EventStream

EVENT STREAM

EventStream

CONTENT

EventStream

EVENT STREAM

<Keep>
<it>
<simple>

Ibrahim Chikani

Linkedin

Github

Nishitha Jogi

Linkedin

Github

PROBLEM STATEMENT

EventStream

Design a probabilistic data structure that tracks the frequency of millions of real-time events with minimal memory and supports accurate percentile queries.

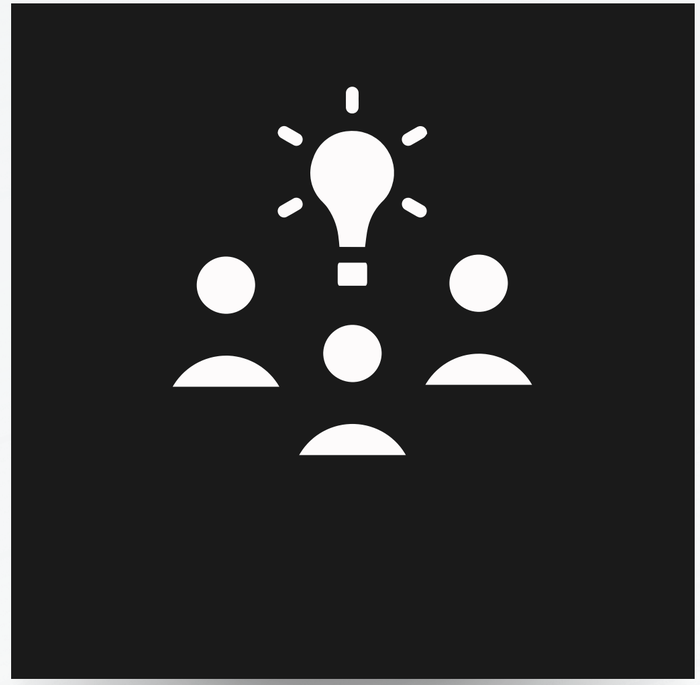
EventStream

WHY THIS PROBLEM?

- With de-centralization being the current trend of technology, solving this problem means contributing to the efficiency and capability of systems we use daily.
- Achieving sub-linear space complexity while maintaining accuracy is a significant technical feat.
- It’s the perfect challenge to push our boundaries and test our technical acumen.

EventStream

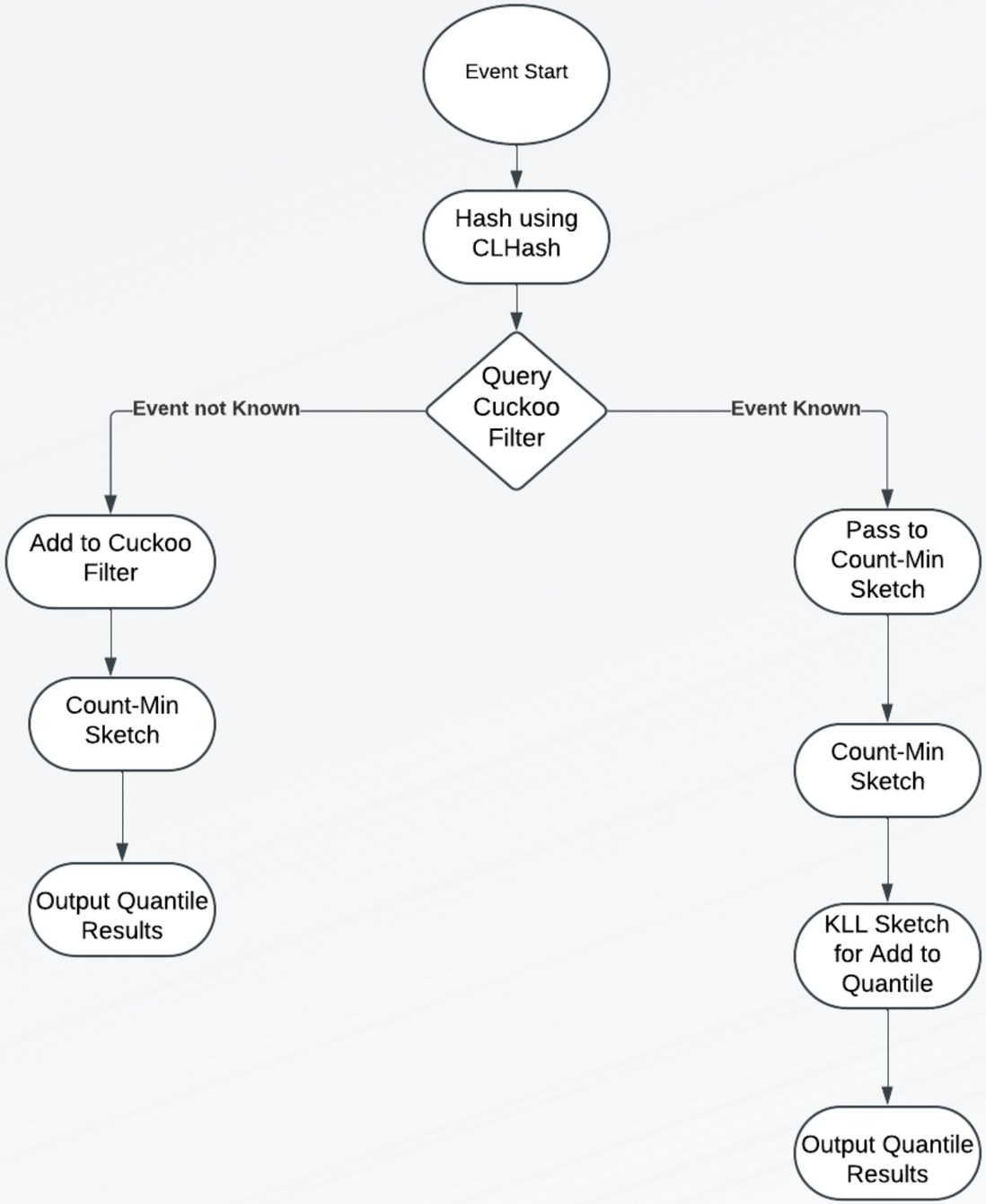
SOLUTION OVERVIEW



- To solve the problem of tracking millions of events in real-time with minimal memory, we propose a system combining a Cuckoo Filter, a Count-Min Sketch, and robust hashing techniques.
- The Cuckoo Filter ensures unique event identification with minimal collisions, while the Count-Min Sketch efficiently tracks frequencies in sub-linear space.
- Robust hash functions like CLHASH/CityHash ensure uniform data distribution, enhancing the accuracy and efficiency of these components.
- Together, they provide a scalable, memory-efficient solution that supports real-time operations and distributed environments.

VISUAL WORKFLOW

EventStream



TECH-STACK

PYTHON

CUCKOO-FILTER
1.0.6

PY-BLOOM

PYHASH

IMPLEMENTATION PLAN

EventStream

Phase	Hours Spent	Key Deliverables
Ideation and Setup	0-6	Architecture, workflow diagram, tools ready.
Basic Implementation	6-30	Fully functional components.
Integration	30-48	Integrated pipeline with basic functionality.
Optimization	48-60	Memory-efficient system, percentile queries.
Testing and Debugging	60-68	Reliable prototype, performance benchmarks.
Presentation Prep	68-72	Demo, pitch, and visuals.

EventStream

What do you aim to achieve with this solution?

We aim to create an efficient way to track real-time events with minimal memory usage while providing basic support for percentile queries. Using probabilistic data structures like the Cuckoo Filter and Count-Min Sketch, the solution will offer a scalable, lightweight approach to handle large volumes of data without overwhelming system resources.

Highlight potential benefits, impact, or results of your project.

- Efficient Memory Usage: The solution keeps memory usage low, making it ideal for environments with limited resources.
- Scalability: It can scale across multiple systems if needed, handling larger datasets over time.
- Real-Time Processing: It provides quick, approximate tracking of events, making it useful for real-time applications where exact accuracy isn’t always required.
- Cost-Effective: The design minimizes memory and storage needs, offering a more affordable way to track large amounts of data.
- Practical Applications: This approach is suitable for things like monitoring systems or basic analytics, where handling a lot of events efficiently is key, even if some accuracy is sacrificed.

EventStream

CHALLENGES

- Ensuring compatibility between Cuckoo Filter and Count-Min Sketch operations.
- Balancing memory usage to accommodate both structures effectively.
- Identifying the use cases where both structures are needed together.
- Understanding trade-offs between memory, accuracy, and speed.

EventStream

WHY THESE TECH?

CLHash

CLHash is a fast, cache-friendly hash function that ensures uniform distribution of events across data structures. It minimizes hash collisions, improving performance for Cuckoo Filters and Count-Min Sketch

Cuckoo Filter

Cuckoo Filter is memory-efficient and supports deletions while maintaining low false-positive rates, making it ideal for set membership checks in real-time systems.

Count-Min Sketch

Count-Min Sketch is highly space-efficient for frequency estimation. It works well in streaming scenarios by providing approximate counts with configurable accuracy.

KLL Sketch

KLL Sketch is specifically designed for efficient quantile estimation in streaming data. It adapts dynamically to data distribution, offering accurate quantiles with low memory overhead.

PROOF OF FEASIBILITY OF THE CHOSEN TECHNOLOGIES

EventStream

Count-Min Sketch

- Proven Use: Frequently used in network traffic monitoring, search engine queries, and data stream mining.
- Performance: Provides frequency estimates in constant time (O(1)) with sublinear memory requirements.
- Accuracy: Balances precision and memory through tunable parameters.

Cuckoo Filter

- Proven Use: Widely adopted for set membership queries in distributed systems, databases, and web caching.
- Performance: Achieves low false-positive rates (~3%) with compact memory.
- Efficiency: Supports fast insertions, deletions, and lookups in constant time (O(1)).

KLL Sketch (Quantile Estimation)

- Proven Use: A state-of-the-art algorithm for quantile approximation in streaming data.
- Performance: Adapts dynamically to data distribution with logarithmic memory overhead.
- Accuracy: Provides precise quantile results with high confidence levels.

EventStream

THANK YOU!