IPADS, SJTU, China http://ipads.se.sjtu.edu.cn/projects/wukong

TALK @ TUE OCT 31
2:45 SESSION 13
Data Analytics

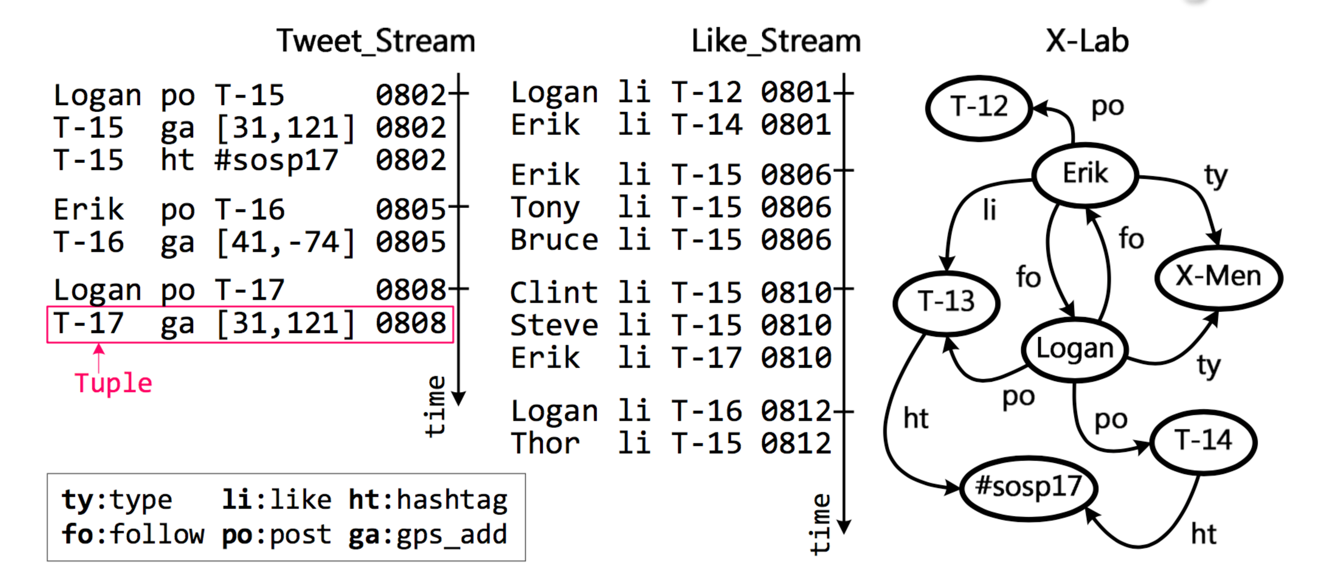
Sub-millisecond Stateful Stream Querying over Fast-evolving Linked Data

Yunhao Zhang, Rong Chen and Haibo Chen

Motivation

Q: Why we need a new streaming system?

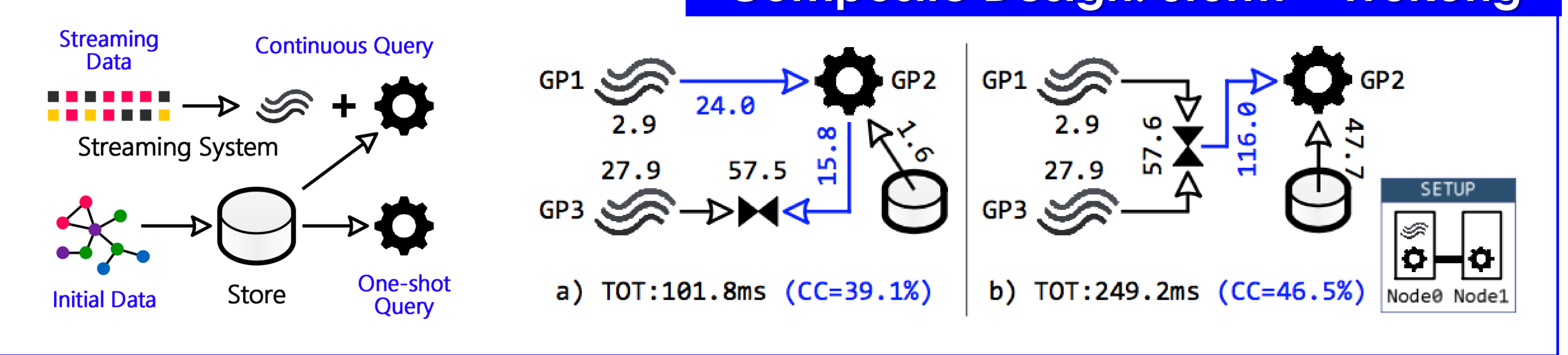
A Motivating Example: Social Networking



Continuous Query

```
REGISTER QUERY Qc
SELECT ?X ?Y ?Z
FROM Tweet_Stream [RANGE 10s STEP 1s]
FROM Like_Stream  [RANGE  5s STEP 1s]
FROM X-Lab
WHERE {                                  Query Condition
  GRAPH Tweet_Stream  { ?X po ?Z }
  GRAPH X-Lab         { ?X fo ?Y }
  GRAPH Like_Stream   { ?Y li ?Z }
}
```

Composite Design: Storm + Wukong



Key Performance Deficiencies

Cross-system cost. Data transformation and transmission dominate latency.

Inefficient query plan. No query plan optimizing both cross-system cost and intermediate result size.

Limited scalability. Stream computation systems provide strong data isolation between each query execution.

Workload Characteristics

Stateful. Some stream knowledge become part of the social graph and server all future queries.

Data Sharing. All queries share the same evolving social graph and the same set of data streams.

Partial Data Access. Each query only cares about a part of the stream ,not the stream as a whole.

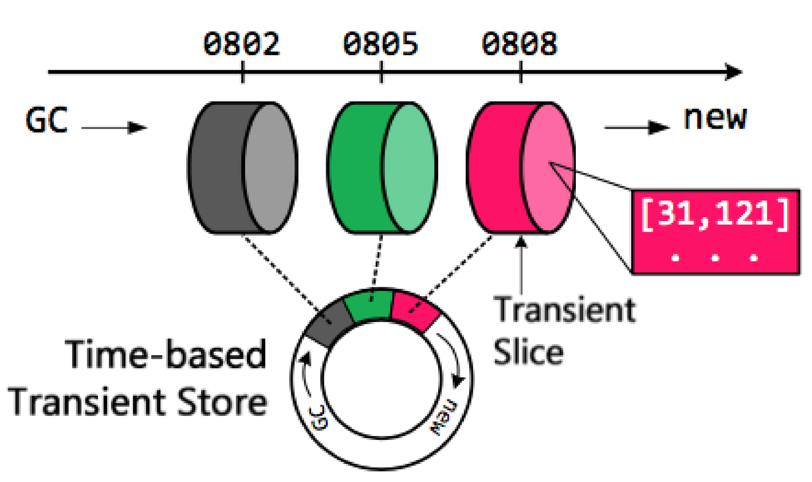
Latency-oriented. Query Latency should be the first-class citizen.

System Design

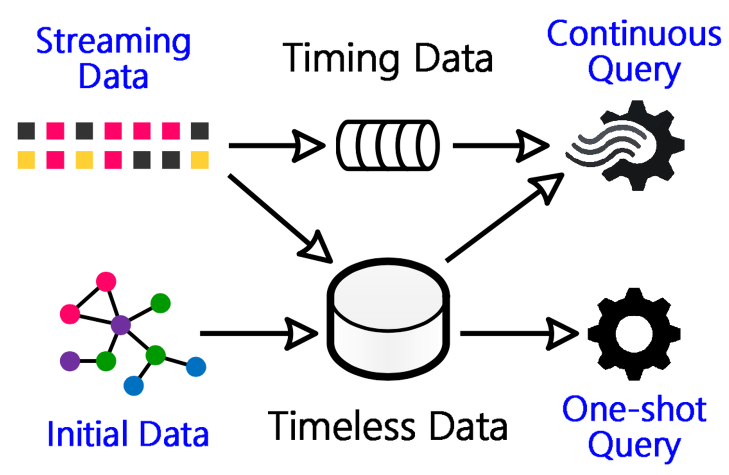
Q1: How to gracefully integrate streaming and stored data?

Continuous persistent store. Extend static graph storage to absorb useful parts of streams (append-only) for timeless data

Time-based transient store. consists of a sequence of transient slices in time order for timing data



Integrated Design



Q2 : How to access streaming data scattered over the entire timeless data?

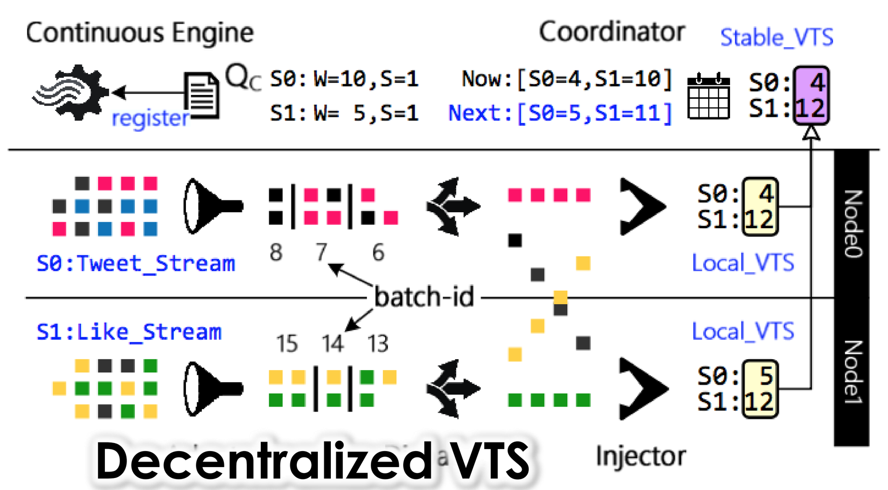
Observation. Continuous persistent store and time-based transient store adopt **similar** internal data structures.

Design decision. Develop a time-related index (***Stream Index***) over two stores, which provide a ***fast-path*** to access streaming data.

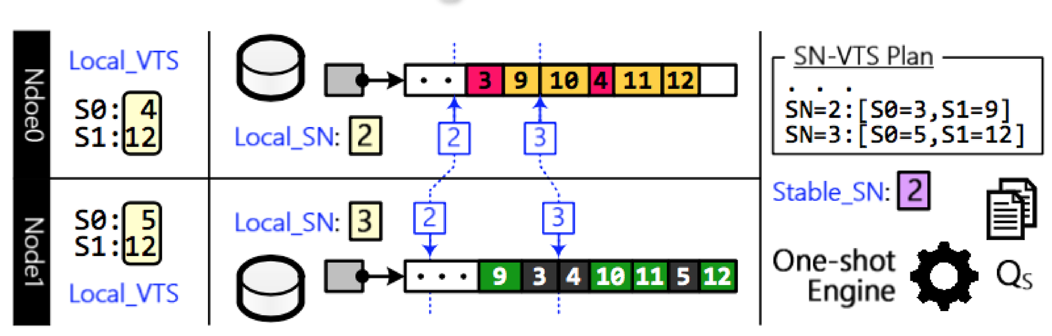
Locality-aware partitioning. Indexes can be **replicated on-demand** to balance memory usage and locally processing.

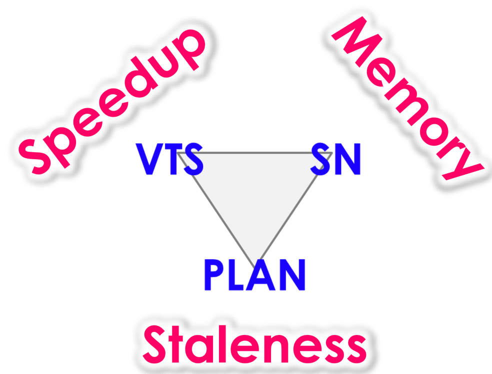
Q3 : How to provide consistent snapshots on dynamic data with high memory efficiency?

Challenges: *1) Inject* streaming data in a distributed fashion; *2) Using* vector timestamps (VTS) for timeless data causes high memory/CPU overhead



Bounded Snapshot Scalarization





Evaluation

Latency

Table 3: The query performance (ms) on a 8-node cluster.

LSBench 3.75B	Wukong+S	Storm+Wukong All	(Storm)	(Wukong)	Spark Streaming
L1	0.10	0.23	0.23	-	219
L2	0.08	1.64	1.02	0.11	527
L3	0.11	2.62	2.97	0.16	712
L4	1.78	31.14	31.14	-	346
L5	3.50	40.77	14.37	3.49	2,215
L6	1.68	49.03	36.16	2.39	1,422
Geo. M	0.46	6.29	-	-	679

Table 4: The further performance comparison on a 8-node cluster.

LSBench 3.75B	Heron+Wukong All	(Heron)	(Wukong)	Structured Streaming	Wukong /Ext
L1	0.24	0.24	-	287	0.19
L2	1.58	0.74	0.11	743	0.14
L3	2.35	1.72	0.15	1,698	0.17
L4	30.92	30.92	-	x	6.91
L5	31.72	13.23	3.73	x	7.36
L6	45.78	24.48	2.65	x	7.33
Geo. M	5.85	-	-	-	1.08

Throughput

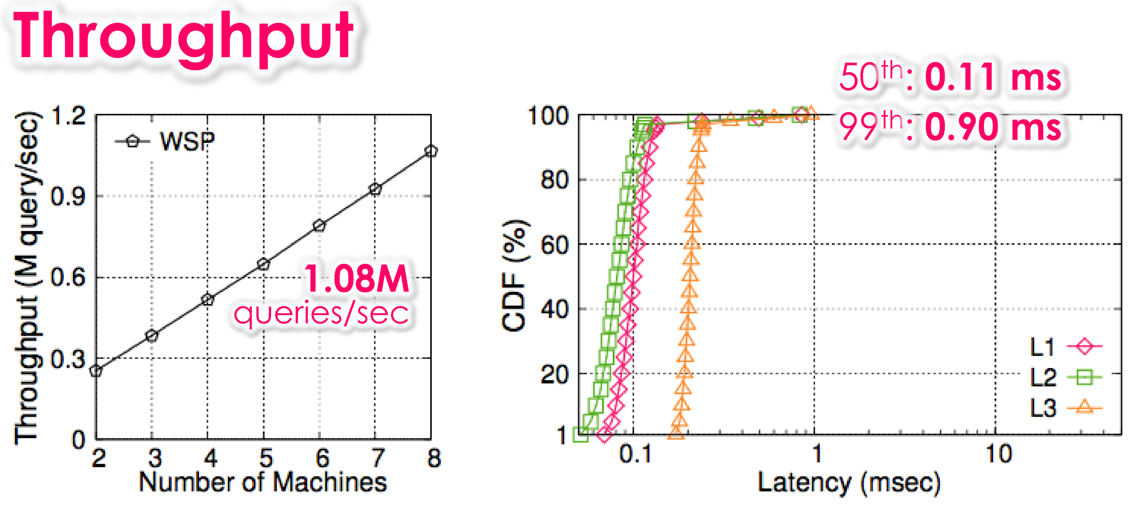


Fig. 14: (a) The throughput of a mixture of 3 classes of queries with the increase of nodes, and (b) the CDF of latency on 8 nodes.

Stream Rate

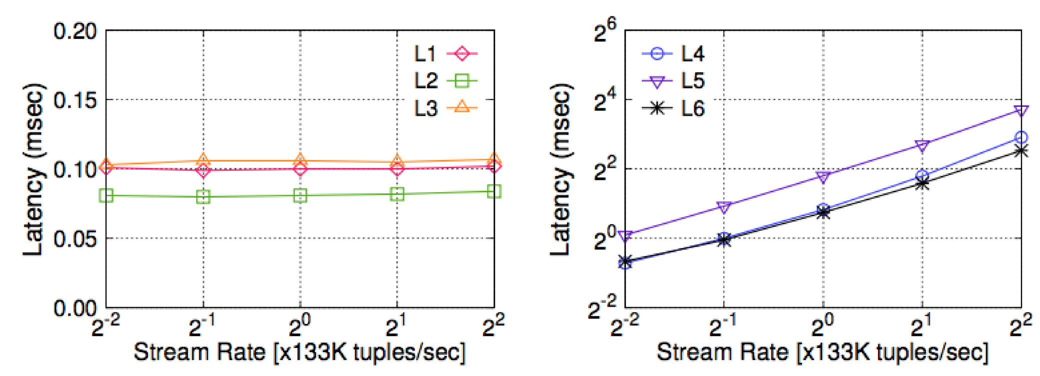


Fig. 13: The latency of queries in group (I) and (II) with the increase of stream rate on LSBench-3.75B.

No-RDMA

Table 5: The performance impact of RDMA on Wukong+S.

LSBench	L1	L2	L3	L4	L5	L6	Geo.M
Wukong+S	0.10	0.08	0.11	1.78	3.50	1.68	0.46
Non-RDMA	0.11	0.08	0.12	6.22	6.14	4.90	0.75
Slowdown	1.1X	1.0X	1.1X	3.5X	1.8X	2.9X	1.6X

One-shot Query

Table 8: The performance comparison (ms) for one-shot query.

LSBench	S1	S2	S3	S4	S5	S6	Geo.M
Wukong	4.04	0.11	0.19	23.1	0.26	60.2	1.77
Wukong+S/Off	4.12	0.12	0.20	24.1	0.28	61.8	1.83
Wukong+S/On	4.31	0.11	0.21	25.5	0.29	64.2	1.93

Related Work

Stream Computation Systems: usually stateless, complete data access and throughput-oriented. *(Apache Storm, S4, Twitter Heron, Spark Streaming, Filnk, etc.*

Stream Processing Engine: widely adopt relational data model, which is shown to be inefficient for highly-linked data. *(Aurora, Borealis, MaxStream, TelegraphCQ, Fjording, etc.)*

Conclusion

Wukong+S: A distributed stream querying engine for both stateful continuous and one-shot queries over fast-evolving linked data

Achieving sub-millisecond latency and throughput exceeding one million queries per second