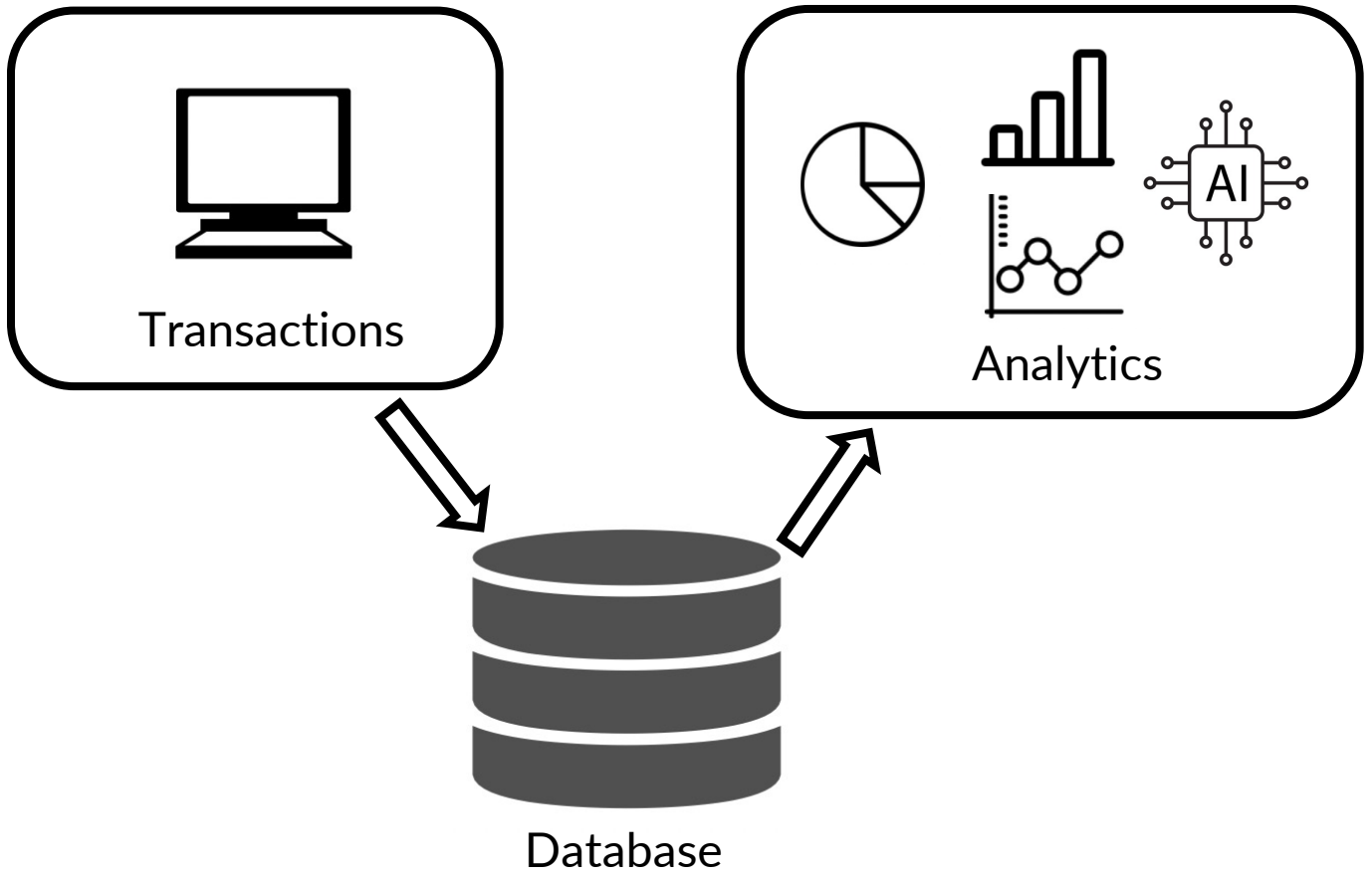
How Good is My HTAP System?

Elena Milkai, Yannis Chronis, Kevin P. Gaffney, Zhihan Guo,
Jignesh M. Patel, Xiangyao Yu

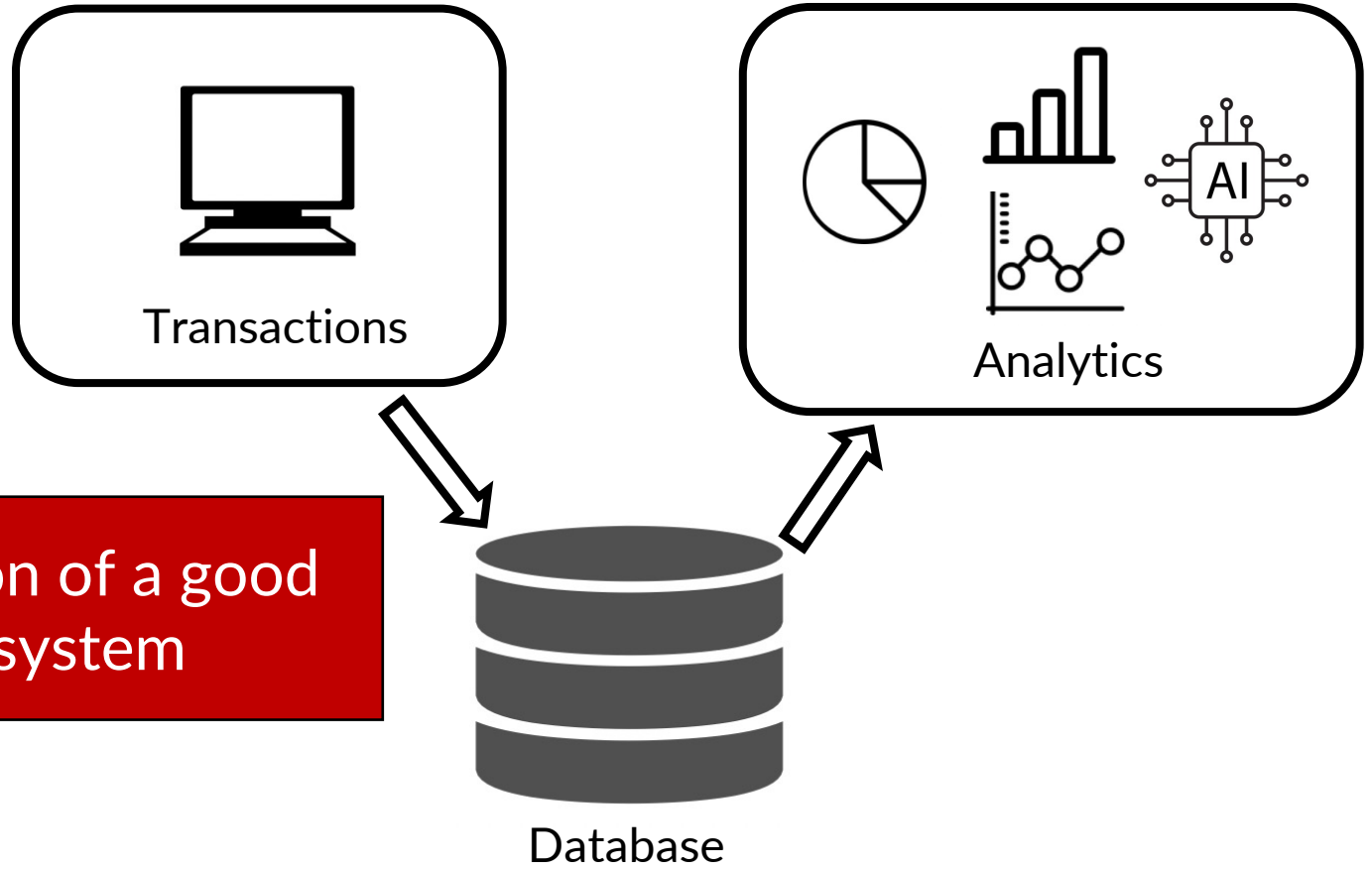
University of Wisconsin-Madison

This work was supported in part by the Semiconductor Research Corporation (SRC) and DARPA.

Hybrid Transactional & Analytical Processing

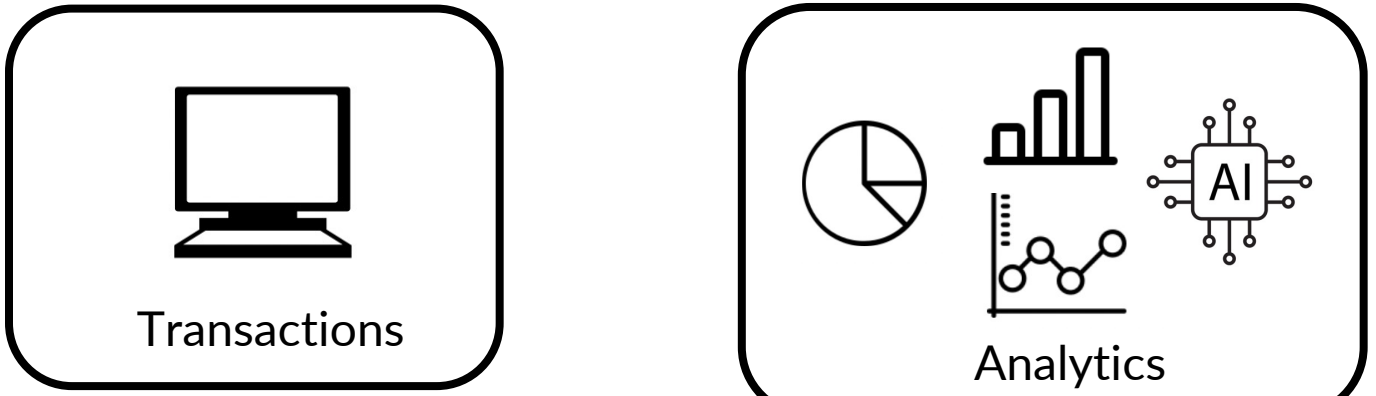


Hybrid Transactional & Analytical Processing



No definition of a good HTAP system

Hybrid Transactional & Analytical Processing



Database

No definition of a good HTAP system

Difficult to characterize and compare HTAP systems

Fundamental Goals of HTAP

Goal 1 **Performance Isolation**

No interference between the transactional (*T*) and analytical (*A*) portions of the workload

Fundamental Goals of HTAP

Goal 1 **Performance Isolation**

No interference between the transactional (*T*) and analytical (*A*) portions of the workload

Goal 2 **Real-Time Analytics**

Analytical queries observe the latest transactional updates ***(fresh data)***

Our Goal & Contributions

Goal

Define a ***good*** HTAP system & provide a systematic methodology to evaluate HTAP systems

Our Goal & Contributions

Goal Define a ***good*** HTAP system & provide a systematic methodology to evaluate HTAP systems

Contributions

- Provide a ***performance metric*** *for* HTAP systems

Our Goal & Contributions

Goal Define a ***good*** HTAP system & provide a systematic methodology to evaluate HTAP systems

Contributions

- Provide a ***performance metric*** *for* HTAP systems
- Quantify and measure ***freshness*** of analytical queries

Our Goal & Contributions

Goal Define a ***good*** HTAP system & provide a systematic methodology to evaluate HTAP systems

Contributions

- Provide a ***performance metric*** *for* HTAP systems
- Quantify and measure ***freshness*** of analytical queries
- New benchmark called ***HATtrick*** to measure performance and freshness

Our Goal & Contributions

Goal Define a ***good*** HTAP system & provide a systematic methodology to evaluate HTAP systems

Contributions

- Provide a ***performance metric*** *for* HTAP systems
- Quantify and measure ***freshness*** of analytical queries
- New benchmark called ***HATtrick*** to measure performance and freshness
- Use ***HATtrick*** to evaluate representative HTAP systems

C1: Measure performance of an HTAP system

Metric 1: ***Throughput Frontier***

Throughput Frontier Definition

- Input: (τ, α) pairs
  - **τ** transactional-clients
  - **α** analytical-clients

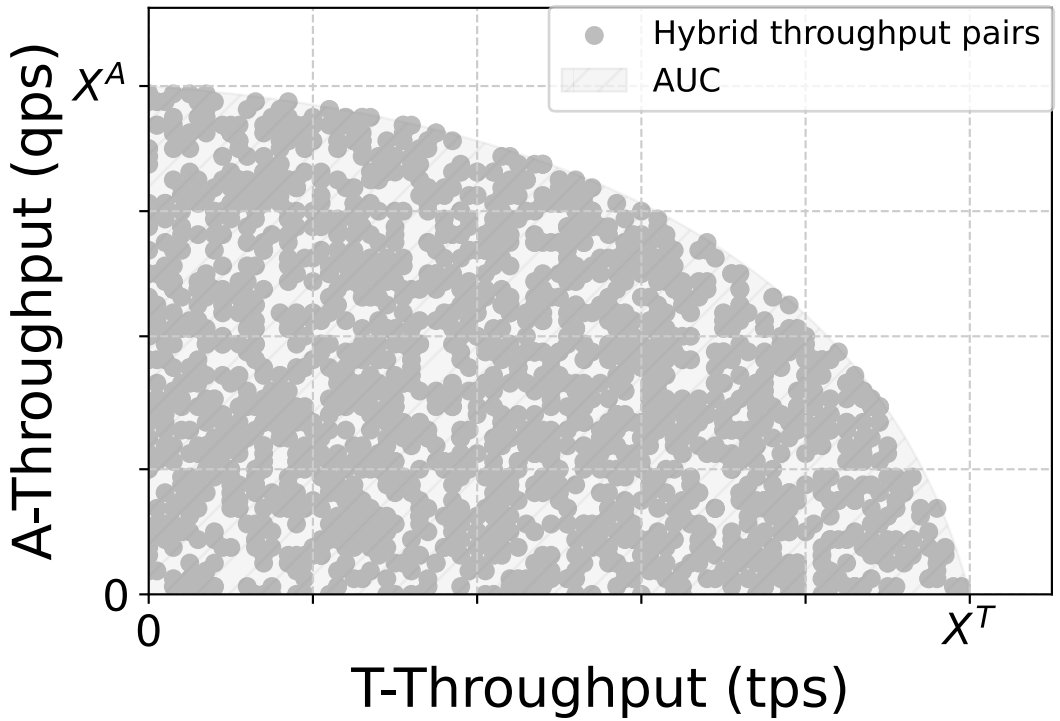
Output: **hybrid throughput** (x_τ , x_α)

Throughput Frontier Definition

- Input: (τ, α) pairs
  - **τ** transactional-clients
  - **α** analytical-clients

Output: **hybrid throughput** (x_τ , x_α)

- Sample all possible (τ, α) mixes
  - Map hybrid throughputs to 2D space

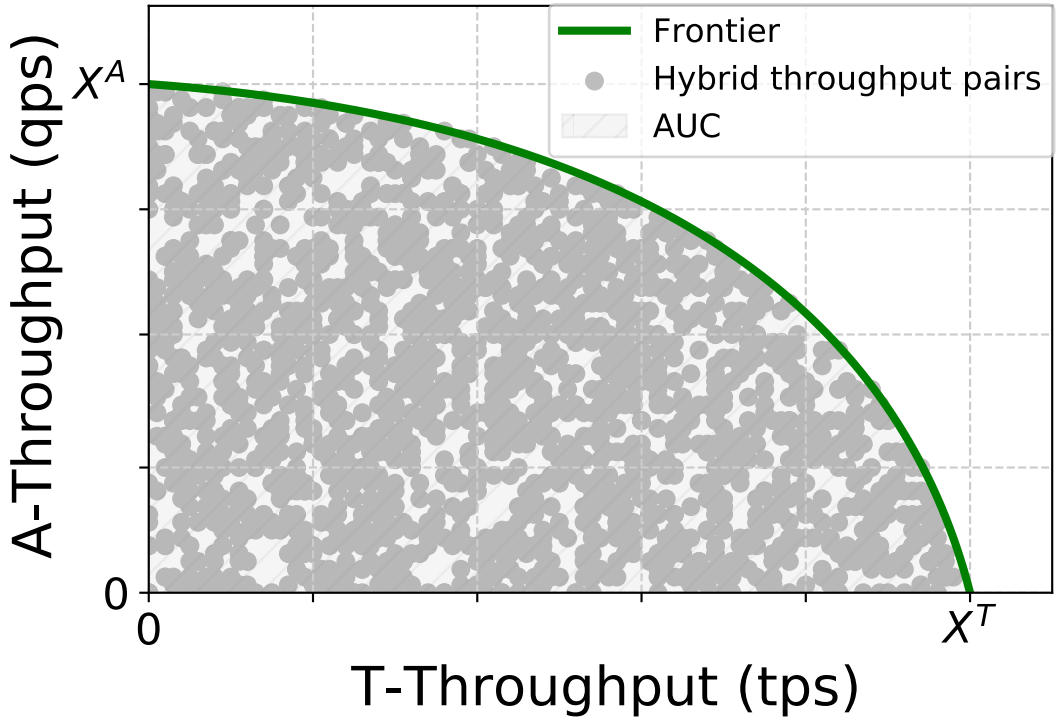


Throughput Frontier Definition

- Input: (τ, α) pairs
  - **τ** transactional-clients
  - **α** analytical-clients

Output: **hybrid throughput** (x_τ , x_α)

- Sample all possible (τ, α) mixes
  - Map hybrid throughputs to 2D space



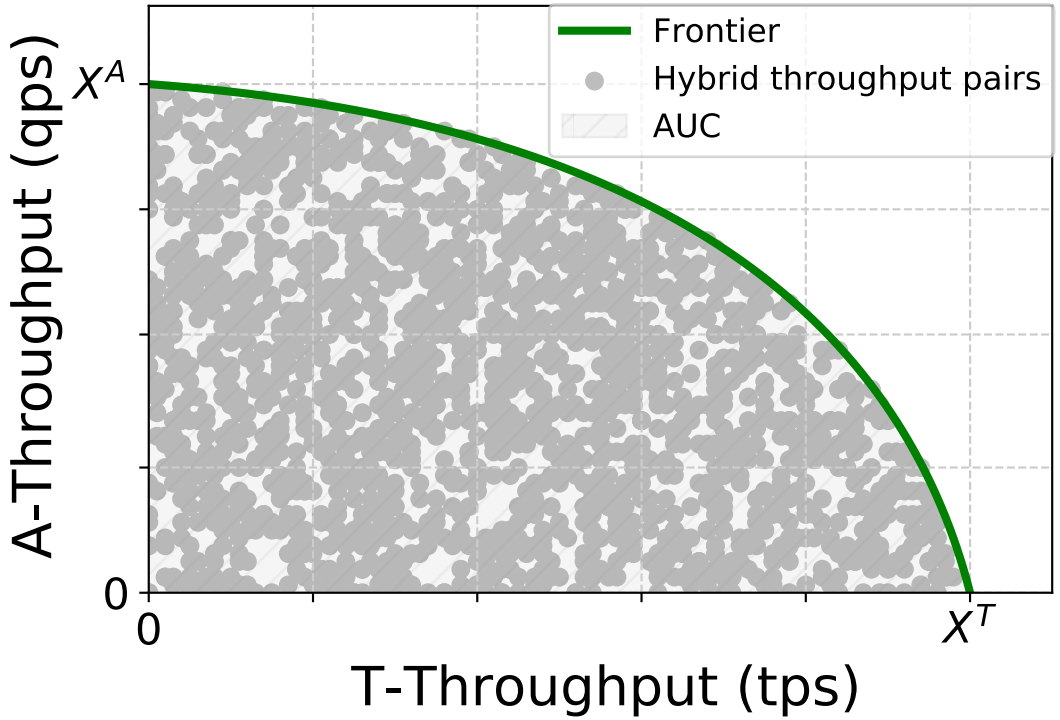
- **Max** hybrid throughput values → ***throughput frontier***

Throughput Frontier Definition

- Input: (τ, α) pairs
  - **τ** transactional-clients
  - **α** analytical-clients

Output: **hybrid throughput** (x_τ , x_α)

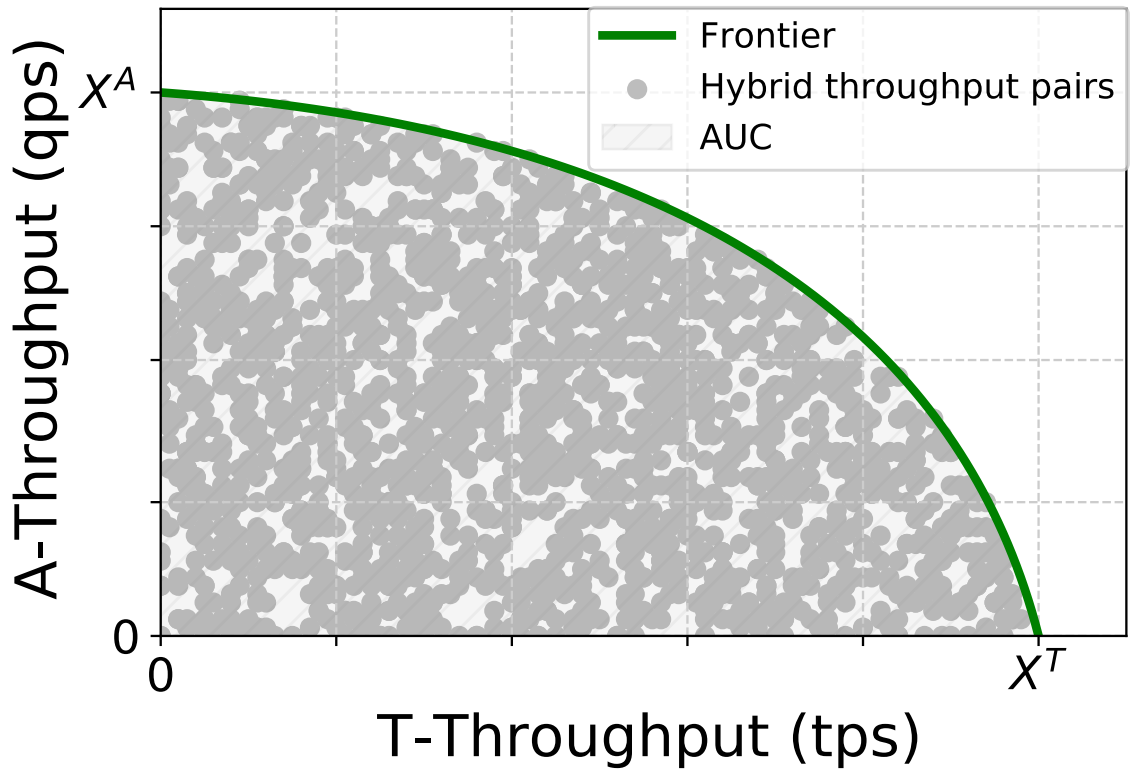
- Sample all possible (τ, α) mixes
  - Map hybrid throughputs to 2D space



- **Max** hybrid throughput values → ***throughput frontier*** captures
  - ***T*** and ***A*** **throughput**
  - **Interference** between *T* & *A* portions of the workload

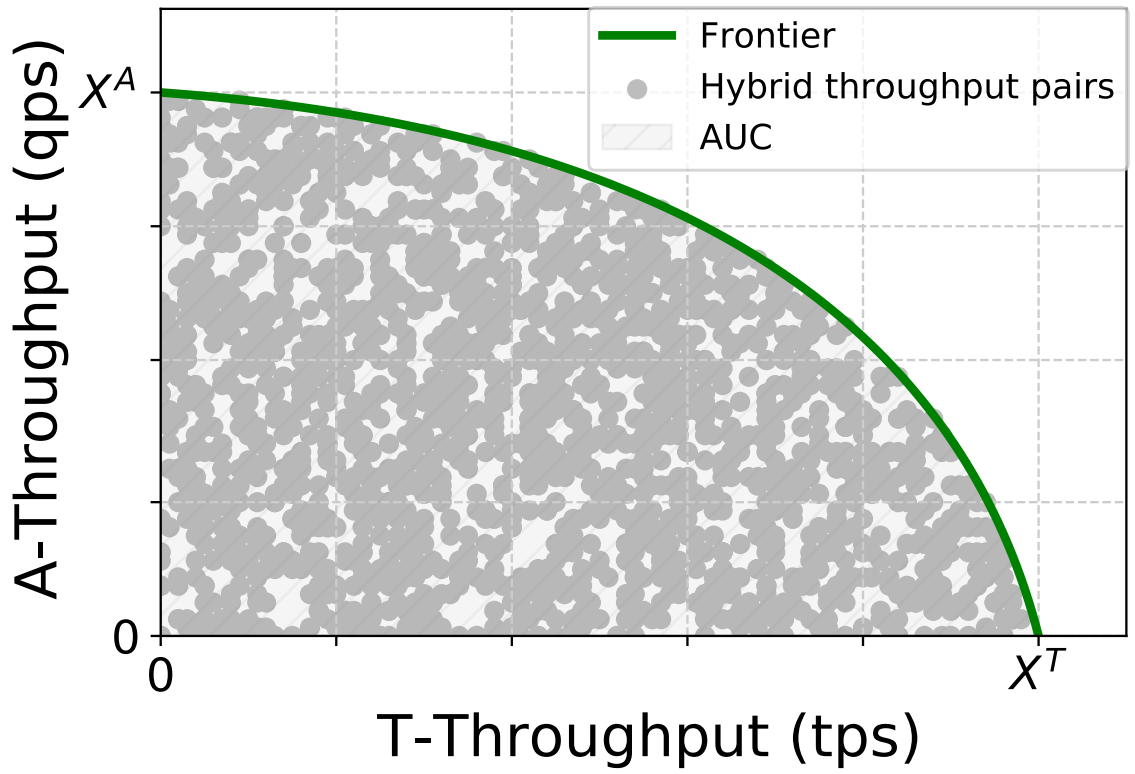
Calculate Throughput Frontier

➢**Sampling method**: Accurate, but time-consuming



Calculate Throughput Frontier

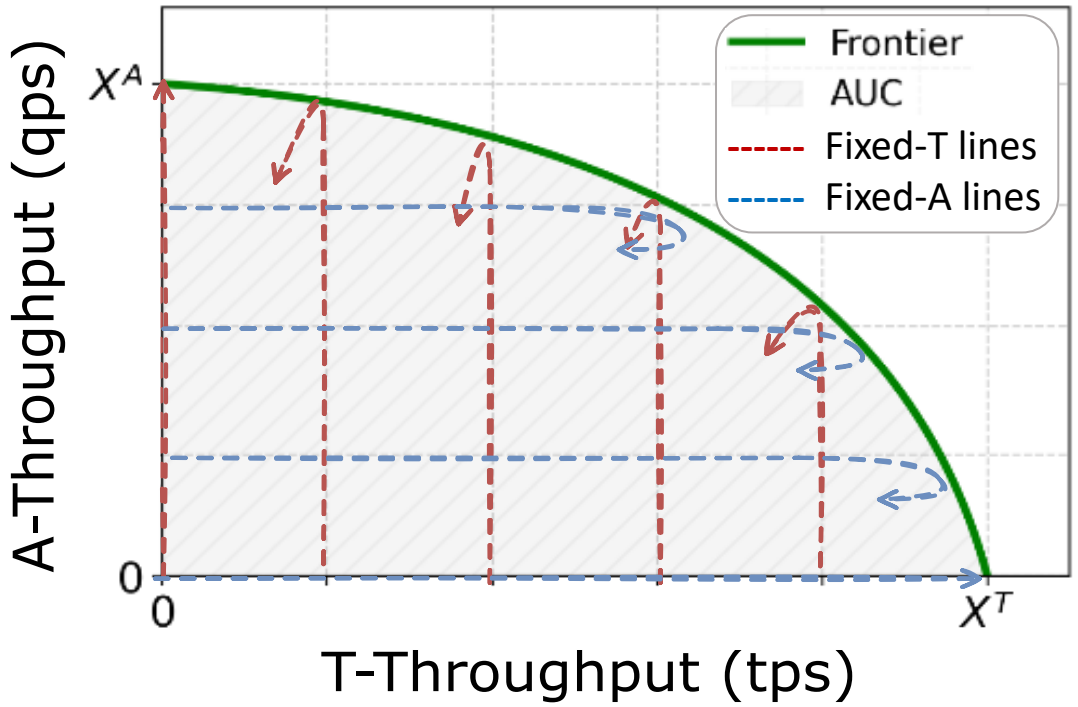
➢**Sampling method**: Accurate, but time-consuming



Calculate Throughput Frontier

➢ **Saturation method**: Systematic way

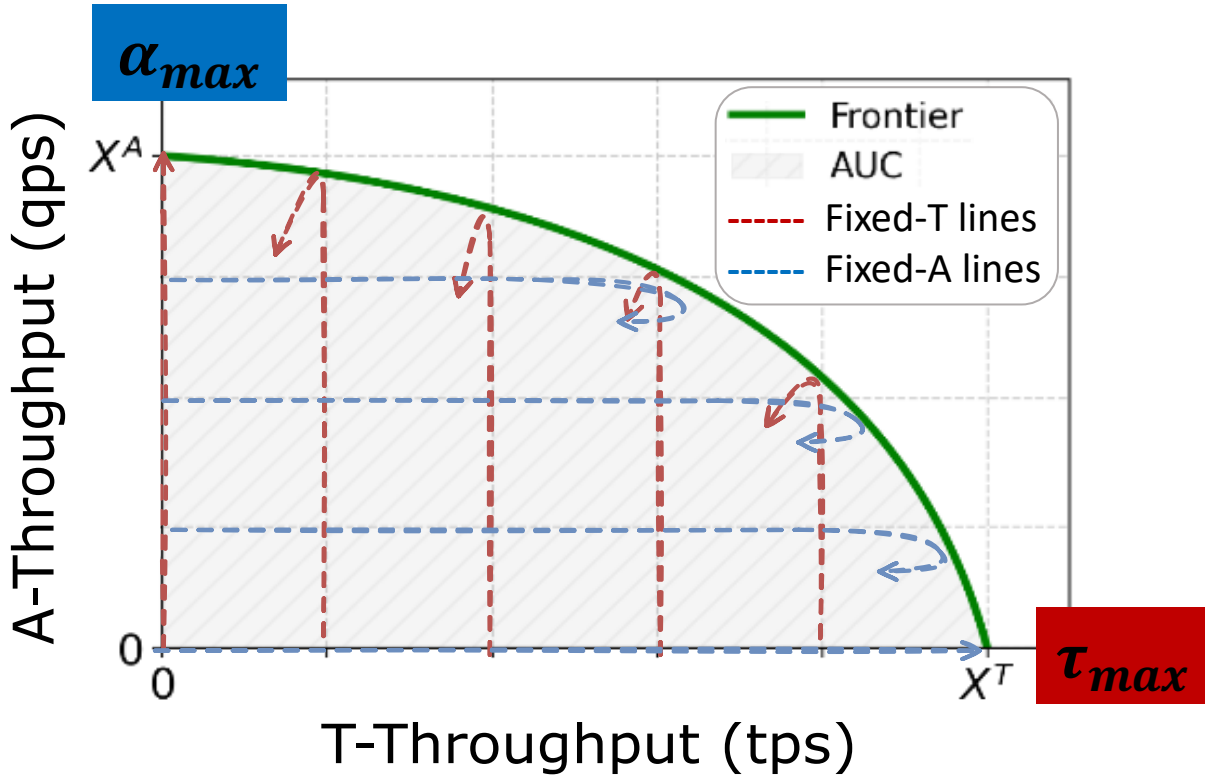
- Constant number of steps



Calculate Throughput Frontier

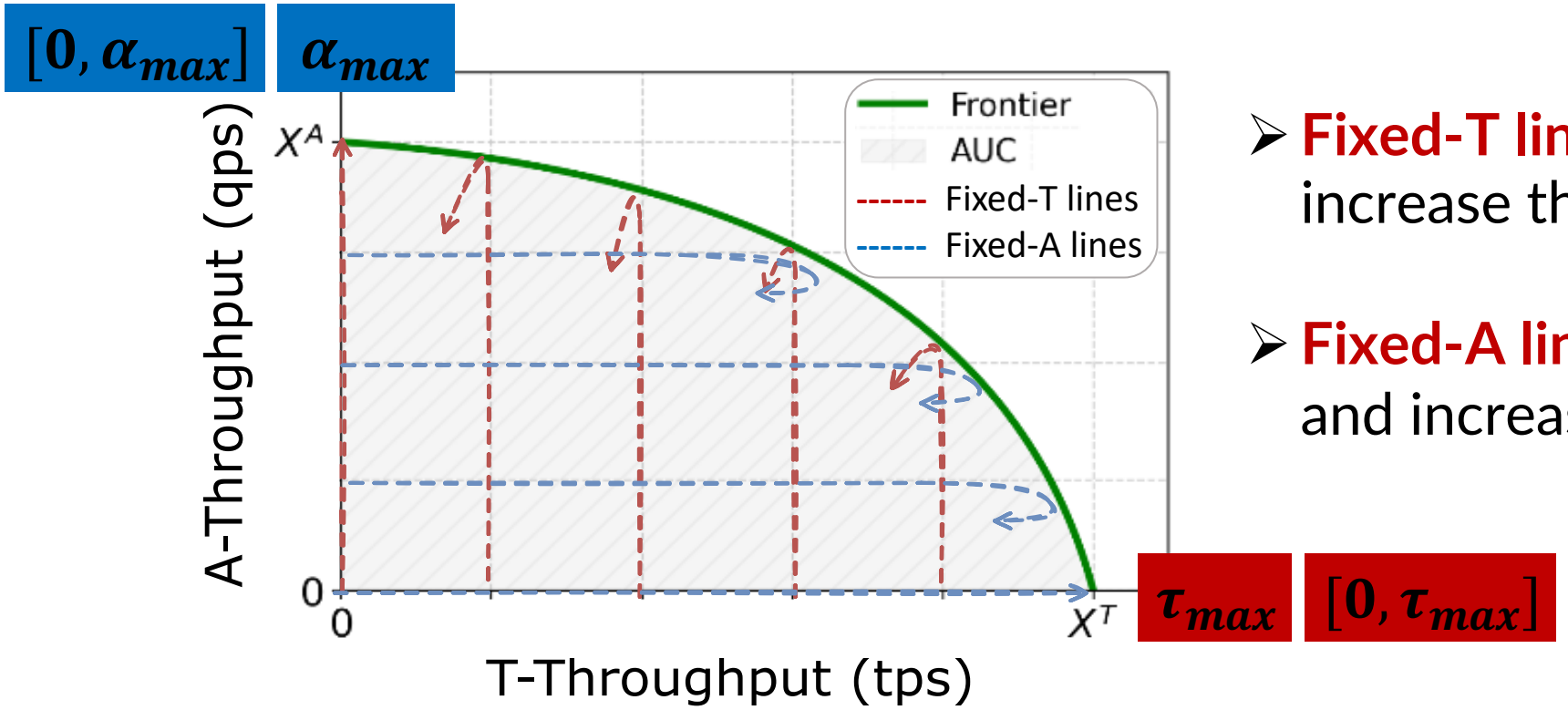
➢ **Saturation method**: Systematic way

- Constant number of steps



Calculate Throughput Frontier

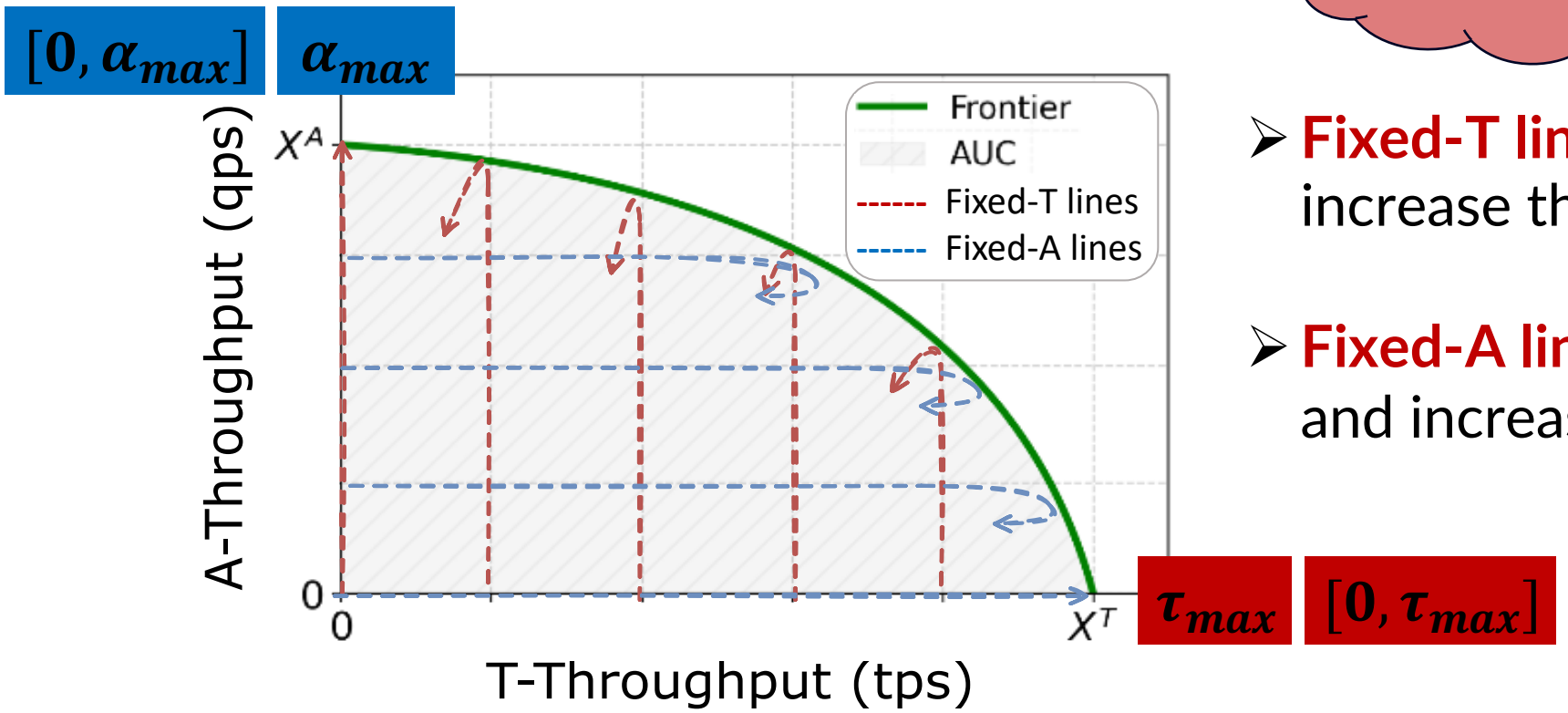
- **Saturation method**: Systematic way
  - Constant number of steps



- **Fixed-T lines**: Fix the # of *T* clients and increase the # of *A* clients
- **Fixed-A lines**: Fix the # of the *A* clients and increase the # of the *T* clients

Calculate Throughput Frontier

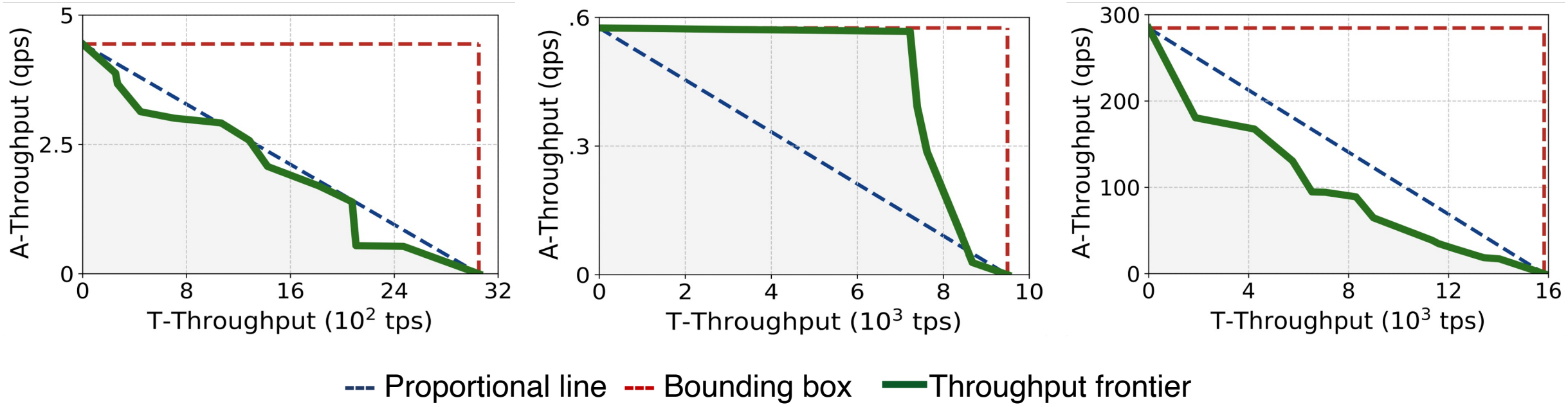
- **Saturation method**: Systematic way
  - Constant number of steps



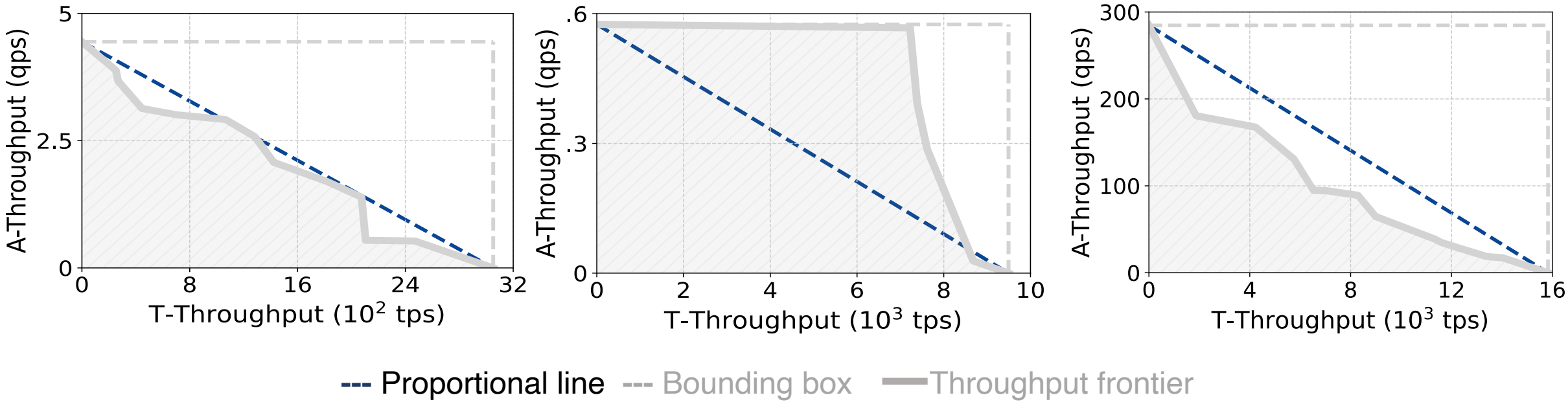
36 points

- **Fixed-T lines**: Fix the # of *T* clients and increase the # of *A* clients
- **Fixed-A lines**: Fix the # of the *A* clients and increase the # of the *T* clients

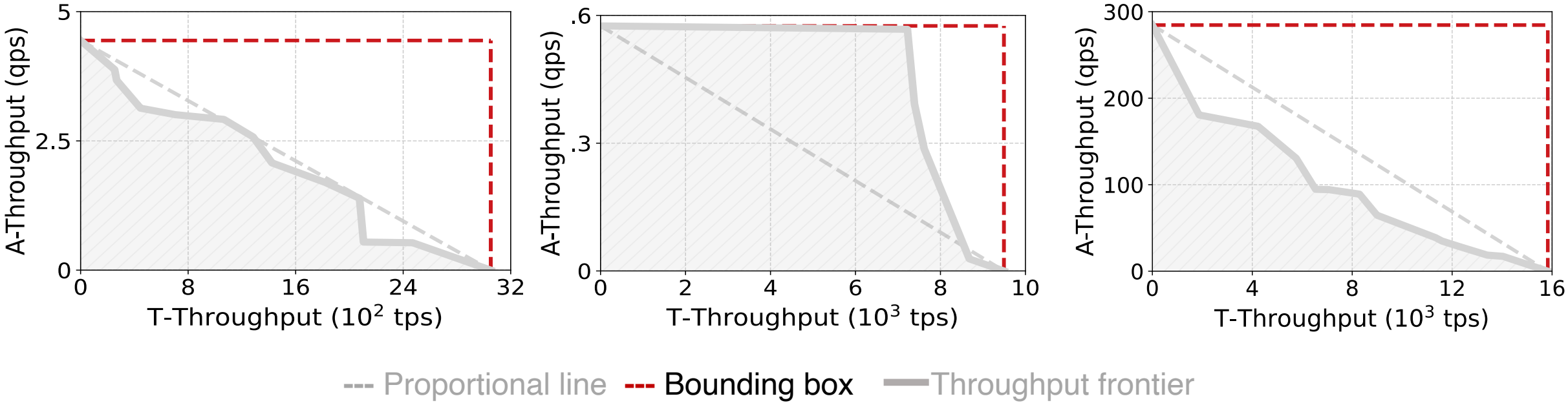
Patterns of Throughput Frontier



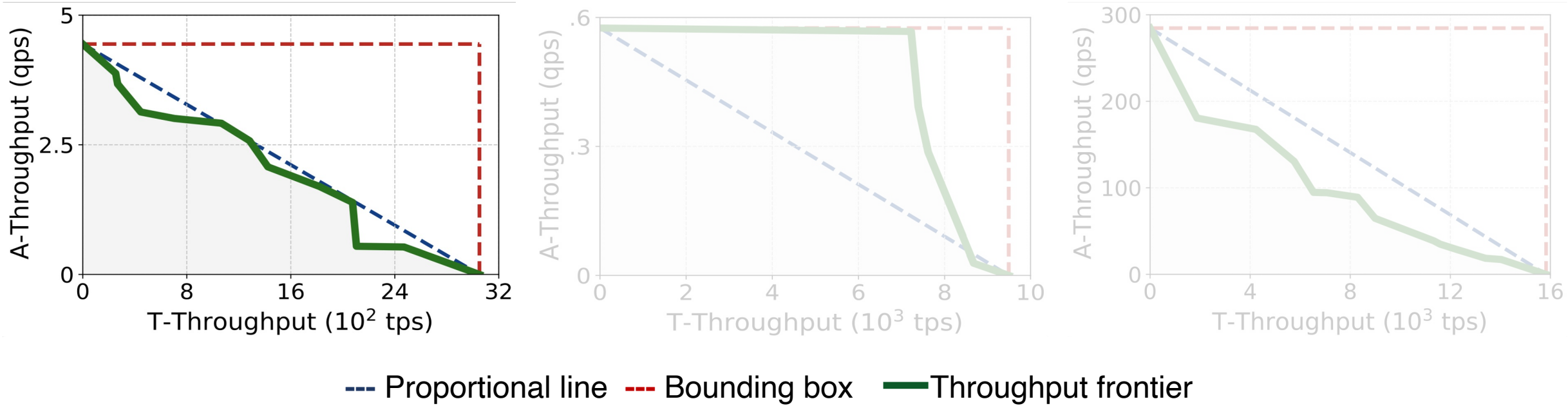
Proportional Line



Bounding Box

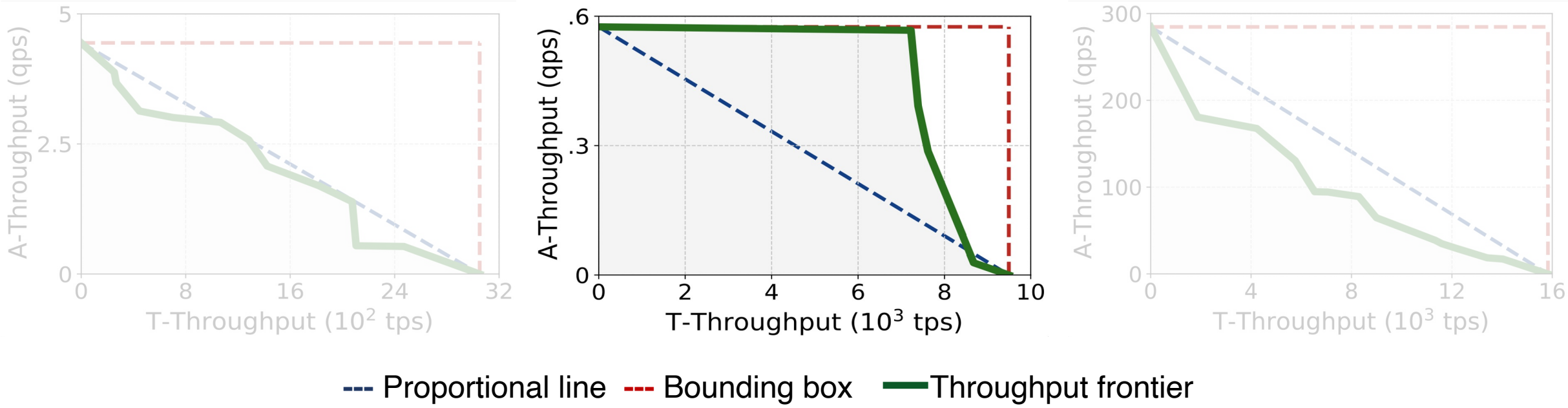


Patterns of Throughput Frontier I



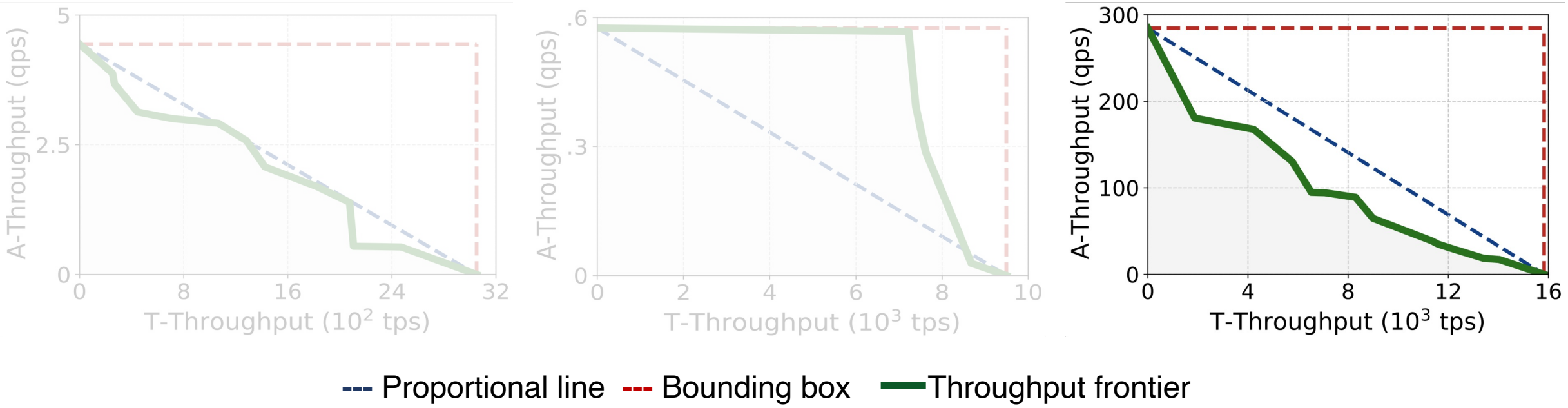
- Pattern 1: **Close to the proportional line**
  - Linear dependence between transactions and analytics
  - Transactions and analytics share resources

Patterns of Throughput Frontier II



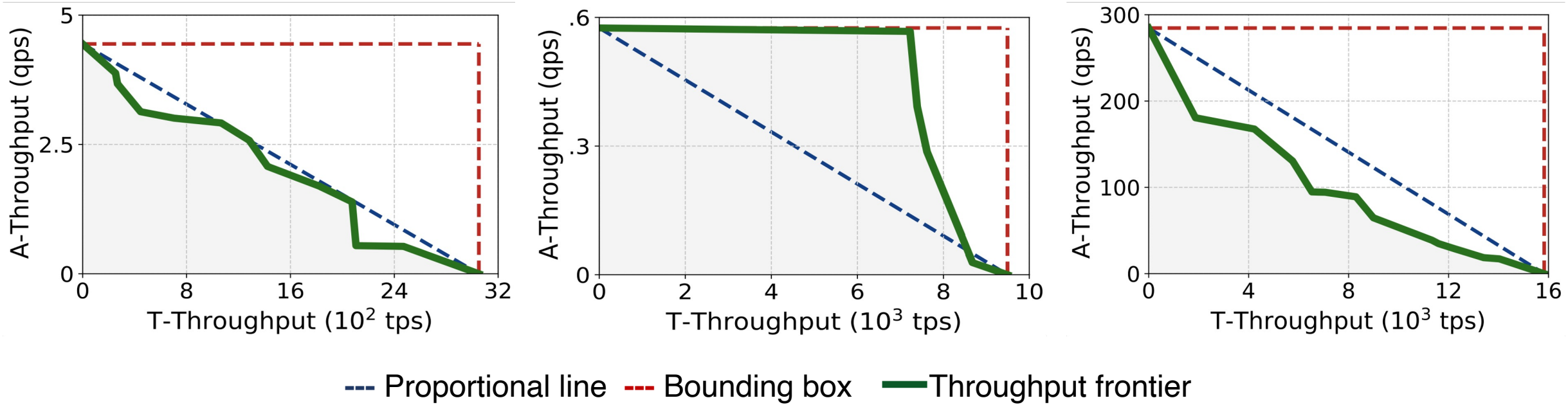
- Pattern 2: **Above the proportional line, close to the bounding box**
  - Independence between transactions and analytics
  - Performance isolation

Patterns of Throughput Frontier III

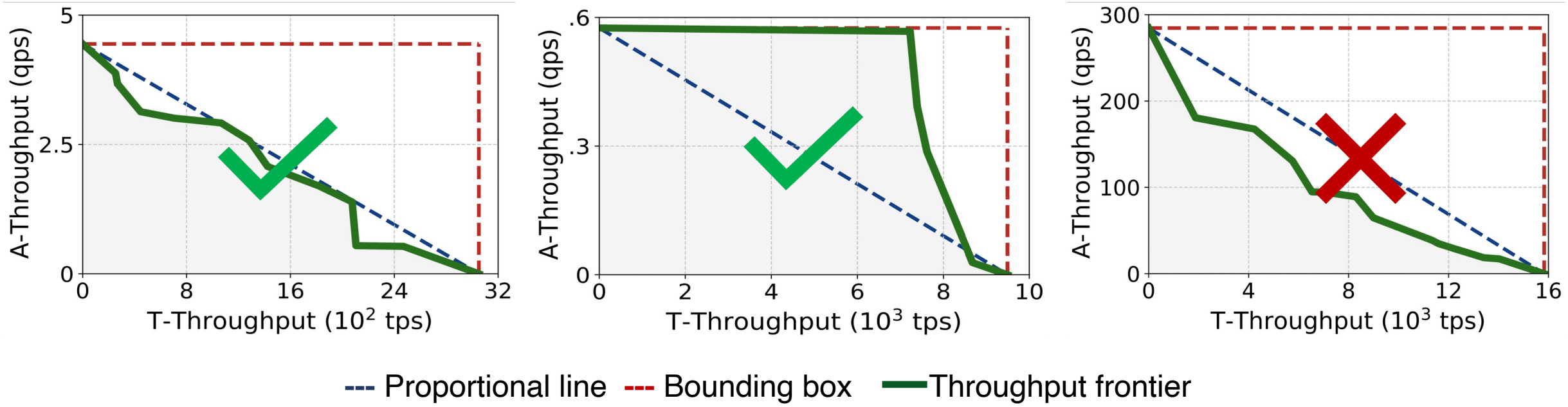


- Pattern 3: **Below the proportional line, close to the axes**
  - Interference between transactions and analytics
  - Contention for resources

Patterns of Throughput Frontier



Patterns of Throughput Frontier



Frontier Interpretation

- Quantifies **absolute performance**

Frontier Interpretation

- Quantifies **absolute performance**
- **Isolation** and **interference** between *T* and *A* workloads

Frontier Interpretation

- Quantifies **absolute performance**
- **Isolation** and **interference** between *T* and *A* workloads
- Diagnose **performance issues**

Frontier Interpretation

- Quantifies **absolute performance**
- **Isolation** and **interference** between *T* and *A* workloads
- Diagnose **performance issues**
- Discover the **architecture design** of an HTAP system

C2: Quantify and measure freshness of an HTAP system

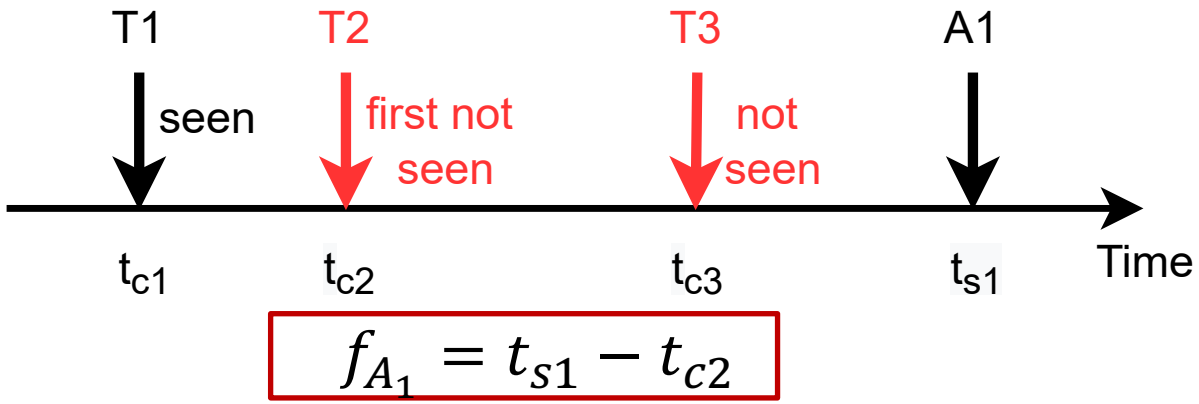
Metric 2: ***Freshness***

Freshness Definition

- **Metric** to extract the **recency** of the data snapshots used when an analytical query runs

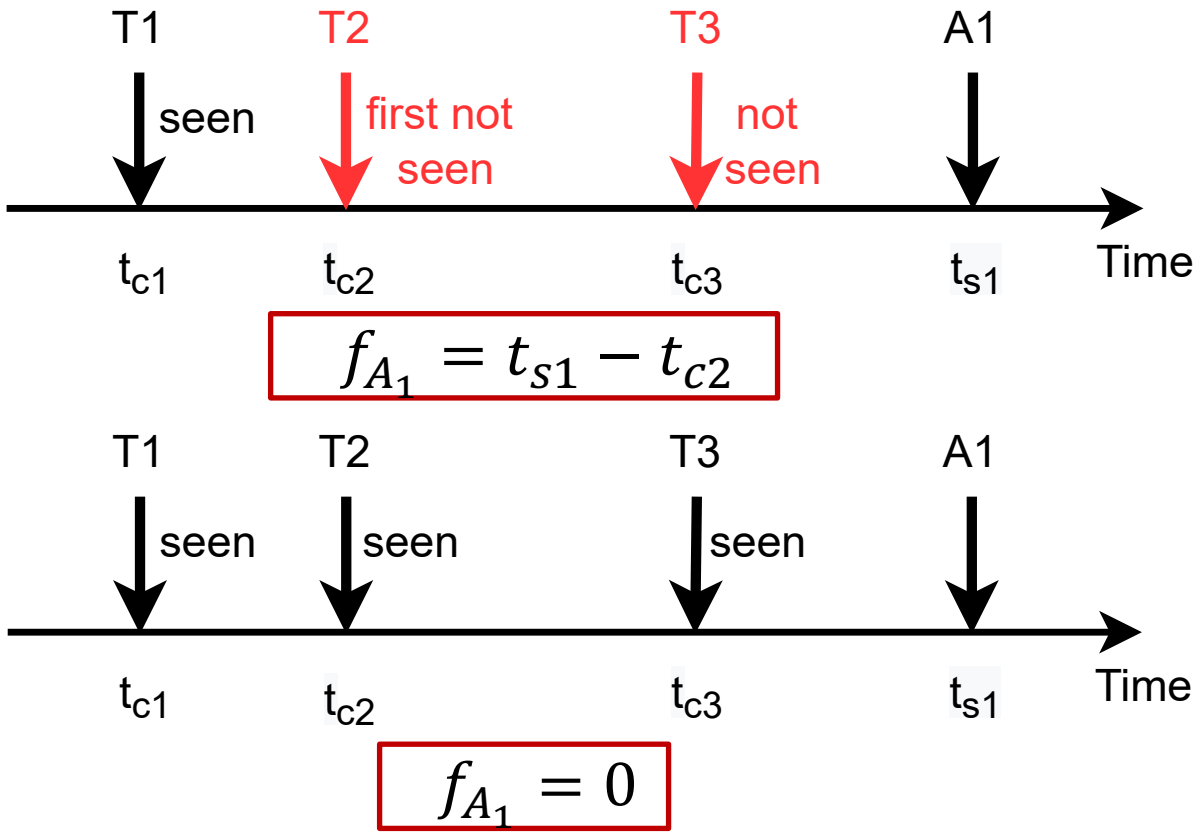
Freshness Definition

- **Metric** to extract the **recency** of the data snapshots used when an analytical query runs



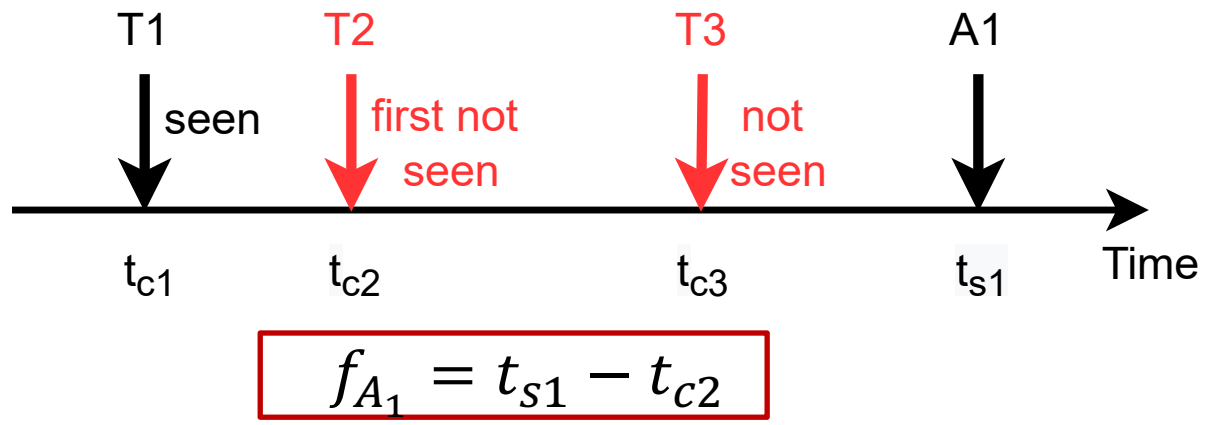
Freshness Definition

- **Metric** to extract the **recency** of the data snapshots used when an analytical query runs

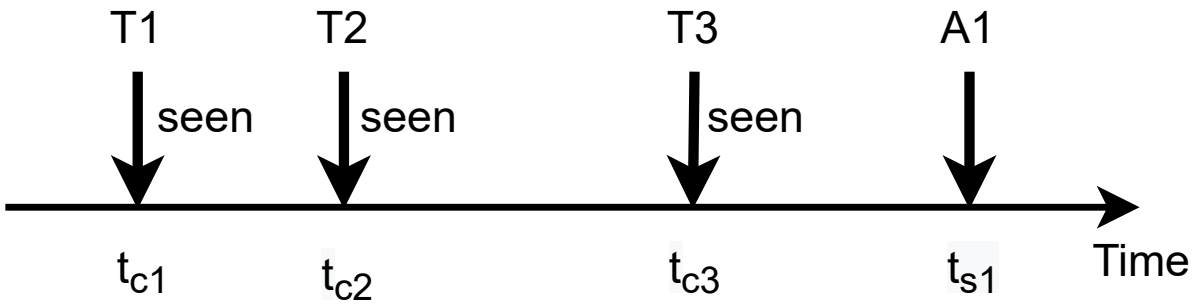


Freshness Definition

- **Metric** to extract the **recency** of the data snapshots used when an analytical query runs



$$f_{A_1} = t_{s1} - t_{c2}$$



$$f_{A_1} = 0$$

Freshness of A_q:

$$f_{A_q} = \max(0, t^{s}_{A_q} - t^{fns}_{A_q})$$

$t^{s}_{A_q}$: start time of the A_q

$t^{fns}_{A_q}$: commit time of **f**irst **n**ot **s**een by A_q

Challenges in Measuring Freshness

$$f_{A_q} = \max(0, t^{S}_{A_q} - t^{fns}_{A_q})$$

Challenges in Measuring Freshness

$$f_{A_q} = \max(0, t^{S}_{A_q} - t^{fns}_{A_q})$$

- Challenge 1: **No globally synchronized clock**

Challenges in Measuring Freshness

$$f_{A_q} = \max(0, t^{S}_{A_q} - t^{fns}_{A_q})$$

- Challenge 1: **No globally synchronized clock**
- Solution: Collect time only on the client side

Challenges in Measuring Freshness

$$f_{A_q} = \max(0, t^{s}_{A_q} - t^{fns}_{A_q})$$

- Challenge 1: **No globally synchronized clock**
- Solution: Collect time only on the client side

- Challenge 2: **Hard to identify first-not-seen transaction**

Challenges in Measuring Freshness

$$f_{A_q} = \max(0, t^{s}_{A_q} - t^{fns}_{A_q})$$

- Challenge 1: **No globally synchronized clock**
- Solution: Collect time only on the client side

- Challenge 2: **Hard to identify first-not-seen transaction**
- Solution: Auxiliary tables storing monotonically increasing IDs
  - Identify seen vs. not seen transactions by a query

C3: Design a new benchmark to measure performance and freshness

HATtrick

HATtrick Benchmark

- **Hybrid** benchmark
  - Analytical component: **Star-Schema benchmark** (SSB)
  - Transactional component: **Adapted** version of TPC-C benchmark
  - **Simpler** than previous HTAP benchmarks

- Throughput frontier & freshness can be added to every hybrid benchmark

- Source code is available at https://github.com/UWHustle/HATtrick

Evaluation

Experimental Configuration

- Systems
  - **Postgres** (single-node and multi-node with streaming replication)
  - **TiDB** (singe-node vs. distributed)
  - **System-X** (single-node)
- Dataset Size
  - SF100 (~80GB)

Systems Comparison

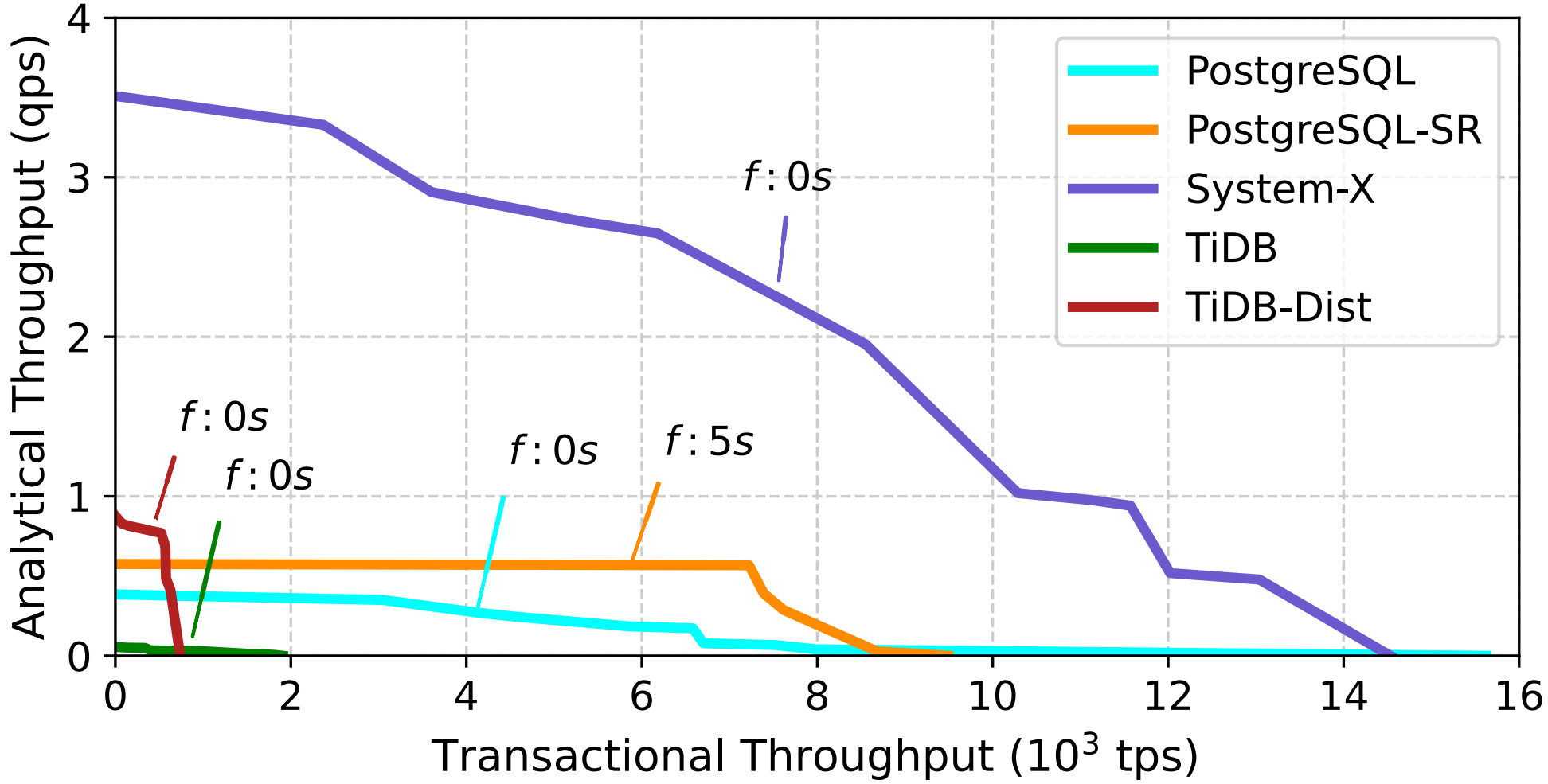


Fig.1: Throughput frontiers and 99th-percentile freshness scores *(f)* for the client mix 50:50 in seconds

Systems Comparison

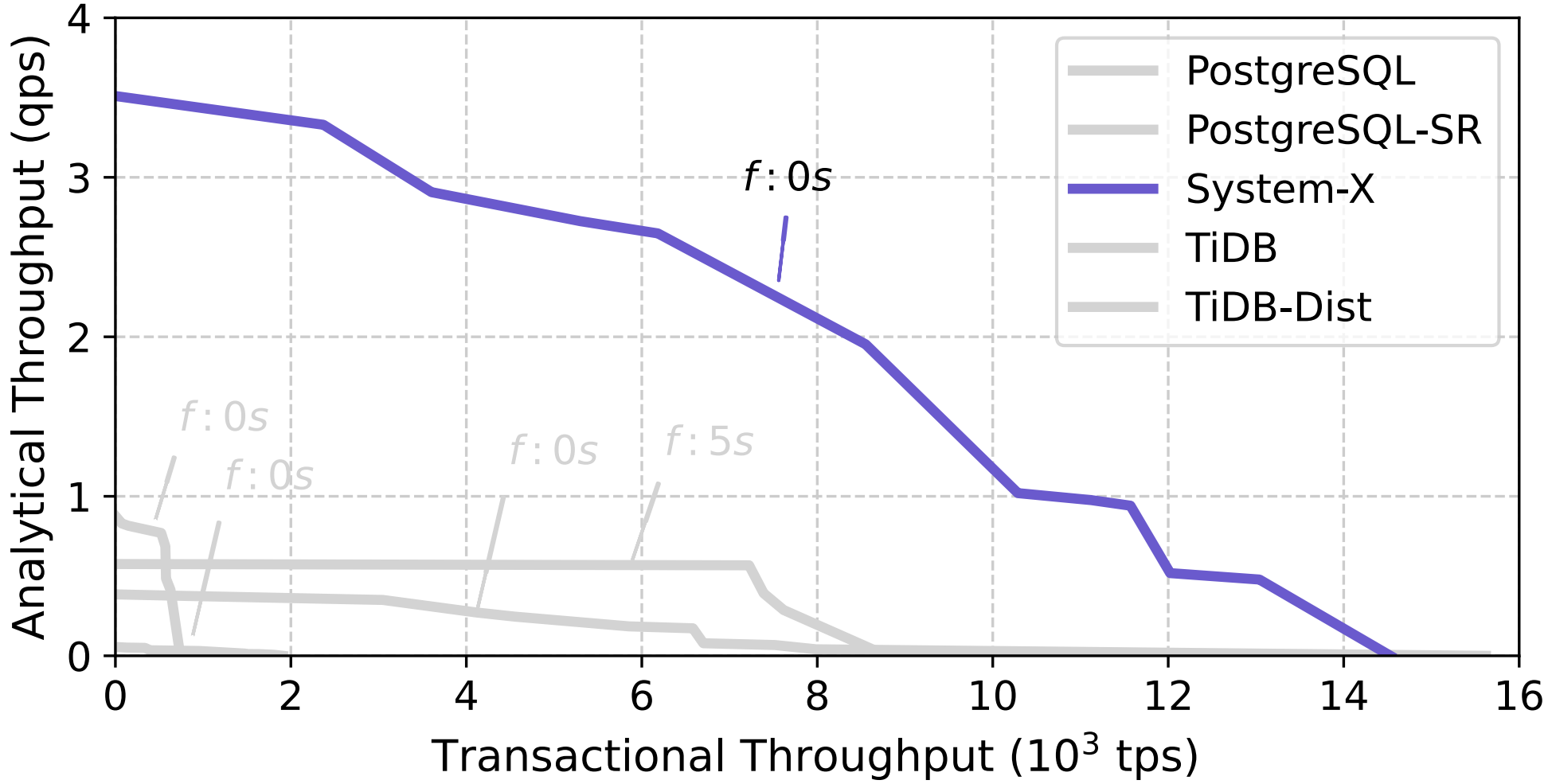


Fig.1: Throughput frontiers and 99th-percentile freshness scores *(f)* for the client mix 50:50 in seconds

Systems Comparison

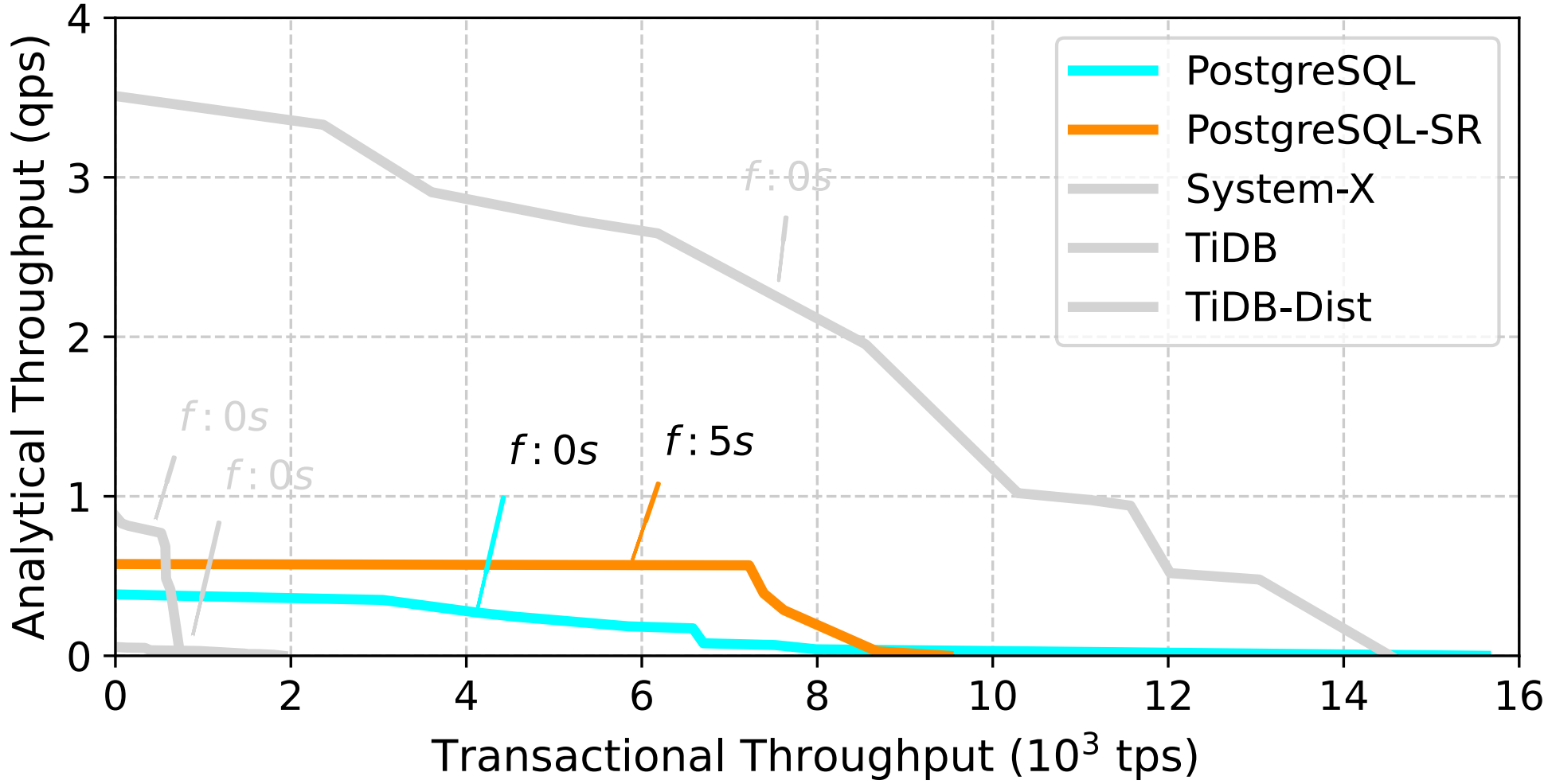


Fig.1: Throughput frontiers and 99th-percentile freshness scores *(f)* for the client mix 50:50 in seconds

Systems Comparison

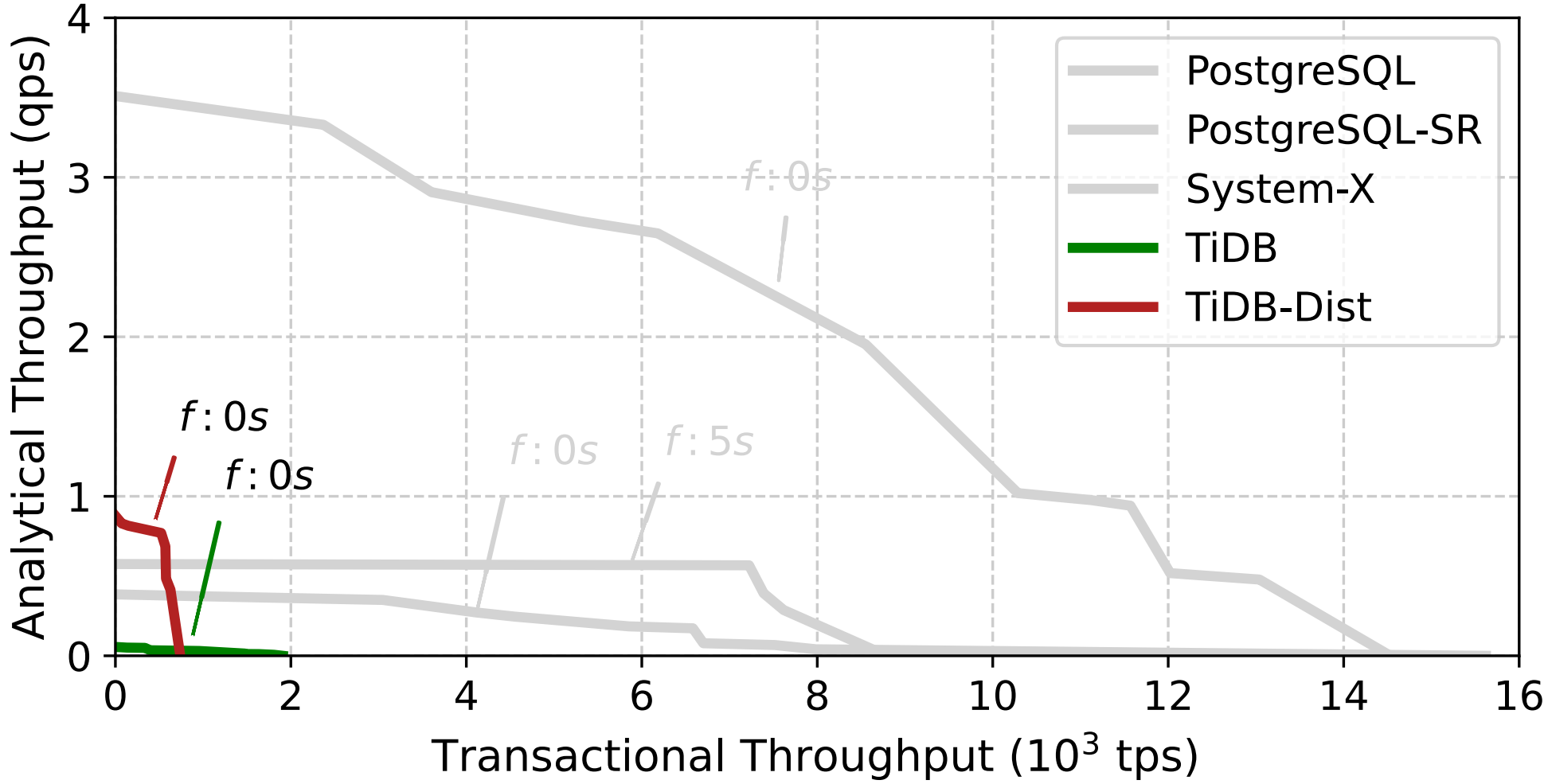


Fig.1: Throughput frontiers and 99th-percentile freshness scores *(f)* for the client mix 50:50 in seconds

Systems Comparison

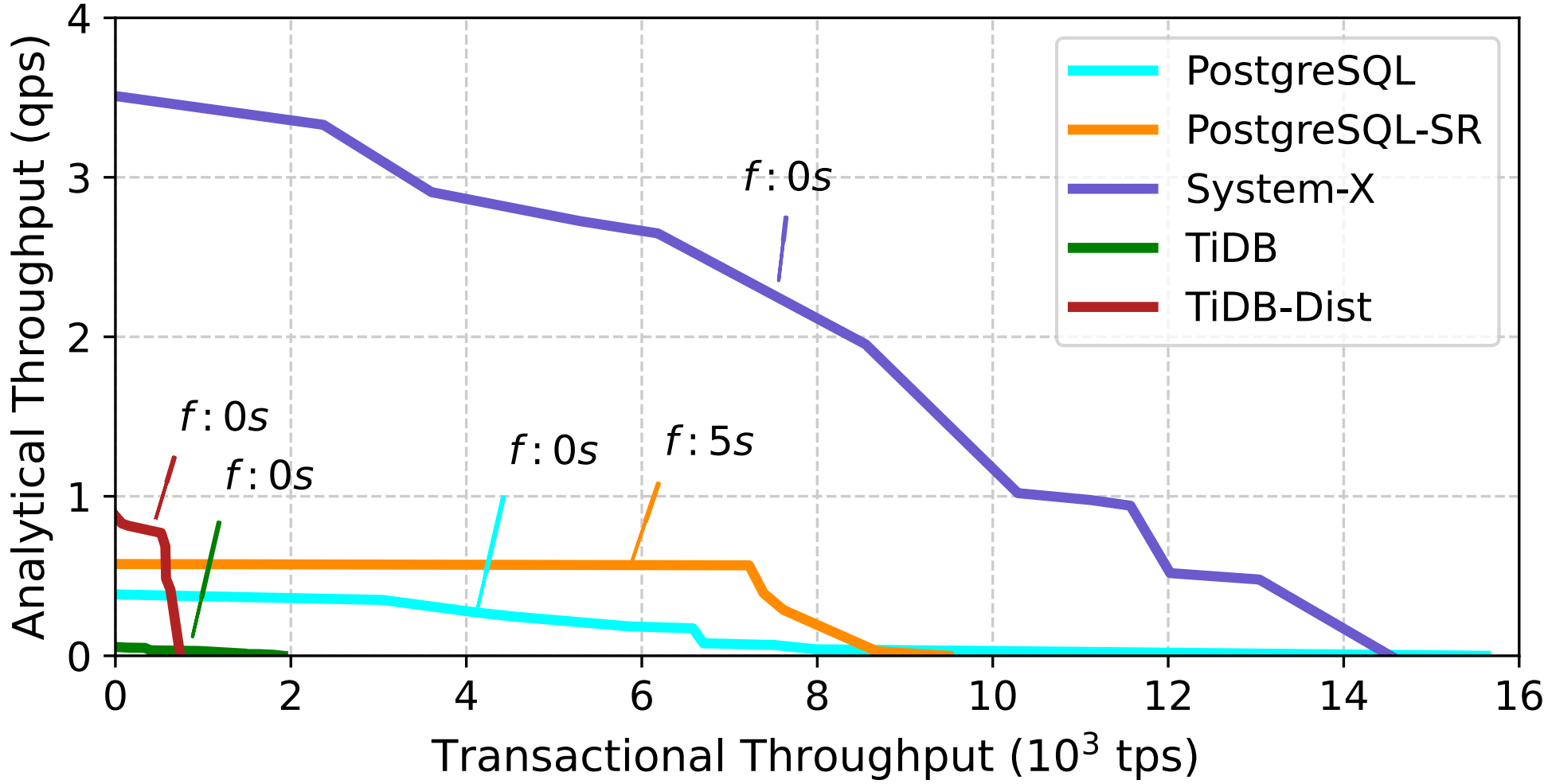
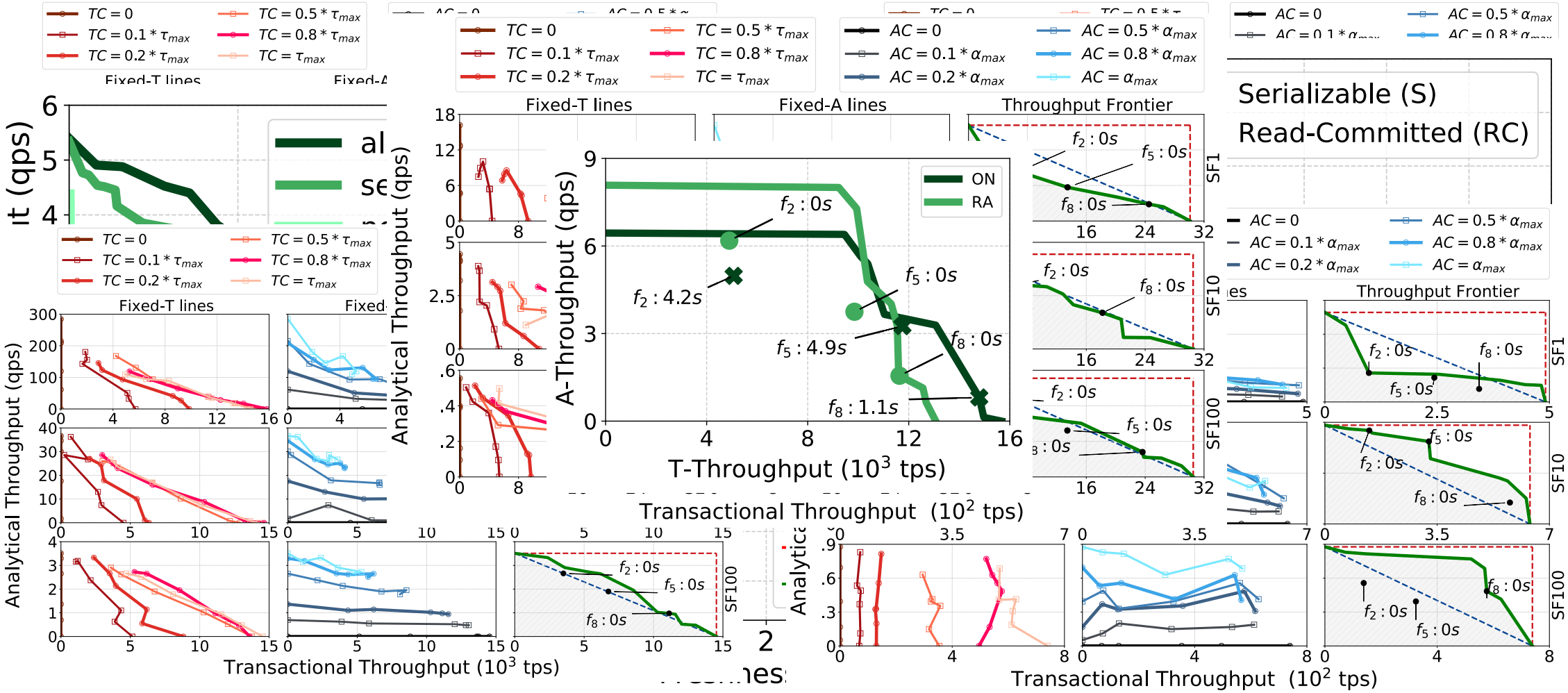


Fig.1: Throughput frontiers and 99th-percentile freshness scores *(f)* for the client mix 50:50 in seconds

More Experiments...



Conclusions

- **Throughput frontier** and **freshness**

Conclusions

- **Throughput frontier** and **freshness**
  - **Within system** configuration, discover trade-offs
  - Easier **across systems** comparison

Conclusions

- **Throughput frontier** and **freshness**
  - **Within system** configuration, discover trade-offs
  - Easier **across systems** comparison

- Lessons learned
  - Fresh analytics come with a **cost** in the *T* or/and *A* performance
  - *T*-throughput is **severely affected** by the increase of *A* clients

Conclusions

- **Throughput frontier** and **freshness**
  - **Within system** configuration, discover trade-offs
  - Easier **across systems** comparison

- Lessons learned
  - Fresh analytics come with a **cost** in the *T* or/and *A* performance
  - *T*-throughput is **severely affected** by the increase of *A* clients

- There is still room for improving current HTAP systems

Thank you

milkai@wisc.edu

This work was supported in part by the Semiconductor Research Corporation (SRC) and DARPA.