GeoTP: Latency-aware Geo-Distributed Transaction Processing in Database Middlewares

Qiyu Zhuang[†], Xinyue Shi[†], Shuang Liu[†], Wei Lu[†], Zhanhao Zhao[†]
Yuxing Chen[‡], Tong Li[†], Anqun Pan[‡], Xiaoyong Du[†]

[†]Renmin University of China [‡]Tencent Inc.

20/05/2025

Content

- Background
- Related work
- System overview
- Technique Details
- Evaluation

Content

- Background
- Related work
- System overview
- Technique Details
- Evaluation

Background

- Geo-distributed applications often process transactions across different databases in various locations

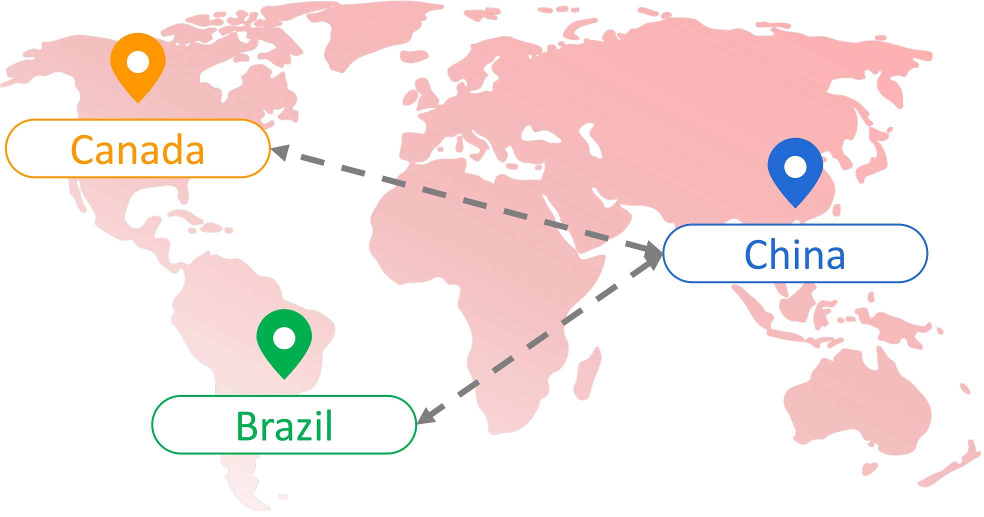
Shopping application: Browse | Order | Payment | Carriage

Data Source 2

Stock		
item	stocks	...
ice wine	10000	...
...		

Data Source 3

Stock		
item	stocks	...
coffee	10000	...
...		



Data Source 1

Bank		
user	balance	...
andrew	$100	...
...		

Stock | Cart

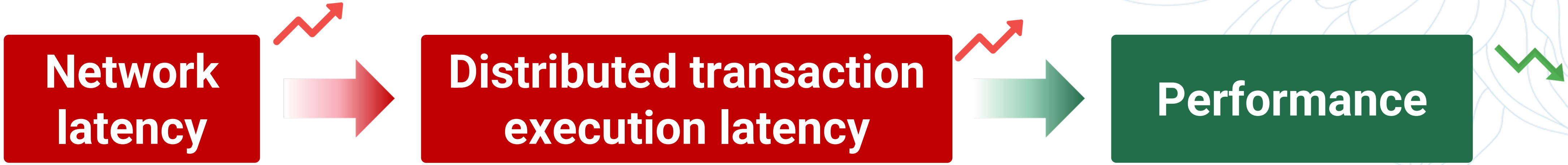
Background

- Geo-distributed applications often process transactions across different databases in various locations

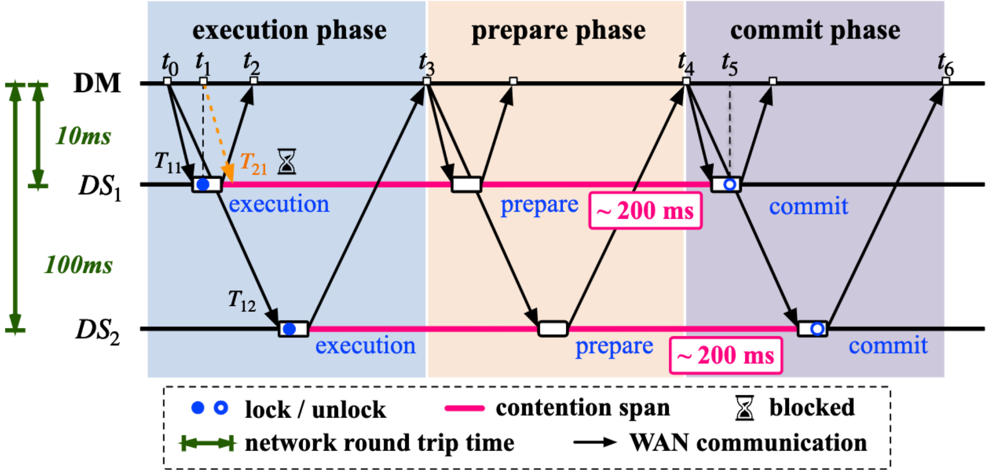
- Database middleware (DM) is required to provide transaction processing across heterogeneous databases **without modifications** and ensure the **transaction atomicity**

Motivation and Challenge

⚠ **High and dynamic wide-area network (WAN) latency !**

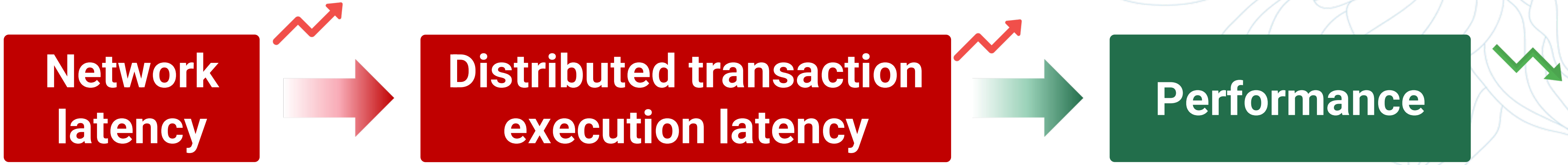


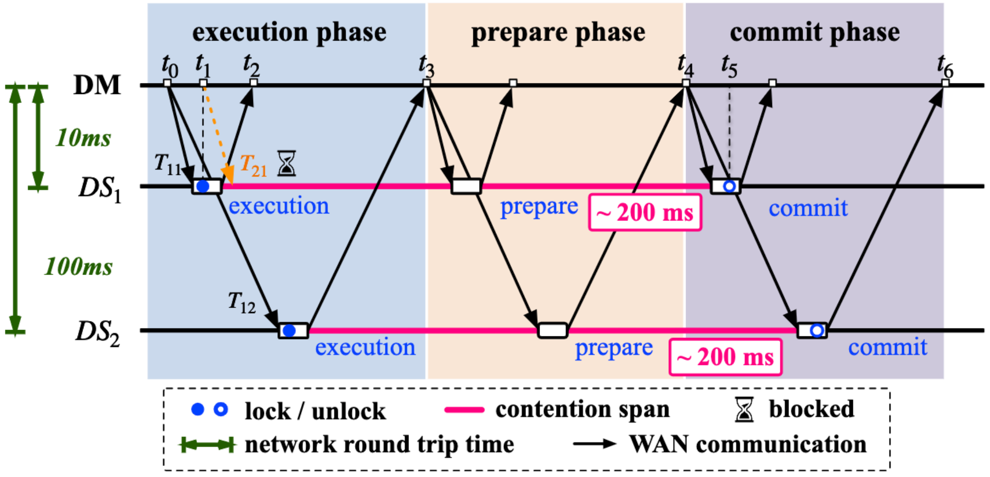
Network latency → Distributed transaction execution latency → Performance

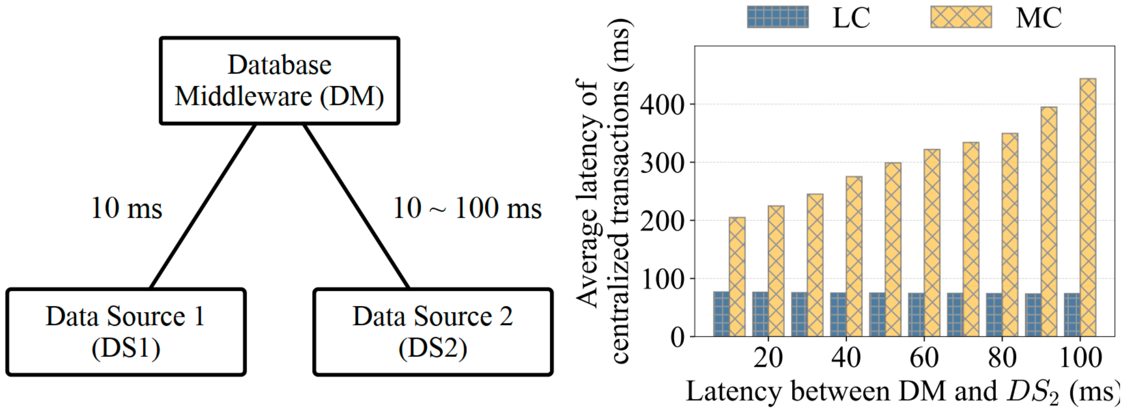


Motivation and Challenge

⚠ High and dynamic wide-area network (WAN) latency !



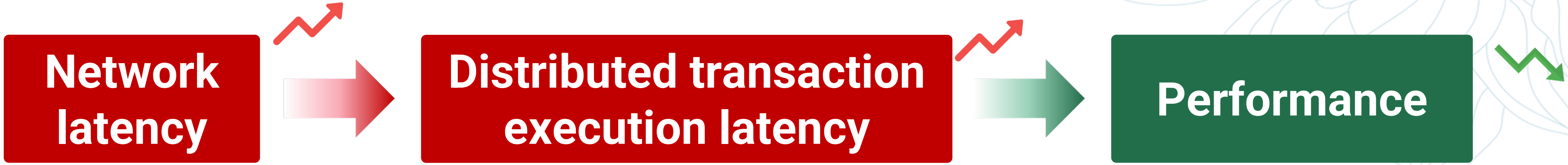




(a) Data source deployment (b) Impact of network latency

Motivation and Challenge

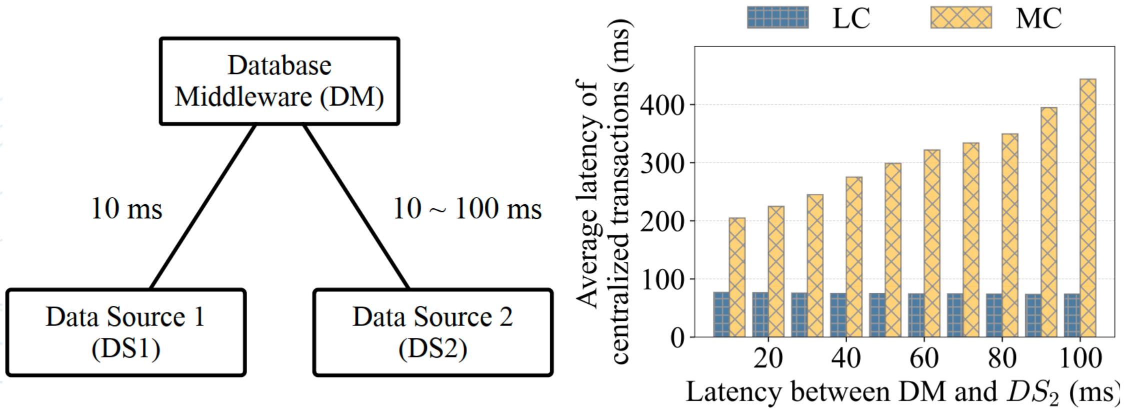
⚠ **High and dynamic wide-area network (WAN) latency !**



Network latency

Distributed transaction execution latency

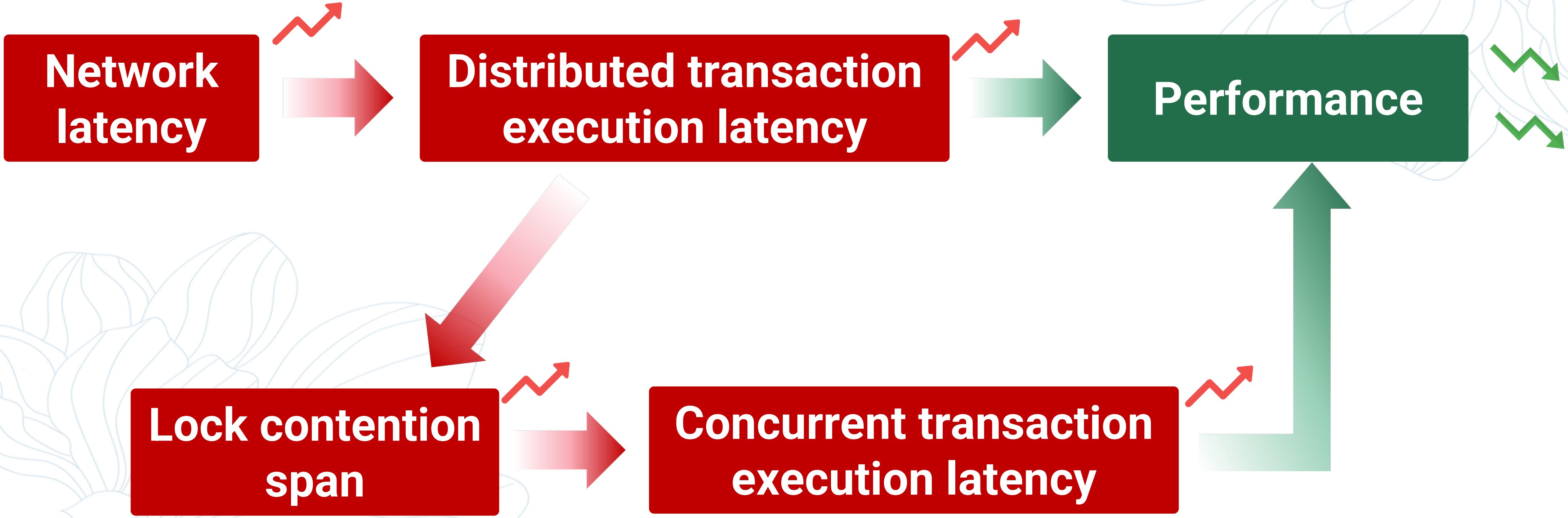
Performance



(a) Data source deployment (b) Impact of network latency

Motivation and Challenge

⚠ High and dynamic wide-area network (WAN) latency !



Network latency

Distributed transaction execution latency

Performance

Lock contention span

Concurrent transaction execution latency

Content

- Background
- Related work
- System overview
- Technique Details
- Evaluation

Related Works

Reduce WAN round trips
(e.g. Carousel [SIGMOD'18], Natto [SIGMOD'22], RedT [VLDB'24], ...)

➢ Eliminating the prepare phase by writing logs during execution or integrating consensus protocols with 2PC

➢ **Limitation:** require rewriting the kernel-level protocol, making them hard to extend to heterogeneous data sources.

Reduce lock contention spans
(e.g. QURO [VLDB'16], DAST [EuroSys'21], Chiller [SIGMOD'21], , ...)

➢ Delay the scheduling of part of the operations in the transaction.

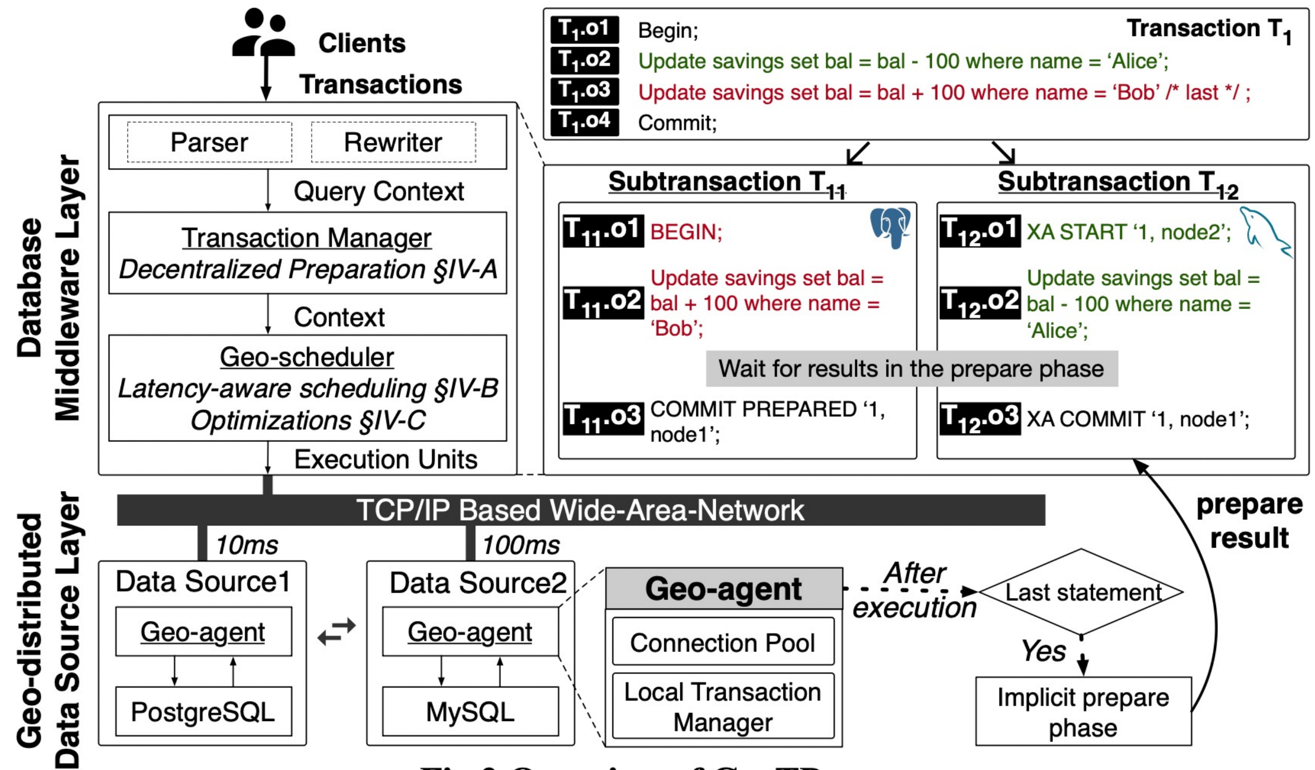
➢ **Limitation:** they overlook the varied latency between cross-region nodes, leaving substantial room for optimizing the lock contention span.

Content

- Background
- Related work
- System overview
- Technique Details
- Evaluation
- Conclusion

System Overview

Two-layer architecture
(Database middleware, Geo-distributed data source)



```
T1.o1 Begin;
T1.o2 Update savings set bal = bal - 100 where name = 'Alice';
T1.o3 Update savings set bal = bal + 100 where name = 'Bob' /* last */ ;
T1.o4 Commit;
```

```
T11.o1 BEGIN;
T11.o2 Update savings set bal = bal + 100 where name = 'Bob';
T11.o3 COMMIT PREPARED '1, node1';
```

```
T12.o1 XA START '1, node2';
T12.o2 Update savings set bal = bal - 100 where name = 'Alice';
T12.o3 XA COMMIT '1, node1';
```

Fig.2 Overview of GeoTP

Key Techniques

- ➢ **Decentralized prepare mechanism**
  offloads the coordination cost required for the prepare phase
- ➢ **Latency-aware scheduling**
  minimize the lock contention span
- ➢ **Optimization in high-contention workloads**
  schedule transactions considering local execution latency

Content

- Background
- Related work
- System overview
- Technique Details
- Evaluation

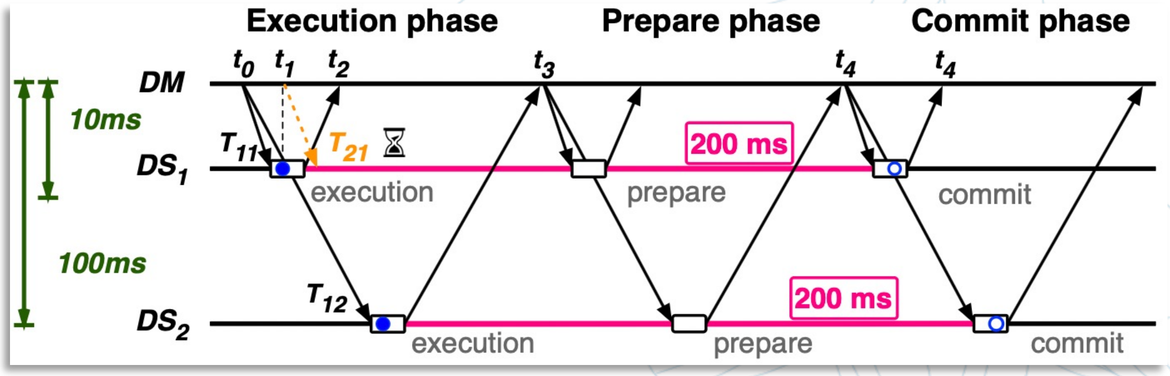
Decentralized Prepare Mechanism

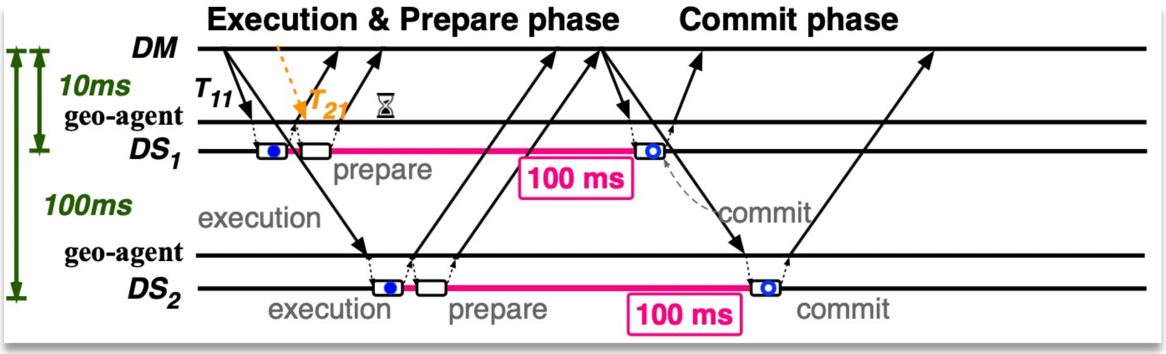
Decentralized prepare

- ➢ Offload the coordination cost of the prepare phase from the DM to data sources.
- ➢ Eliminate one WAN round trip without database modifications.

Early abort

- ➢ Abort subtransactions bypassing the DM.
- ➢ Reduce half of the WAN round trip when transactions are required to abort.





Latency-aware Scheduling

- Lock contention span is the time span between the acquire and release the lock.

$$LCS(T_{ij}) = \check{t}^{T_{ij}}_{last} - \hat{t}^{T_{ij}}_{1st}$$

- Significant differences in network latencies often lead to unnecessary lock contention spans.
- Optimize the start time point for each subtransaction based on network latency to minimize the lock contention span.

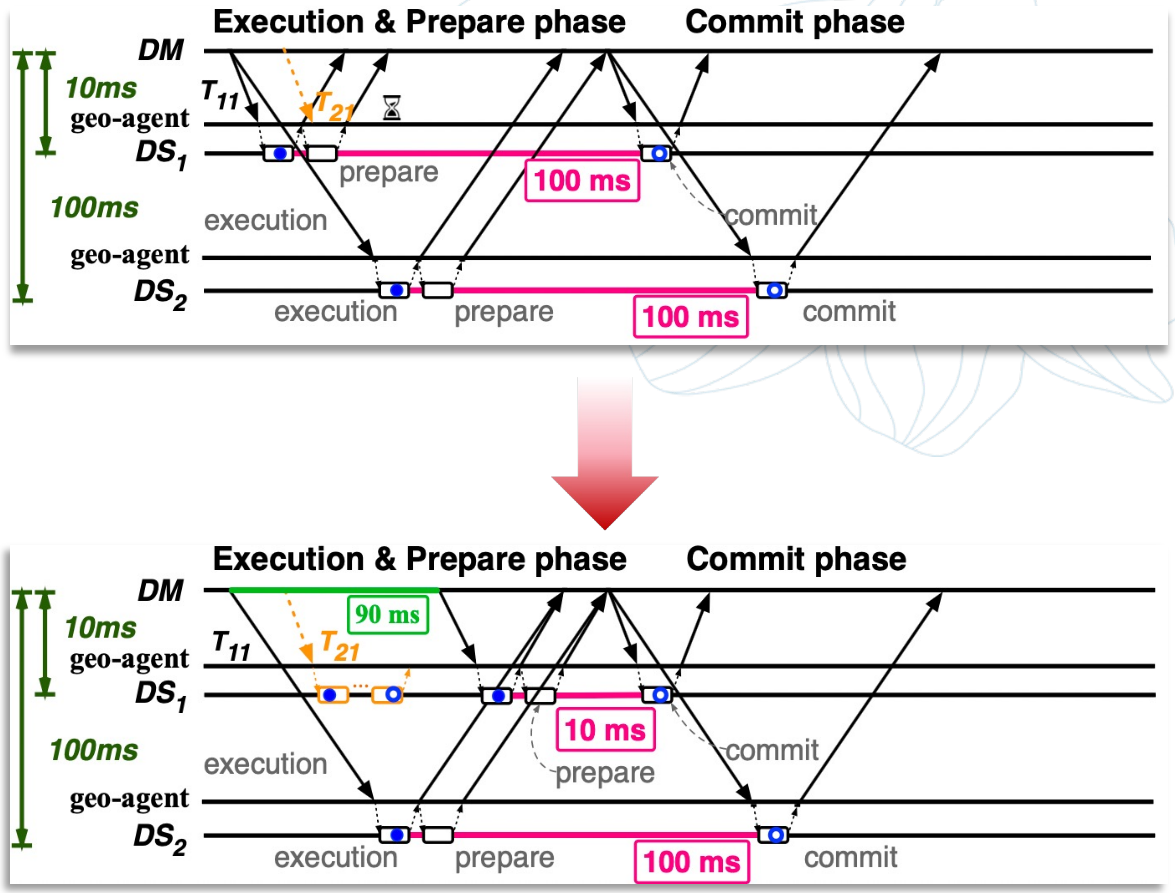
$$\underset{\hat{t}^{T_{ij}}_{1st}}{\arg\min}\, LCS(T_{ij}) \Rightarrow \underset{\hat{t}^{T_{ij}}_{1st}}{\arg\min}\Big(\max_{\forall T_{is} \in T_i} \tau_{is} - \hat{t}^{T_{ij}}_{1st}\Big)$$

$$s.t. \quad \hat{t}^{T_{ij}}_{1st} + \tau_{ij} \leq \max_{\forall T_{is} \in T_i} \tau_{is}$$

$$\hat{t}^{T_{ij}}_{1st} = \max_{\forall T_{is} \in T_i} \tau_{is} - \tau_{ij}$$

Latency-aware Scheduling

- Lock contention span is the time span between the acquire and release the lock.
- Significant differences in network latencies often lead to unnecessary lock contention spans.
- Optimize the start time point for each subtransaction to minimize the lock contention span.



Optimizations

Challenges in high-contention workloads:

- The lock contention span is also influenced by the time required for local execution of subtransactions.

- Transaction are more likely to be rollbacked.

Optimizations:

Forecast local execution latency by collecting hotspot data access metadata.

$$t^{T_{ij}}_{start} = \max_{\forall T_{is} \in T_i} (\tau_{is} + \widehat{LEL}(T_{is})) - (\tau_{ij} + \widehat{LEL}(T_{ij}))$$

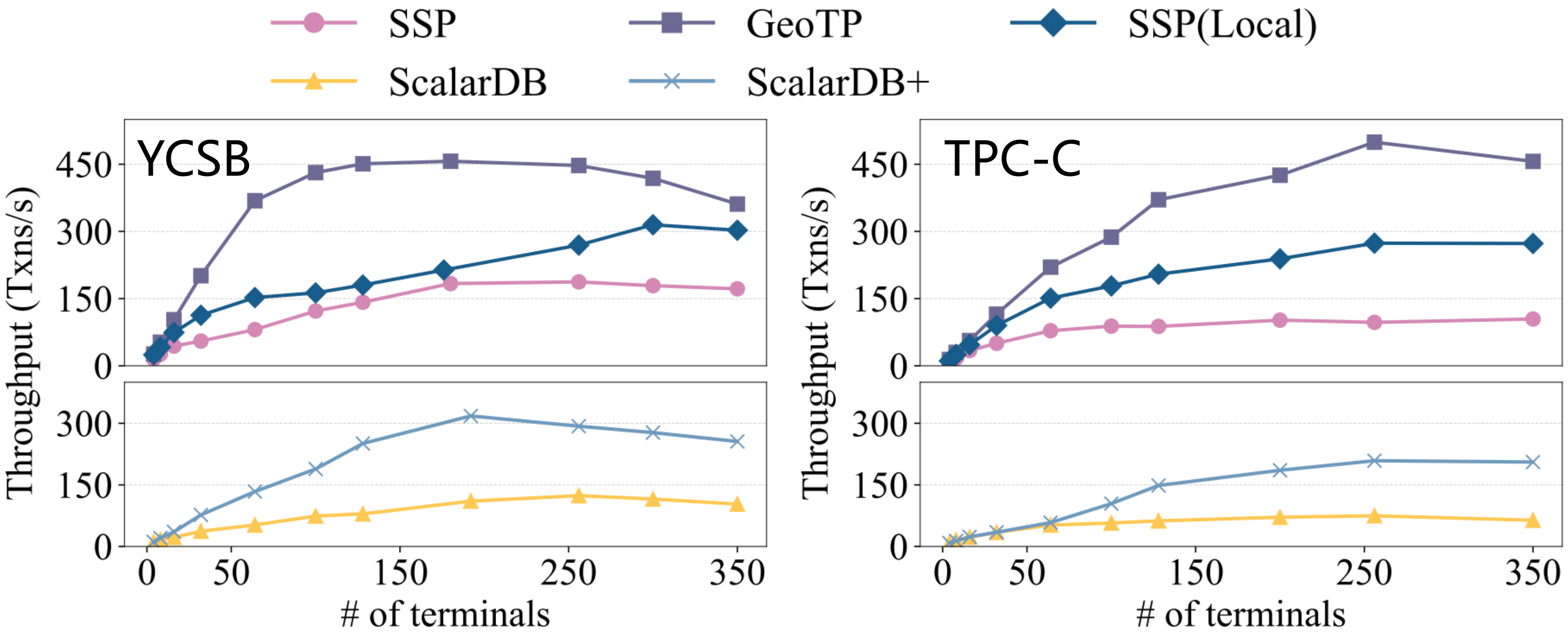
Blocking transactions with high abort rates in the DM.

Content

- Background
- Related work
- System overview
- Technique Details
- Evaluation

Evalution

GeoTP outperforms state-of-the-art approaches (SSP(Local), SSP, ScarlarDB) by up to **2.65×**, **5.14×**, and **7.15×**, respectively.



Evalution

Ablation Study

GeoTP outperforms SSP, achieving up to **17.7×** higher throughput and **84.3%** lower p99 latency, with O2 and O3 providing key benefits under medium and high contention scenarios.

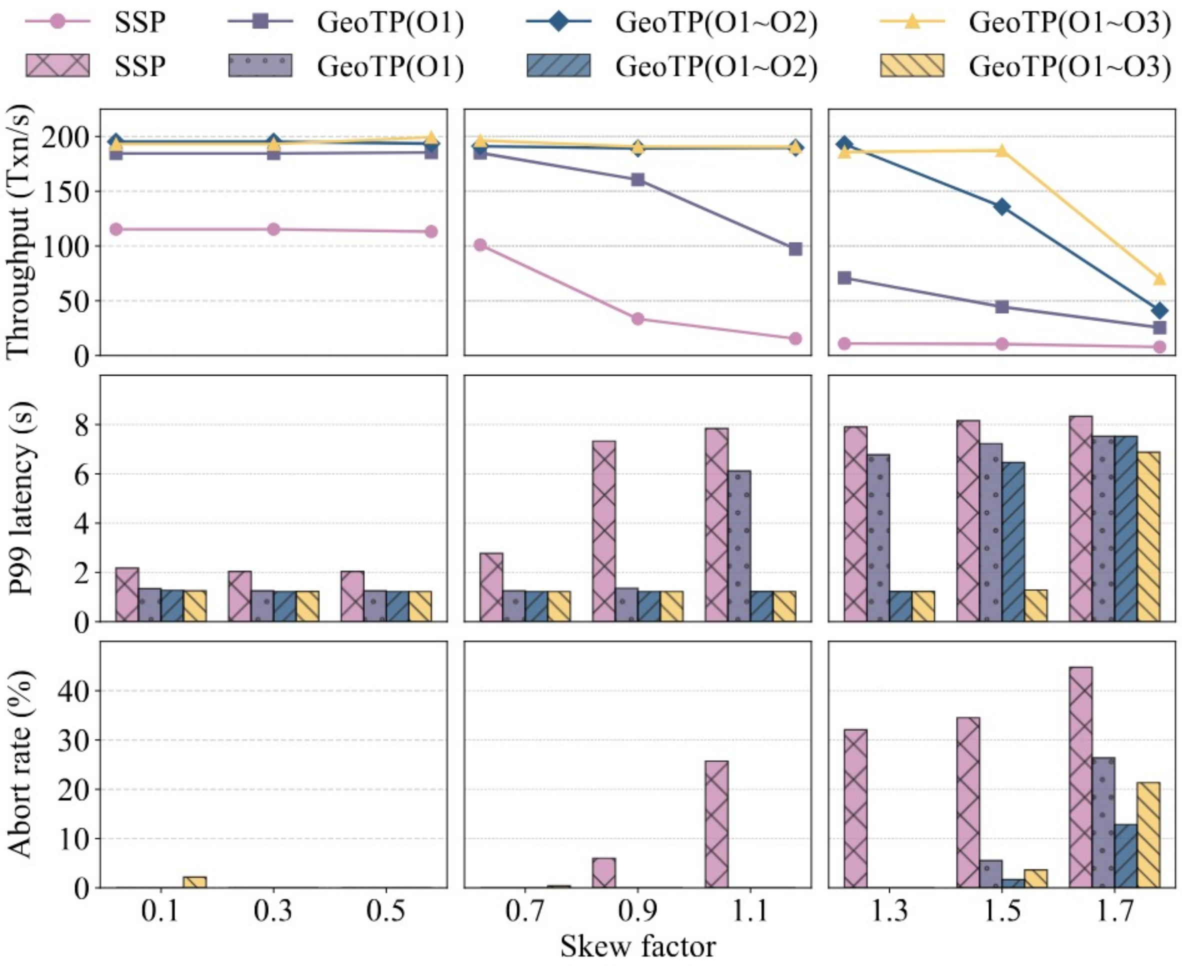
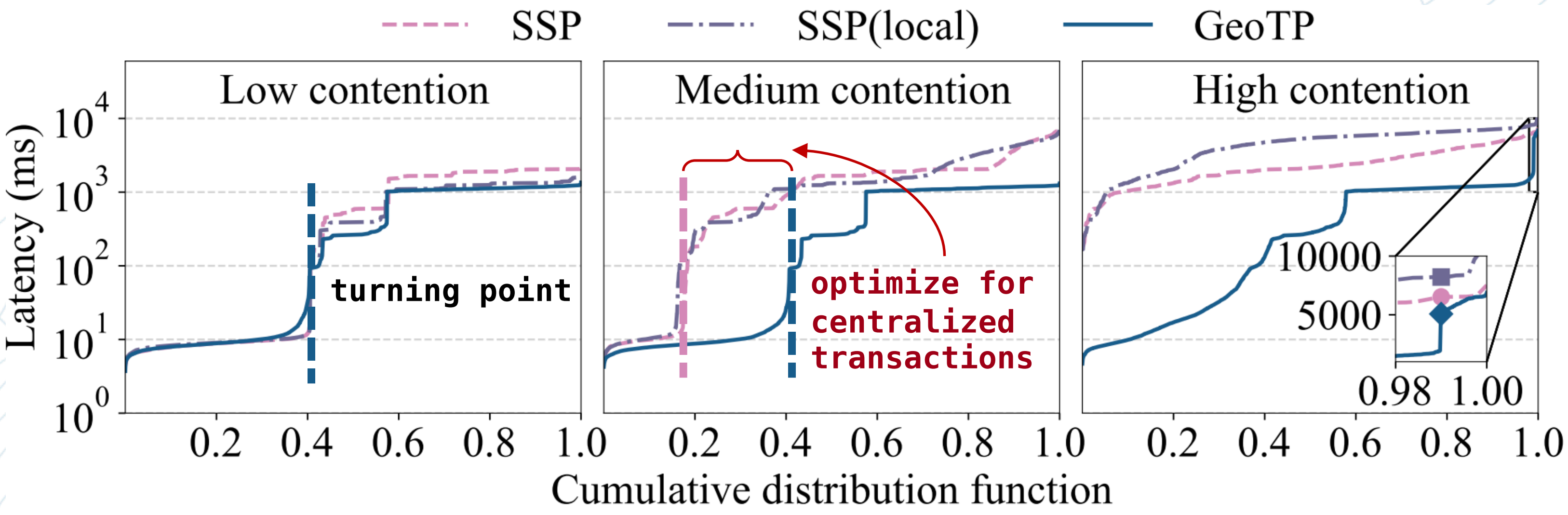


Fig. 12: Impact of optimizations over YCSB

Evalution

GeoTP reduces distributed transaction latency and delays the point where latency starts to spike, outperforming baselines by up to **35.9%.**



Conclusion

- We propose GeoTP, a latency-aware geo-distributed transaction processing method in database middlewares.
- GeoTP adopts decentralized prepare and latency-aware scheduling to reduce WAN overhead and lock contention without modifying database kernels.

Thanks!