HTAPBench: Hybrid Transactional and Analytical Processing Benchmark

HASLab@INESCTEC

International Conference on Performance Engineering (ICPE2017)

1st March, 2022

Large-scale Real-time Analytics Applications[1]

Require distributed data management systems that handle fast concurrent transactions and analytics on the recent data at the same time.

- Real-time inventory/pricing.
- Recommendations from mobile apps.
- Fraud detection.

[1] Hybrid Transactional/Analytical Processing: A Survey, 2017, [ÖTT17]

Cases Of HTAP System (From The Point Of Architecture)[2]

Avoid offline Extract-Transform-Load(ETL) to transfer data.

- Dual System: F1 Lightning[3] catches transaction logs via Change-Data-Capture(CDC).
- Single Layout: Directly builds an HTAP system derived from one specialized system(e.g. OLTP).
- Dual Layout: TiDB[4] introduces learner roles to generate columnar stores for real-time OLAP queries.

[2] Retrofitting High Availability Mechanism to Tame Hybrid Transaction/Analytical Processing, 2021, [SCCZ21]

[3] F1 Lightning: HTAP as a Service, 2020, [YRX+20]

[4] TiDB: a Raft-based HTAP database, 2020, [HLC+20]

Benchmark In HTAP System

Design the workload according to their own system, simple workload and not objective.

HTAP System Benchmark – CH-benCHmark[5]

Run transactional and analytical workload in parallel on a shared set tables, based on TPC-C and TPC-H standard benchmark.

[5] The mixed workload CH-benCHmark, 2011, [CFG+11]

TPC-C

It uses mixture of read-only and update-intensive business transactions: New-Order, Payment, Order-Status, Delivery and Stock-Level, and takes New-Order transaction per minute(tpmC) as metric.

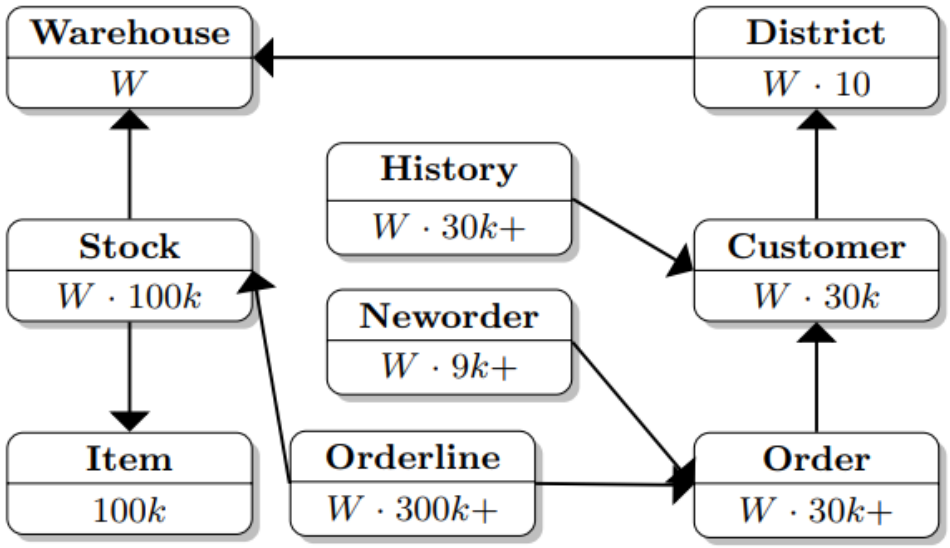


Figure 1: TPC-C Schema

TPC-H

It uses 22 business queries with complex data operations and takes the total number of completed queries per hour(QphH) as metric.

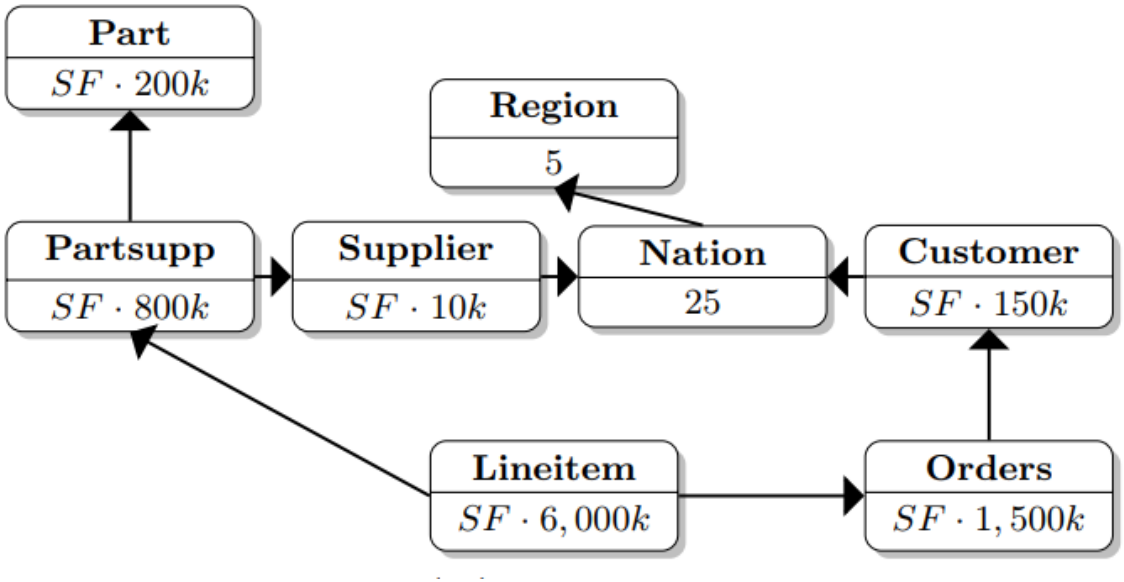


Figure 2: TPC-H Schema

CH-benCHmark Schema

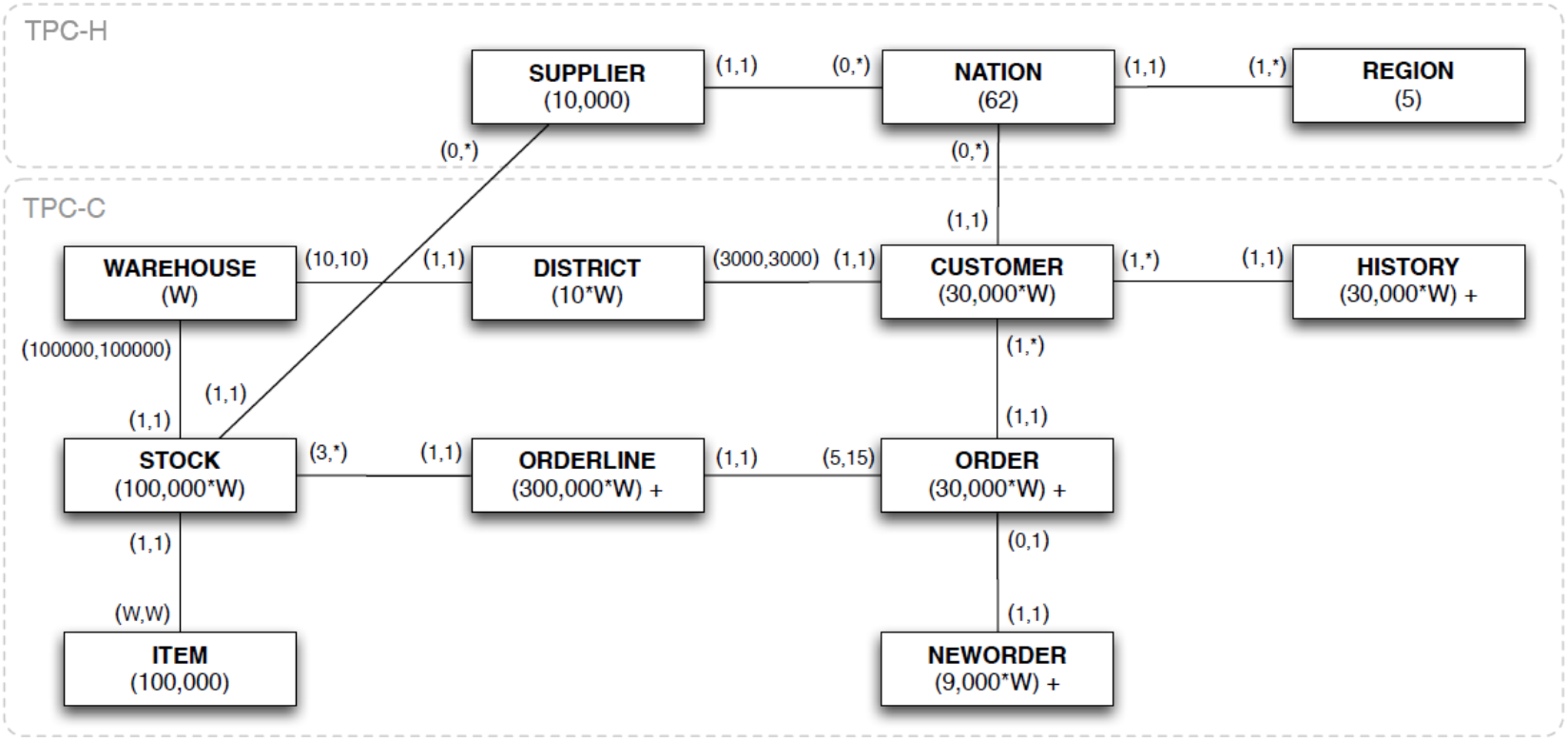


Figure 3: TPC-H Schema

Comparation Of Query 5 Between TPC-H And CH-benCHmark

```
1  SELECT n_name,SUM(
2    l_extendedprice*(1-l_discount)
3  )
4  AS revenue
5  FROM customer,orders,lineitem,
6    supplier,nation,region
7  WHERE c_custkey=o_custkey
8  AND l_orderkey=o_orderkey
9  AND l_suppkey=s_suppkey
10 AND c_nationkey=s_nationkey
11 AND s_nationkey=n_nationkey
12 AND n_regionkey= r_regionkey
13 AND r_name=REGION
14 AND o_orderdate>=DATE
15 AND o_orderdate<DATE+INTERVAL
16 GROUP BY n_name
17 ORDER BY revenue
18 DESC
```

```
1  SELECT n_name,SUM(ol_amount)
2  AS revenue
3  FROM customer,order,orderline,
4    stock,supplier,nation,region
5  WHERE c_id=o_c_id
6  AND c_w_id=o_w_id
7  AND c_d_id=o_d_id
8  AND ol_o_id=o_id
9  AND ol_w_id=o_w_id
10 AND ol_d_id=o_d_id
11 AND ol_w_id = s_w_id
12 AND ol_i_id = s_i_id
13 AND mod((s_w_id*s_i_id),10000)
14   =su_suppkey
15 AND ascii(SUBSTRING(c_state,1,1))
16   =su_nationkey
17 AND su_nationkey=n_nationkey
18 AND n_regionkey=r_regionkey
19 AND r_name=REGION
20 AND o_entry_d>=DATE
21 GROUP BY n_name
22 ORDER BY revenue
23 DESC
```

CH-benCHmark Drawbacks[6]

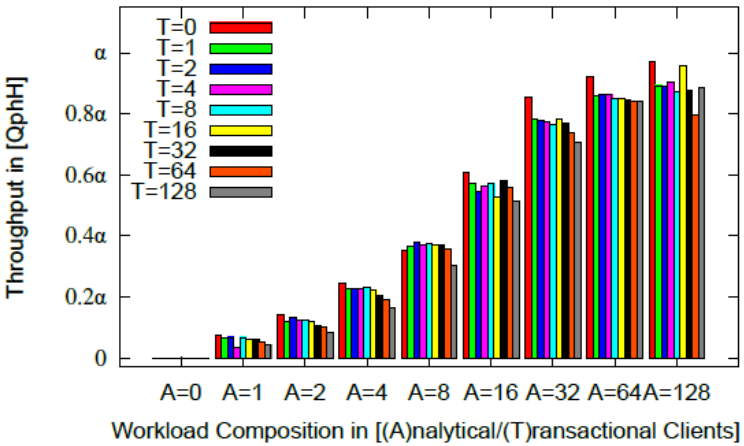


Figure 4: QphH SAP HANA

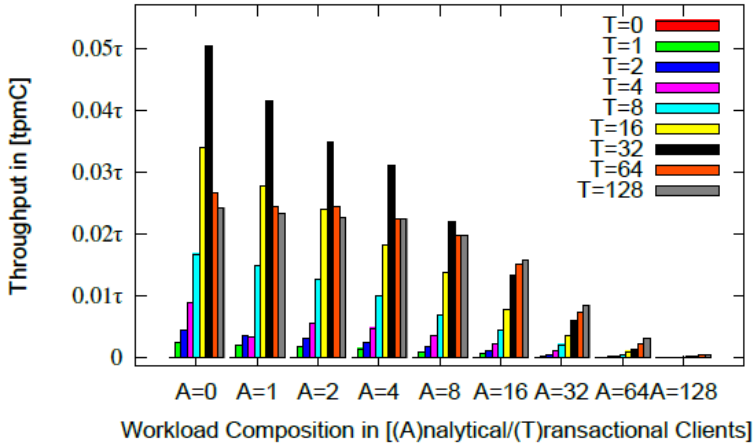


Figure 5: TpmC SAP HANA

- Metrics are not unified.
- Results are not comparable across runs.

[6] Scaling up mixed workloads: a battle of data freshness, flexibility, and scheduling, 2014, [PWM+14]

HTAPBench Contribution

Gartner states that an HTAP system should prioritize a sustained transactional throughput, delivering at the same time scalable analytical processing without disrupting the operational activity.

- Privode homogeneous and comparable results across executions.
- Introduce Client Balancer to coordinate hybrid workloads.
- Define a unified metric for HTAP systems geared toward the execution of constantly increasing OLAP requests limited by an admissible impact on OLTP performance.

Architecture Overview

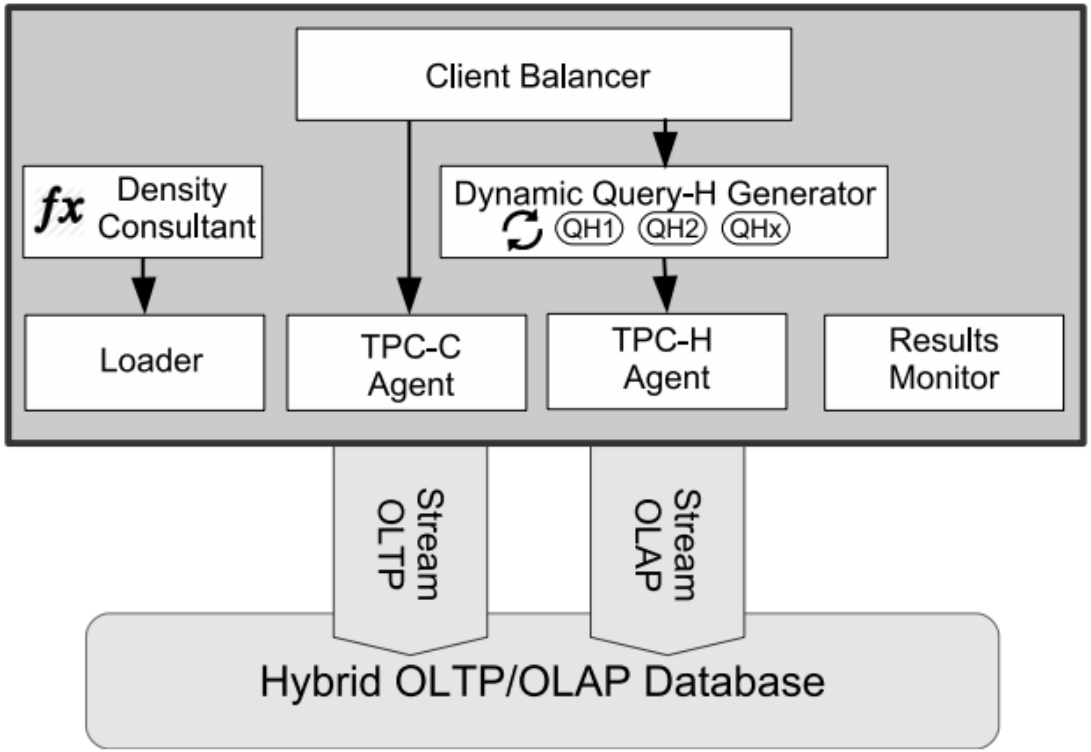


Figure 6: HTAPBench Architecture

Date Field

Filtering operations limit the number of considered rows.

```
1  -- New-Order
2  INSERT INTO OORDER(
3      O_ID, O_D_ID, O_W_ID, O_C_ID,
4      O_ENTRY_D, O_OL_CNT, O_ALL_LOCAL
5  )
6  VALUES (?, ?, ?, ?, ?, ?, ?)
7  -- Payment
8  INSERT INTO HISTORY(
9      H_C_D_ID, H_C_W_ID, H_C_ID, H_D_ID,
10     H_W_ID, H_DATE, H_AMOUNT, H_DATA
11 )
12 VALUES (?,?,?,?,?,?,?,?)
13 -- Delivery
14 UPDATE ORDERLINE
15 SET OL_DELIVERY_D = ?
16 WHERE OL_O_ID = ?
17 AND OL_D_ID = ?
18 AND OL_W_ID = ?
```

```
1  SELECT ol_number,
2  sum(ol_quantity) AS sum_qty,
3  sum(ol_amount) AS sum_amount,
4  avg(ol_quantity) AS avg_qty,
5  avg(ol_amount) AS avg_amount,
6  count(*) AS count_order
7  FROM orderline
8  WHERE ol_delivery_d > DATE
9  GROUP BY ol_number
10 ORDER BY ol_number
```

Result Set Not Homogeneous

The populate stage promotes bursts of transactions inserting data, causing a high concentration of timestamps in a short time period. While during the execution stage, transaction rate is regulated by TPC-C.

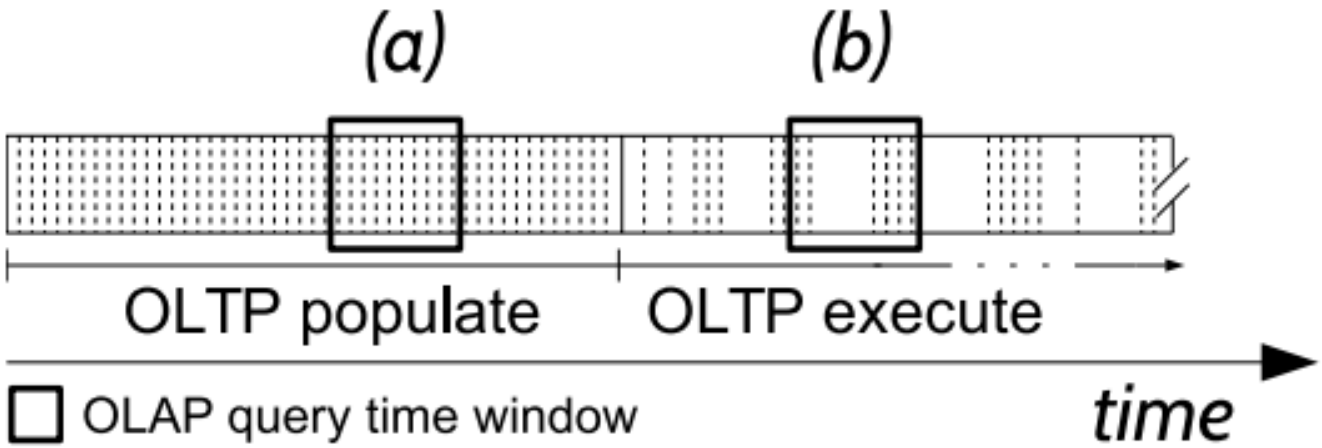


Figure 7: Timestamp Density Difference

Regulate the populate stage date density by the configured transaction mix within TPC-C to achieve the result set homogeneity.

Density Function

$$target(tpmC) = target(tps) \times 60 \times \frac{\%NewOrder}{100} \quad (1)$$

$$txnMix = \frac{\%NewOrder + \%Payment + 10 \times \%Delivery}{100} \quad (2)$$

$$d(T_s/s) = tps \times txnMix \quad (3)$$

Table 1: Density Observation Results

tpmC	Total Observed(T_s)	Expected $d(T_s/s)$	Experimental $d(T_s/s)$
635	108051	30.24	30.01
741	125500	35.14	34.86
886	150114	42.02	41.69

Calculation Example

target(tpmC)=635
according to Equation 1, target(tps)≈23.52
according to Equation 2, txnMix≈1.28
according to Equation 3, $d(T_s/s) \approx 30.24$

The density function results in Table 1 are only 3% apart from experimental observation. During the populate stage, the Loader is equipped with a clock that initiates with the system time, then computes how much time should elapse between clock ticks(ΔT_s).

$$\Delta T_s(ms) = \frac{1}{d(T_s/s)} \times 1000 \tag{4}$$

This particular query restricts the result set to orders placed within a one-year time frame and window frames are not kept static to ensure new regions on the dataset would be queried.

Query 6 In CH-benCHmark

```
1    SELECT SUM(ol_amount)
2    AS revenue
3    From order_line
4    WHERE ol_delivery_d between [Date] and [Date + 1year]
5    AND ol_quantity between [Amount a] and [Amout b]
```

Client Balancer

When the OLTP agent ensures that the target tps is stable, the Client Balancer will periodically assess whether or not the system is capable of handling an extra OLAP worker. This assessment relies on a proportional integral feedback controller.

$$output = K_P \Delta tps + K_I \int \Delta t \tag{5}$$

Meaning Behind Equation 5

Algorithm 1 Client Balancer

```
1: procedure
2:     wait(Δt)
3:     error ← target_tps − measured_tps
4:     integral ← integral + error × 1/Δt
5:     output ← K_P × error + K_I × integral
6:     previous_error ← error
7:
8:     if output > (target_tps × margin) and ¬saturated
   then
9:         start OLAP worker
10:    else
11:        saturated ← true
```

Figure 8: Client Balancer Algorithm

Execution Logic

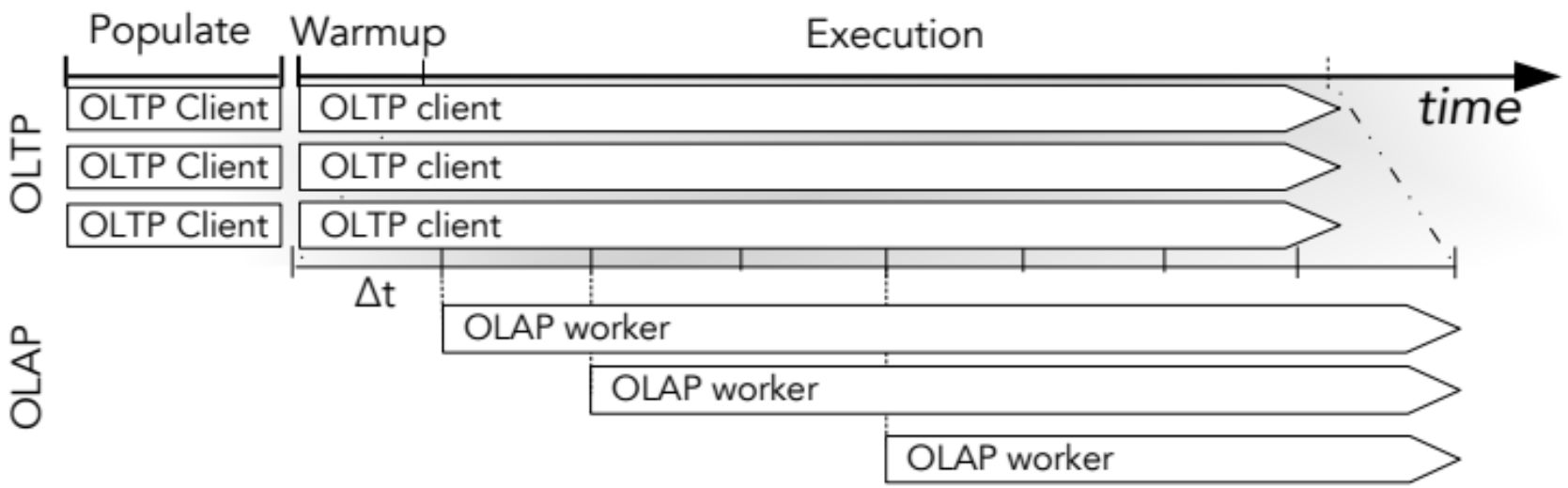


Figure 9: HTAPBench Execution

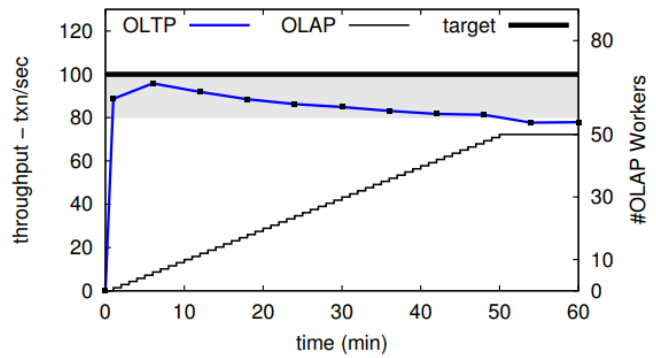
Unified Metric

QpHpW, short for "Queries of type H per Hour per Worker", means the number of analytical queries executed per OLAP worker regarding a system that is able to sustain the configured tps. A higher QpHpW maps a system where each OLAP worker is able to compute more queries per analytical worker.

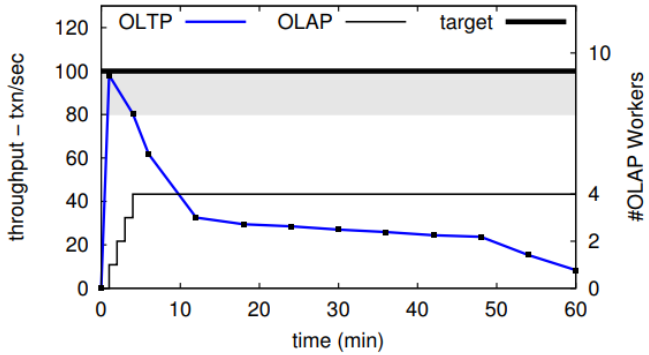
$$QpHpW = \frac{QphH}{\#OLAPworkers}@tpmC \tag{6}$$

Experiment Configuration

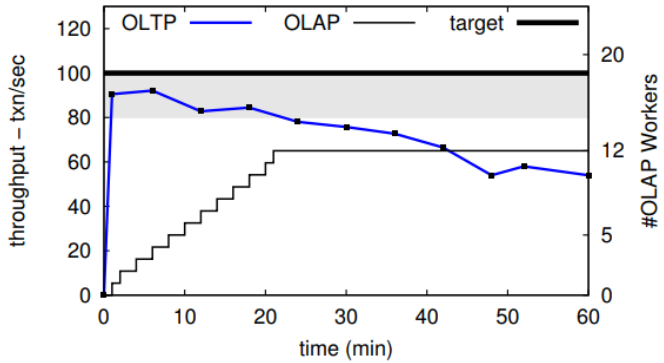
Client Balancer Δt 60 seconds, Target Tps 100, Execution Time 60 minutes.



(a) OLTP System Throughput



(b) OLAP System Throughput



(c) HTAP System Throughout

	# OLAP workers	QpH	QpHpW
OLTP	50	7	0.14 @ 756
OLAP	4	123	30.75 @ 217
Hybrid	12	169	14.14 @ 530

(d) Overview Results For Different System

Unified Metric

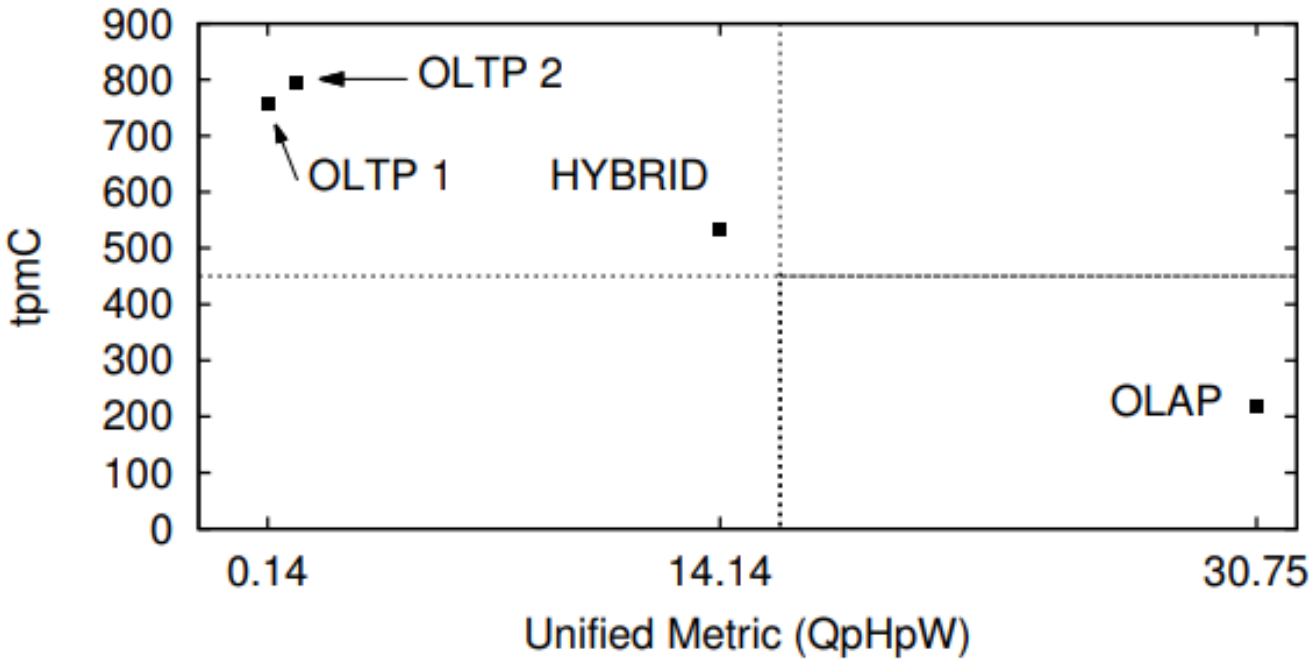
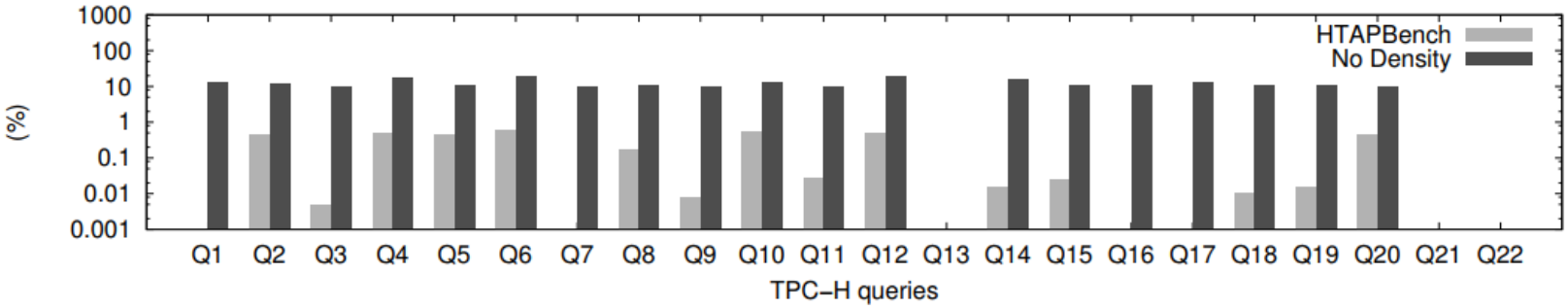


Figure 10: Quadrant Field Plot For Unified Metric

Homogeneity And Reproducibility



- Simple extension to CH-benCHmark.
- Deficiency in consideration on data freshness and isolation.
- Opaque experiments and non-unified experimental environment.

[CFG+11] Richard Cole, Florian Funke, Leo Giakoumakis, Wey Guy, Alfons Kemper, Stefan Krompass, Harumi Kuno, Raghunath Nambiar, Thomas Neumann, Meikel Poess, et al.
The mixed workload ch-benchmark.
In *Proceedings of the Fourth International Workshop on Testing Database Systems*, pages 1–6, 2011.

[HLC+20] Dongxu Huang, Qi Liu, Qiu Cui, Zhuhe Fang, Xiaoyu Ma, Fei Xu, Li Shen, Liu Tang, Yuxing Zhou, Menglong Huang, et al.
Tidb: a raft-based htap database.
Proceedings of the VLDB Endowment, 13(12):3072–3084, 2020.

[ÖTT17] Fatma Özcan, Yuanyuan Tian, and Pinar Tözün.
Hybrid transactional/analytical processing: A survey.
In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1771–1775, 2017.

[PWM+14] Iraklis Psaroudakis, Florian Wolf, Norman May, Thomas Neumann, Alexander Böhm, Anastasia Ailamaki, and Kai-Uwe Sattler.
Scaling up mixed workloads: a battle of data freshness, flexibility, and scheduling.
In *Technology Conference on Performance Evaluation and Benchmarking*, pages 97–112. Springer, 2014.

[SCCZ21] Sijie Shen, Rong Chen, Haibo Chen, and Binyu Zang.
Retrofitting high availability mechanism to tame hybrid transaction/analytical processing.
In *15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21)*, pages 219–238, 2021.

[YRX+20] Jiacheng Yang, Ian Rae, Jun Xu, Jeff Shute, Zhan Yuan, Kelvin Lau, Qiang Zeng, Xi Zhao, Jun Ma, Ziyang Chen, et al.
F1 lightning: Htap as a service.
Proceedings of the VLDB Endowment, 13(12):3313–3325, 2020.

Thanks!