Emerging Cloud Applications

Introduction

1. Study streaming applications with structured and amorphous data, as well as Internet of things

Streaming Applications with Structured Data

1. Stream processing fundamentals
    - Raw stream data (e.g., stock prices, tweets, facebook posts, news feeds, etc.) push to the cloud
    - Application in the cloud organizes the raw data into meaningful events
    - Application entertains continuous queries from users (e.g., sending stock ticker, twitter feeds, facebook feeds, etc.)
2. Stream processing of structured data
    - Aurora and Medusa (MIT, Brown, Brandeis)
        - Stuctured data streams as input data
        - Queries built from well-defined operators
            * Select, aggregate
            * Operator accepts input streams and produces output streams
        - Stream processor
            * Schedules operators respecting QoS constraints
    - Medusa is a federated version of Aurora
    - TelegraphCQ (Berkeley)
        - Similar in goals and principles to Aurora
3. Concrete example: Tweets
    - Real-time streaming
    - Input: Raw tweets
    - Output: Tweet feeds to users
    - Scale with number of tweets and number of followers
    - Compute real time active user counts (RTAC)
        - Measure real-time engagement of users to tweets and ads
    - "Computation on the feeds"
        - Data transformation, filtering, joining, aggregation across twitter streams
        - Machine learning algorithms over streaming data
            * Regression, association, clustering
        - Goals
            * Offer user services, revenue, growth, search, content discovery
4. Heron architecture for Twitter
    - Topology: Application graph (DAG)
        - Vertices: computation
        - Edges: Data tuples streams
        - Typically 3-8 stages of processing steps
    - Architecture
        - Central scheduler accepts topologies
        - Launches topologies on to Zookeeper cluster resources using Mesos
            * Each topology runs as a Mesos job with multiple containers
            * Master, stream manager, metrics manager, Heron instances (app logic)

Fault Tolerance for Streaming Applications

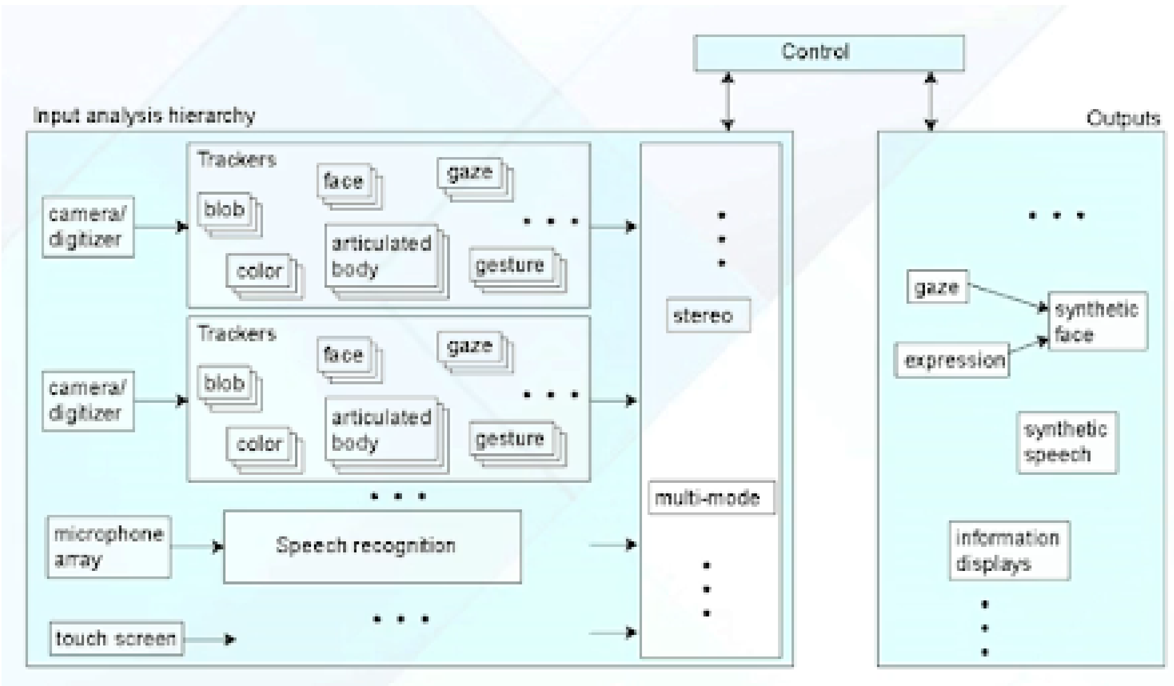
1. General processing requirements for streaming apps
    - Real time processing
    - Persistent state
    - Out of order data
    - Constant latency as you scale up

- Exactly once semantics
- Spam filtering
- Intrusion detection
- Ad customization
- Question: Instead of inventing for every app, can we have a generic framework for many if not ALL apps?

2. Millwheel
   - Applications considered as a directed graph
     - User defined topology for connecting the nodes
     - Nodes are computation
     - Edges are "intermediate results" called records delivered from one computational component of the app to the next
   - Focus of MillWheel is fault tolerance at the framework level
     - Any node can fail without affecting the correctness of the data
     - Guaranteed delivery of records
     - API guarantees idempotent handling of record processing
       * App can assume that a record is delivered "exactly once"
     - Automatic checkpointing at fine granularity
   - MillWheel (just like MapReduce for batch jobs) handles all the heavy lifting for streaming jobs
     - Scalable execution of the app with fault tolerance
     - Dynamically grow and shrink the graph

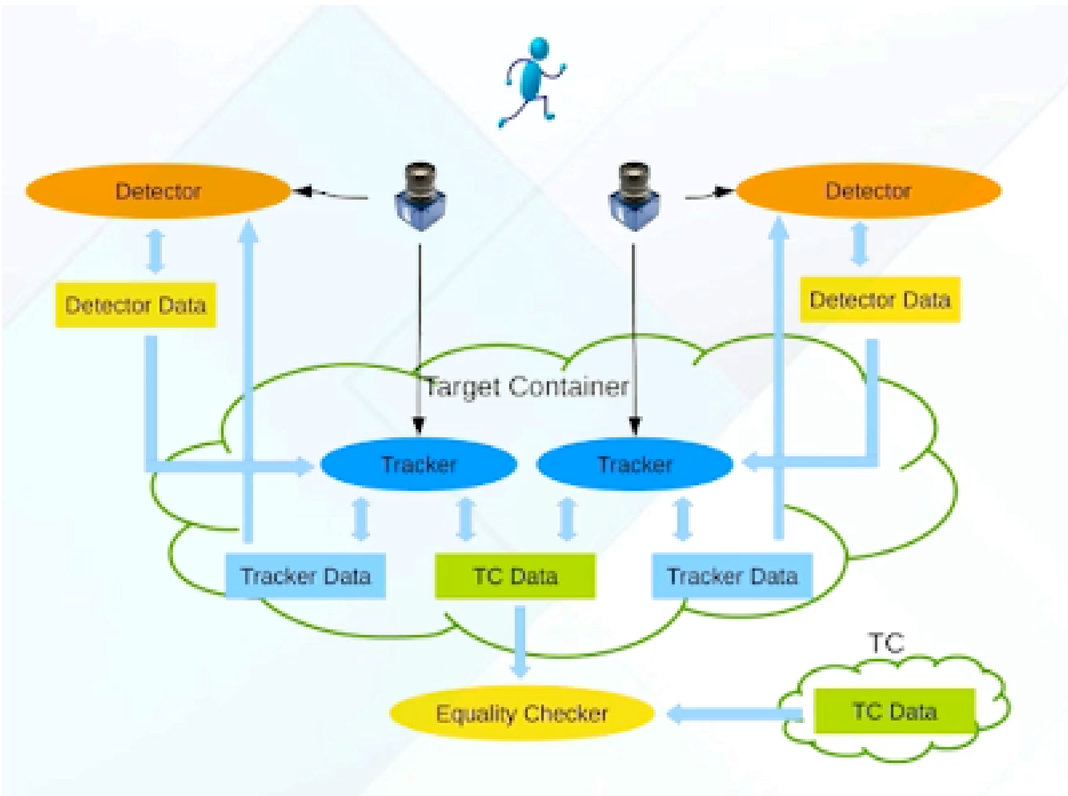
Streaming Applications with Amorphous Data

1. Stream-based programming models
   - Amorphous data: Data is unstructured, such as with cameras
     - Know it's an array of pixels but have no idea about its contents
   - IBM System S, Slipstream, Stampede, TargetContainer
2. Stampede
   - Channels are entities that sit between computation steps
   - Space Time Memory: Channels are the space axis, have frames as a function of time
     - Immutable data structure (only append to channels)



Stampede Architecture

3. TargetContainer
   - Object being tracked isn't always stationary, so we need to correlate targets among cameras
     - This makes the problem a distributed systems problem instead of just a computer vision problem
   - Domain expert provides trackers and equality checkers so the framework can correlate blobs between cameras and frames



TargetContainer Architecture

Internet of Things and Situation Awareness

1. Rise of Internet of Things
    - IoT: Many devices collecting and communicating data to each other
    - Situation awareness applications
        - Predictive maintenance
        - Enable new knowledge
        - Agriculture
        - Smart grid
        - Energy saving
        - Transportation and connected vehicles
        - Intelligent buildings
        - Healthcare
        - Defense
        - Industrial automation
        - Smart home
        - Enhance safety and security
    - Argonne National Lab is developing the "array of things," a suite of cameras and sensors that can be deployed on light poles
        - Deployed in Chicago and Atlanta
2. Future Internet Applications on IoT
    - Common characteristics
        - Dealing with real-world data streams
        - Real-time interaction among mobile devices
        - Wide-area analytics
    - Requirements

    - Dynamic scalability
    - Low-latency communication
    - Efficient in-network processing
3. Concluding thoughts
    - Emerging applications pose new challenges
    - Streaming apps on the cloud thus far
        - Human in the loop
        - Latency at "human perception" speeds
    - Future
        - Machine to machine communication
            * Connected cars
        - Latency at "machine perception" speeds
    - Utility computing today is performed on the cloud
        - Good for handling streaming, but bad for computation
        - A self-driving car needs to operate with very low latency, so it can't communicate with the cloud for every frame of data

Conclusion

1. Cloud computing started out as offering computational services at scale for throughput-oriented applications
2. Streaming applications with structured data are the next wave of apps
    - Twitter, Facebook, Netflix
3. IoT demands stringent latency requirements and geo-distributed processing that cloud computing is unable to accomodate