CS 591 K1:
Data Stream Processing and Analytics

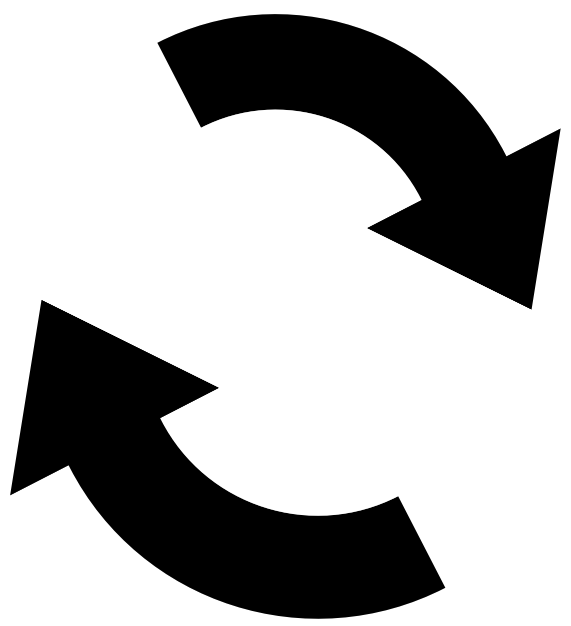
Spring 2021

Vasiliki (Vasia) Kalavri
vkalavri@bu.edu

What is this course about?

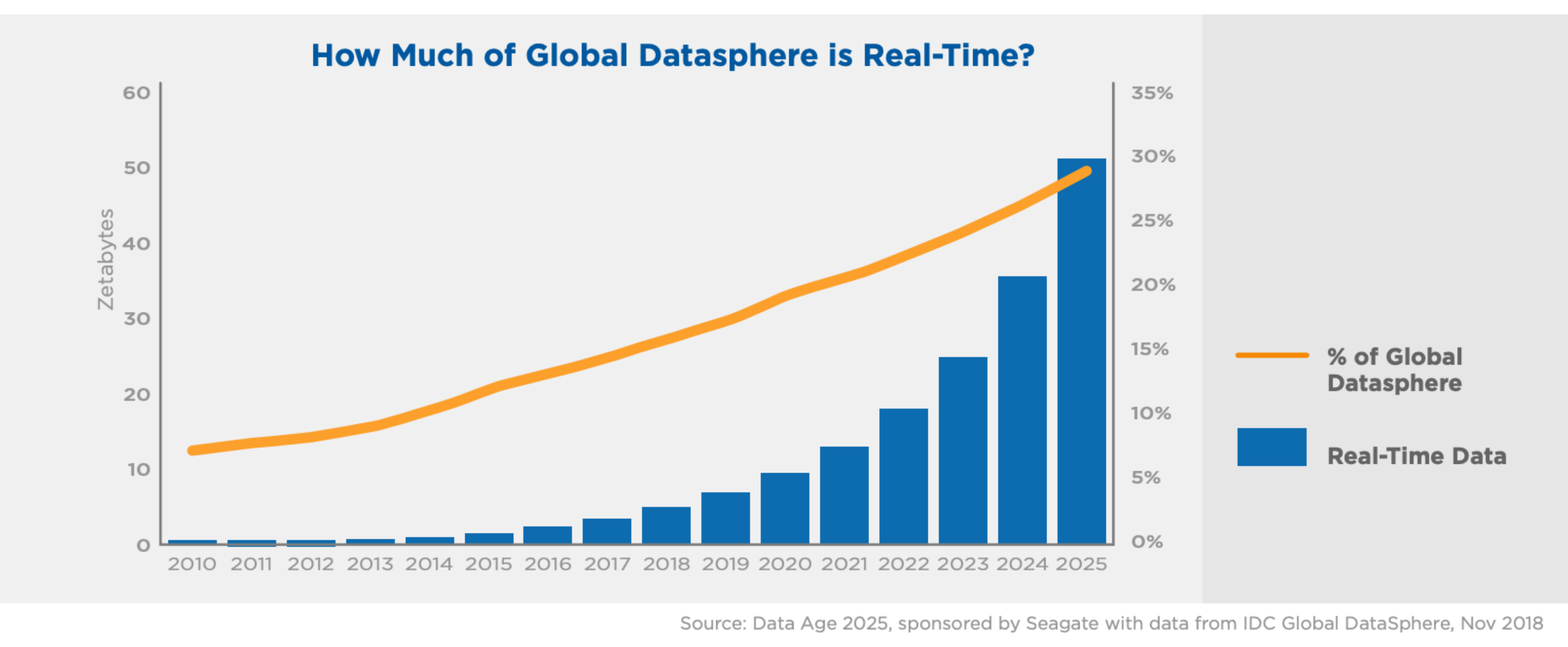
The **design**
and **architecture** of modern
distributed streaming Systems

Architecture and design
Scheduling and load management
Scalability and elasticity
Fault-tolerance and guarantees
State management



Operator semantics
Window optimizations
Filtering, counting, sampling
Graph streaming algorithms

Fundamental Algorithms
for **representing**, **summarizing**,
and **analyzing** data streams



By **2025**,

30% of all data will be **real-time** data

we will be able to store less than **15%** of all data

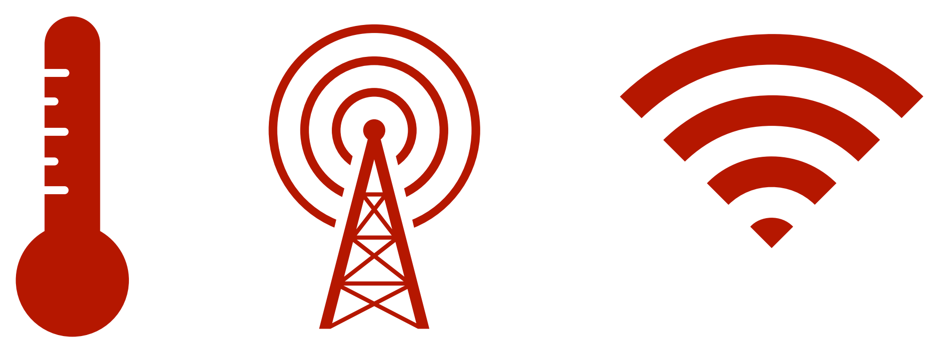
A data stream is a data set that is produced incrementally over time, rather than being available in full before its processing begins.

- Data streams have **unknown**, possibly **unbounded length**
- They **arrive continuously** instead of being available a-priori
- They bear an arrival and/or a generation **timestamp**
- They are produced by external sources, i.e. the system has **no control** over their **arrival order** or the **data rate**

Can you give me some examples of streaming data sources?

Sensor measurements analysis

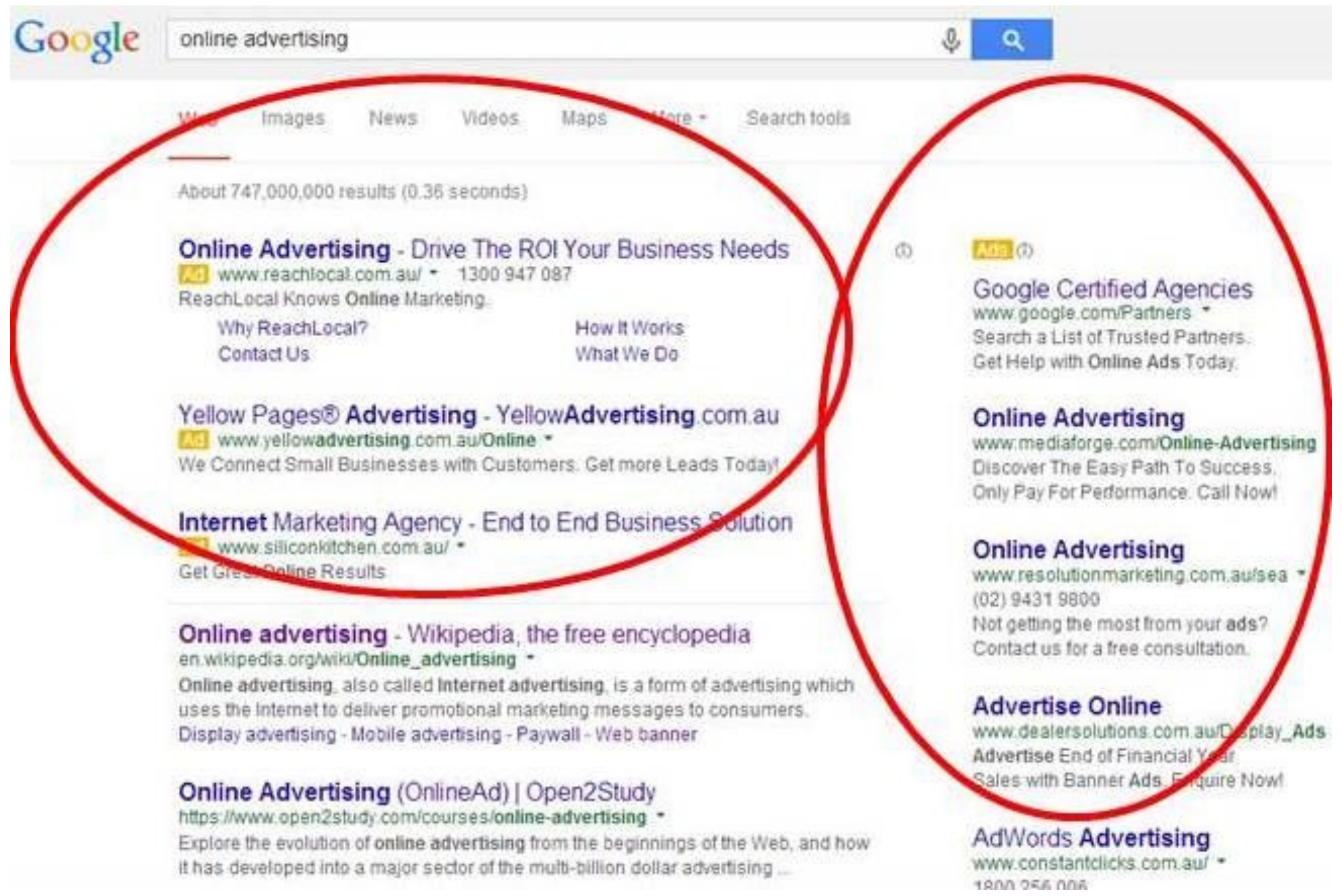
- Monitoring applications
- Complex filtering and alarm activation
- Aggregation of multiple sensors and joins



- **Examples**
  - Real-time statistics, e.g. weather maps
  - Monitor conditions to adjust resources, e.g. power generation
  - Monitor energy consumption for billing purposes

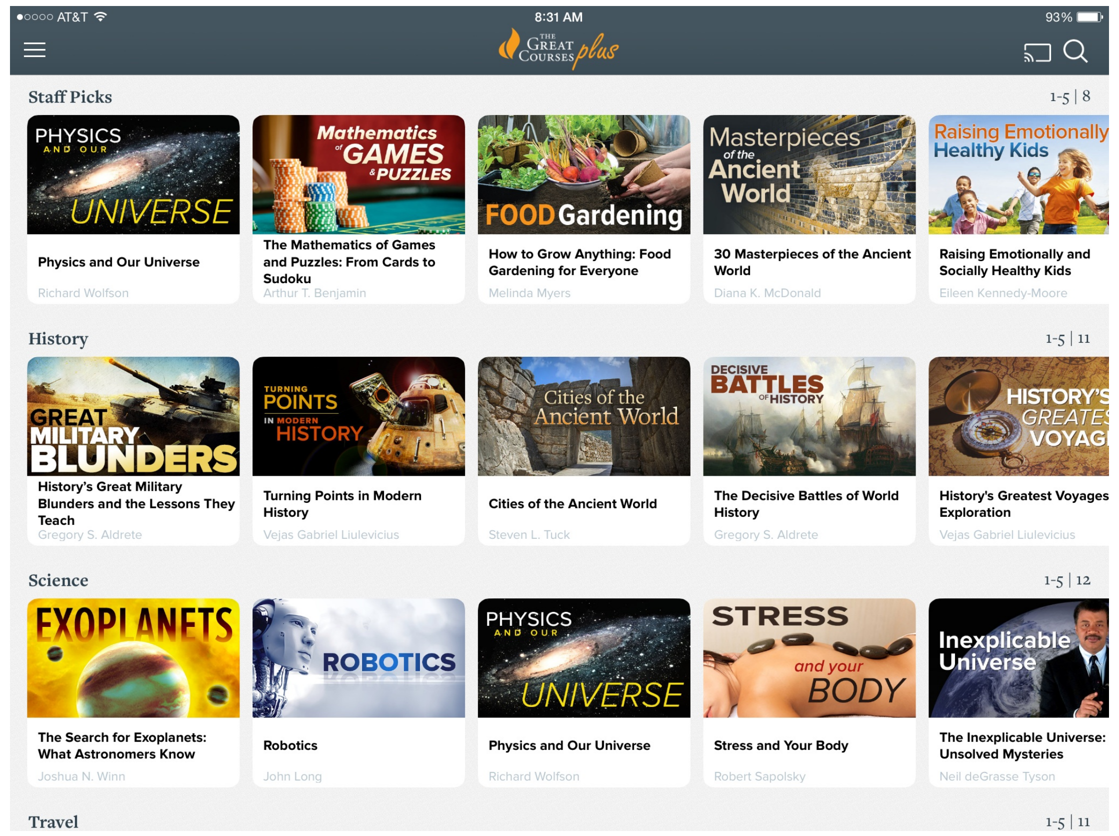
Web activity analysis

- Visualization and aggregation
  - impressions, clicks, transactions, likes, comments
- Analytics on user activity
  - Filtering, aggregation, joins with static data (e.g. user profile data)



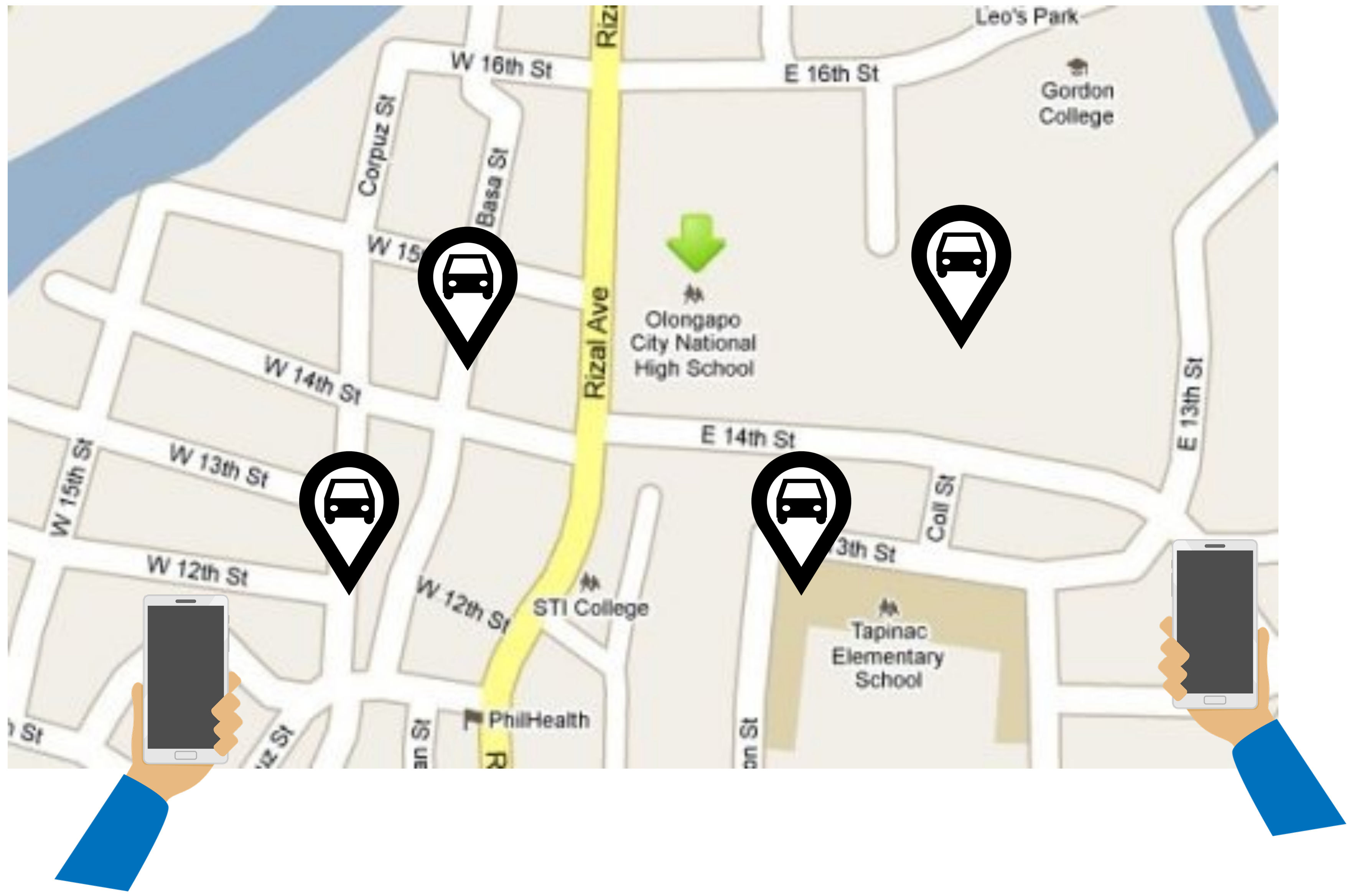
Examples

- online A/B testing
- trending topics
- sentiment analysis, e.g., reaction to just published campaign
- online recommendations of products, articles, people



Online traffic management

- Analysis of real-time vehicle locations to improve traffic conditions
- Provide real-time scheduling information for public transport
- Optimize transport network flow and recommend alternative routes



Stock trading

- Discover correlations, identify trends, forecast future values

Examples

- Find all stocks priced between $20 and $200, where the spread between the high tick and the low tick over the past 30 minutes is greater than 3% of the last price, and where in the last 5 minutes the average volume has surged by more than 300%.
- Find all stocks trading above their 200-day moving average with a market cap greater than $5 Billion that have gained in price today by at least 2%, and are within 2% of today's high.

Financial transaction analysis

- Fraud detection, online risk calculation

Example: Someone steals your phone and sings in your banking app. The app allows transfers of up to €1000 and so the thief makes transfers of €1000 to a "fake account" until either you're out of money or the activity is detected.

- Features to detect fraudulent activity like this:
  - The transaction amount.
  - The number of recent (e.g. the last hour) transactions.
  - Whether money was sent to this recipient account for the first time in the past 24 hours (in other words, to an “unknown” recipient account).

Read more: https://www.ververica.com/blog/real-time-fraud-detection-ing-bank-apache-flink

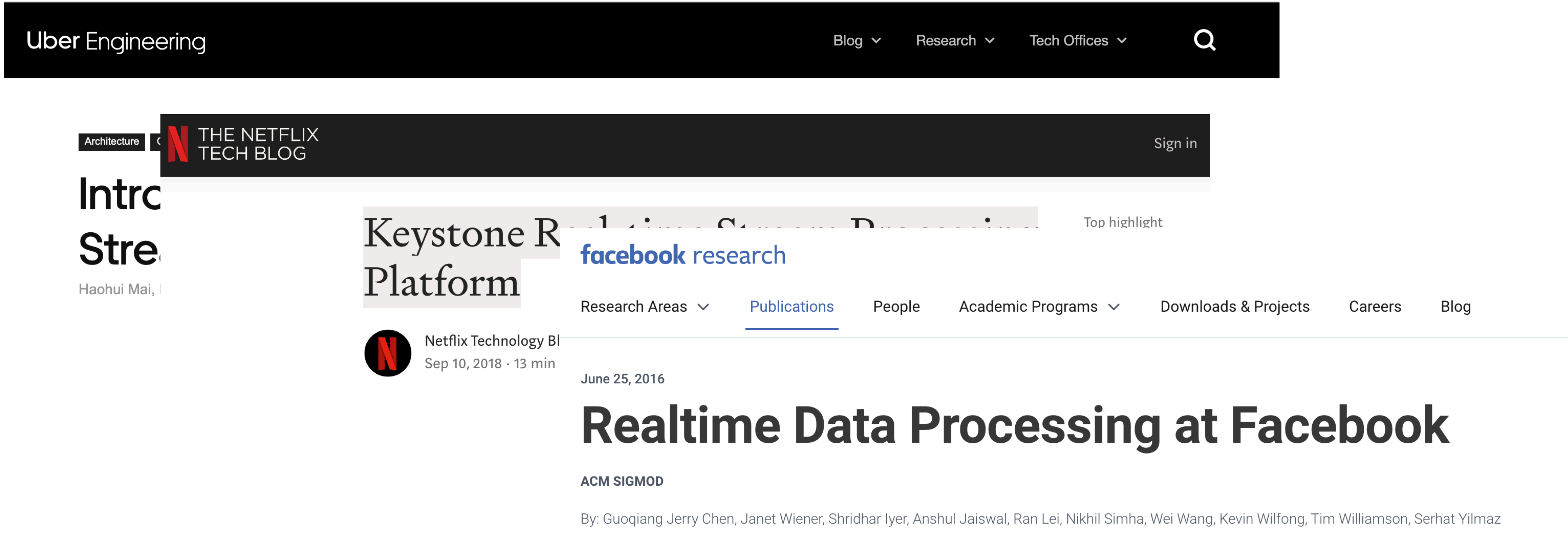
Call monitoring

- Service monitoring, e.g. source and destination phone numbers, their first and last cell towers

Examples:

- Location-based services
- Monitor cell tower load
- Continuously maintain call signatures for fraud detection
    - call frequency
    - top-K cell towers used

Stream processing is an *established* technology in the data analytics stack of the modern business



aws

Amazon Kinesis Data Analytics

Get actionable insights from streaming data in real-time

Get started with Amazon Kinesis Data Analytics

Google Cloud

Dataflow

Dataflow is a fully managed streaming analytics service that minimizes latency, processing time, and cost through autoscaling and batch processing. With its serverless approach to resource provisioning and management, you have access to virtually limitless capacity to solve your biggest data processing challenges, while paying only for what you use.

Alibaba Cloud | Worldwide Cloud Services Partner

Alibaba Cloud > Products > Realtime Compute（StreamCompute）

Realtime Compute

Realtime Compute offers a highly integrated platform for real-time data processing, which optimizes the computing of Apache Flink. With Realtime Compute, we are striving to deliver new solutions to help you upgrade your big data capabilities in your digital transformations.

IBM

IBM Streaming Analytics for IBM Cloud

Leverage continuously available data from all sources to discover opportunities faster

Microsoft Azure

Azure Stream Analytics

Serverless real-time analytics, from the cloud to the edge

Lightbend

cloudflow

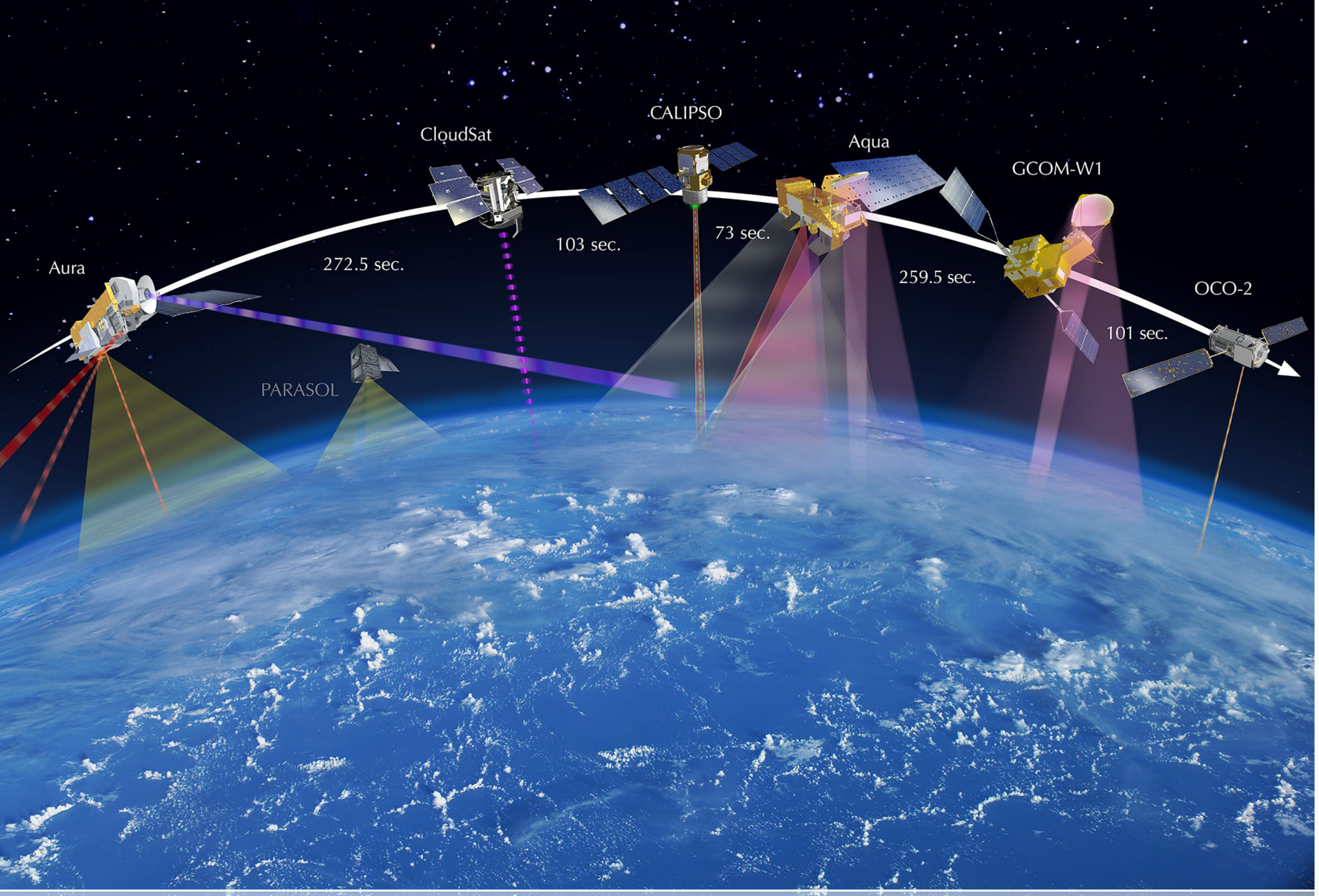
Compose streaming data apps faster than ever before

Alibaba City Brain analyzes vehicle locations to:

- clear paths for emergency response vehicles
- provide scheduling information for public transport
- recommend alternative routes

Traffic light adjustment in real time

Read more: https://edition.cnn.com/2019/01/15/tech/alibaba-city-brain-hangzhou/index.html



Fault-detection for NASA's Deep Space Network

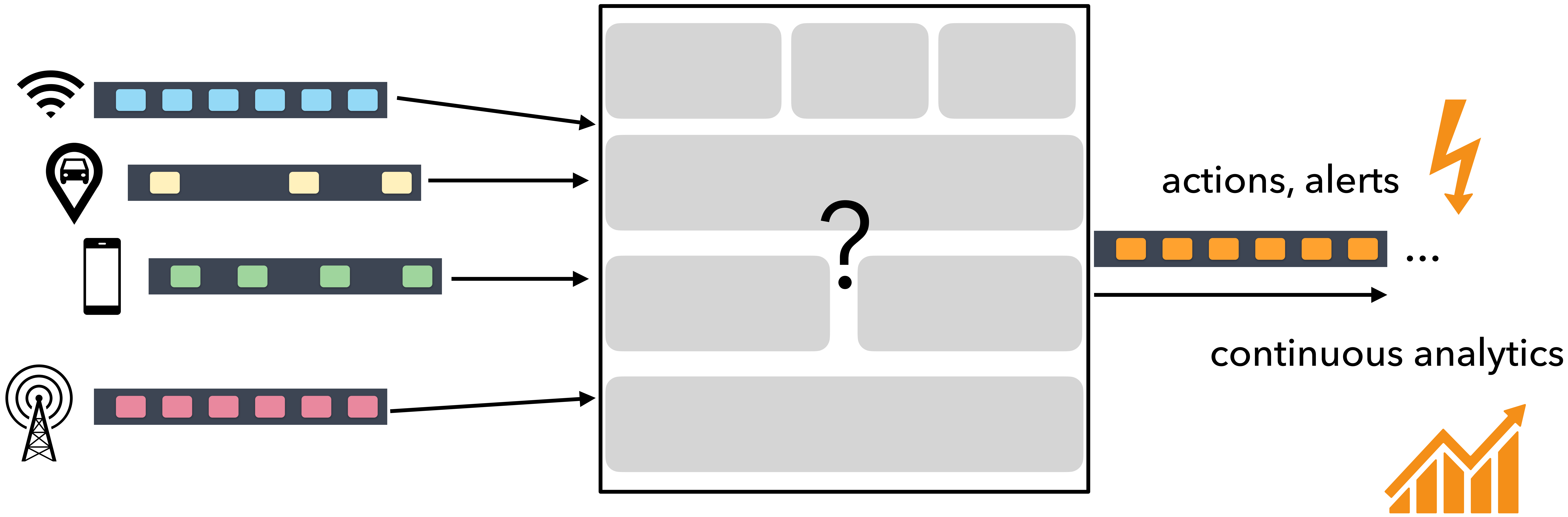
Read more: https://www.confluent.io/kafka-summit-san-francisco-2019/mission-critical-real-time-fault-detection-for-nasas-deep-space-network-using-apache-kafka/

NASA's DSN Complex Event Processing analyzes real-time network data, predicted antenna pointing parameters, and physical hardware logs to:

- ingest, filter, store, and visualize all of the DSN's monitor and control data
- ensure the successful DSN tracking, ranging, and communication integrity of dozens of concurrent deep-space missions

What are the main challenges in stream processing?

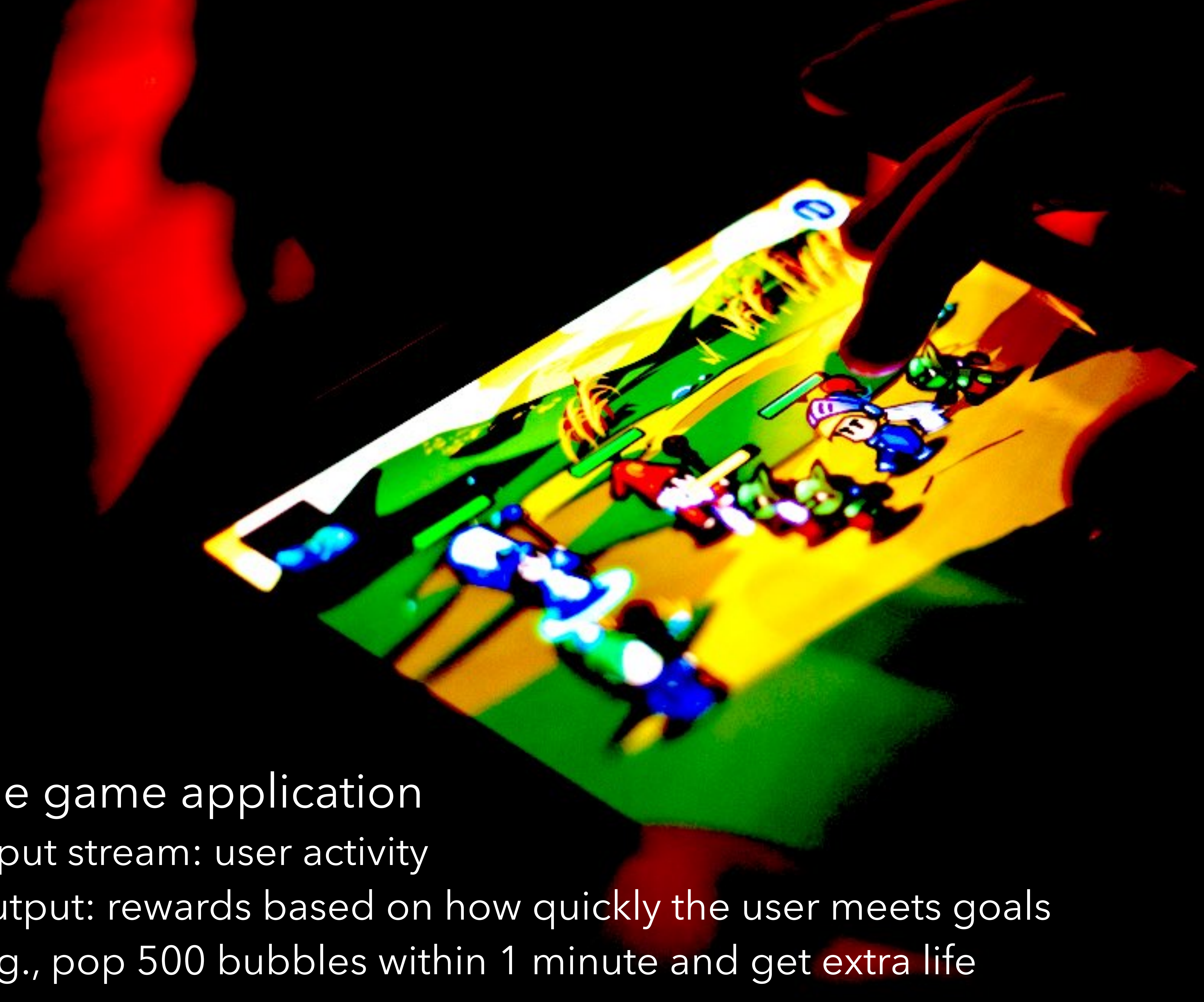
Building a stream processor…



#1 Time

Consider a set of 1000 sensors deployed in different locations inside a forest. The sensors monitor temperature and smoke levels and generate a measurement every 5 seconds.

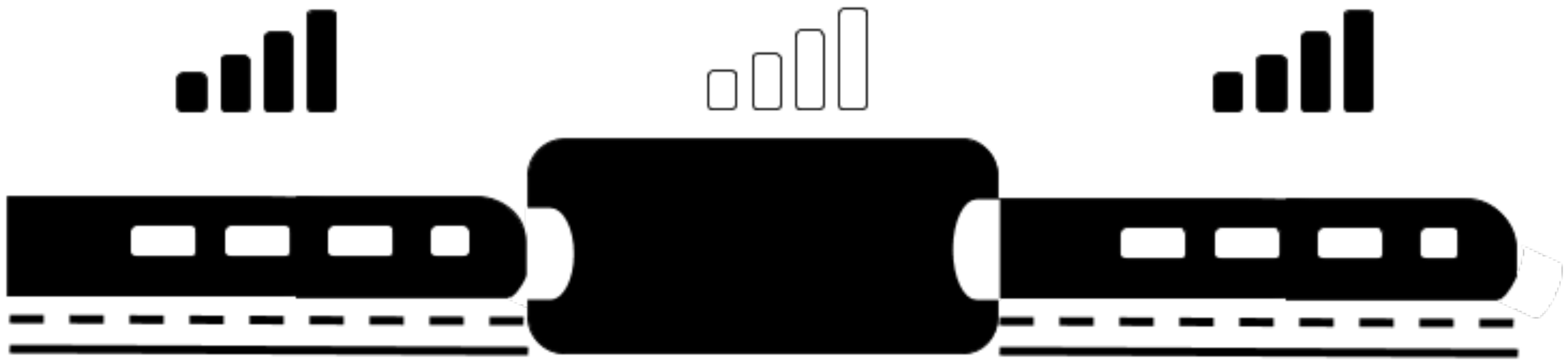
Write a program that every 1 minute emits the average temperature over the last 10 minutes.



Mobile game application

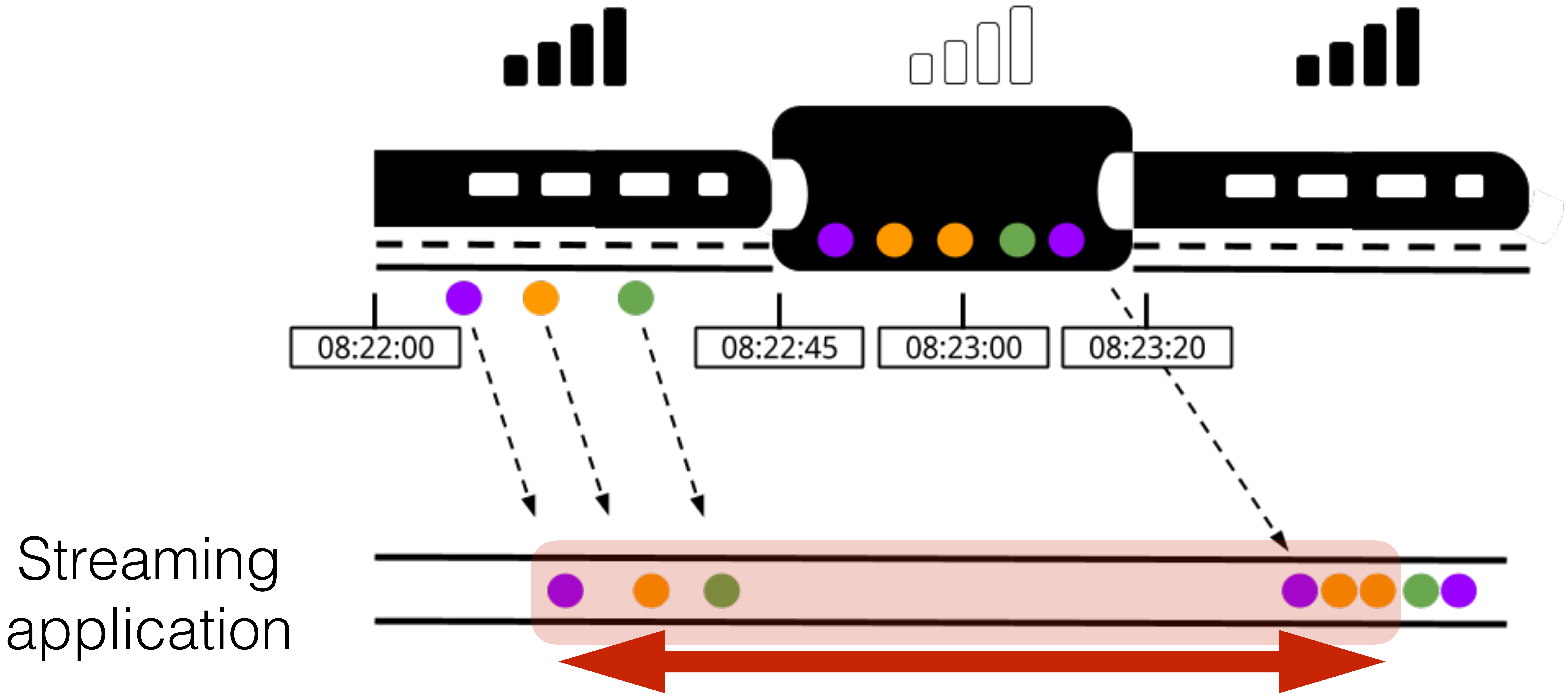
- input stream: user activity
- output: rewards based on how quickly the user meets goals
- e.g., pop 500 bubbles within 1 minute and get extra life

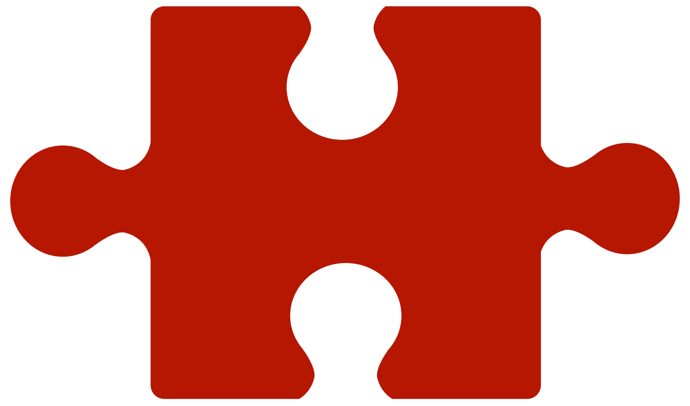
What’s the meaning of one minute?



Streaming
application

What’s the meaning of one minute?



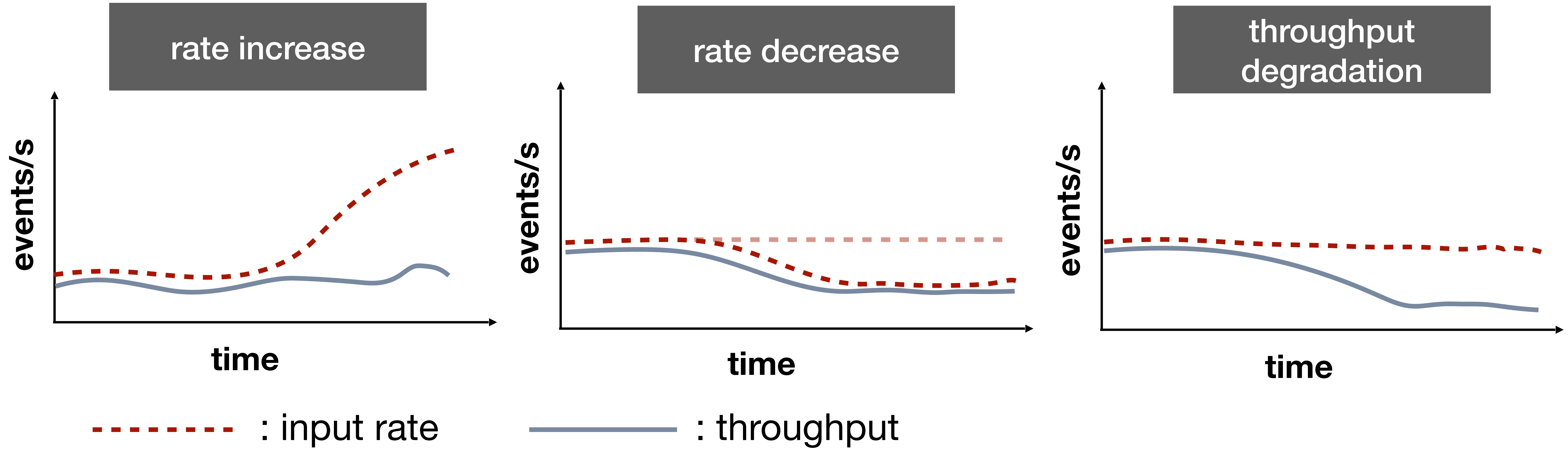


- **What if you were in a plane and not on a train?**
- **What if you never came back online?**
- **How long do we have to wait before we decide that we have seen all events?**

#2 Workload variation

Streaming applications are **long-running**

- Workload will change
- Conditions might change
- State is accumulated over time



Optimization is a continuous process

- Change parallelism
  - scale out to process increased load
  - scale in to save resources
- Fix bugs or change business logic
- Optimize execution plan
- Change operator placement
  - skew and straggler mitigation
- Migrate to a different cluster or software version

online monitoring

low-latency reactions

Control: When and how much to adapt?

- Detect environment changes: external workload and system performance
- Identify bottleneck operators, straggler workers, skew
- Enumerate scaling actions, predict their effects, and decide which and when to apply

Mechanism: How to apply the re-configuration?

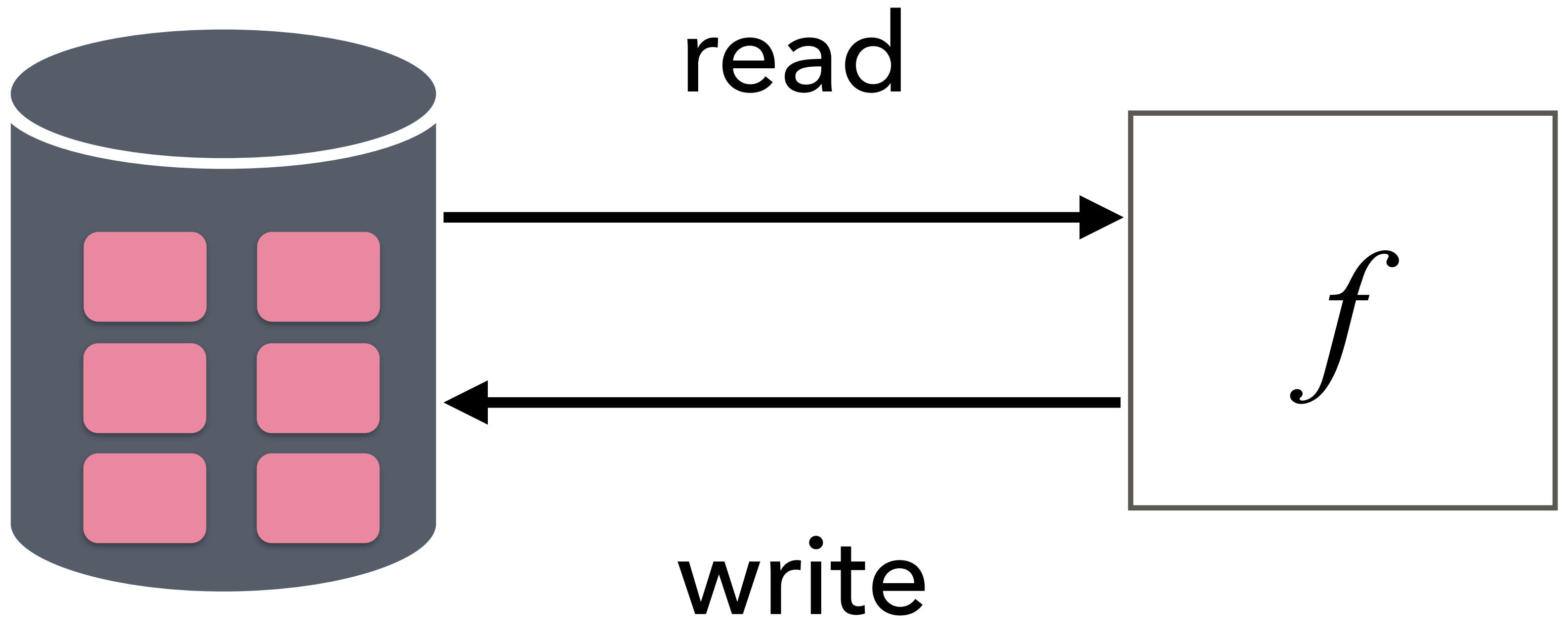
- Allocate new resources, spawn new processes or release unused resources, safely terminate processes
- Adjust dataflow channels and network connections
- Re-partition and migrate state in a consistent manner
- Block and unblock computations to ensure result correctness

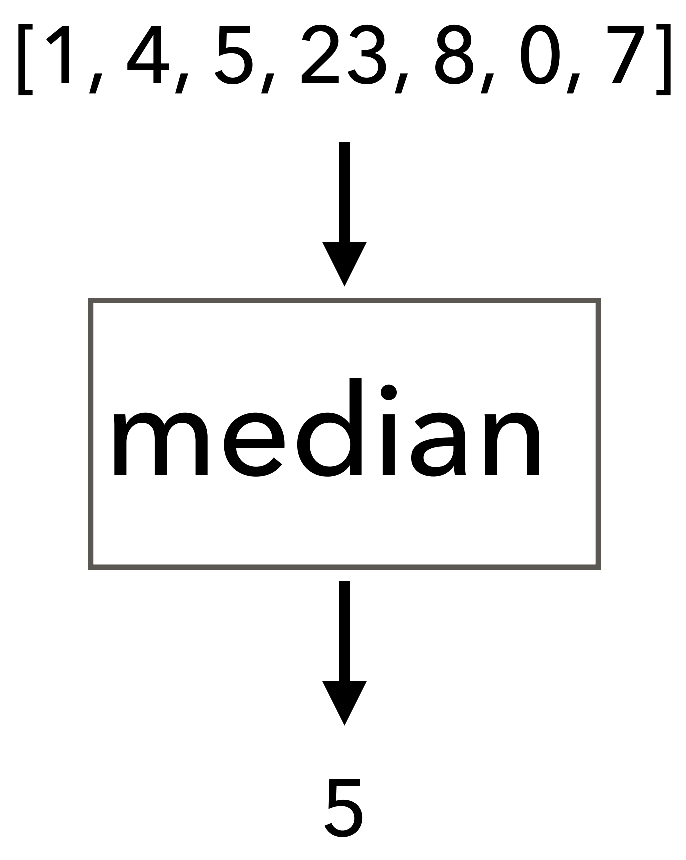
#3 State

Using pseudocode (or the programming language of your choice), write a program that reads a stream of integers and computes:

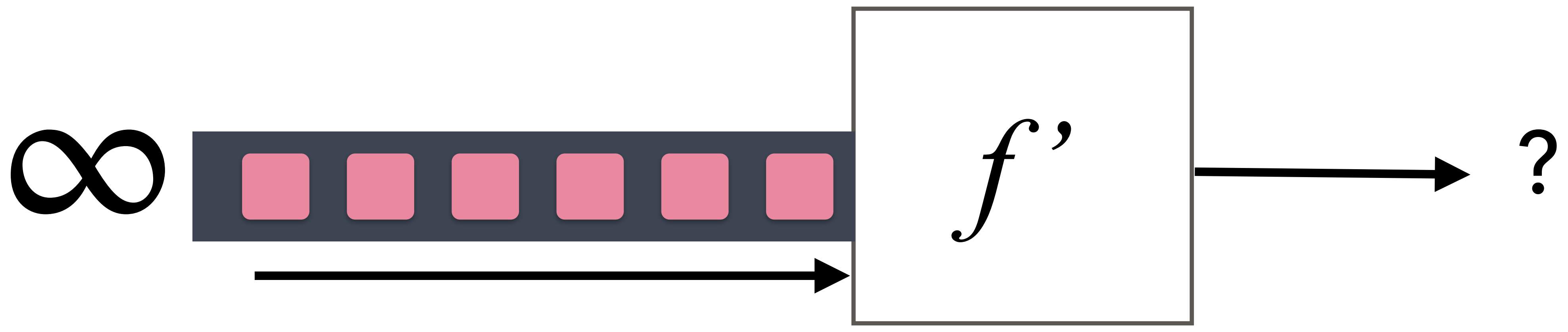
1. the **maximum** number seen so far
2. the **average** of all numbers seen so far
3. the **median** of all numbers seen so far

Complete data accessible in persistent storage



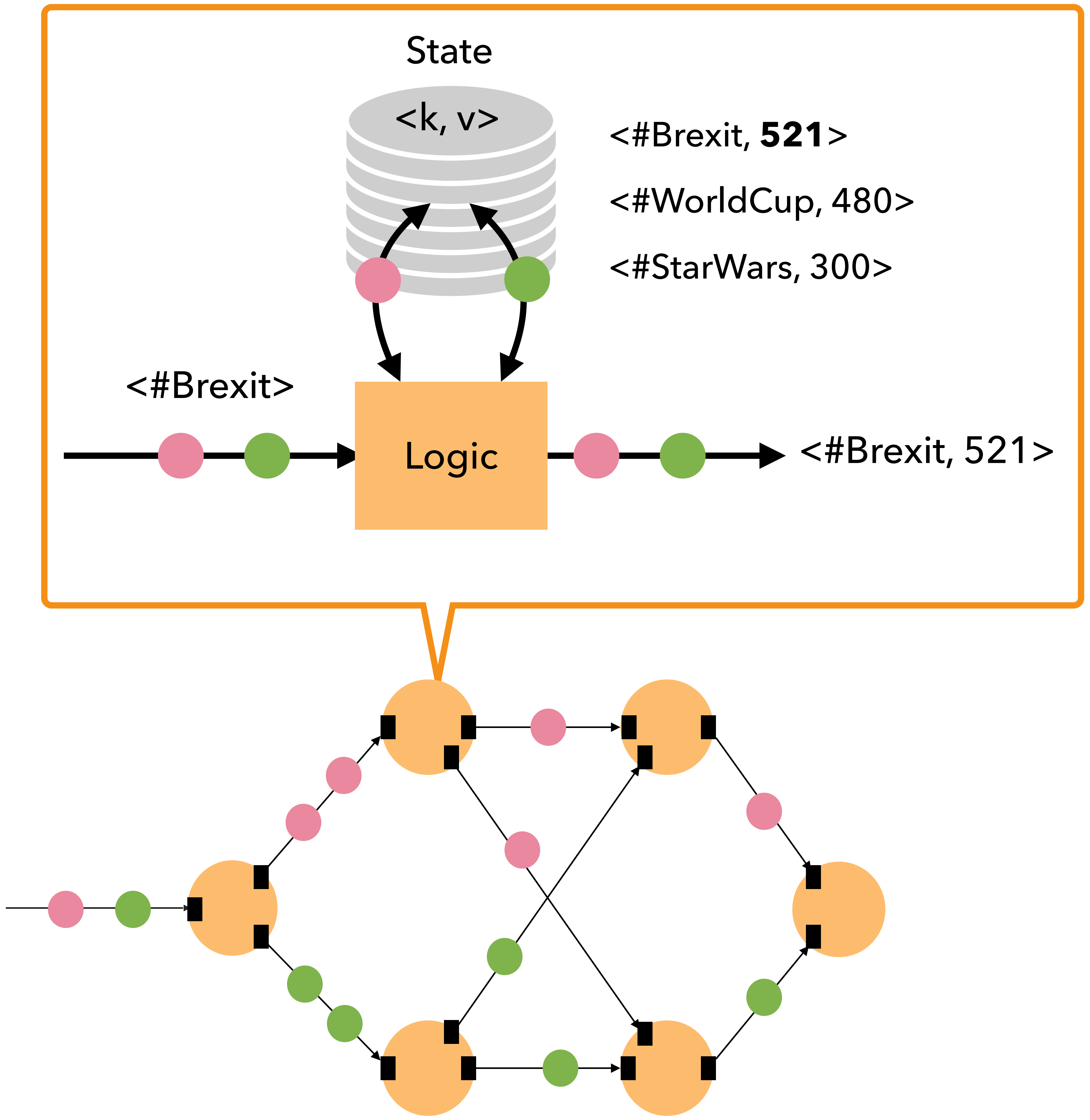


Continuously arriving, possibly unbounded data



- We cannot store the entire stream
- No control over arrival rate or order

Streaming operators accumulate **state**



- Stateful operators **maintain state** that reflect part of the stream history they have seen
  - windows, continuous aggregations…

https://www.ververica.com/blog/blink-flink-alibaba-search

Alibaba Group

Blink is a forked version of Flink that we have been maintaining to fit some of the unique requirements we have at Alibaba. At this point, Blink is running on a few different clusters, and each cluster has about 1000 machines, so large-scale performance is very important to us.

Distributed streaming systems *will* fail

- how can we guard state against failures and guarantee correct results after recovery?
- how can we ensure minimal downtime and fast recovery?
- how can we hide recovery side-effects from downstream applications?

Logistics

Course Objectives

At the end of the course, you will hopefully:

- know **when** to use stream processing vs other technology
- be able to comprehensively **compare features** and **processing guarantees** of streaming systems
- be proficient in using Apache Flink and Kafka to build **end-to-end**, **scalable**, and **reliable** streaming **applications**
- have a solid **understanding** of how stream processing systems work and what factors affect their **performance**
- be aware of the **challenges** and **trade-offs** one needs to consider when **designing** and **deploying** streaming applications

Semester Project

- In teams of 3-4 students
- Research-focused
- Deliverables
  - Design document
  - Midterm demo
  - Final demo & poster
  - Gitlab repository
- Send me you **top-3** preferences
  - by **Feb 1st** 11:59pm EST
  - Piazza private message and include your timezone!

Looking for something more related to your research? Let's discuss during OH.

Alternative: The DEBS Grand Challenge

https://2021.debs.org/call-for-grand-challenge-solutions/

Important Dates

Release of the challenge, initial data set, and a reference implementation **December 15th, 2020**

API Endpoint for development **February 15th, 2021**

Announcement of challenge parameters (batchsize, latency/throughput) **February 15th, 2021**

Evaluation platform (VMs) **February 15th, 2021**

Deadline for uploading the final solution to the evaluation platform **April 5th, 2021**

Deadline for short paper submission **April 19th, 2021**

Notification of acceptance **May 3rd, 2021**

Paper review & presentation

- Each team will be assigned one paper to review (individually) and present (as a team).
- See https://vasia.github.io/dspa21/readings.html for the list of papers.

Readings

- The 8 Requirements of Real-Time Stream Processing
  http://cs.brown.edu/~ugur/8rulesSigRec.pdf

- Streaming 101: The world beyond batch
  www.oreilly.com/ideas/the-world-beyond-batch-streaming-101