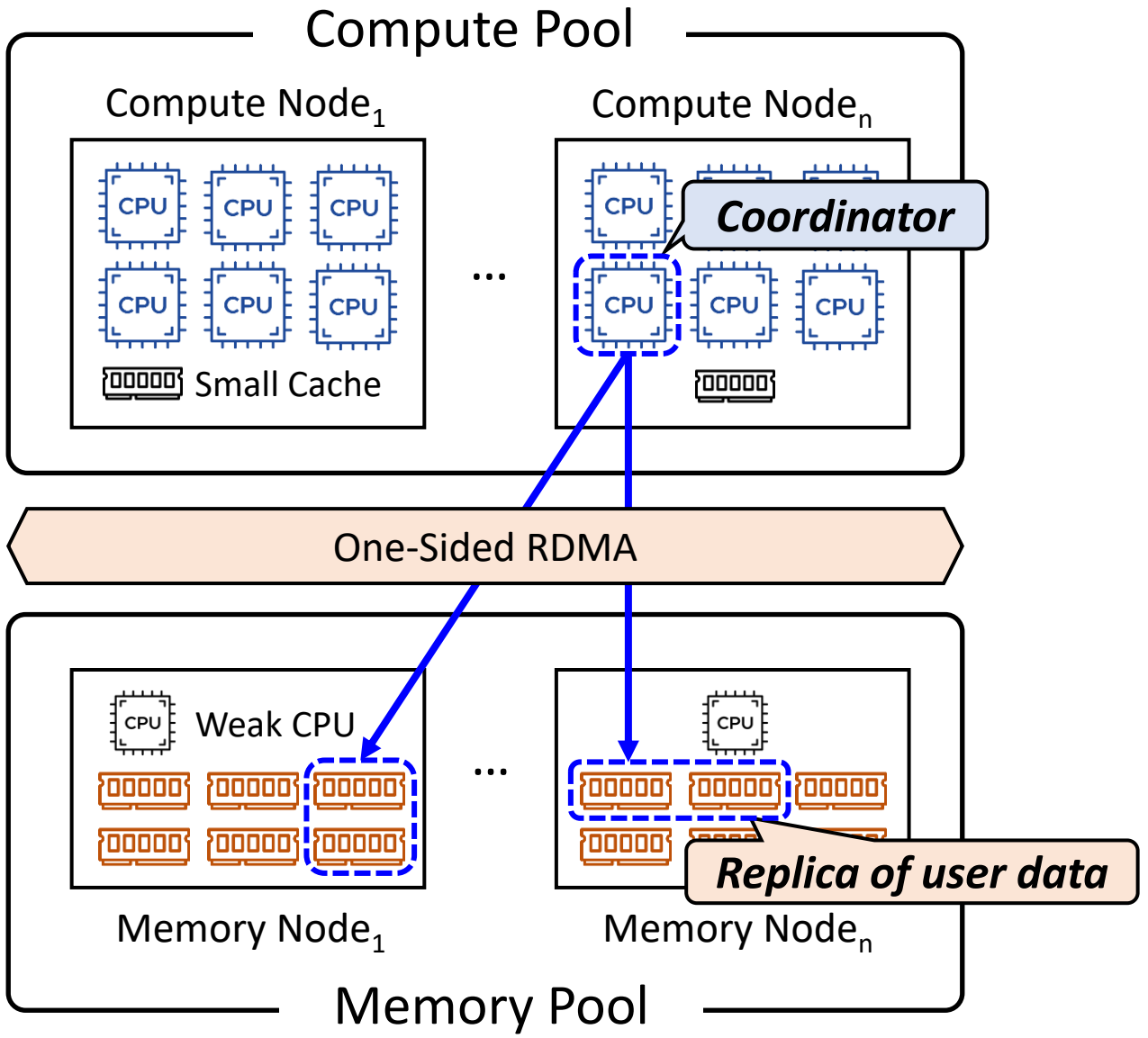
Distributed Transaction on DM



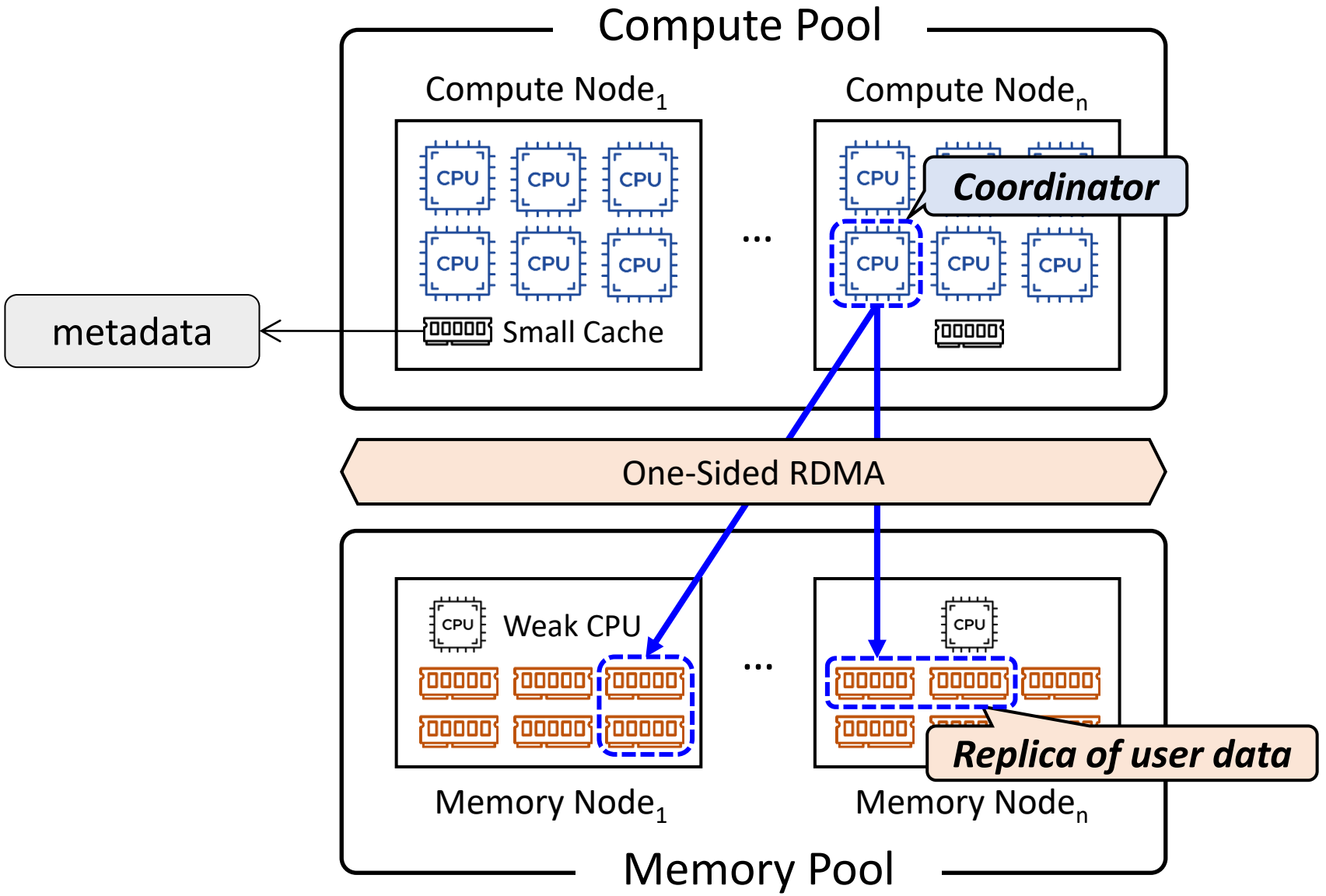
Distributed Transaction on DM



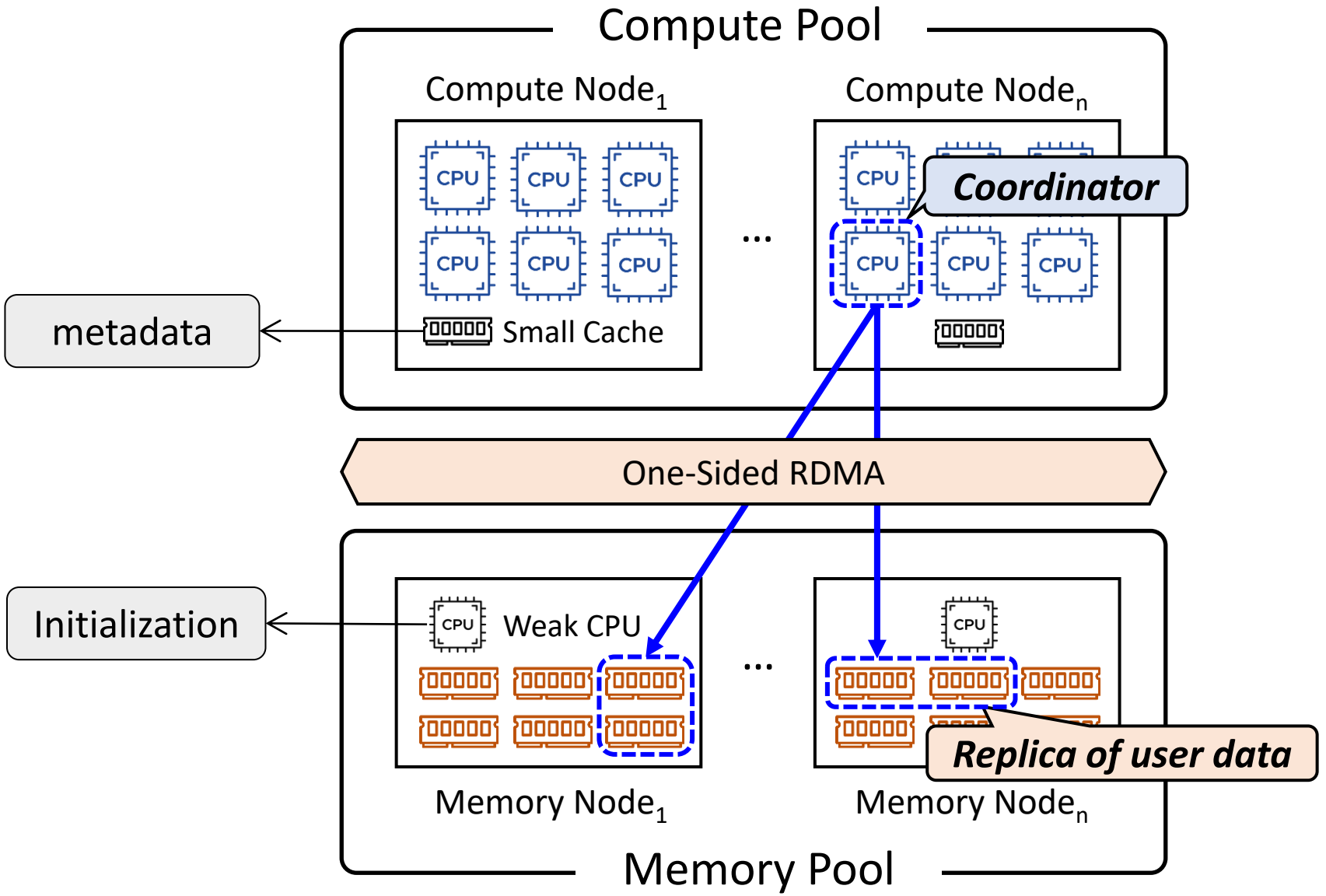
Distributed Transaction on DM



Distributed Transaction on DM



Distributed Transaction on DM



Existing Studies

- Single-versioning distributed transaction system for DM[1]

[1] FORD@FAST'22

Existing Studies

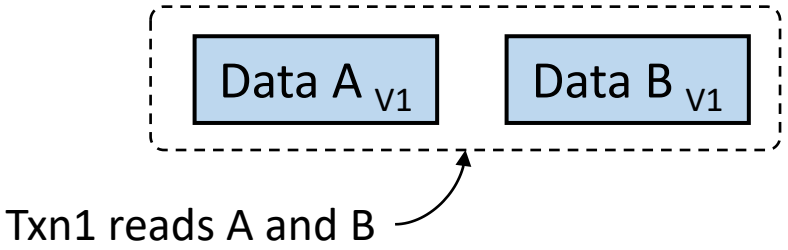
- Single-versioning distributed transaction system for DM[1]
  - ☹ Limited concurrency

[1] FORD@FAST'22

Existing Studies

- Single-versioning distributed transaction system for DM[1]

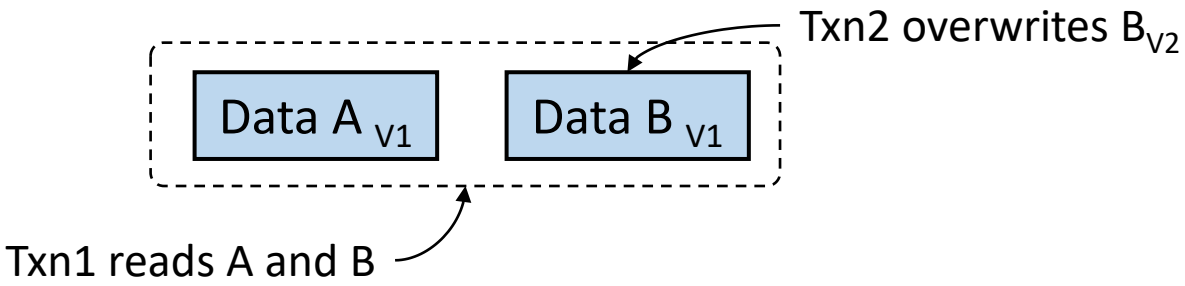
☹ Limited concurrency



[1] FORD@FAST'22

Existing Studies

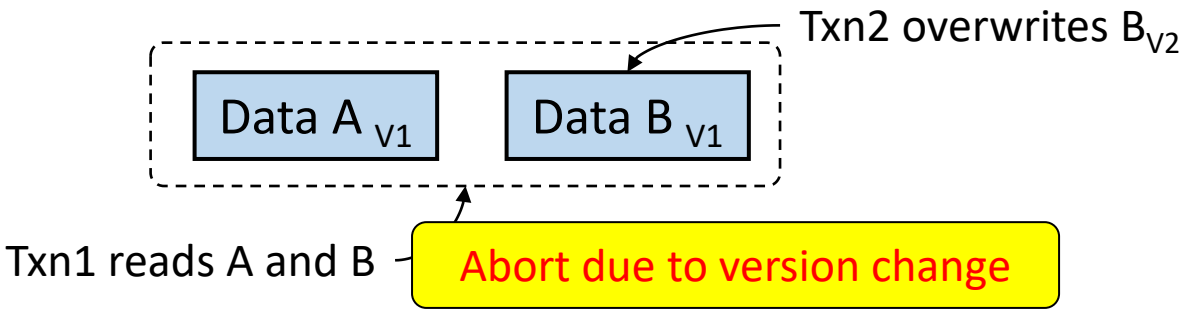
- Single-versioning distributed transaction system for DM[1]
  - ☹ Limited concurrency



[1] FORD@FAST'22

Existing Studies

- Single-versioning distributed transaction system for DM[1]
  - ☹ Limited concurrency



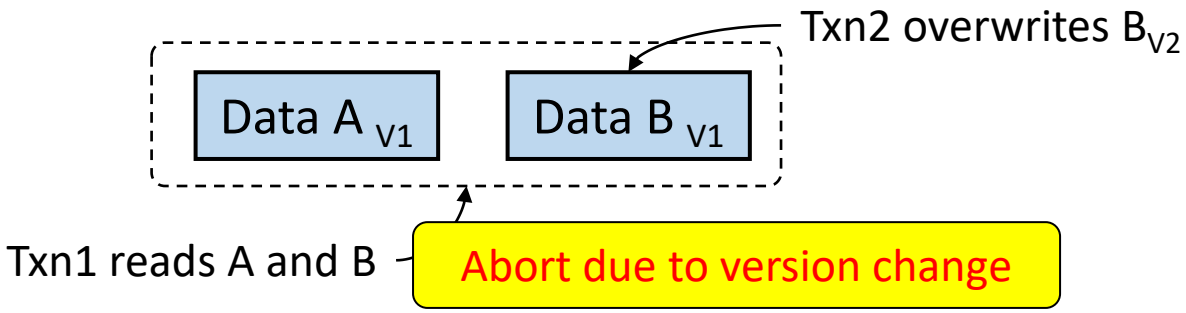
[1] FORD@FAST'22

Existing Studies

- Single-versioning distributed transaction system for DM[1]

☹ Limited concurrency

☹ High logging overhead

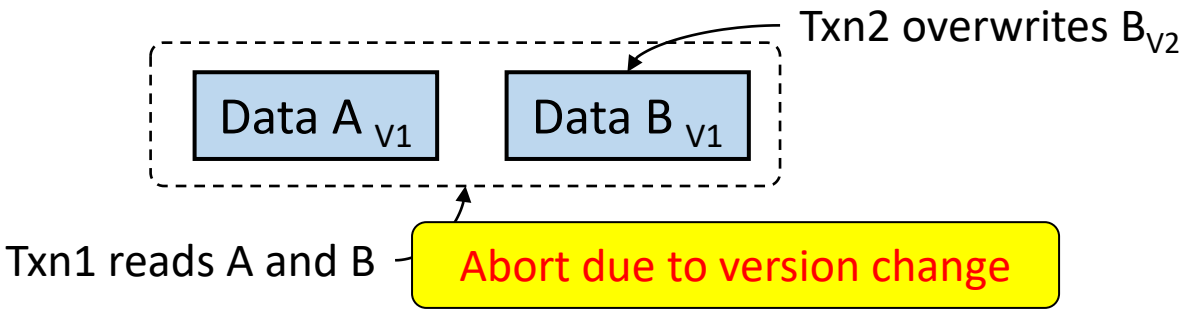


[1] FORD@FAST'22

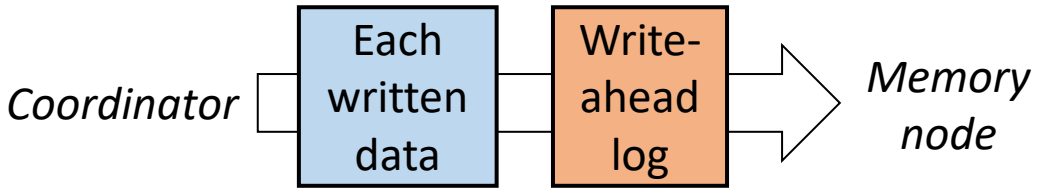
Existing Studies

- Single-versioning distributed transaction system for DM[1]

☹ Limited concurrency



☹ High logging overhead

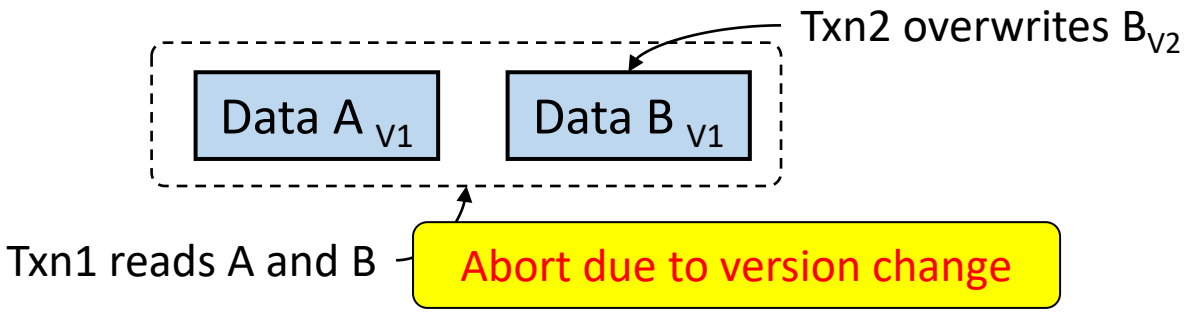


[1] FORD@FAST'22

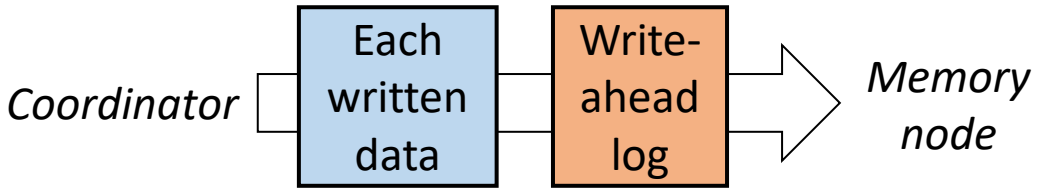
Existing Studies

- Single-versioning distributed transaction system for DM[1]

☹ Limited concurrency



☹ High logging overhead



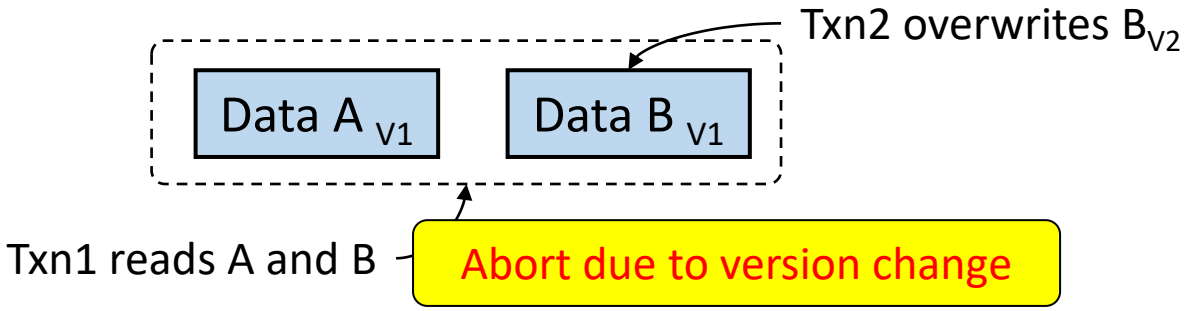
- Multi-versioning helps address limitations of single-versioning

[1] FORD@FAST'22

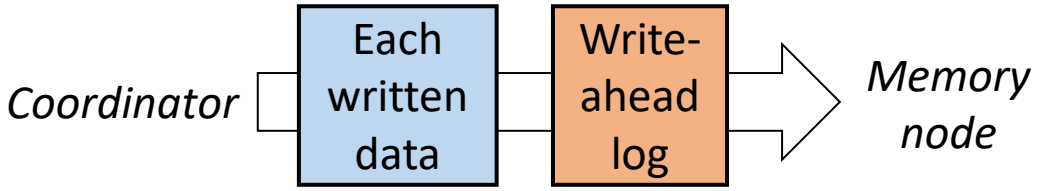
Existing Studies

- Single-versioning distributed transaction system for DM[1]

☹ Limited concurrency

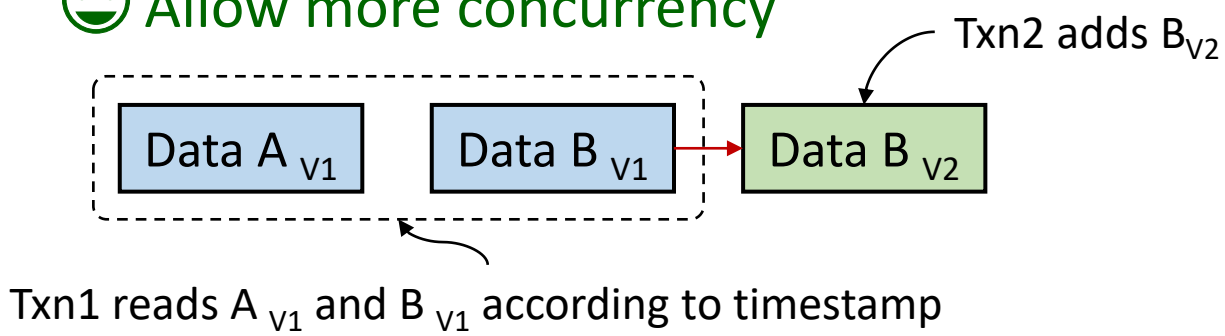


☹ High logging overhead



- Multi-versioning helps address limitations of single-versioning

☺ Allow more concurrency

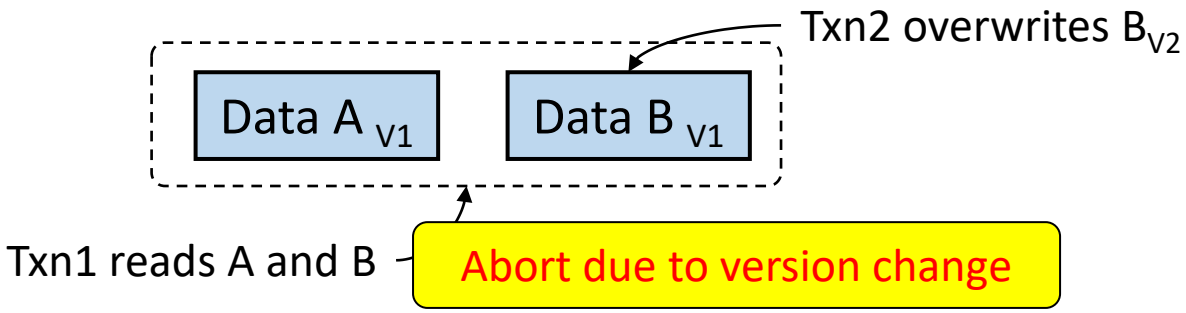


[1] FORD@FAST'22

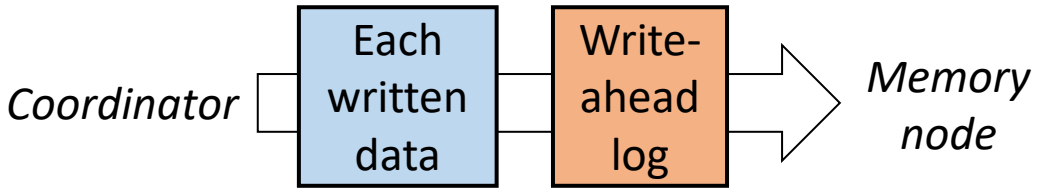
Existing Studies

- Single-versioning distributed transaction system for DM[1]

☹ Limited concurrency

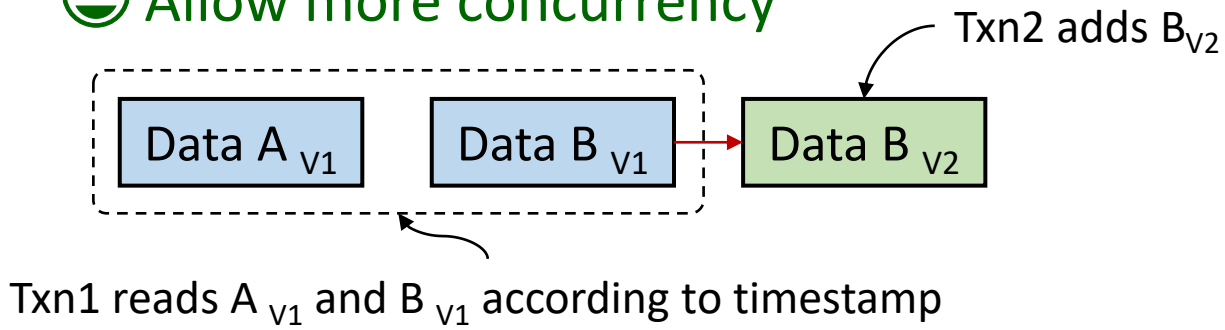


☹ High logging overhead

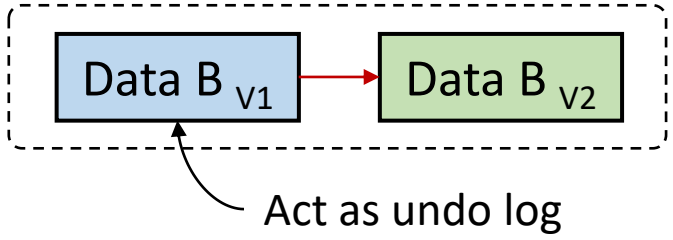


- Multi-versioning helps address limitations of single-versioning

☺ Allow more concurrency



☺ Reduce logs



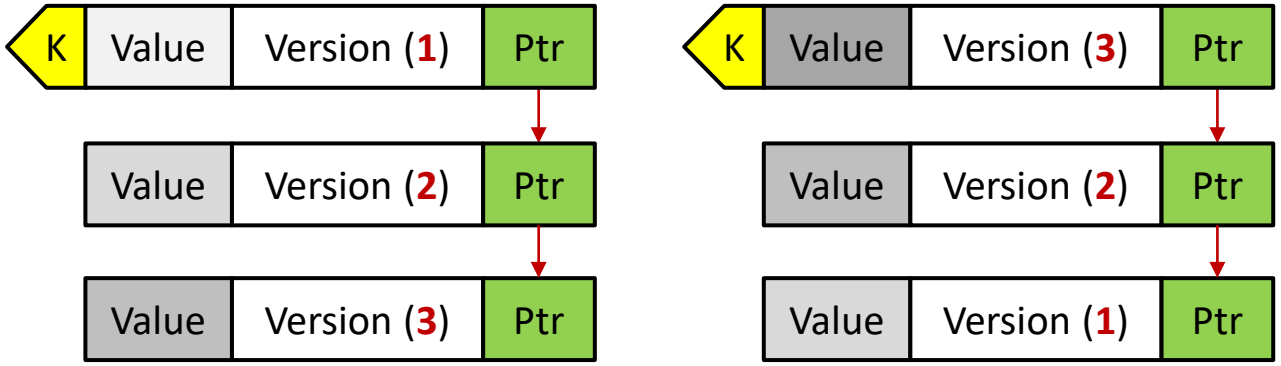
[1] FORD@FAST'22

Does Multi-Versioning Work on DM?

- Existing systems are based on monolithic servers

Does Multi-Versioning Work on DM?

- Existing systems are based on monolithic servers
  - ☹ Inefficient linked version chain
    - Works in monolithic servers but does not fit DM



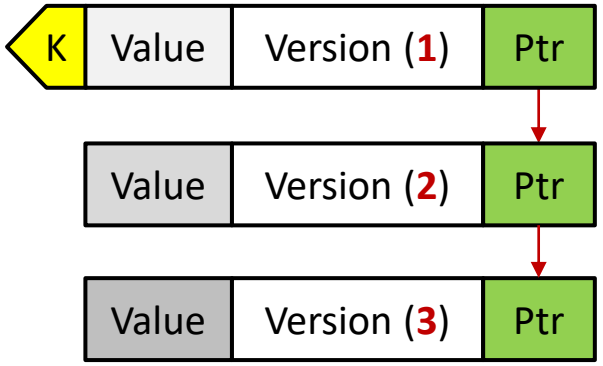
Old-to-new chain[1-3]

New-to-old chain[4-6]

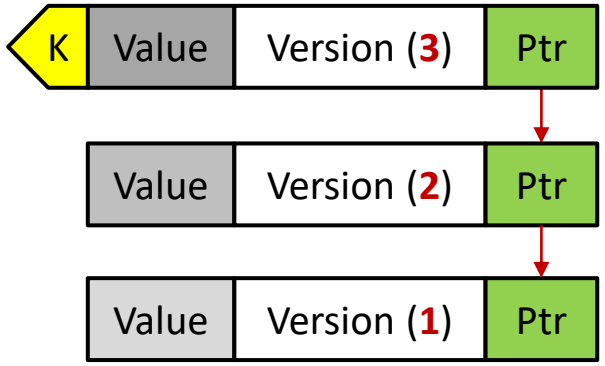
[1] DST@NSDI'21 [2] Hekaton@SIGMOD'13 [3] Aurogon@FAST'22
[4] FaRMv2@SIGMOD'19 [5] Neumann et al.@SIGMOD'15 [6] NAM-DB@VLDB'17

Does Multi-Versioning Work on DM?

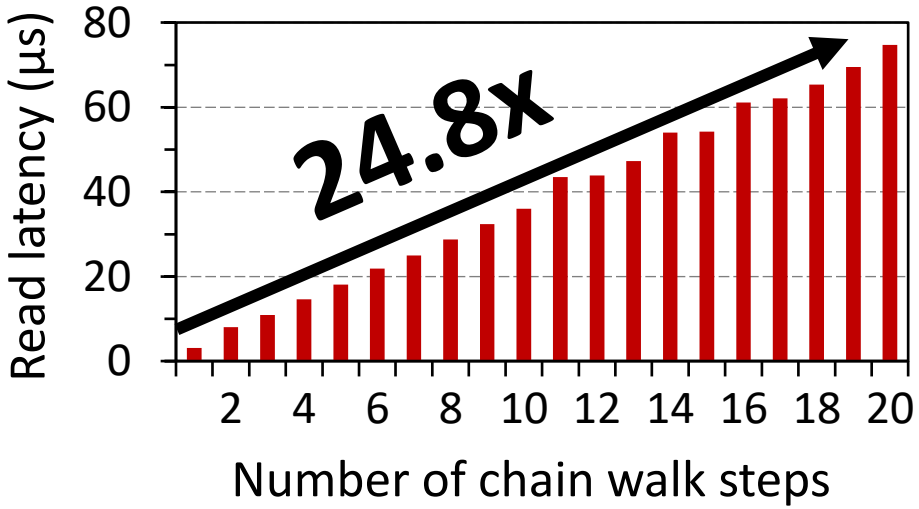
- Existing systems are based on monolithic servers
  - ☹ Inefficient linked version chain
    - Works in monolithic servers but does not fit DM



Old-to-new chain[1-3]



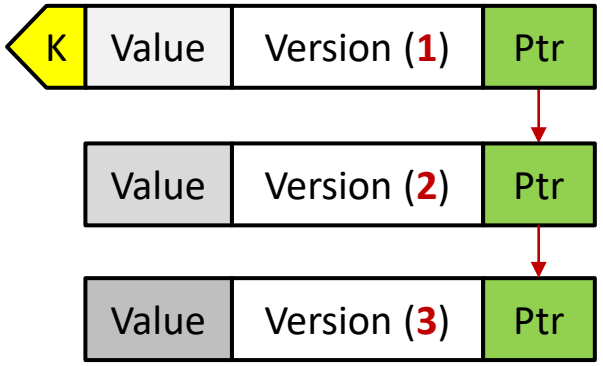
New-to-old chain[4-6]



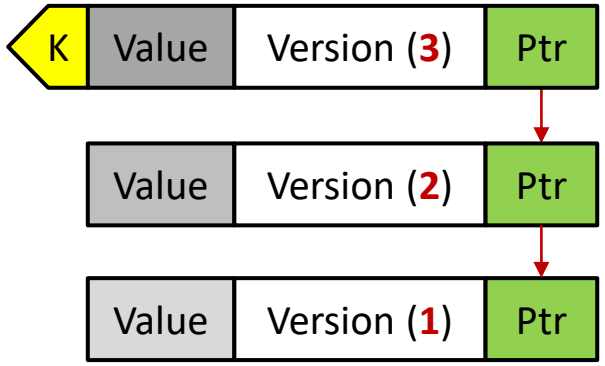
[1] DST@NSDI'21 [2] Hekaton@SIGMOD'13 [3] Aurogon@FAST'22
[4] FaRMv2@SIGMOD'19 [5] Neumann et al.@SIGMOD'15 [6] NAM-DB@VLDB'17

Does Multi-Versioning Work on DM?

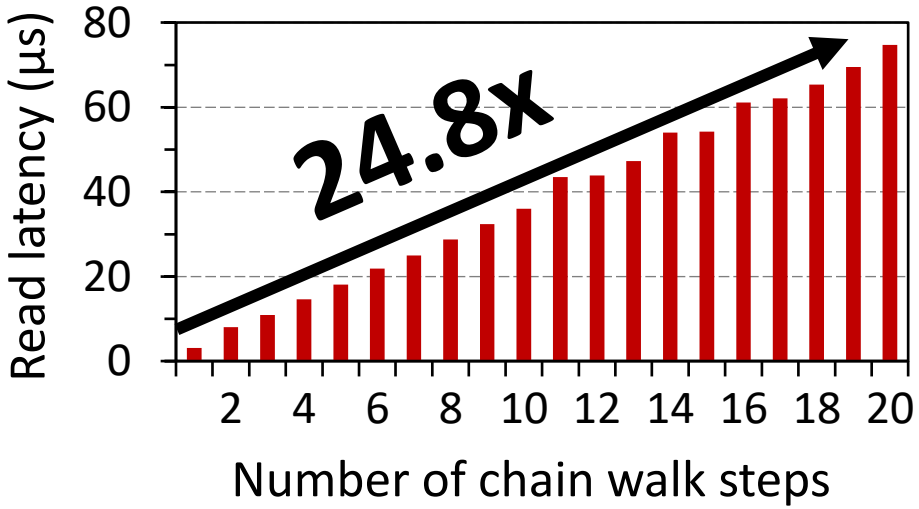
- Existing systems are based on monolithic servers
  - ☹ Inefficient linked version chain
    - Works in monolithic servers but does not fit DM



Old-to-new chain[1-3]



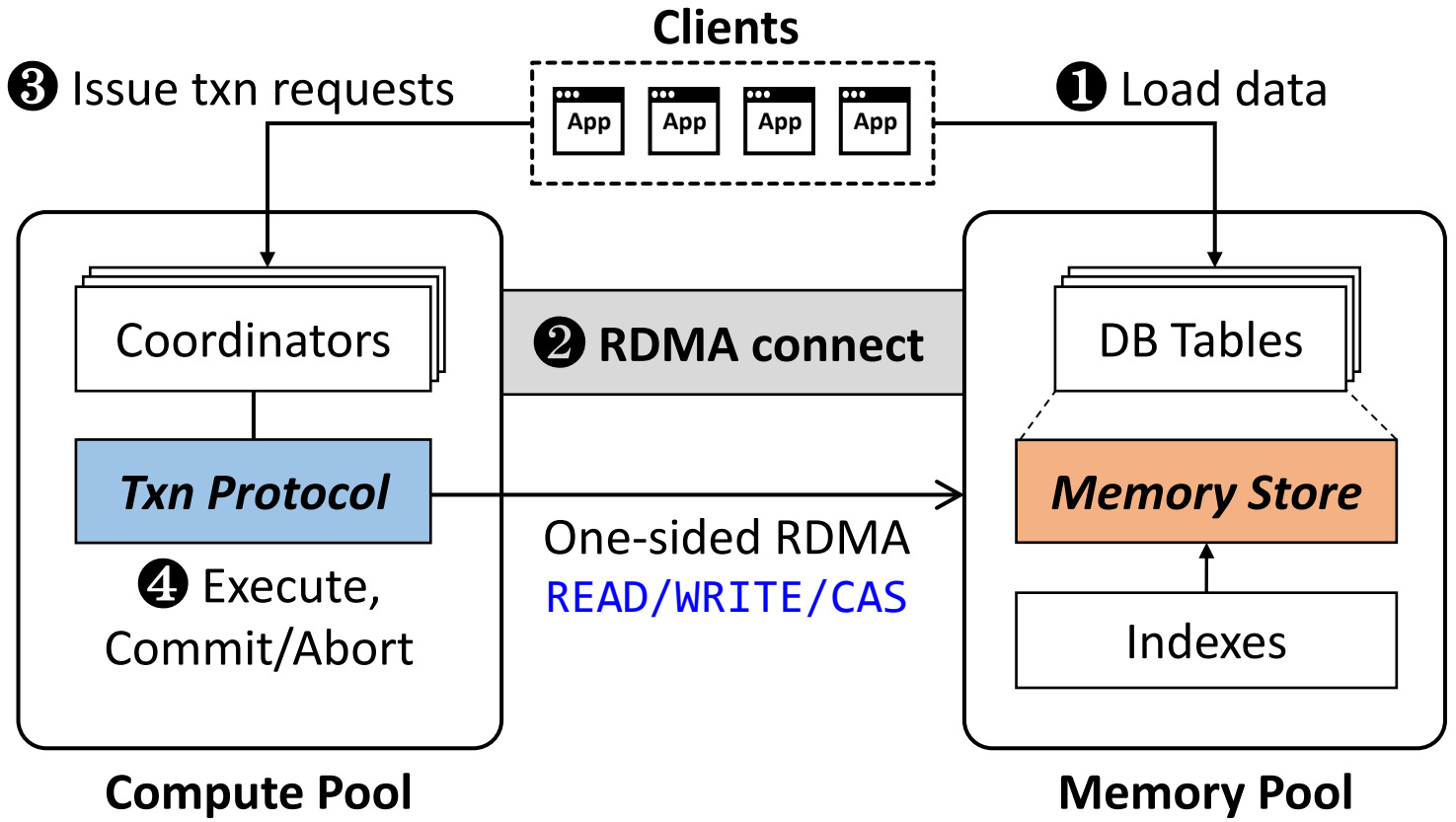
New-to-old chain[4-6]



  - ☹ Incompatible transaction protocol
    - Frequently consumes CPU of each data node[1,4,5]
    - Memory node stores data but only has weak CPU

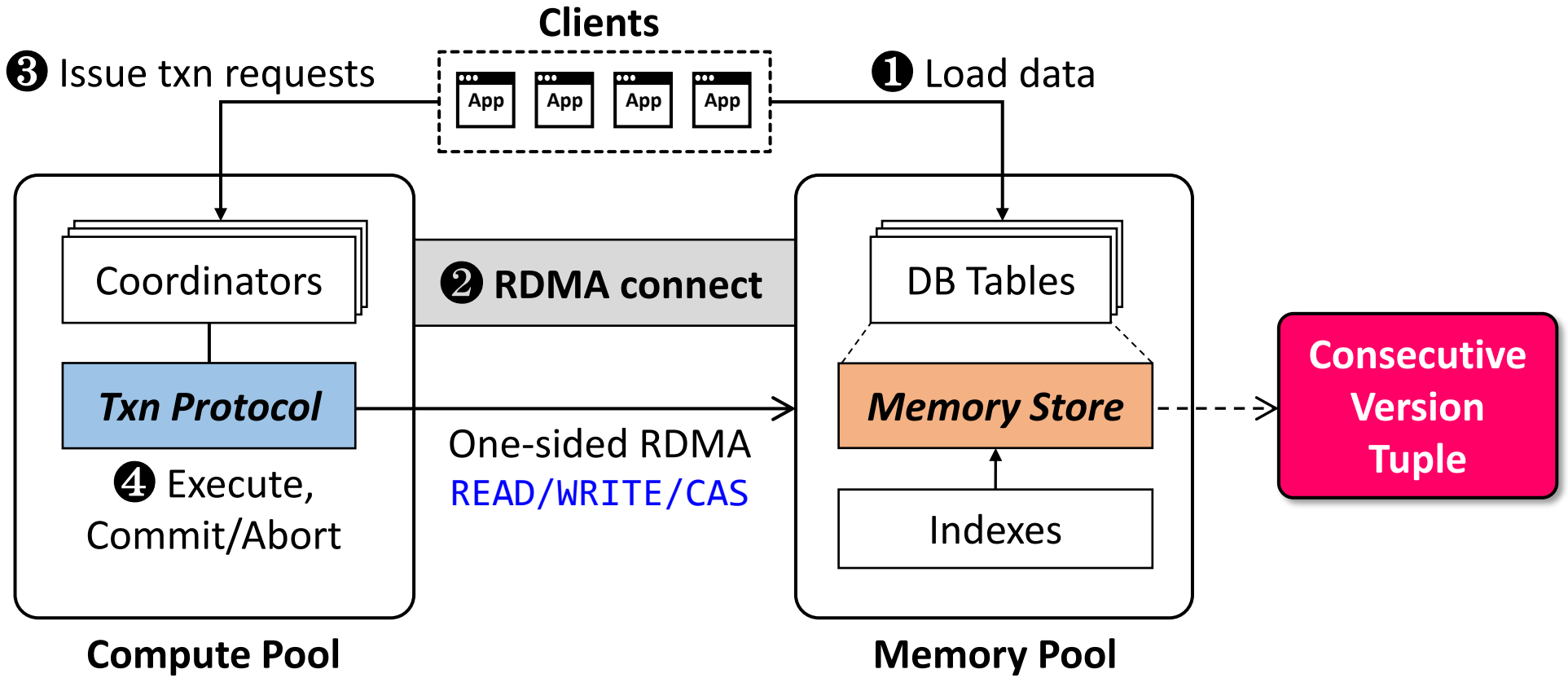
[1] DST@NSDI'21 [2] Hekaton@SIGMOD'13 [3] Aurogon@FAST'22
[4] FaRMv2@SIGMOD'19 [5] Neumann et al.@SIGMOD'15 [6] NAM-DB@VLDB'17

Motor: Enabling Multi-Versioning for DM



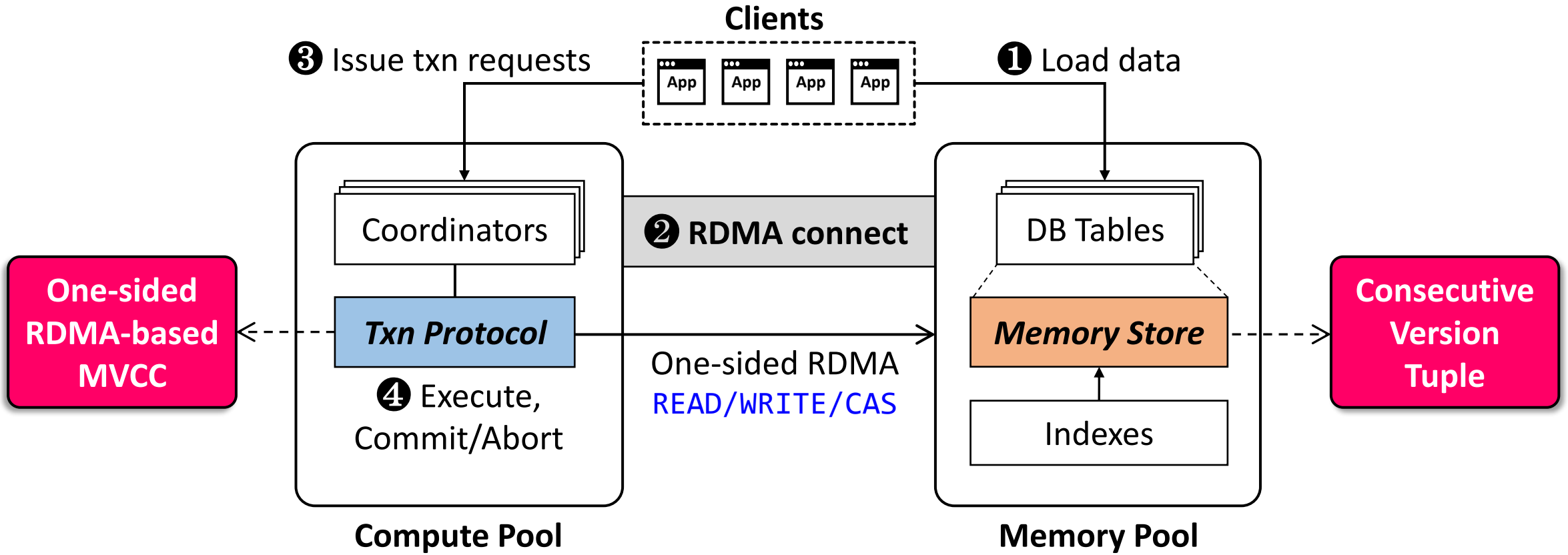
System Overview

Motor: Enabling Multi-Versioning for DM



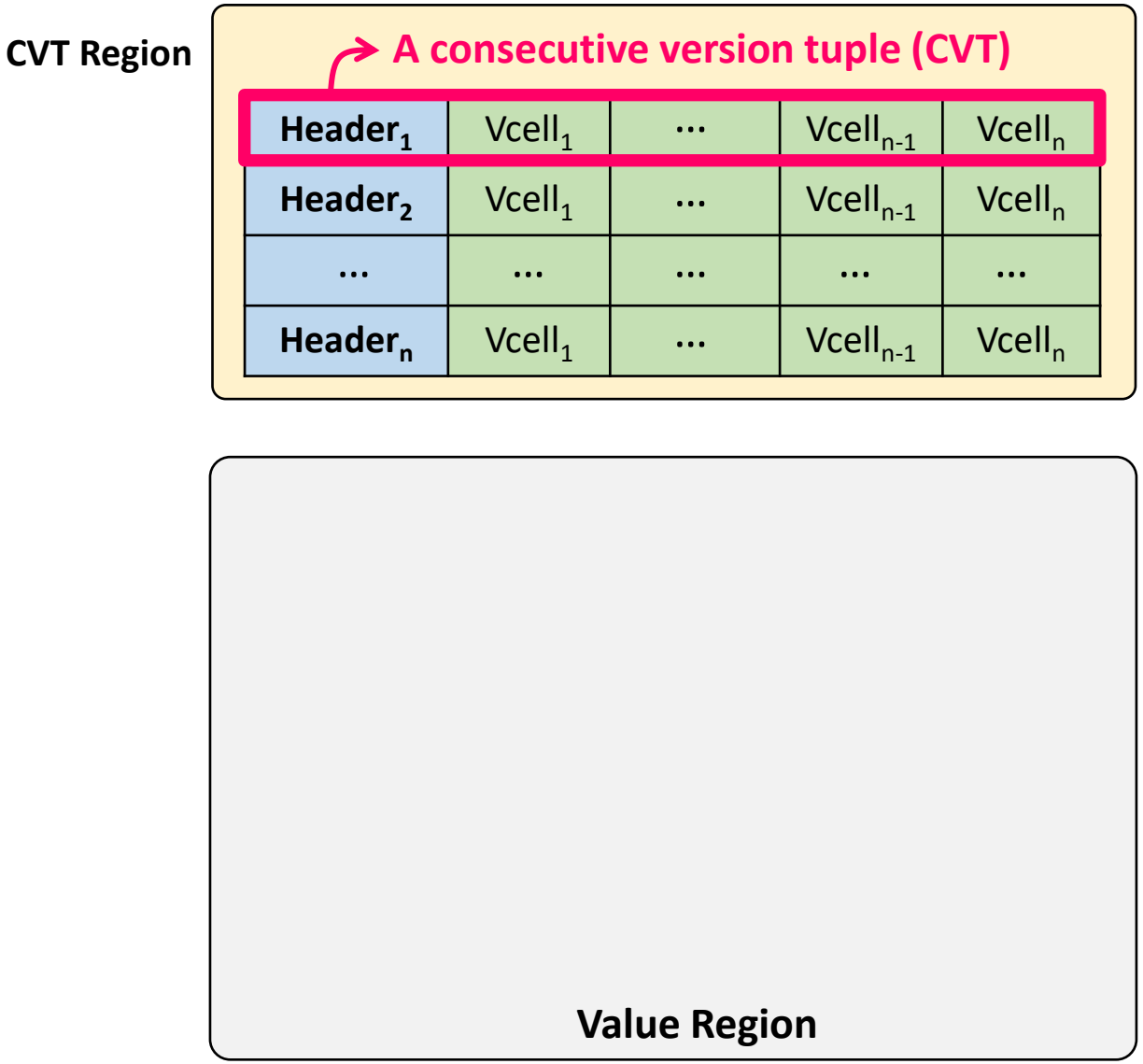
System Overview

Motor: Enabling Multi-Versioning for DM

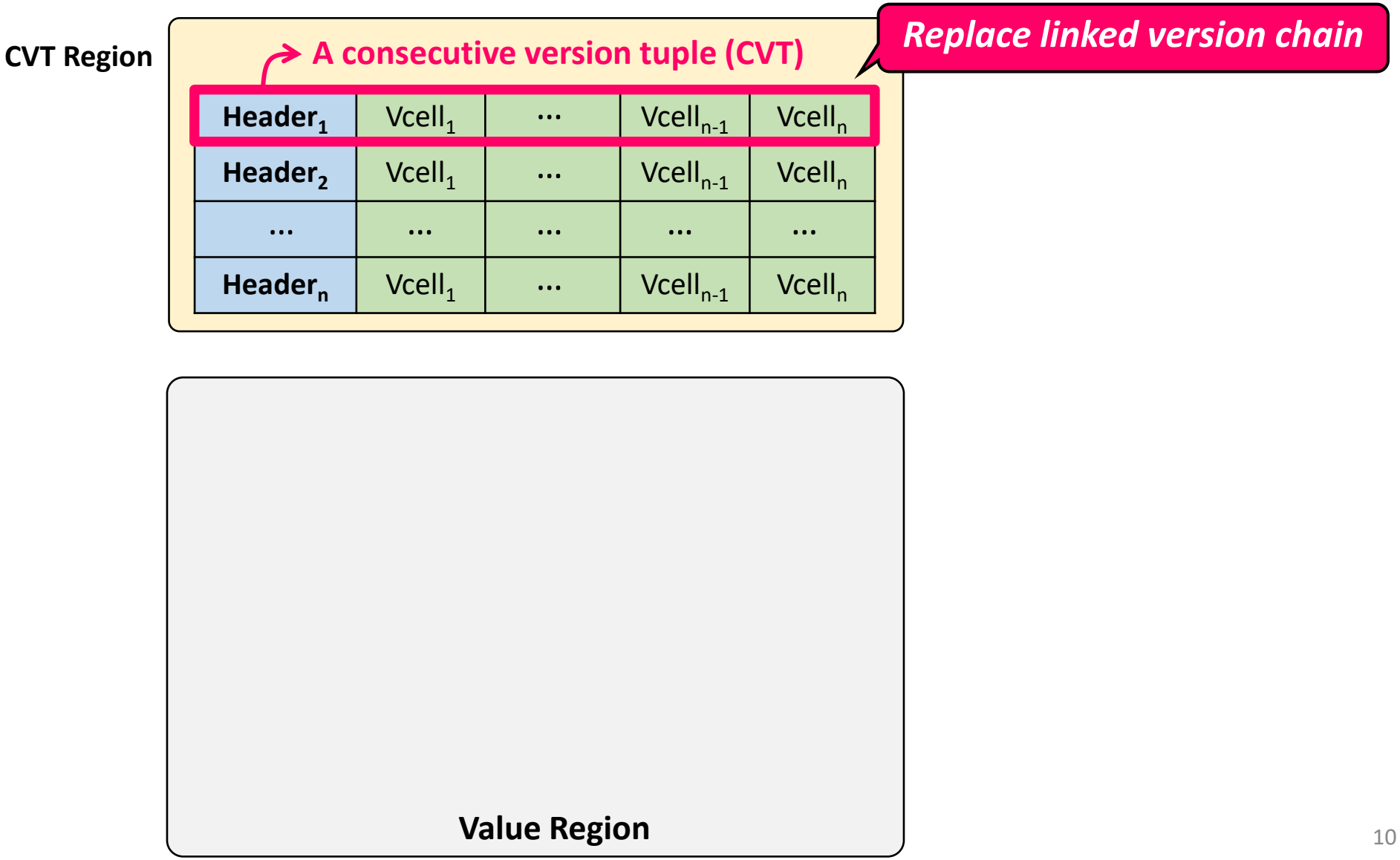


System Overview

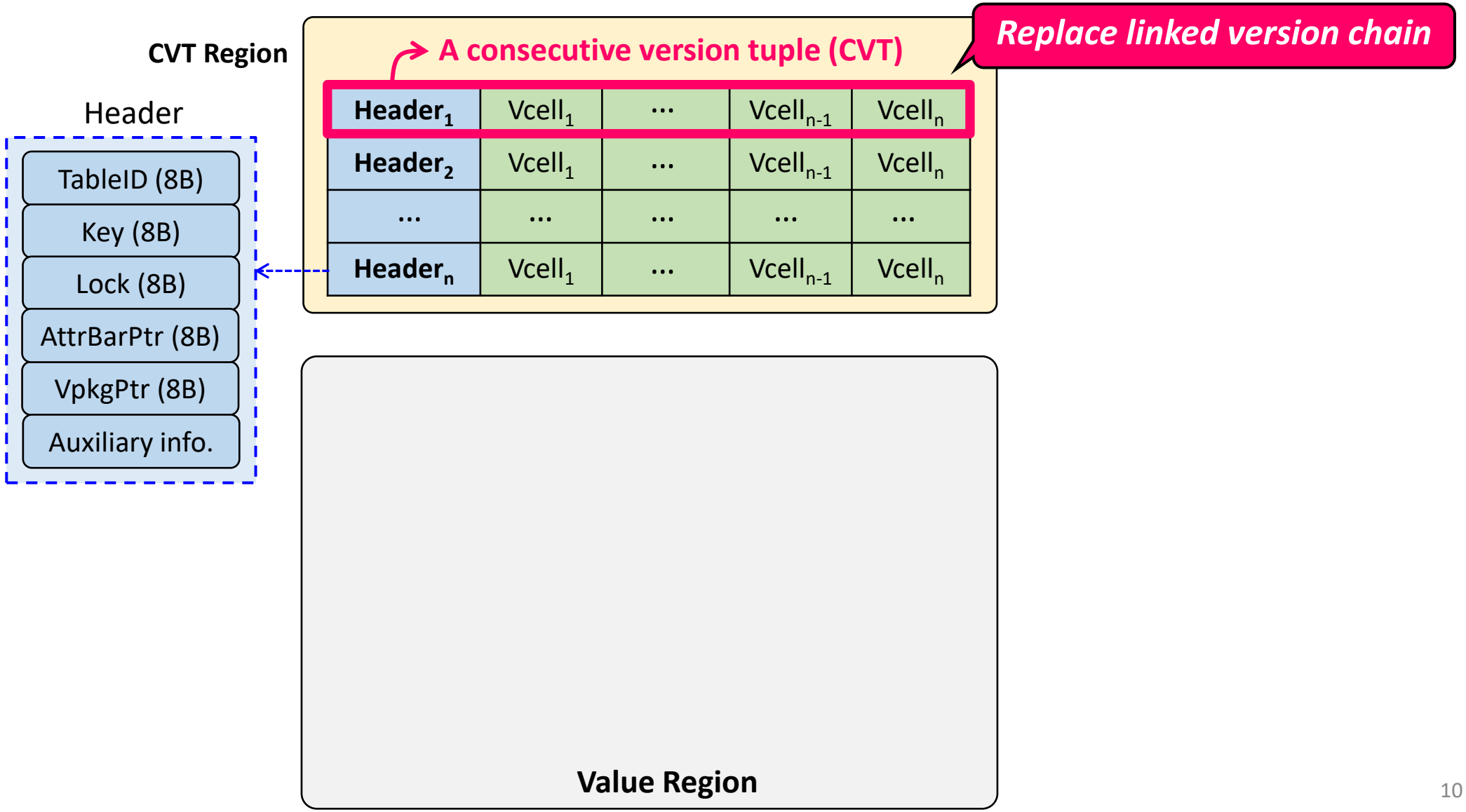
Consecutive Version Tuple



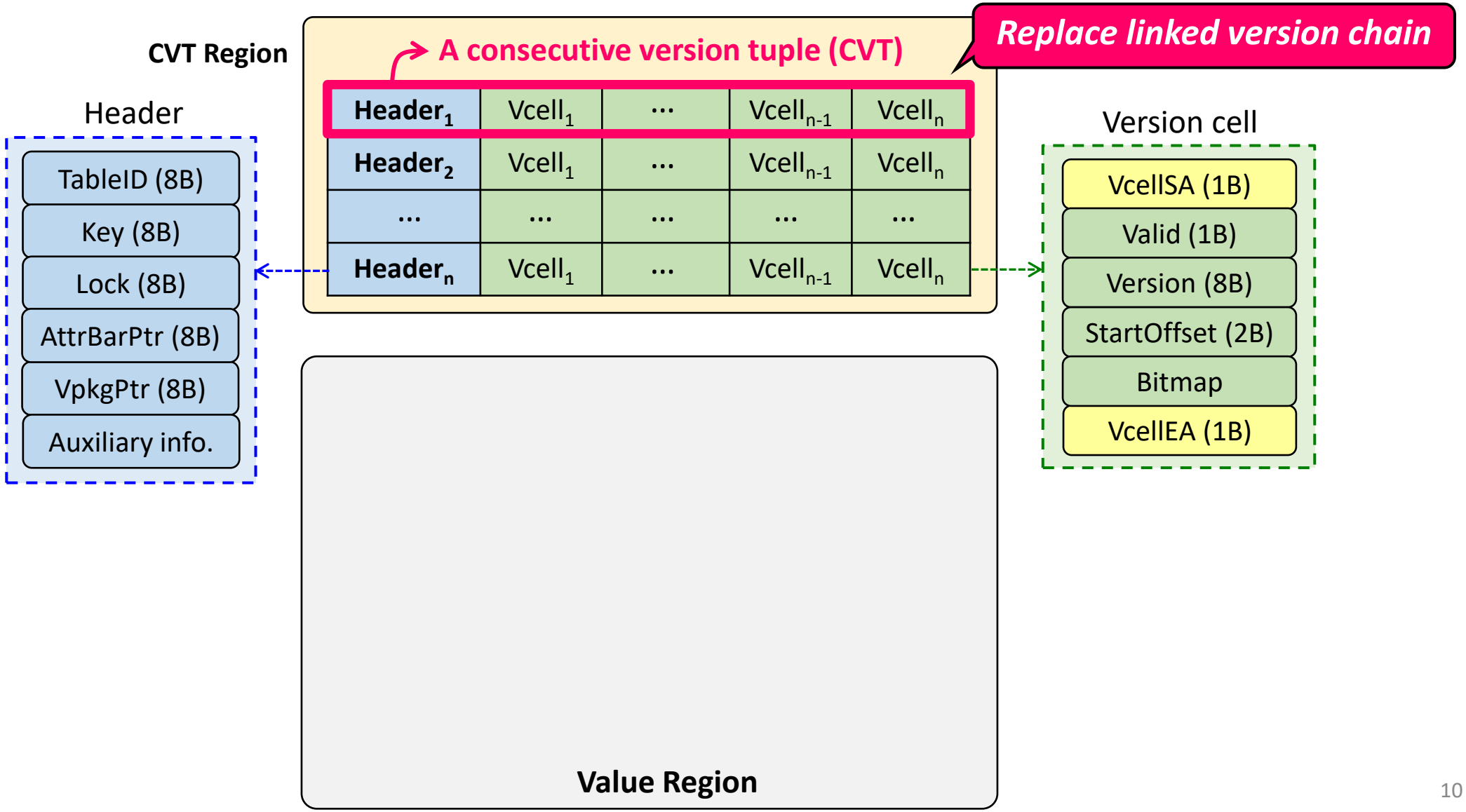
Consecutive Version Tuple



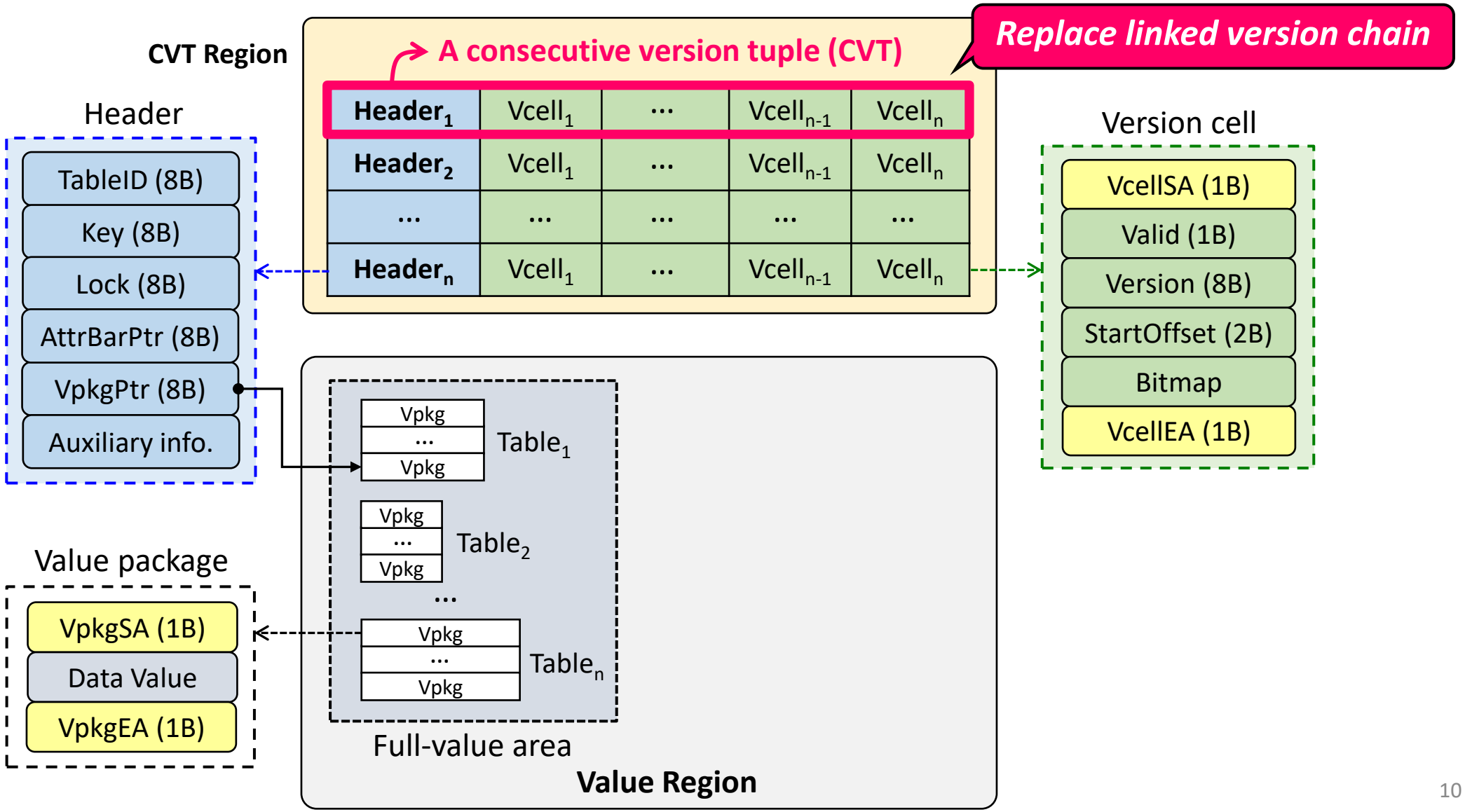
Consecutive Version Tuple



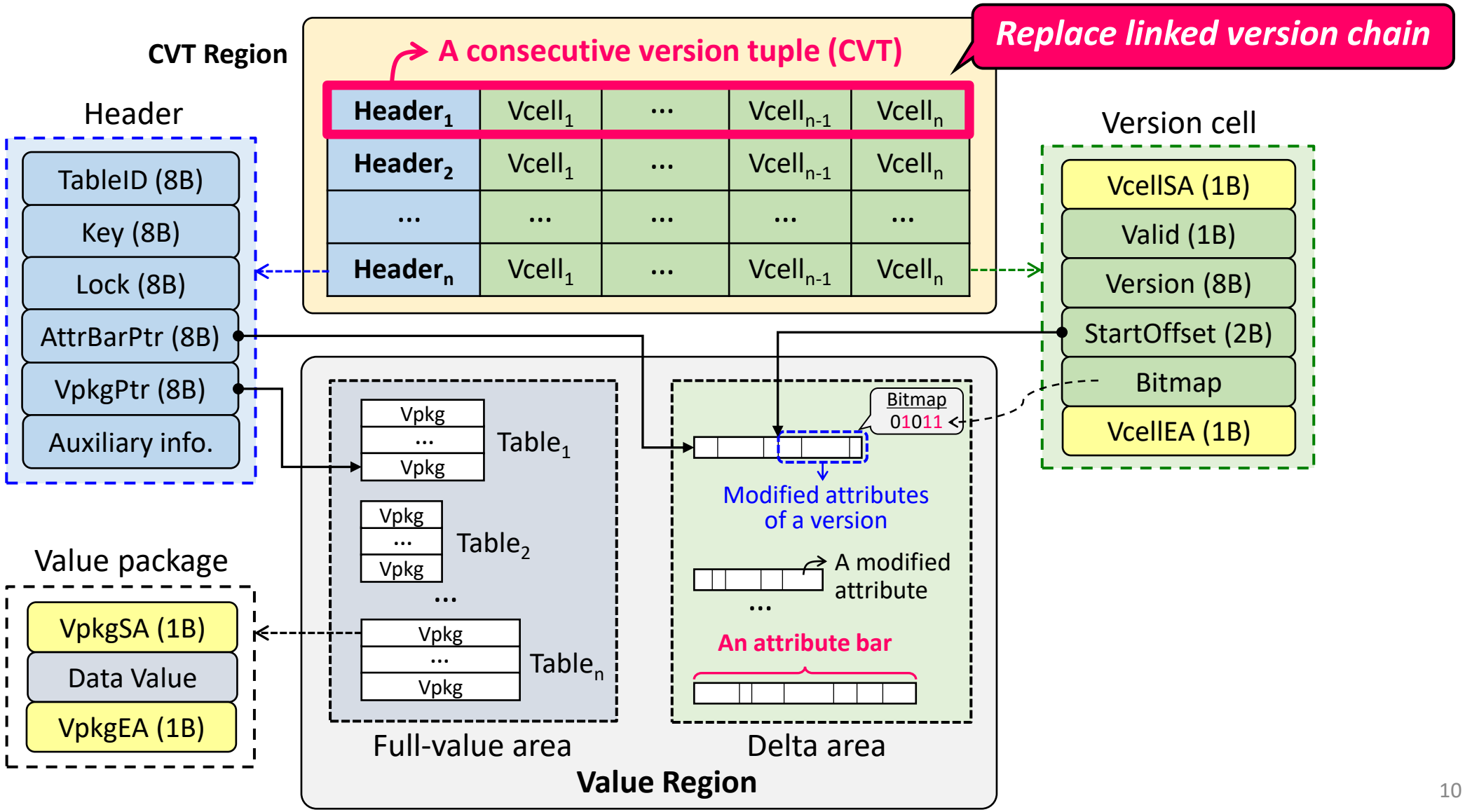
Consecutive Version Tuple



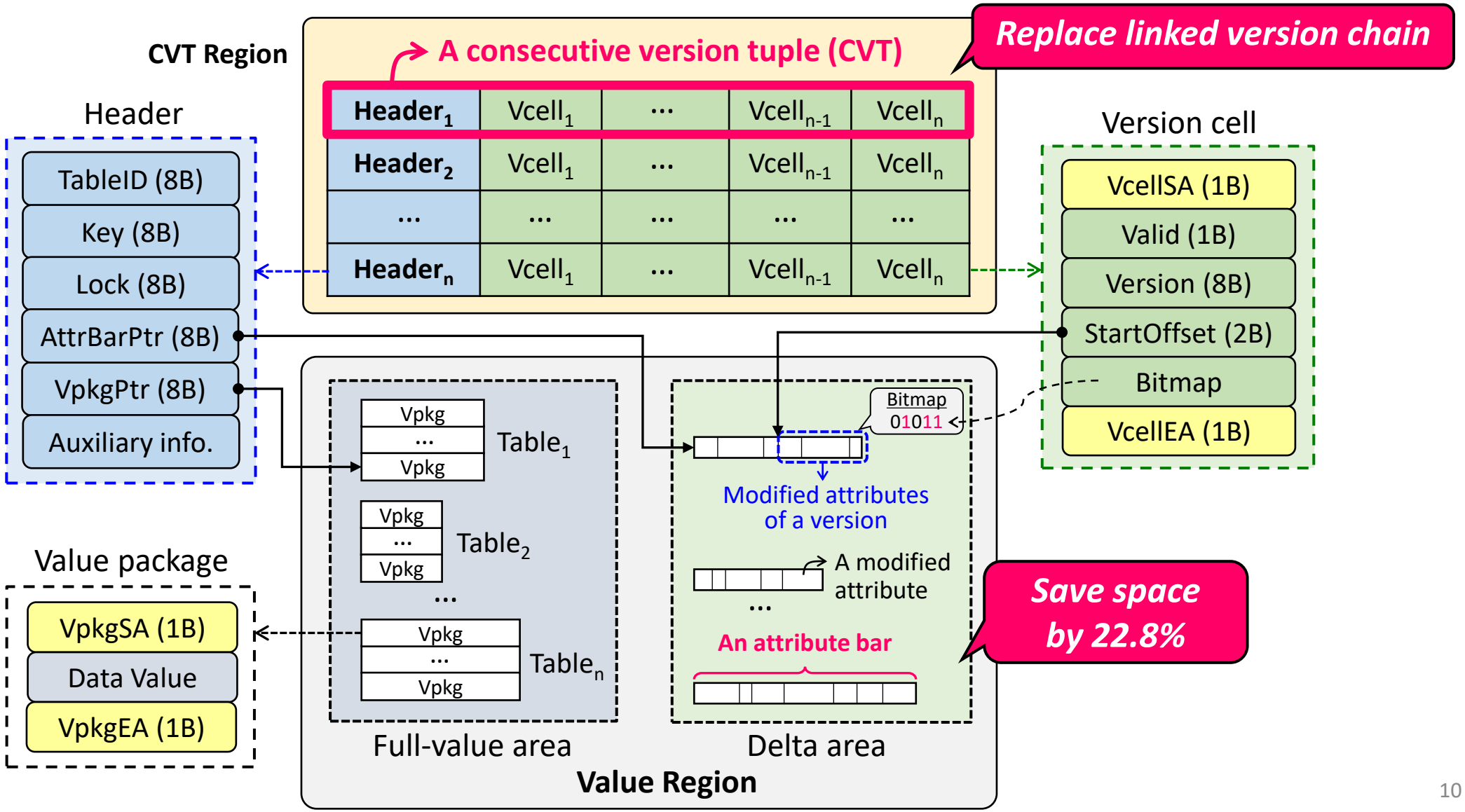
Consecutive Version Tuple



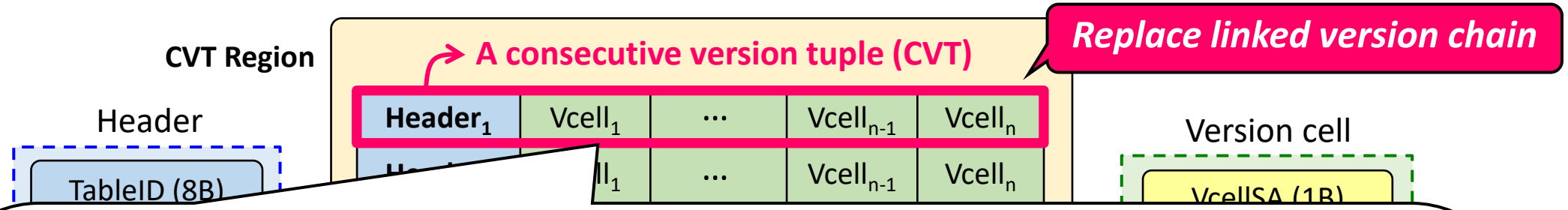
Consecutive Version Tuple



Consecutive Version Tuple

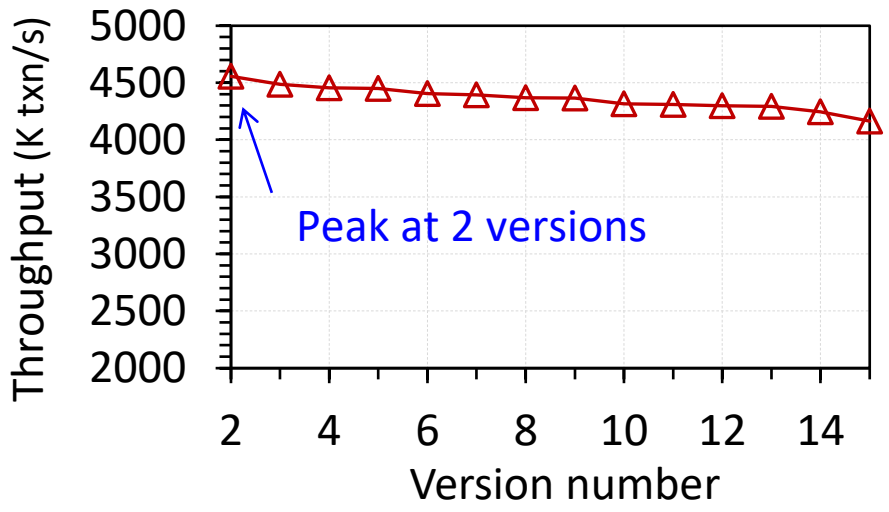


Consecutive Version Tuple

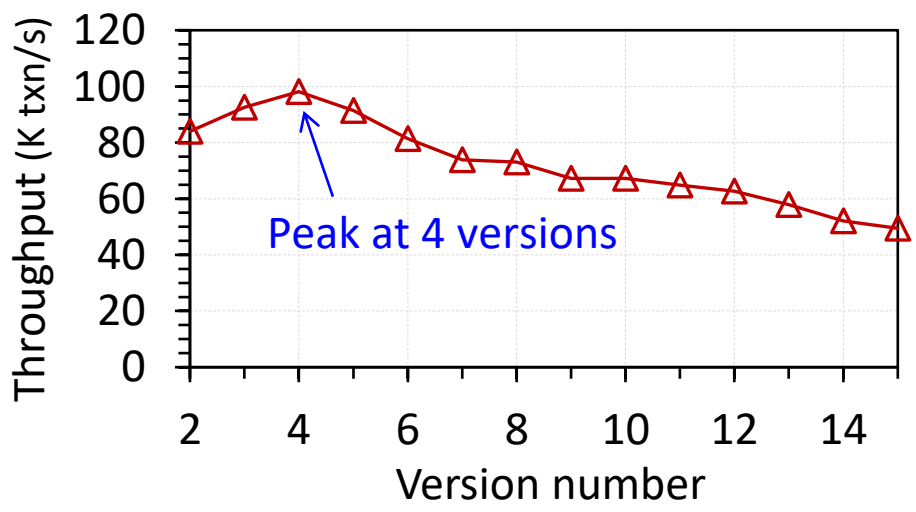


Number of Versions: depending on workload characteristics

- Read-write contention
- Number of accessed records



TATP (low contention, short txns)



TPCC (high contention, long txns)

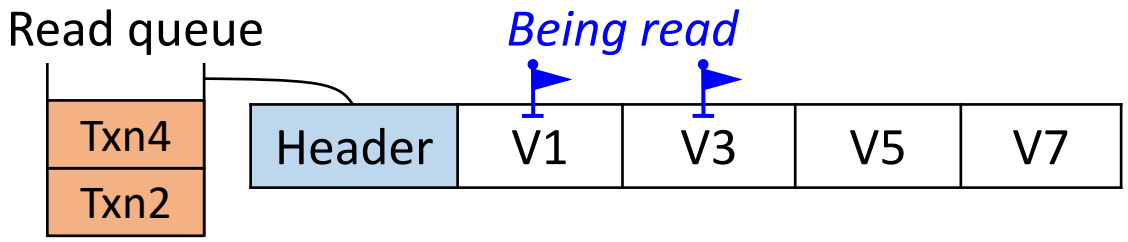
Coordinator-Active Garbage Collection

- A CVT runs out of space – GC required
- Prior systems track transaction states[1-2]
  - CPU in memory nodes is too weak to frequently track

[1] Steam@VLDB'19 [2] FaRMv2@SIGMOD'19

Coordinator-Active Garbage Collection

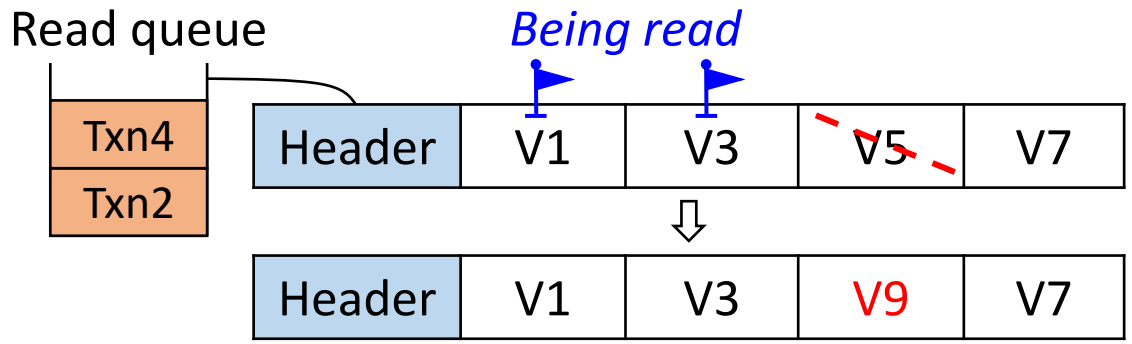
- A CVT runs out of space – GC required
- Prior systems track transaction states[1-2]
  - CPU in memory nodes is too weak to frequently track



[1] Steam@VLDB'19 [2] FaRMv2@SIGMOD'19

Coordinator-Active Garbage Collection

- A CVT runs out of space – GC required
- Prior systems track transaction states[1-2]
  - CPU in memory nodes is too weak to frequently track

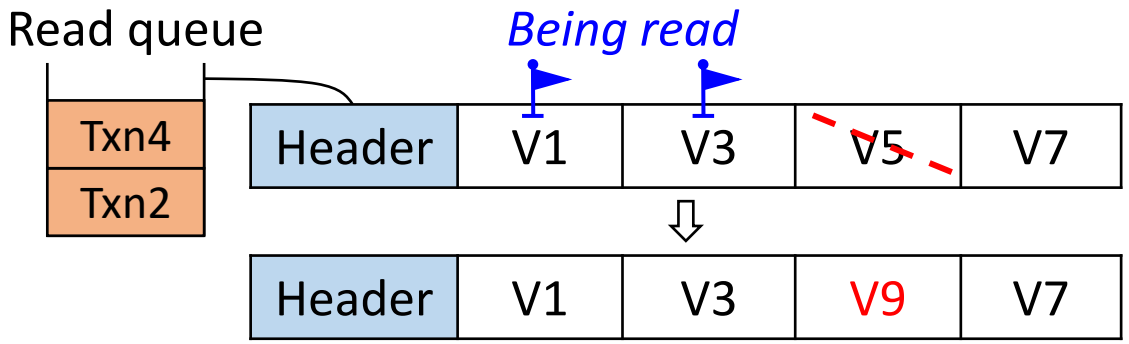


Skip the versions being read

[1] Steam@VLDB'19 [2] FaRMv2@SIGMOD'19

Coordinator-Active Garbage Collection

- A CVT runs out of space – GC required
- Prior systems track transaction states[1-2]
  - CPU in memory nodes is too weak to frequently track



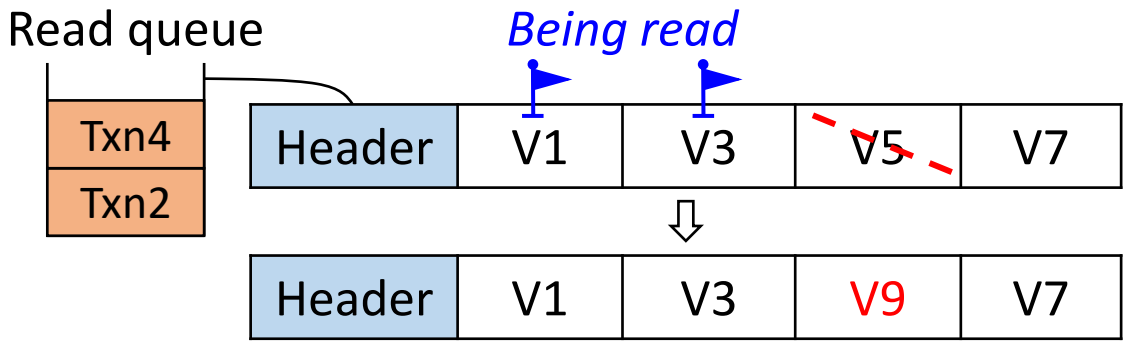
Skip the versions being read

High overhead for compute nodes to maintain remote states

[1] Steam@VLDB'19 [2] FaRMv2@SIGMOD'19

Coordinator-Active Garbage Collection

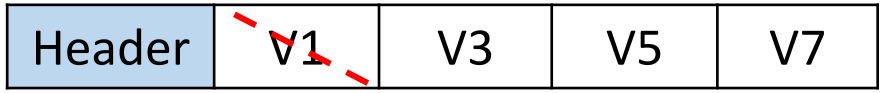
- A CVT runs out of space – GC required
- Prior systems track transaction states[1-2]
  - CPU in memory nodes is too weak to frequently track



Skip the versions being read

High overhead for compute nodes to maintain remote states

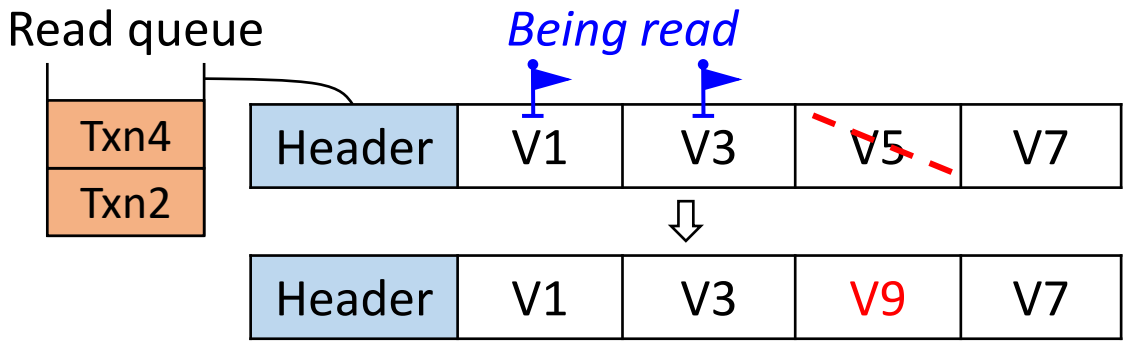
Preemptive selection



[1] Steam@VLDB'19 [2] FaRMv2@SIGMOD'19

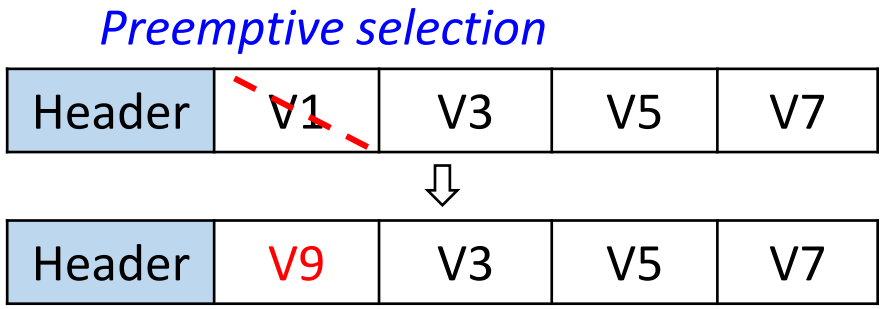
Coordinator-Active Garbage Collection

- A CVT runs out of space – GC required
- Prior systems track transaction states[1-2]
  - CPU in memory nodes is too weak to frequently track



Skip the versions being read

High overhead for compute nodes to maintain remote states

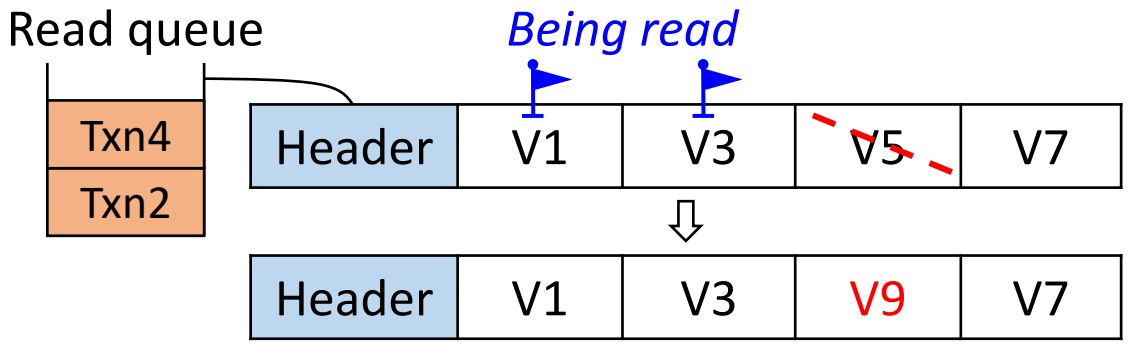


Overwrite the oldest version

[1] Steam@VLDB'19 [2] FaRMv2@SIGMOD'19

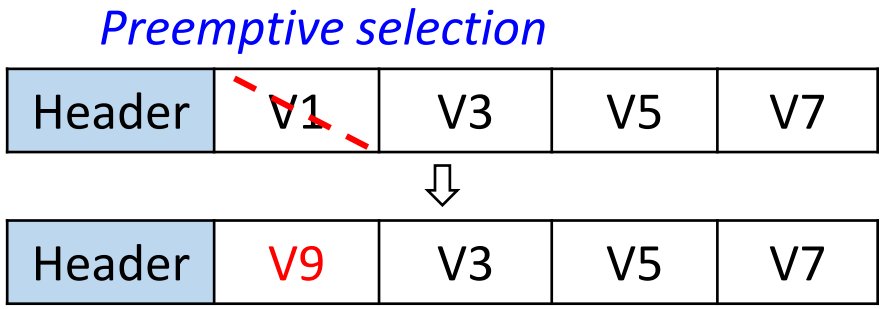
Coordinator-Active Garbage Collection

- A CVT runs out of space – GC required
- Prior systems track transaction states[1-2]
  - CPU in memory nodes is too weak to frequently track



Skip the versions being read

High overhead for compute nodes to maintain remote states



Overwrite the oldest version

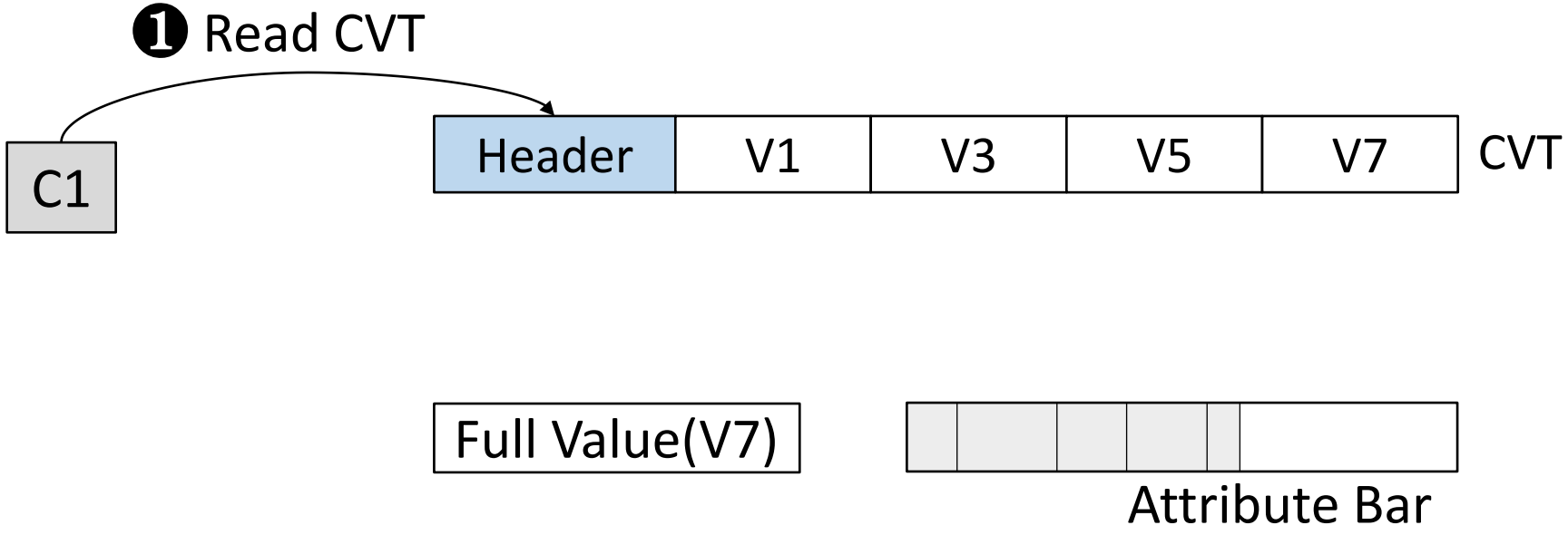
Simple, no tracking

Low abort rate with fast RDMA

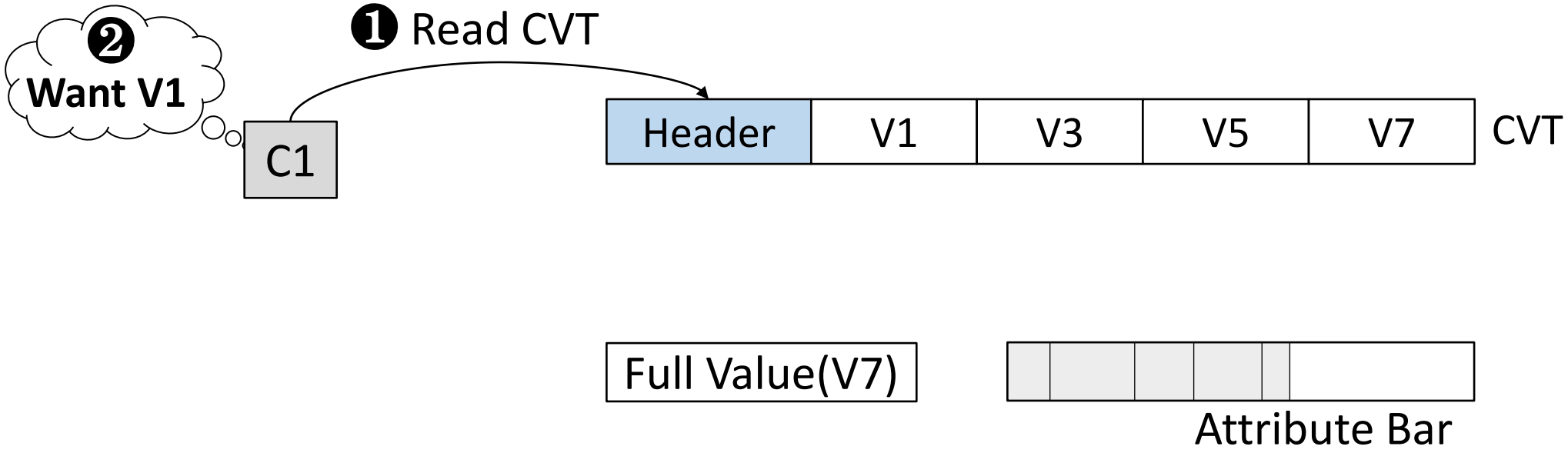
[1] Steam@VLDB'19 [2] FaRMv2@SIGMOD'19

Anchor-Assisted Read

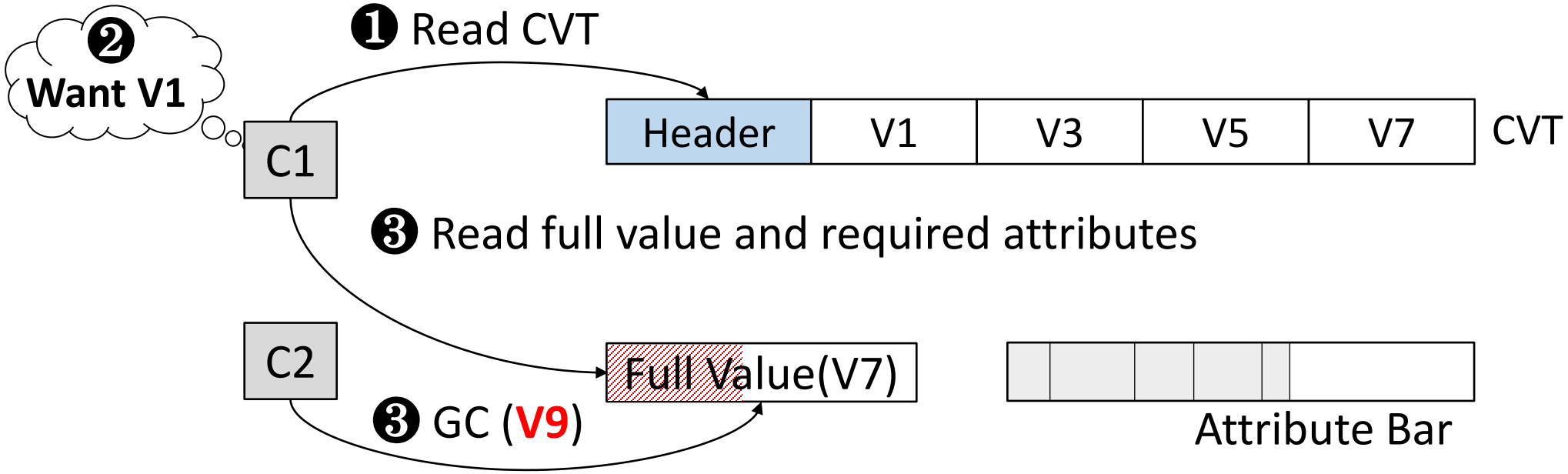
Anchor-Assisted Read



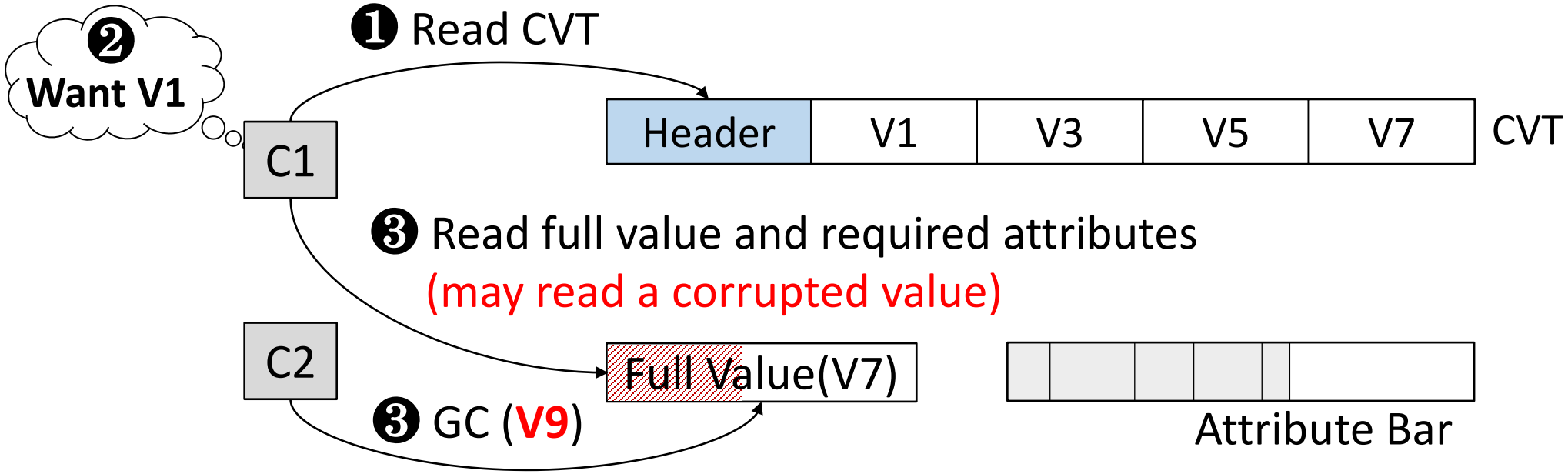
Anchor-Assisted Read



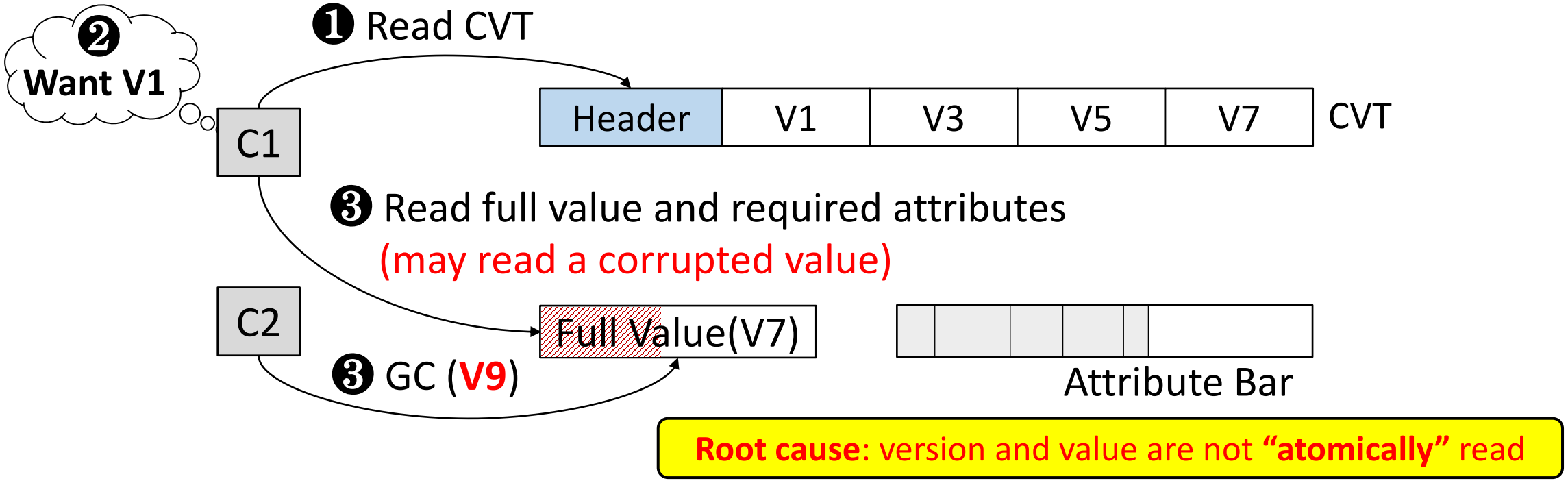
Anchor-Assisted Read



Anchor-Assisted Read

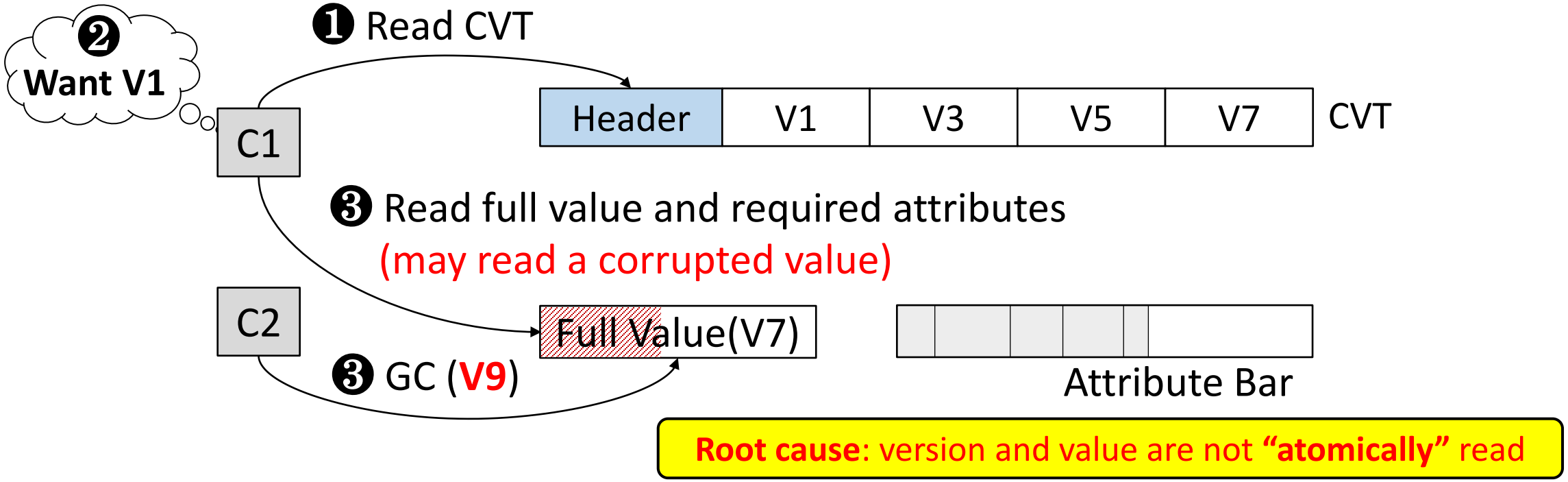


Anchor-Assisted Read

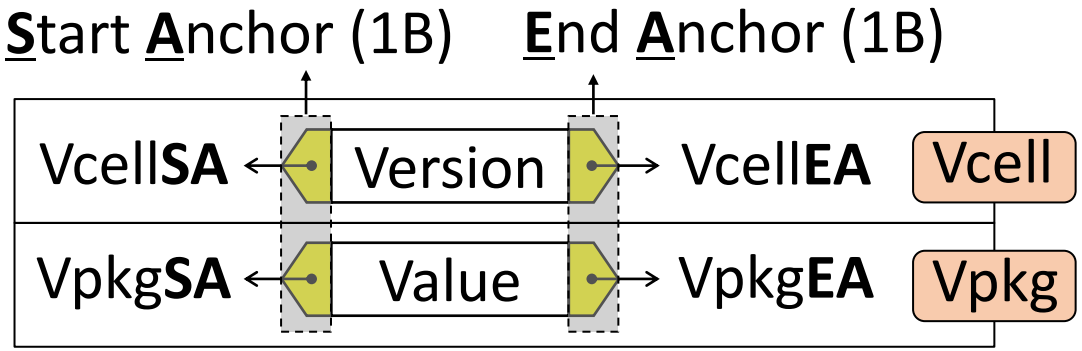


Root cause: version and value are not **"atomically"** read

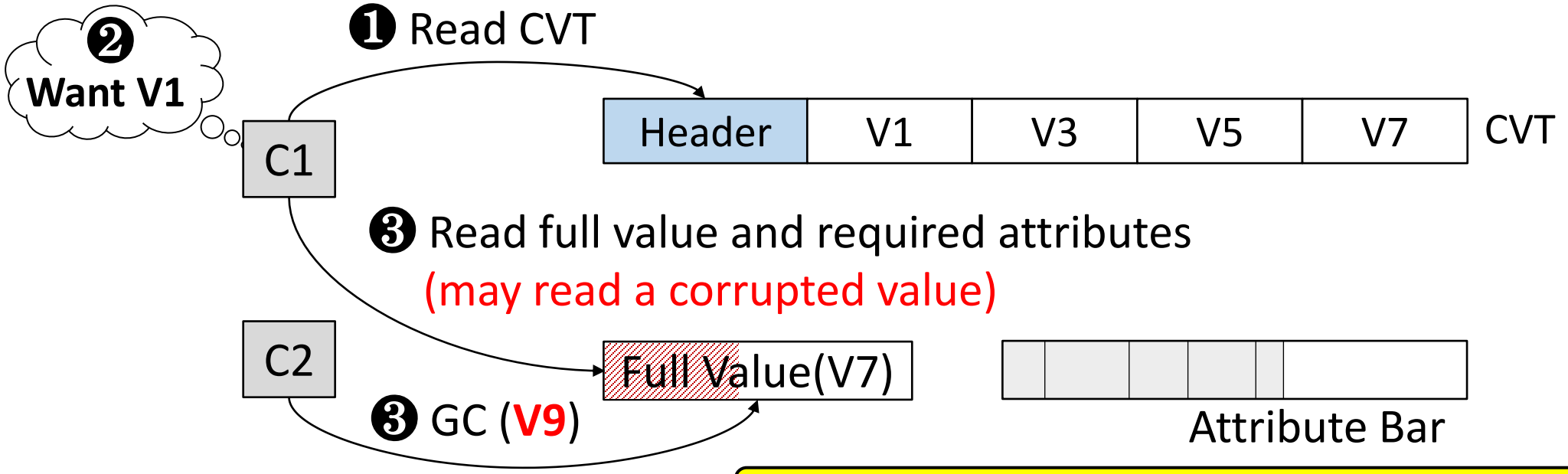
Anchor-Assisted Read



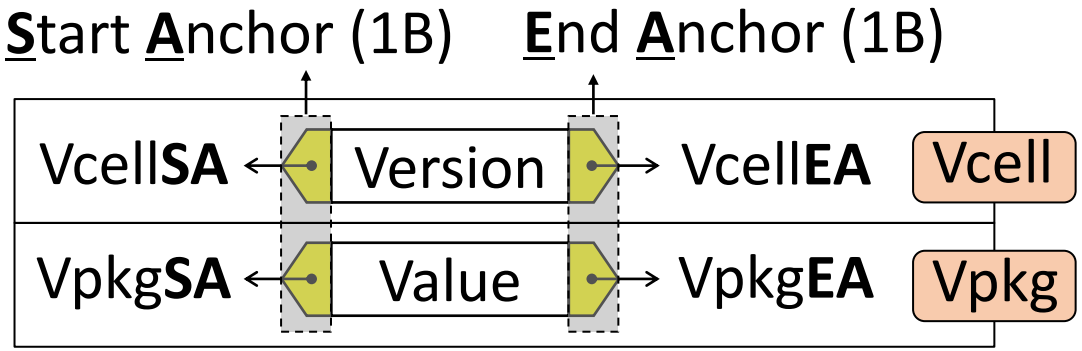
Root cause: version and value are not **"atomically"** read



Anchor-Assisted Read

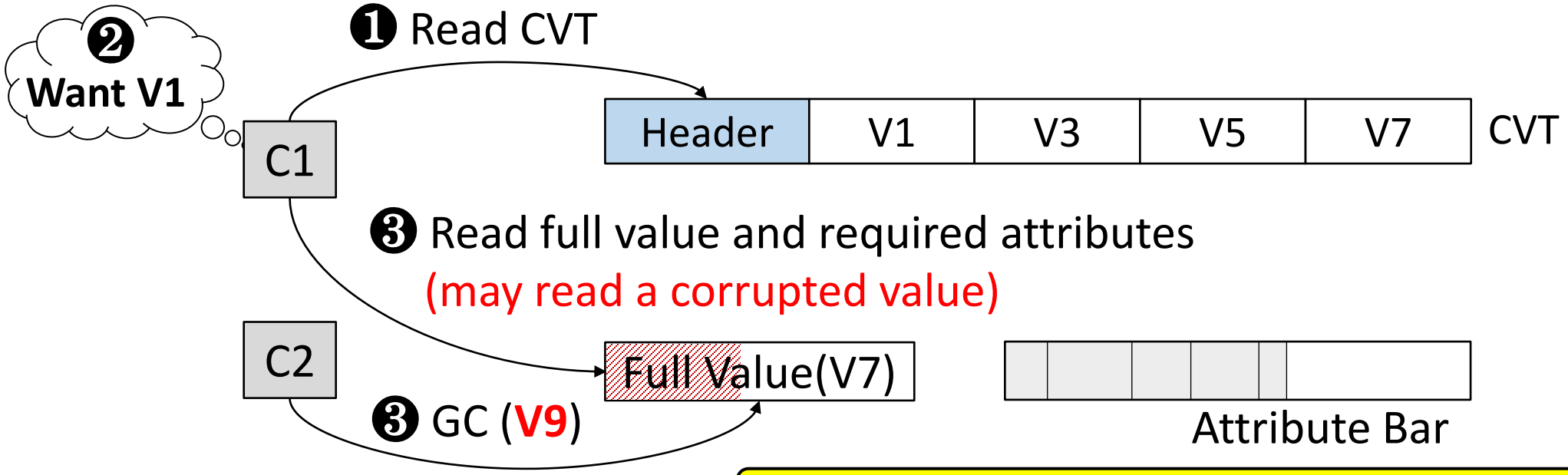


Root cause: version and value are not **"atomically"** read

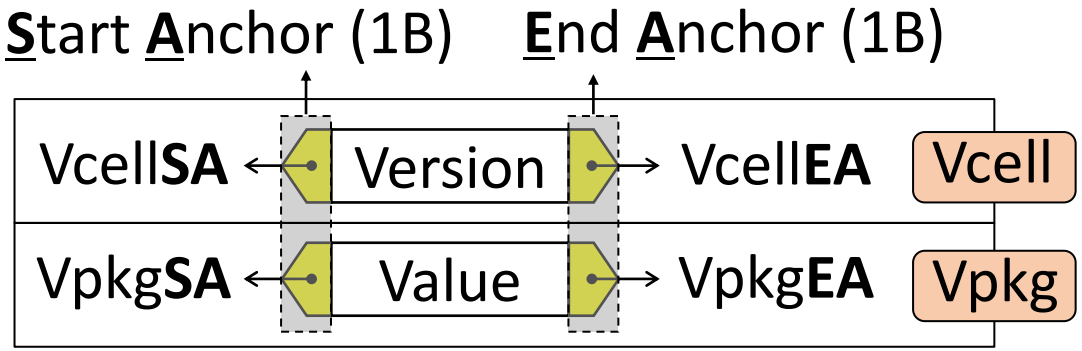


- **Writer:** Vpkg → attributes → Vcell

Anchor-Assisted Read

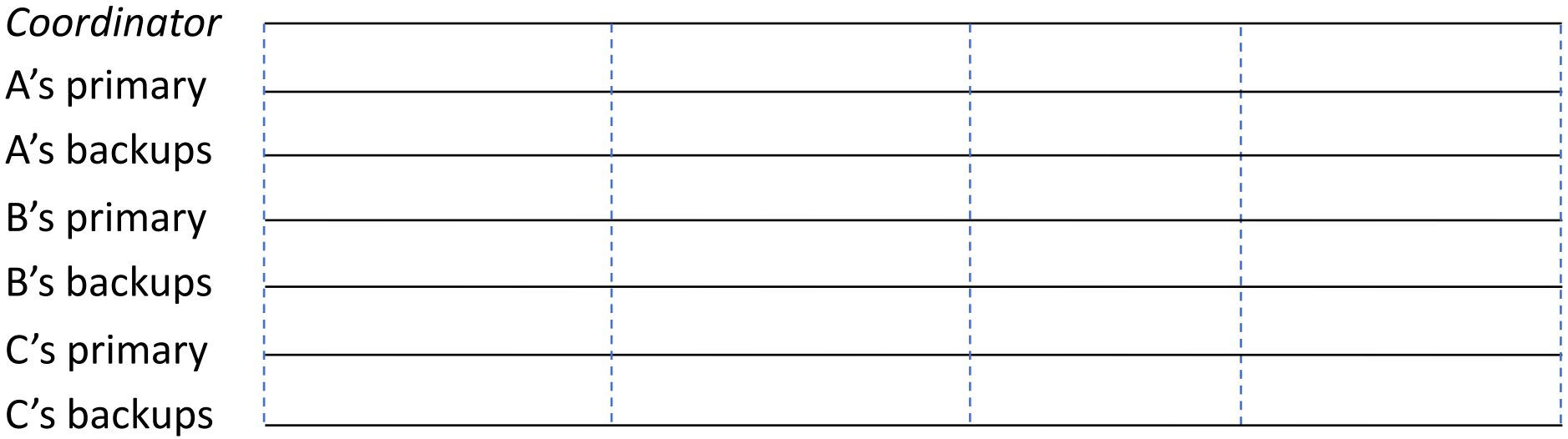


Root cause: version and value are not **"atomically"** read



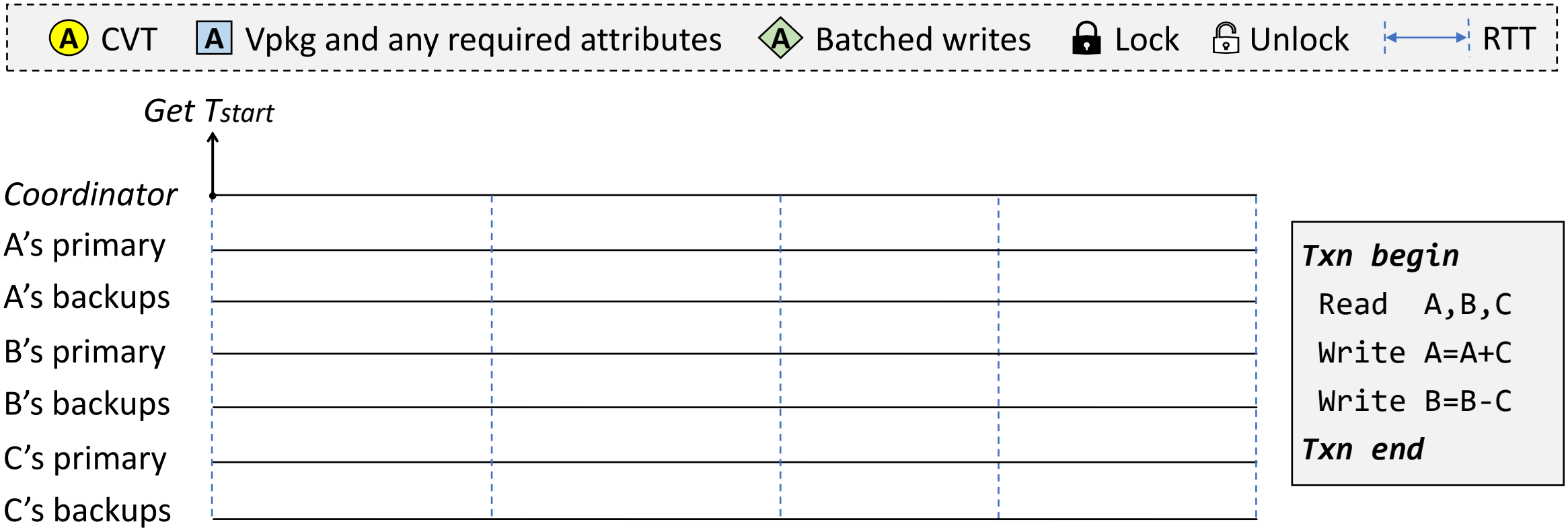
- **Writer:** Vpkg → attributes → Vcell
- **Reader:** check "all anchors are equal"
  VcellSA = VcellEA = VpkgSA = VpkgEA ✓

One-Sided RDMA-Based MVCC



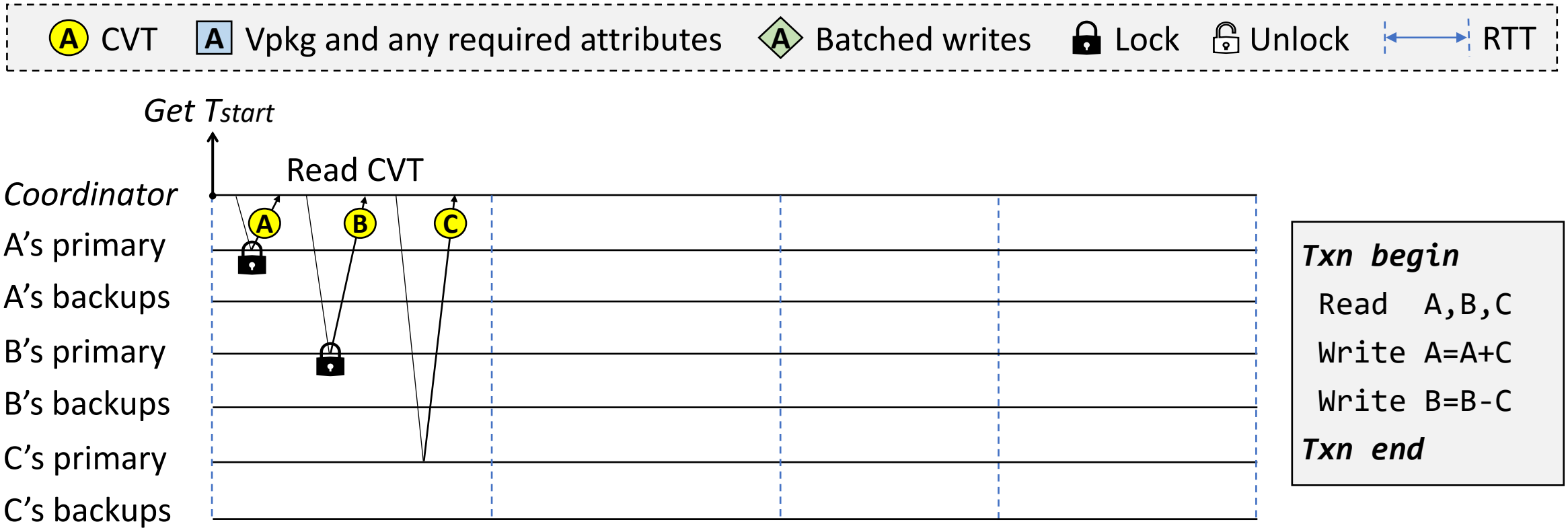

```
Txn begin
 Read  A,B,C
 Write A=A+C
 Write B=B-C
Txn end
```

One-Sided RDMA-Based MVCC



* Tstart/Tcommit stands for start/commit timestamp

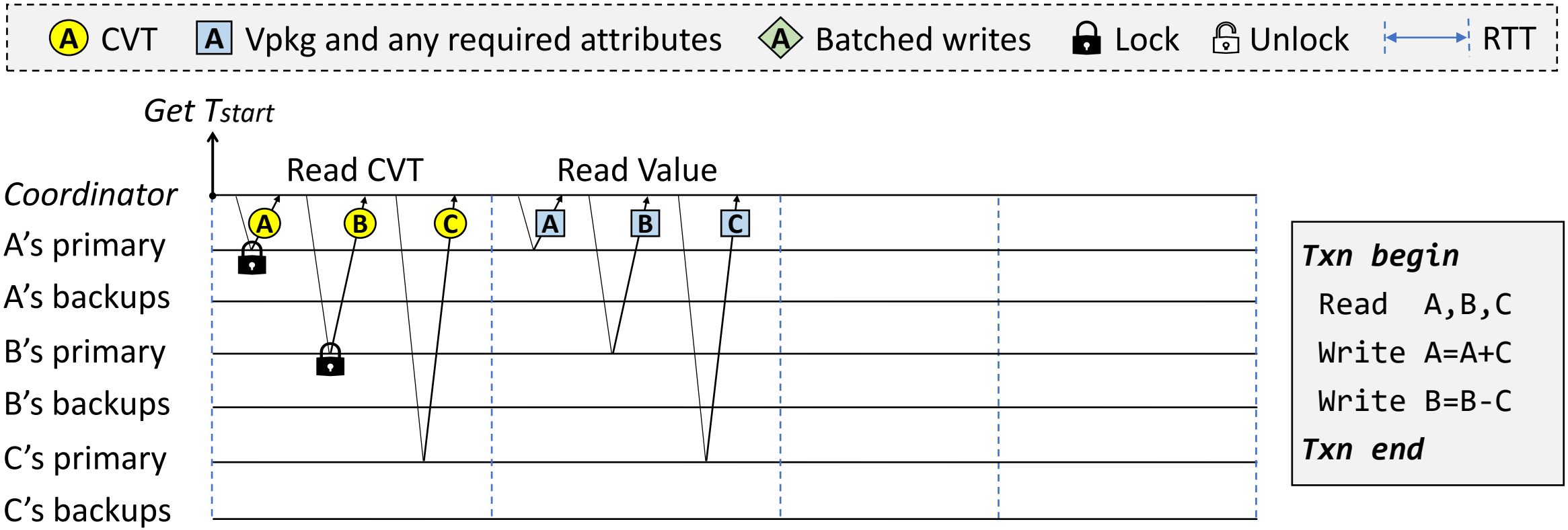
One-Sided RDMA-Based MVCC




```
Txn begin
 Read  A,B,C
 Write A=A+C
 Write B=B-C
Txn end
```

* Tstart/Tcommit stands for start/commit timestamp

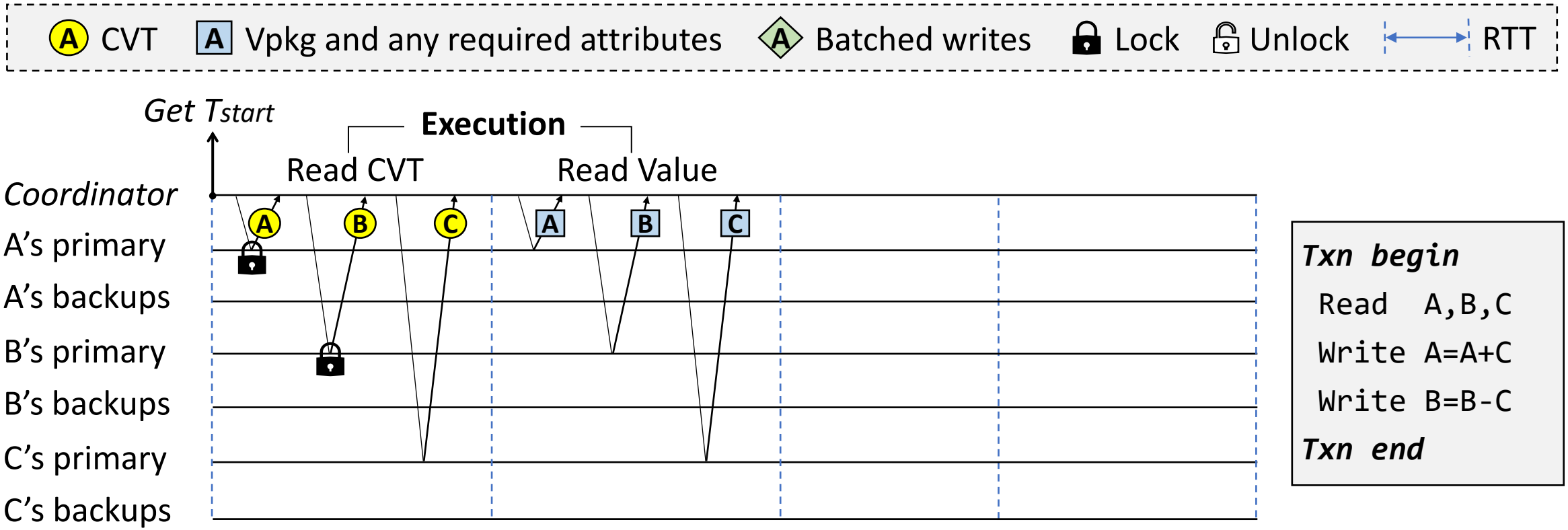
One-Sided RDMA-Based MVCC




```
Txn begin
 Read  A,B,C
 Write A=A+C
 Write B=B-C
Txn end
```

* Tstart/Tcommit stands for start/commit timestamp

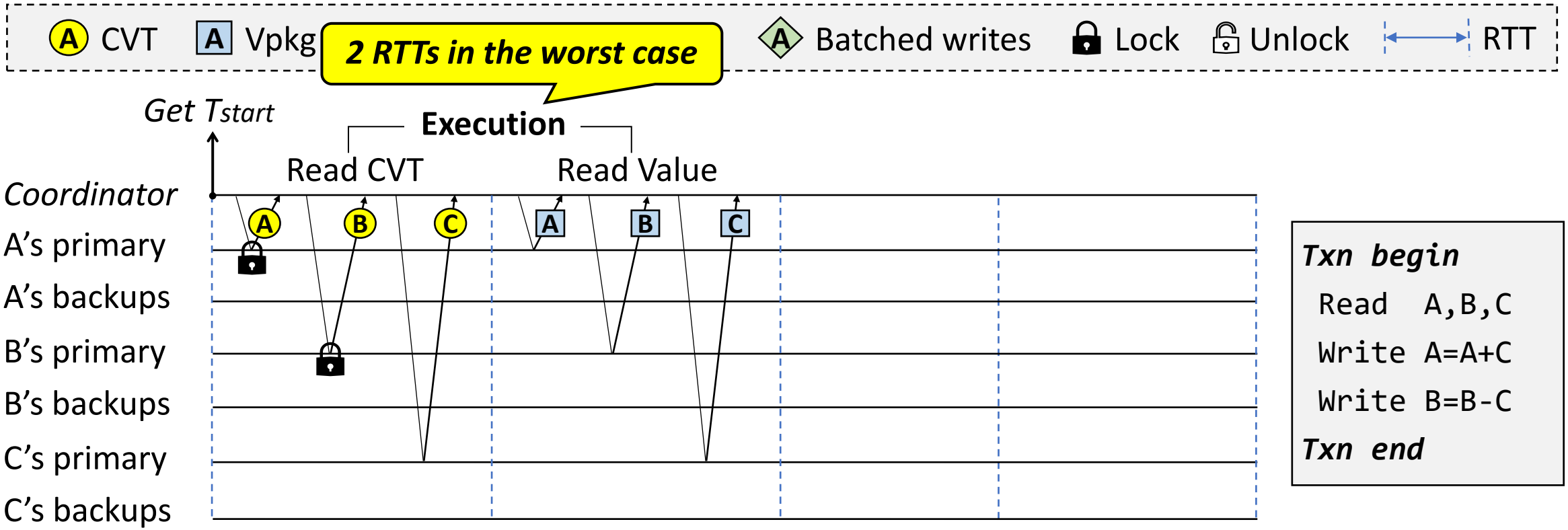
One-Sided RDMA-Based MVCC




```
Txn begin
 Read  A,B,C
 Write A=A+C
 Write B=B-C
Txn end
```

* Tstart/Tcommit stands for start/commit timestamp

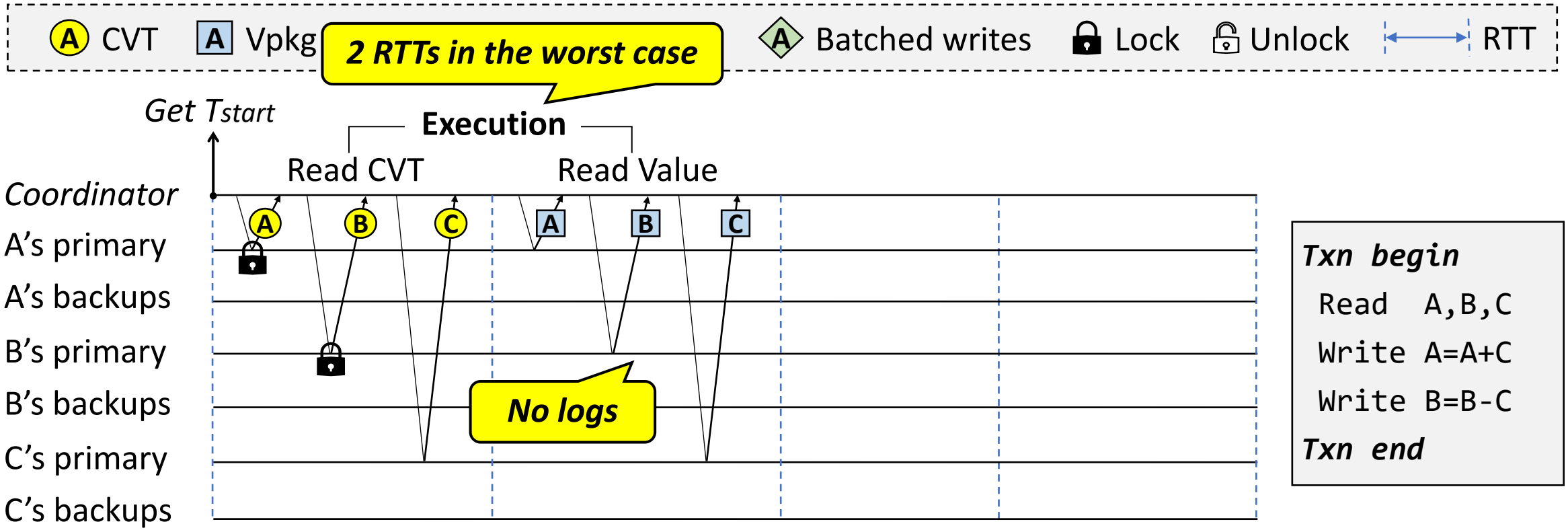
One-Sided RDMA-Based MVCC




```
Txn begin
 Read  A,B,C
 Write A=A+C
 Write B=B-C
Txn end
```

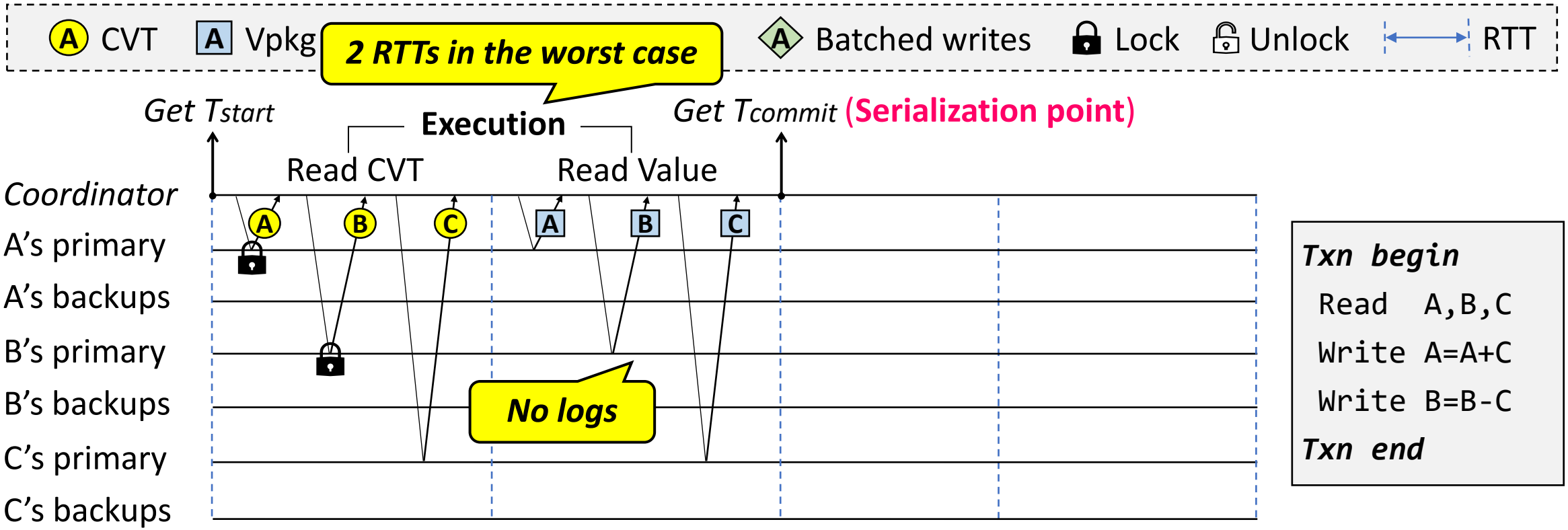
* Tstart/Tcommit stands for start/commit timestamp

One-Sided RDMA-Based MVCC



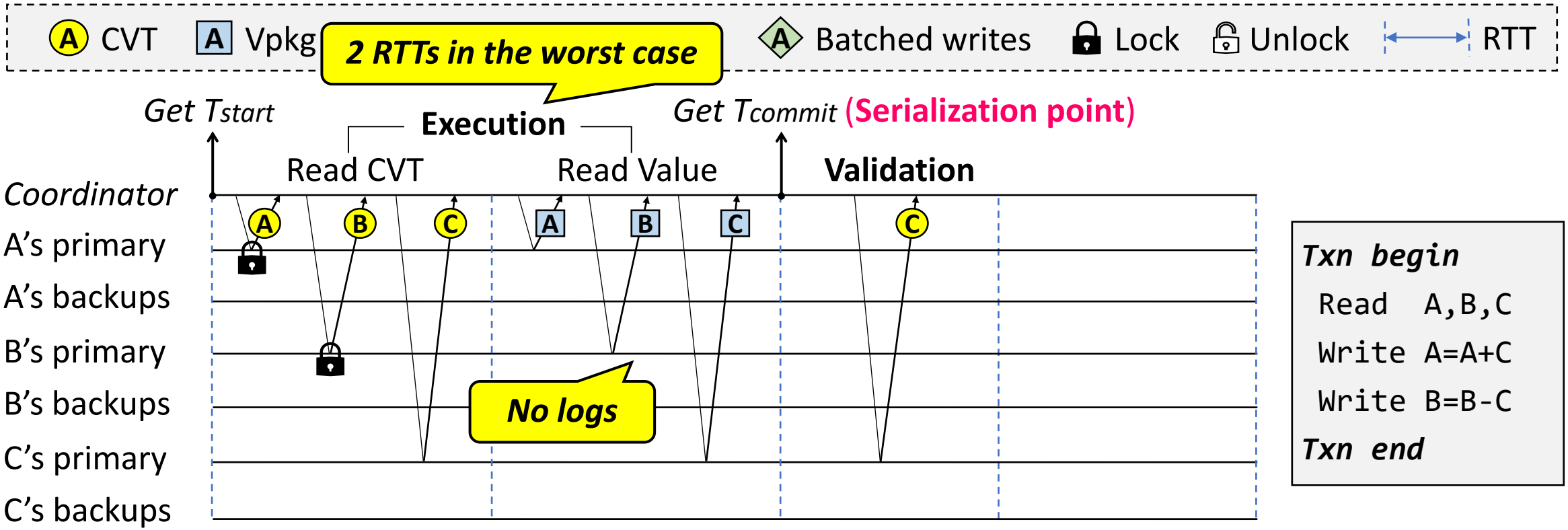
* Tstart/Tcommit stands for start/commit timestamp

One-Sided RDMA-Based MVCC



* Tstart/Tcommit stands for start/commit timestamp

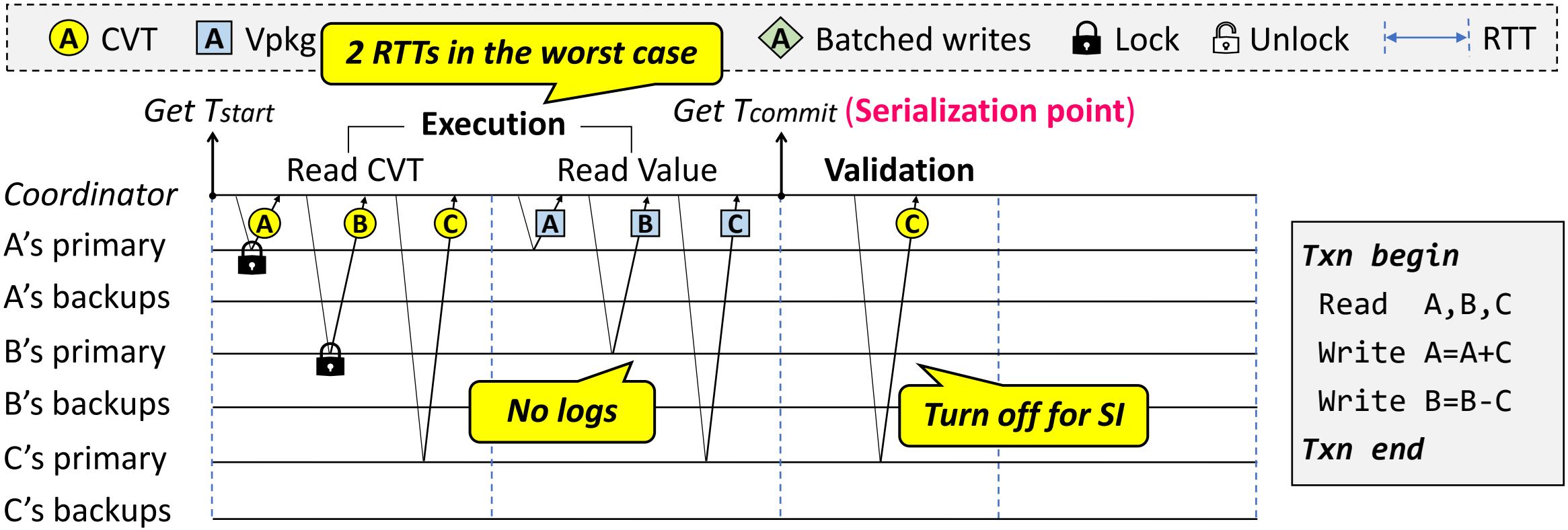
One-Sided RDMA-Based MVCC




```
Txn begin
 Read  A,B,C
 Write A=A+C
 Write B=B-C
Txn end
```

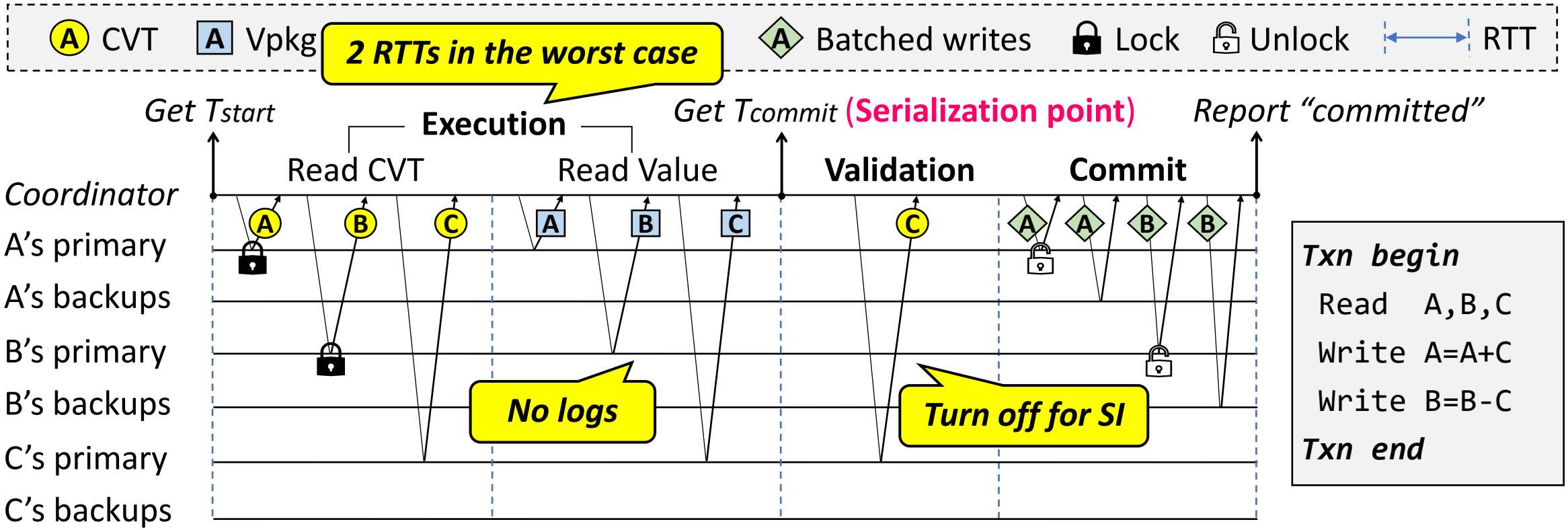
* Tstart/Tcommit stands for start/commit timestamp

One-Sided RDMA-Based MVCC



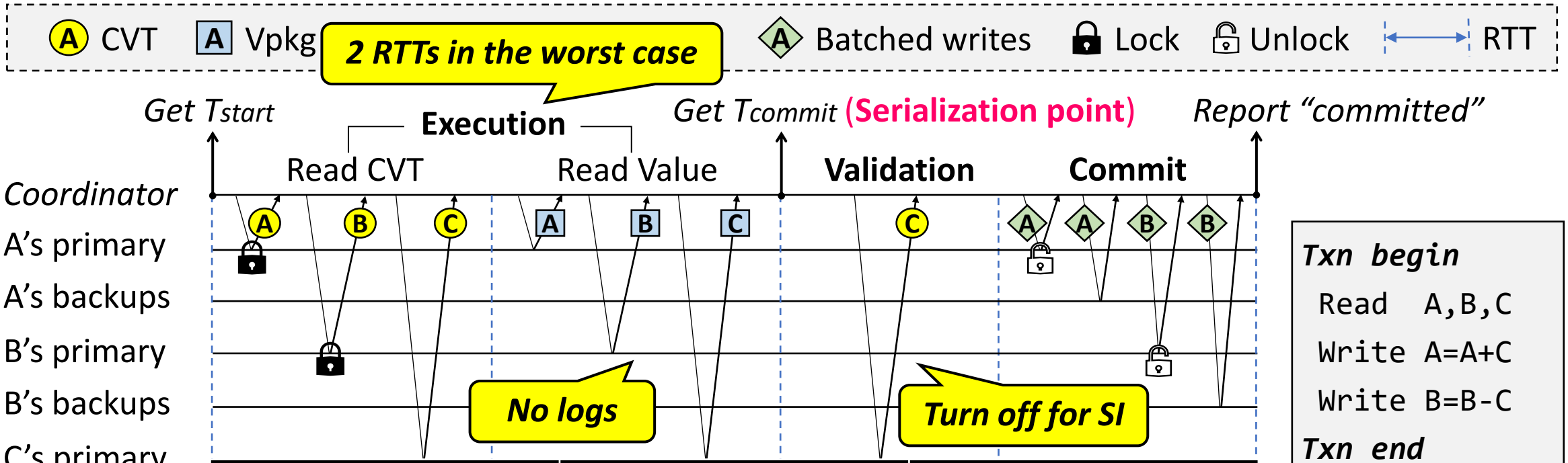
* Tstart/Tcommit stands for start/commit timestamp

One-Sided RDMA-Based MVCC



* Tstart/Tcommit stands for start/commit timestamp

One-Sided RDMA-Based MVCC




```
Txn begin
 Read  A,B,C
 Write A=A+C
 Write B=B-C
Txn end
```

System	RTTs	Data Version
DrTM+H@OSDI'18	4	Single Version
FORD@FAST'22	3	Single Version
FaRMv2@SIGMOD'19	5	Multiple Version
Motor	**≤ 4**	**Multiple Version**

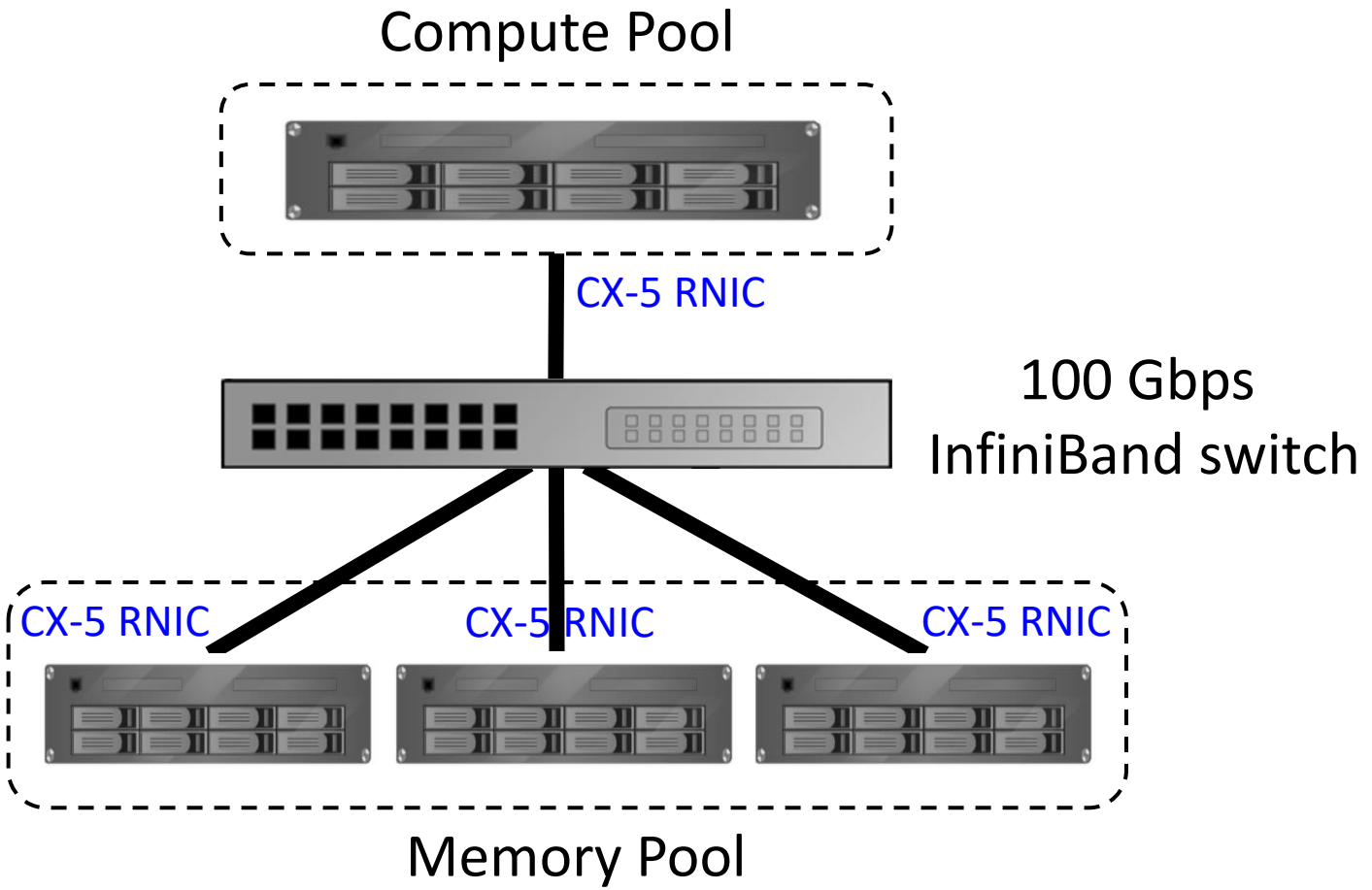
Deliver MV benefits without long RTTs

* Tstart/Tcommit stands for start/commit timestamp

Evaluation

➢ Workloads

- KV store
  - 8B key + 40B value
  - Skewed (skewness tunable)
- TATP
  - RO/RW: 80%/20%, max 48B
- SmallBank
  - RO/RW: 15%/85%, 16B
- TPCC
  - RO/RW: 8%/92%, max 672B

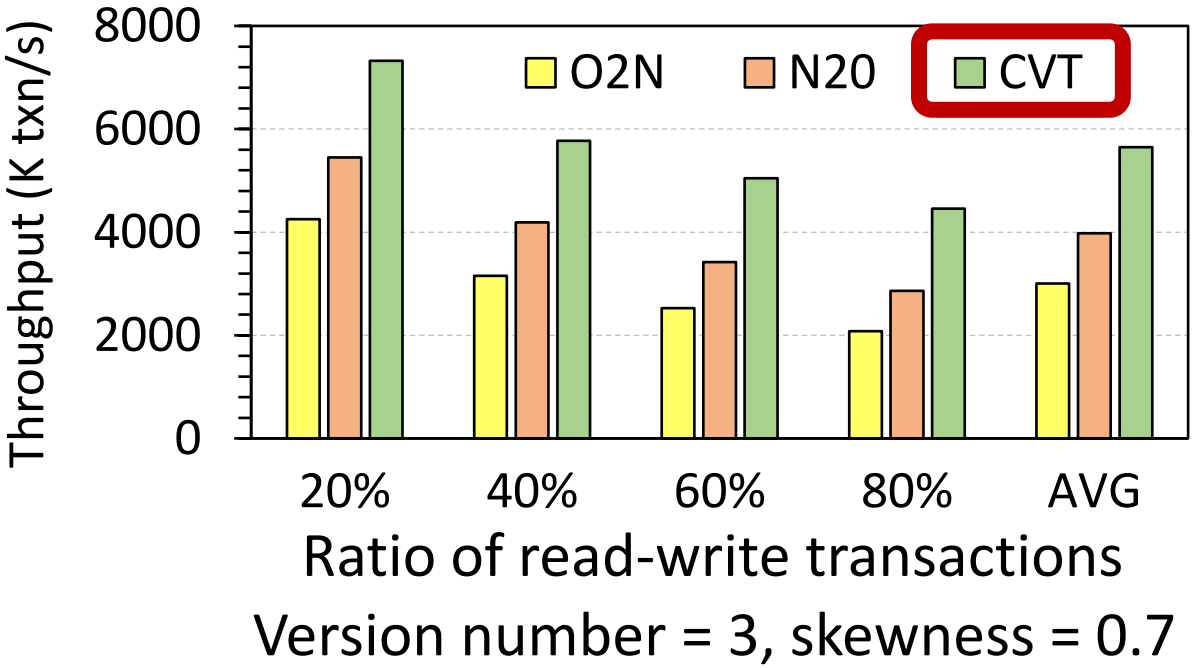


➢ Comparisons

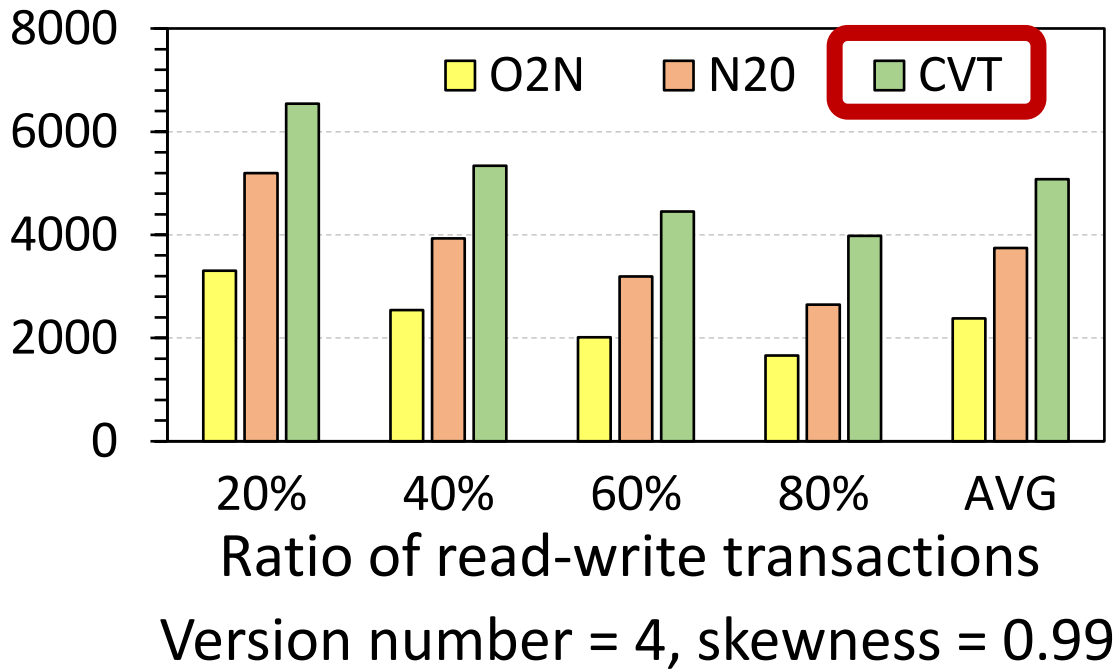
- FaRMv2@SIGMOD'19 (referred as FaRMv2-DM)
- FORD@FAST'22

Performance of Version Structures

- KV store



Version number = 3, skewness = 0.7

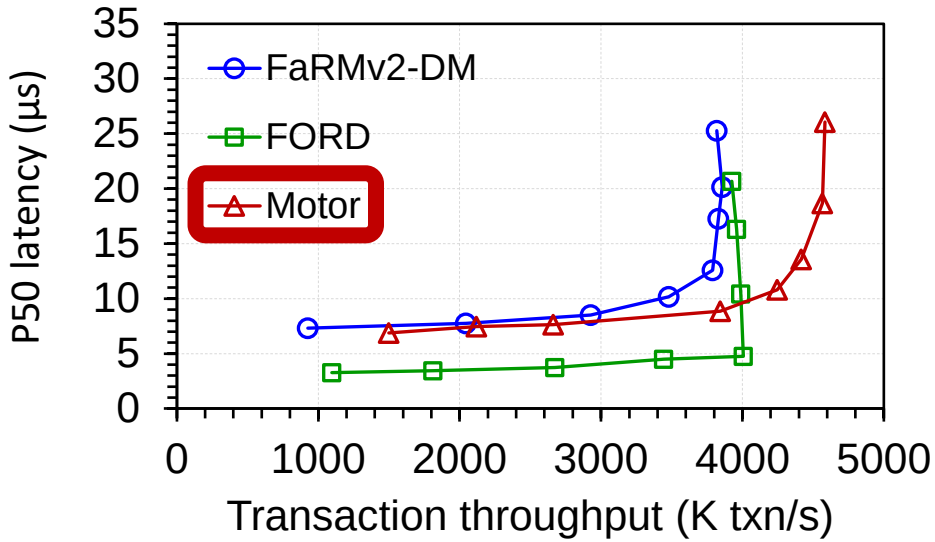


Version number = 4, skewness = 0.99

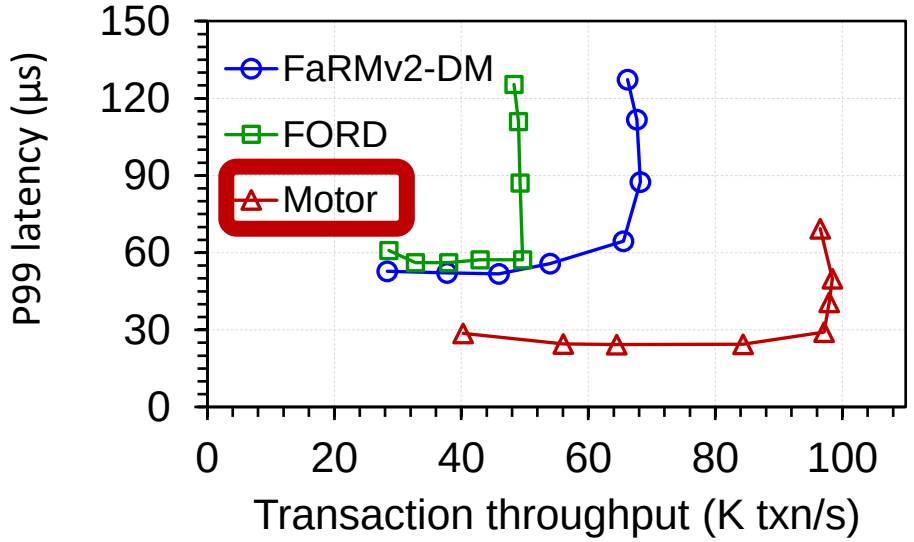
CVT improves throughput by
- 1.7–2.4x over O2N
- 1.3–1.6x over N2O

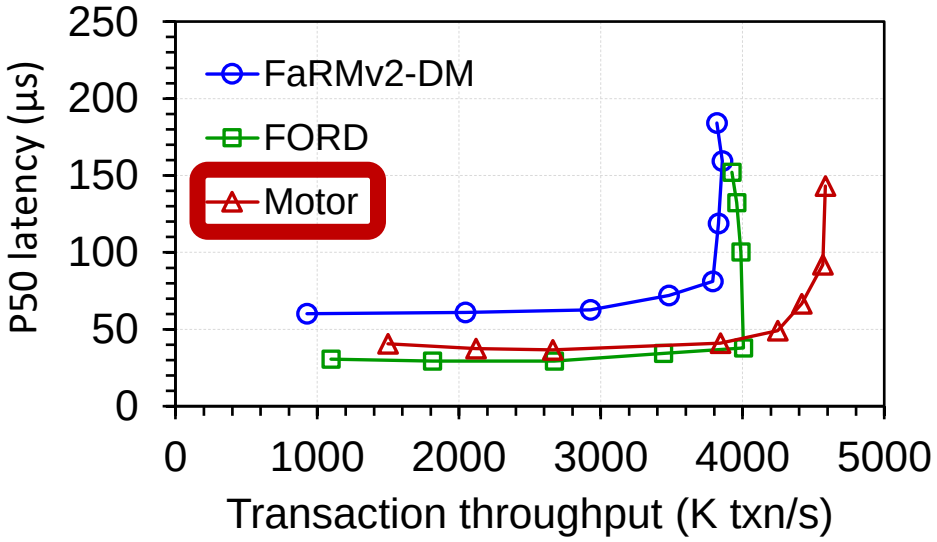
End-to-End Performance

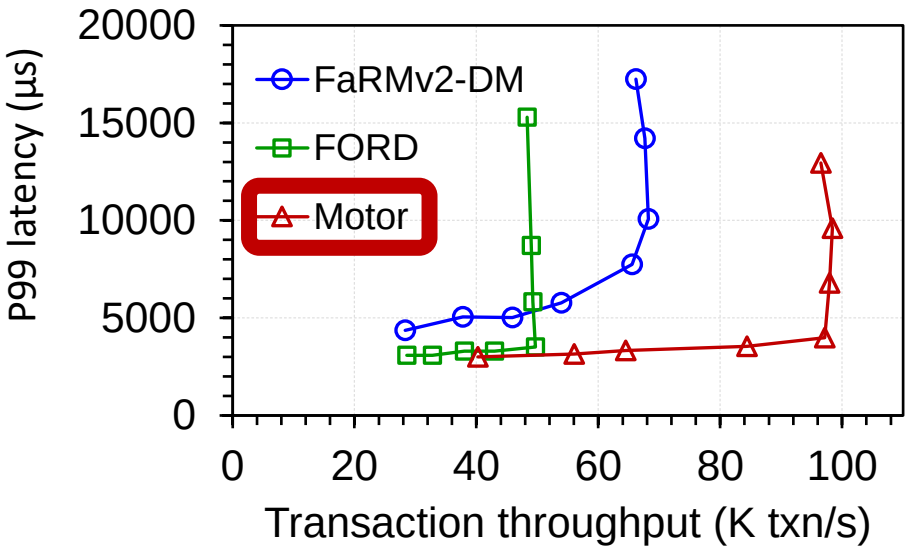
TATP (read-intensive)



TPCC (write-intensive)

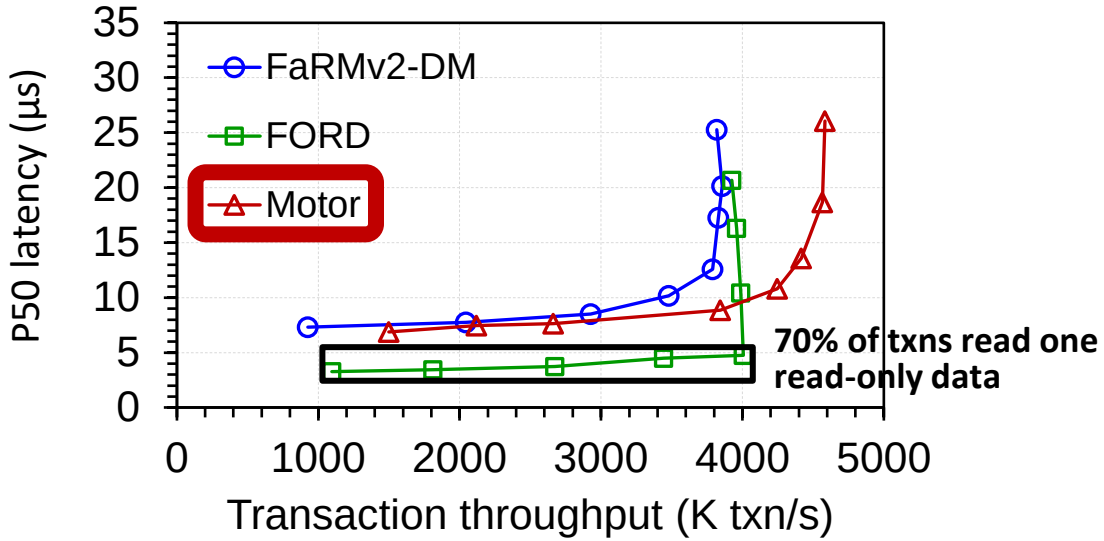




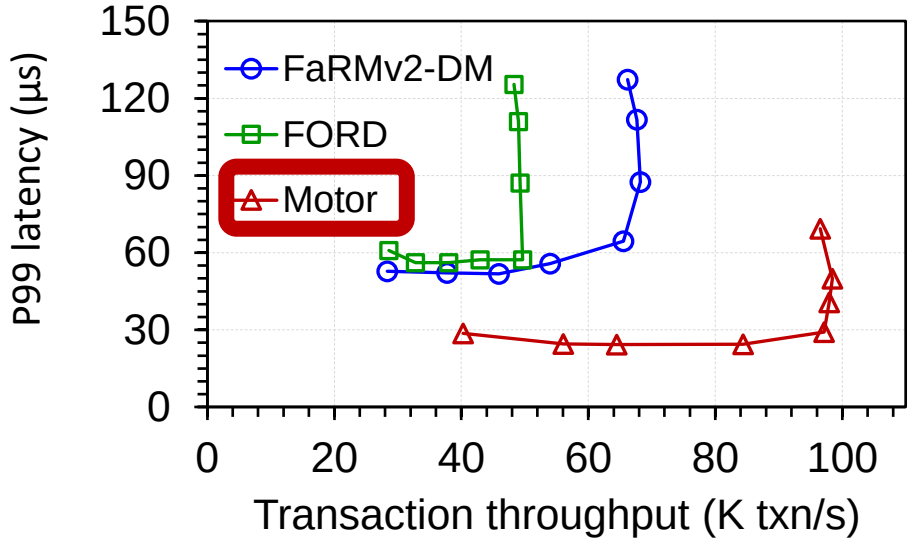


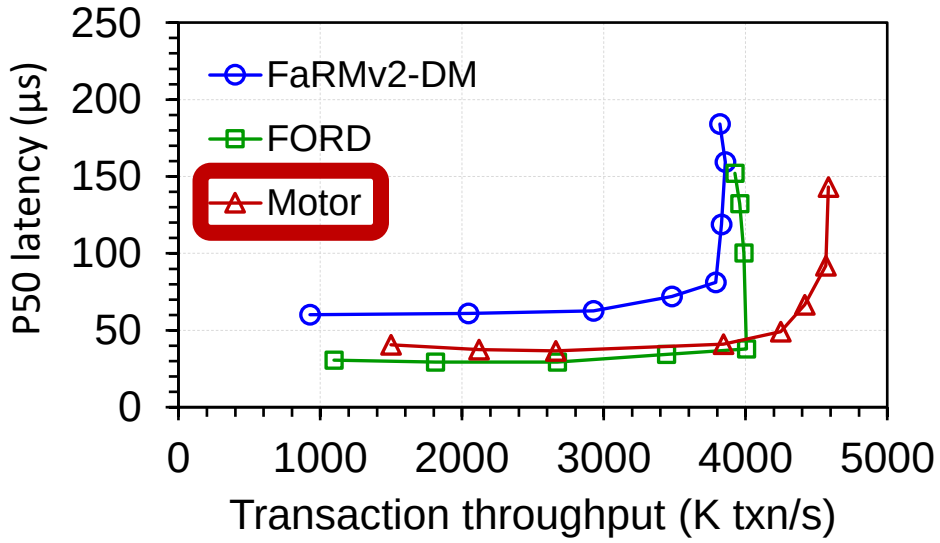
End-to-End Performance

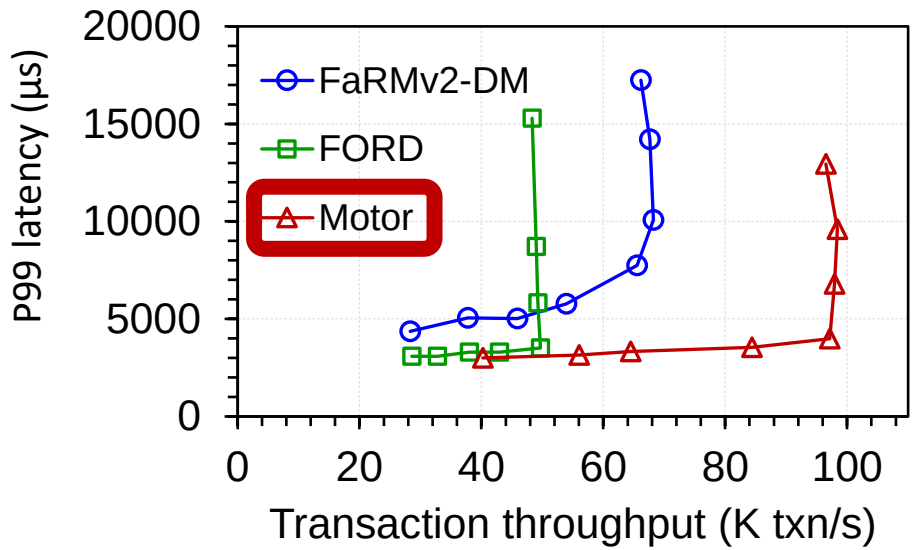
TATP (read-intensive)



TPCC (write-intensive)

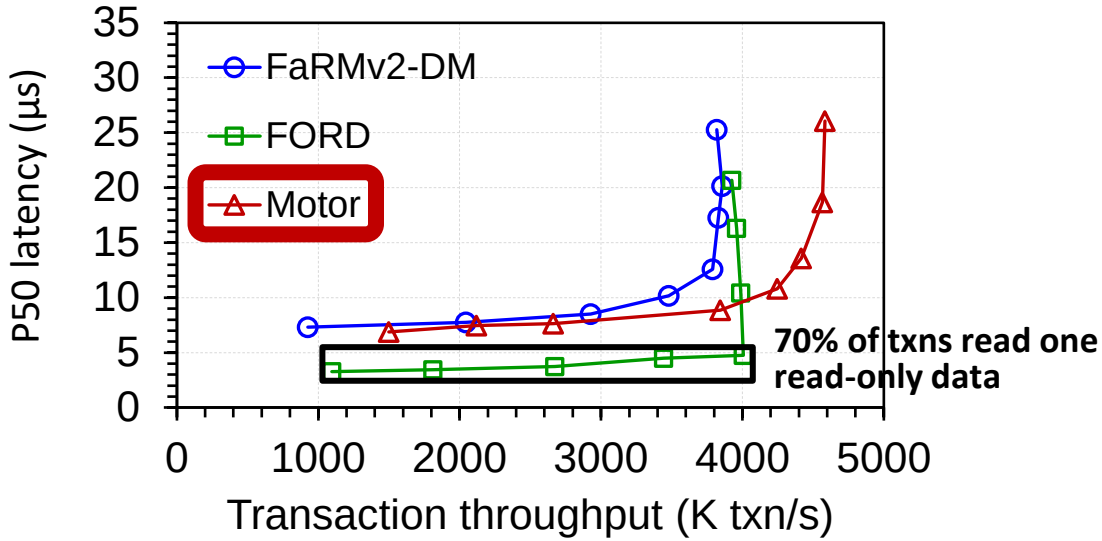




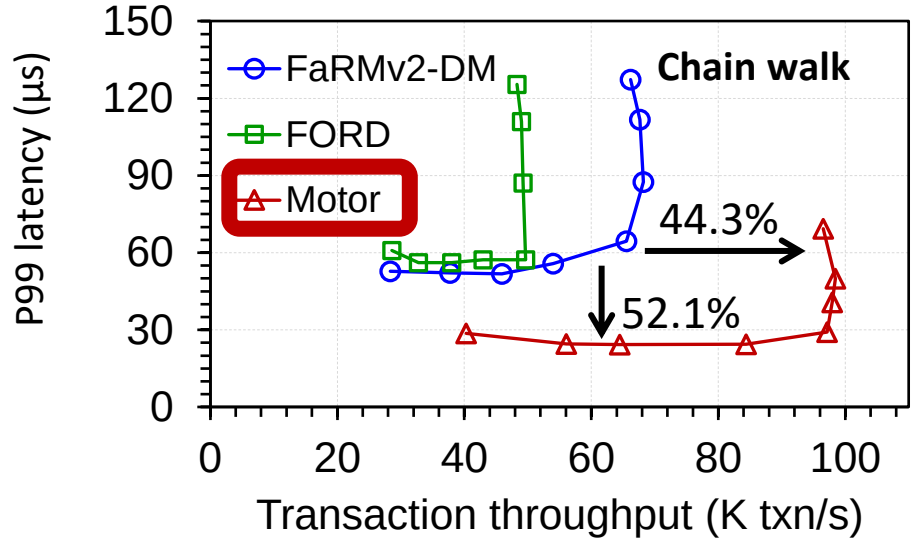


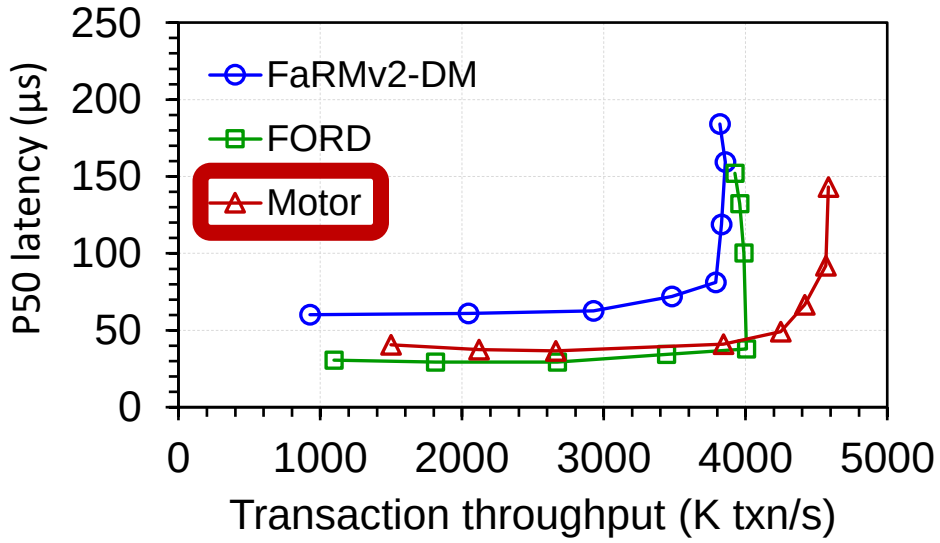
End-to-End Performance

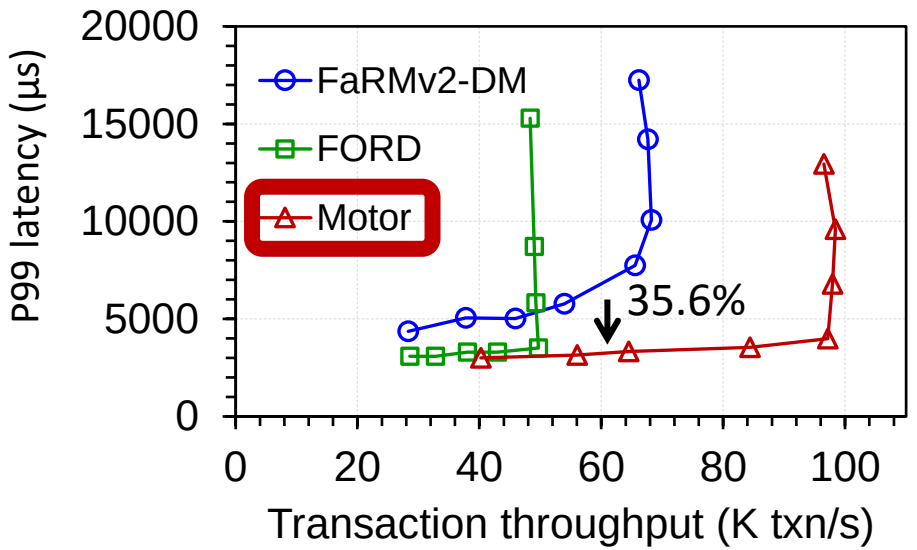
TATP (read-intensive)



TPCC (write-intensive)

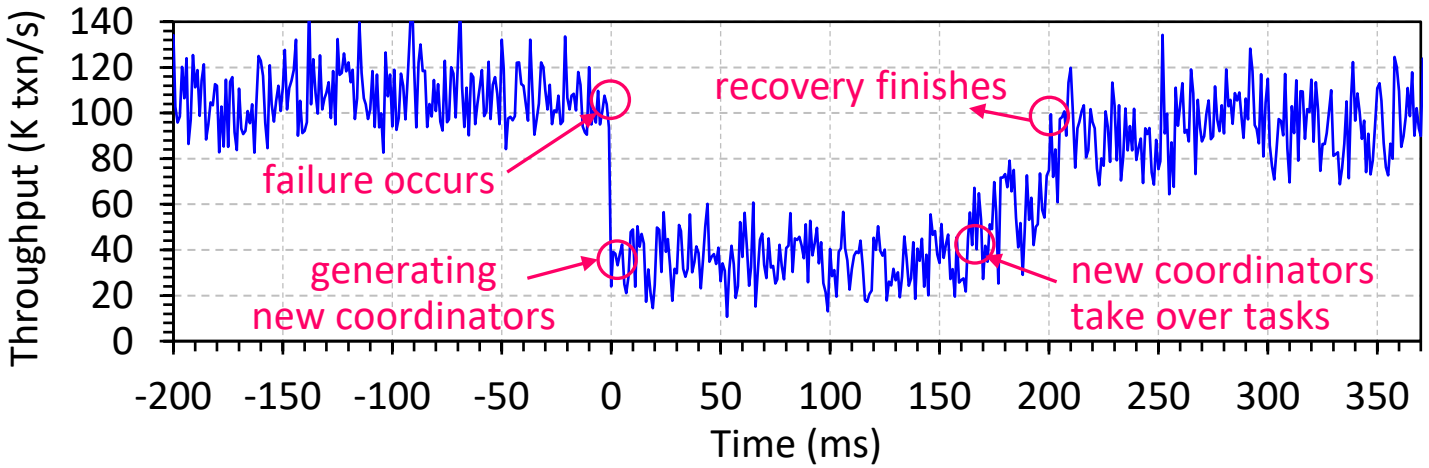




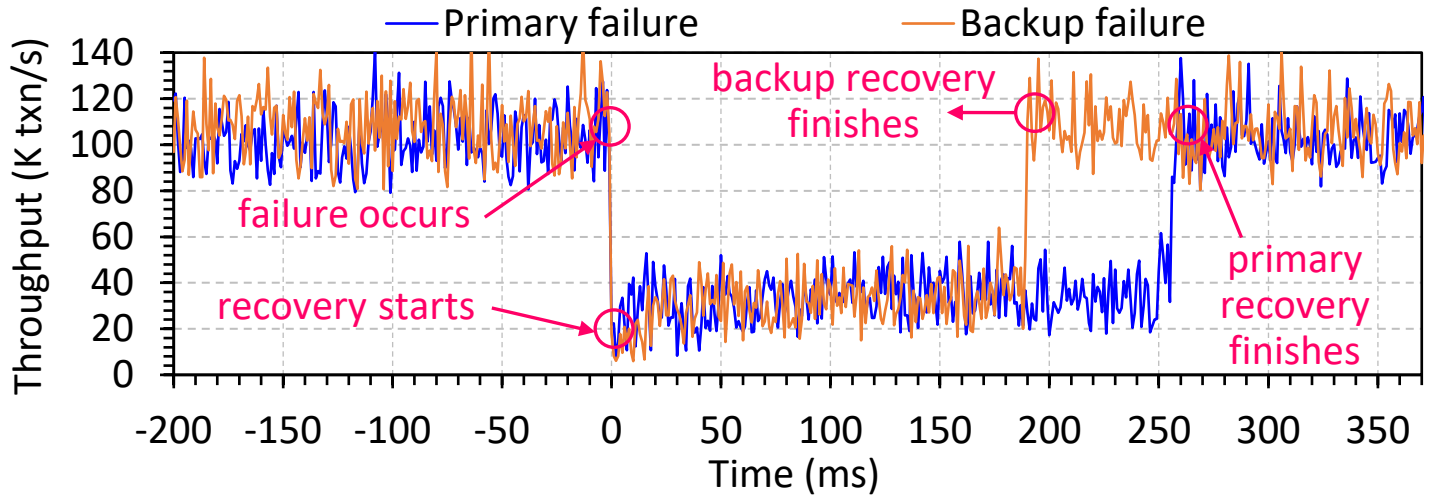


Failure Recovery

➢TPCC



Tolerating coordinator failures using local operation logs



Tolerating replica failures using data migration

Conclusion

- Existing multi-versioning distributed transactions do not fit DM
  - Inefficient linked version chain
  - Incompatible transaction protocol
- ***Motor:*** a holistic multi-versioning design for DM
  - Consecutive version tuple structure (memory pool)
  - One-sided RDMA MVCC based on CVT (compute pool)
- **Benefits**

High Throughput **Low Latency** **Low Memory Overhead**

https://github.com/minghust/motor

Thank you! Q&A