15-319 / 15-619 Cloud Computing

Recitation 15

May 1st 2018

Overview

- **Last week's reflection**
  - Team Project Phase 3
- **This week's schedule**
  - Phase 3 report
    - Due **TODAY** May 1, 23:59:59 ET
  - Project 4.3
    - Due **FRIDAY** May 4, 23:59:59 ET
  - Course survey (2% bonus!)
    - Deadline **Saturday** May 5, 23:59:59 ET

Project 4

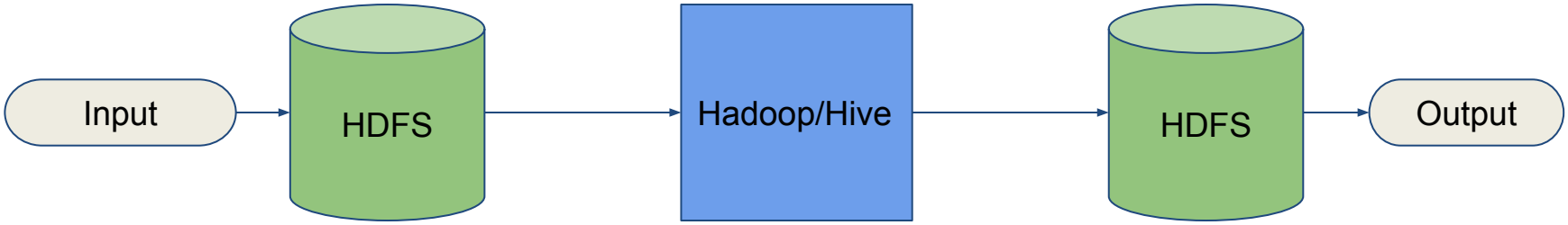
- Project 4.1
  - Batch Processing with MapReduce
- Project 4.2
  - Iterative Processing with Apache Spark
- Project 4.3
  - Stream Processing with Kafka & Samza

Stream vs Batch Processing

Batch Processing		Stream Processing
Iterative or non-iterative Data or graph-parallel	Nature	Event-parallel
Run once every few hours or days	Frequency	Process events in real-time
Historical data analysis	Use Case	IOT sensor data, web event data

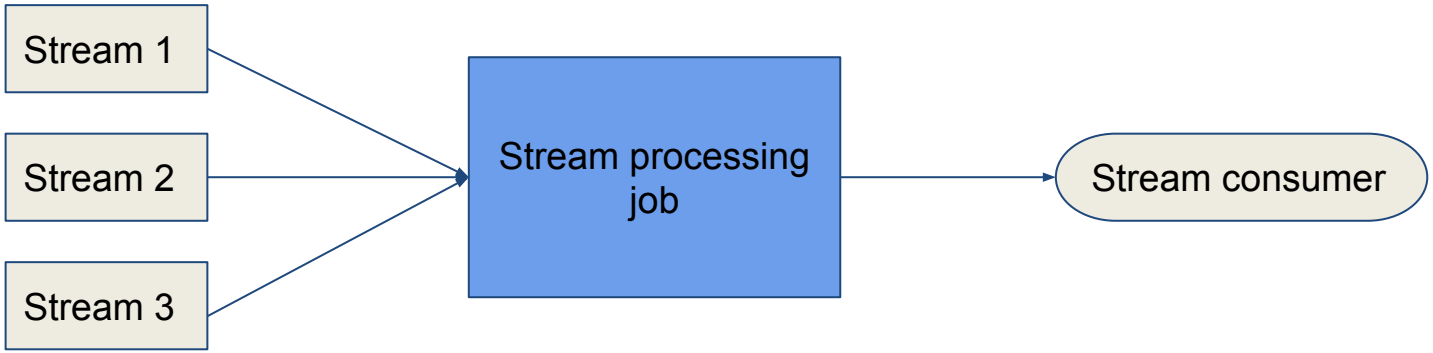
Typical Batch Processing Job

- Input is collected into batches and processing is performed on the input data
- Output is consumed later at any point of time - the data does not lose much of its “value” with time



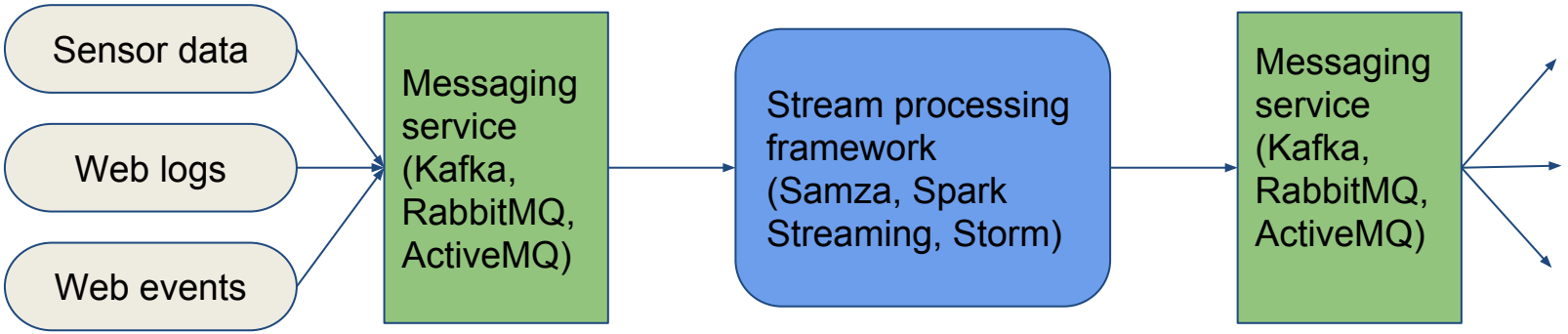
Typical Stream Processing Job

- Data is processed immediately (few seconds)
- Processed data is used by downstream consumers for real time decision/analytics immediately



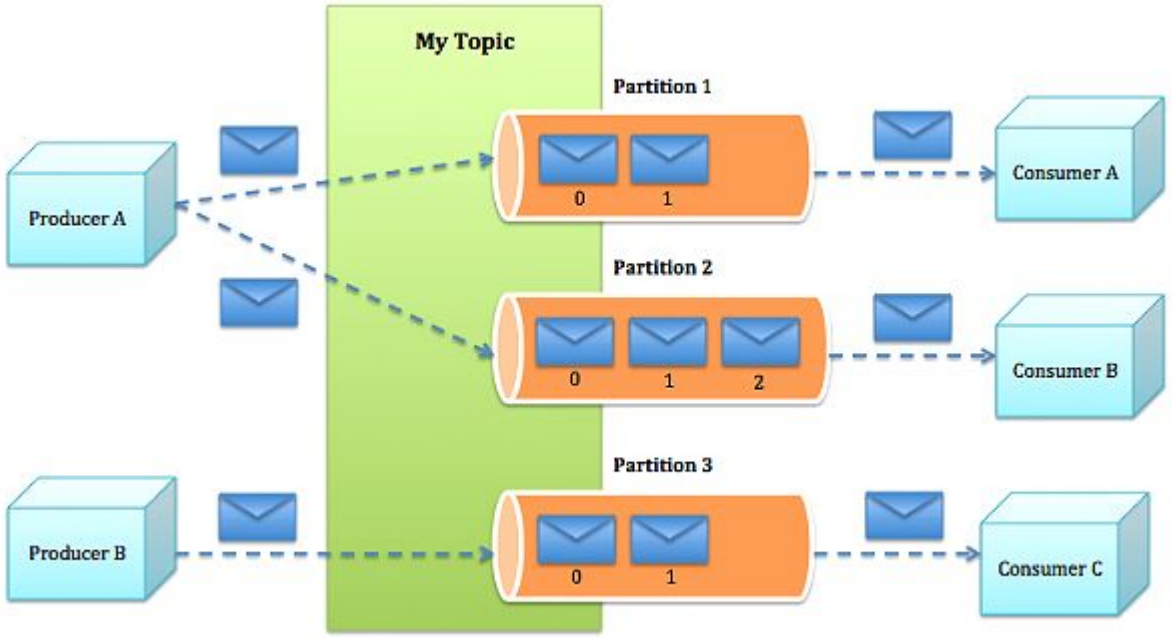
Stream Processing Components

- An event producer - Sensors, web logs, web events
- A messaging service - Kafka, RabbitMQ, ActiveMQ
- A stream processing framework - Samza, Storm, Spark Streaming



Apache Kafka

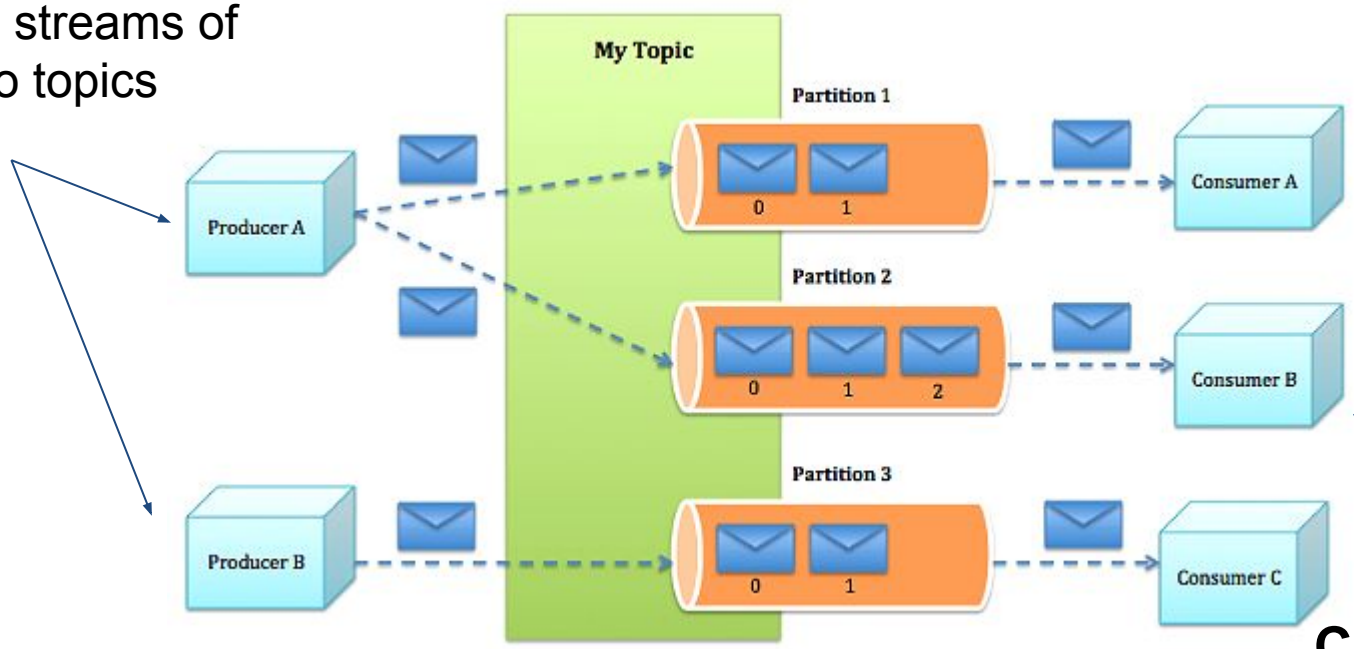
- Developed at LinkedIn as a distributed messaging system.



Apache Kafka

Topic
Category that stores messages

Producer
Process that sends streams of data to topics



Partition
Enables topics to be processed in parallel

Consumer
Process that reads data streams from topics

Semantic Partitioning in Kafka

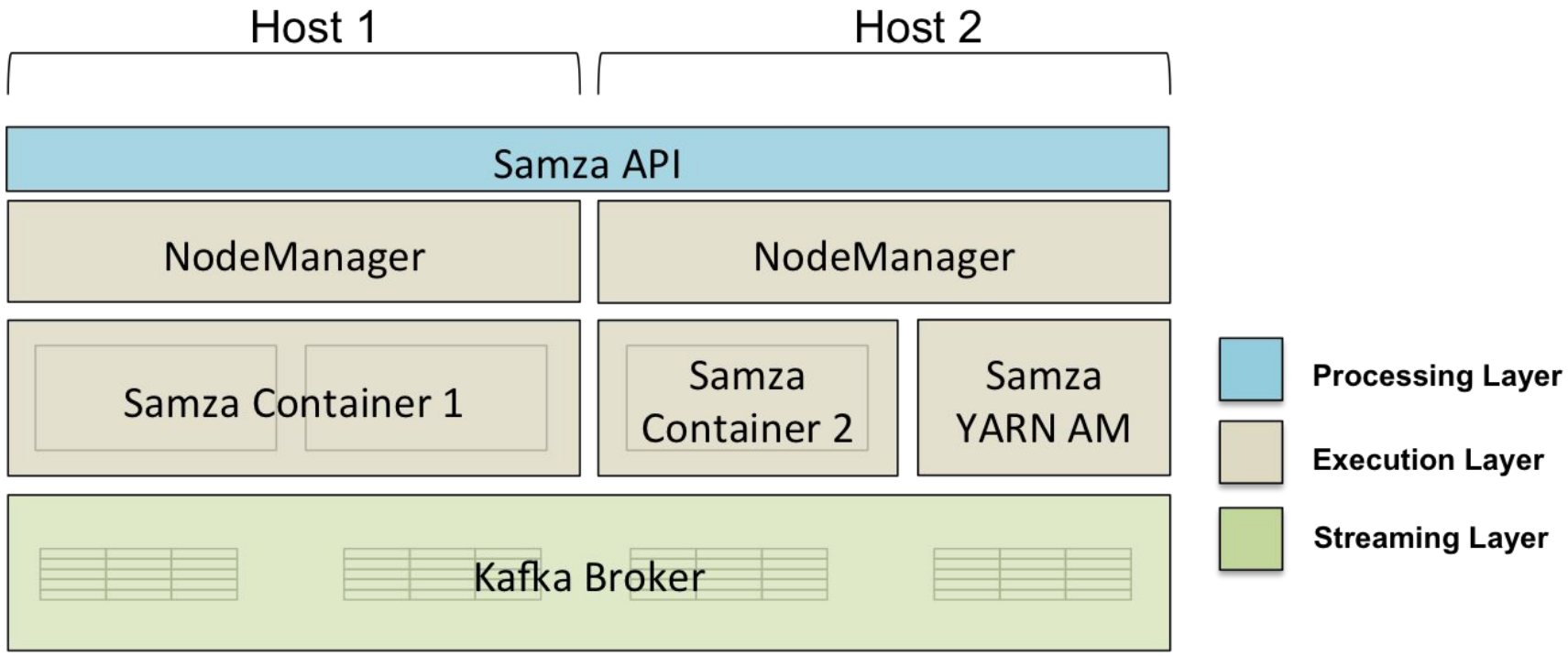
- Each topic is partitioned for scalability across all nodes in the Kafka cluster
- Default partitioning attempts to load balance
- Streams can also be partitioned semantically by a user-defined key where all messages with the same key arrive at the same partition
- Total ordering over records within a partition is guaranteed
- Fault-tolerance via replication
  - One leader and zero/more followers
  - Replication factor
  - ISR (in-sync replicas)

Kafka APIs

- Producer API
  - Used by applications to send streams of data to topics in a Kafka cluster
- Consumer API
  - Used by applications to read streams of data
- Kafka Streams
  - Process and analyze data in Kafka
  - Input and output data are stored in Kafka clusters
- Kafka Connect
  - Stream data between Kafka and other systems

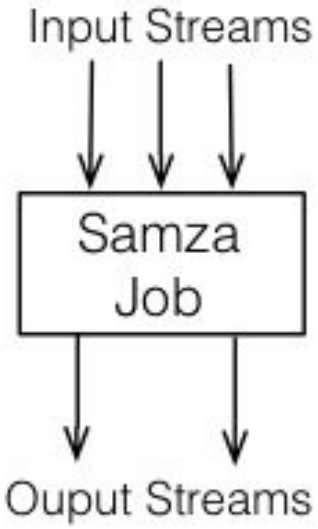
Apache Samza

- Stream processing framework developed at LinkedIn
- Consists of 3 layers:
  - Streaming, execution and processing (Samza) layer
- Most common use: Kafka for streaming, YARN for execution



Apache Samza

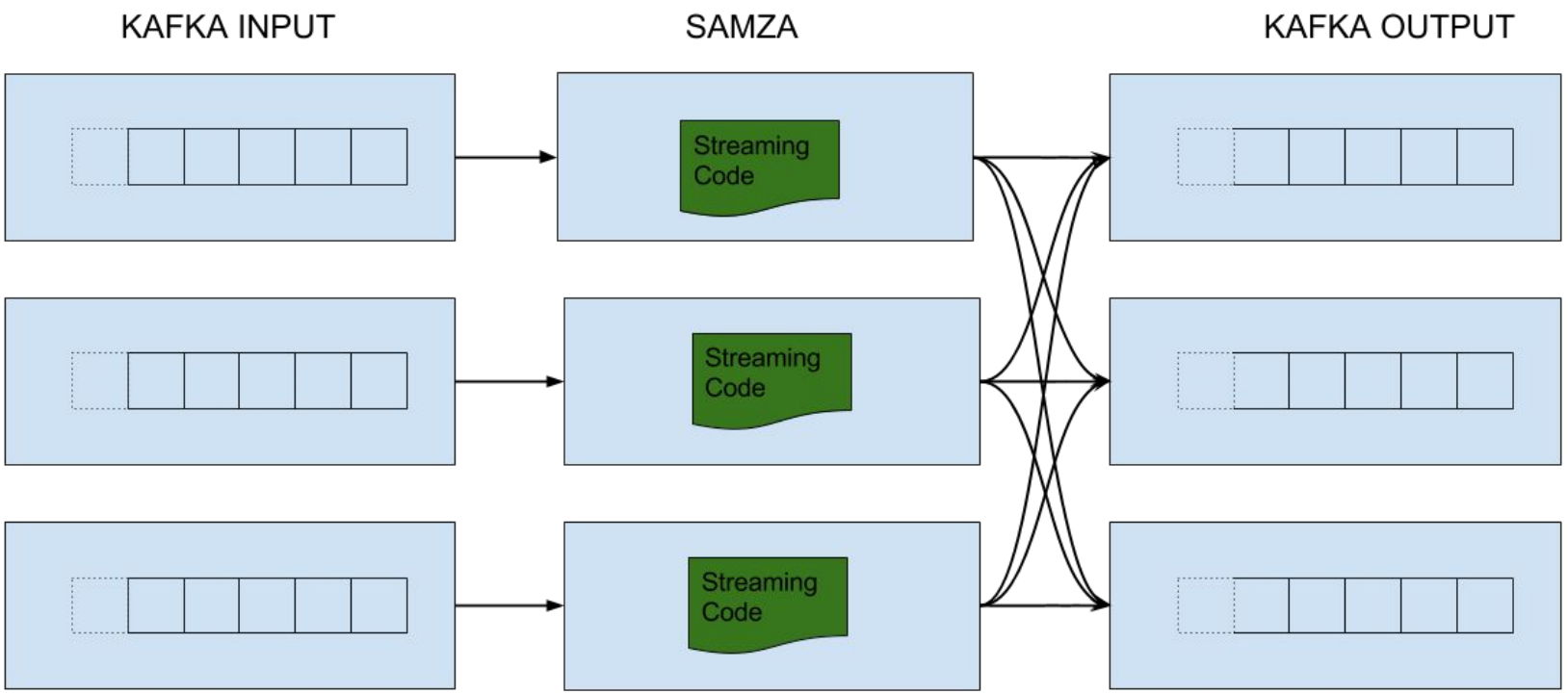
- Programmer uses Samza API to perform stream processing
- Each partition in Kafka is assigned to a single Samza task instance



Stateful Stream Processing in Apache Samza

- Motivation: calculate sum, average or count across events
- State in remote data store? - slow
- State in local memory? - machine might crash
- Solution - persistent KV store provided by Samza
  - Changes to KV store persisted to a different stream (usually Kafka) - replay on failure
  - RocksDB currently supported as a persistent KV store
    - You MUST use a persistent KV store for P4.3!

Putting Kafka and Samza Together



Project 4.3 Scenario

- Scenario
  - Ride sharing company in NYC
  - Match clients with the best available driver
- Dataset
  - File representing a stream of driver-locations and events

Project 4.3 Task Summary

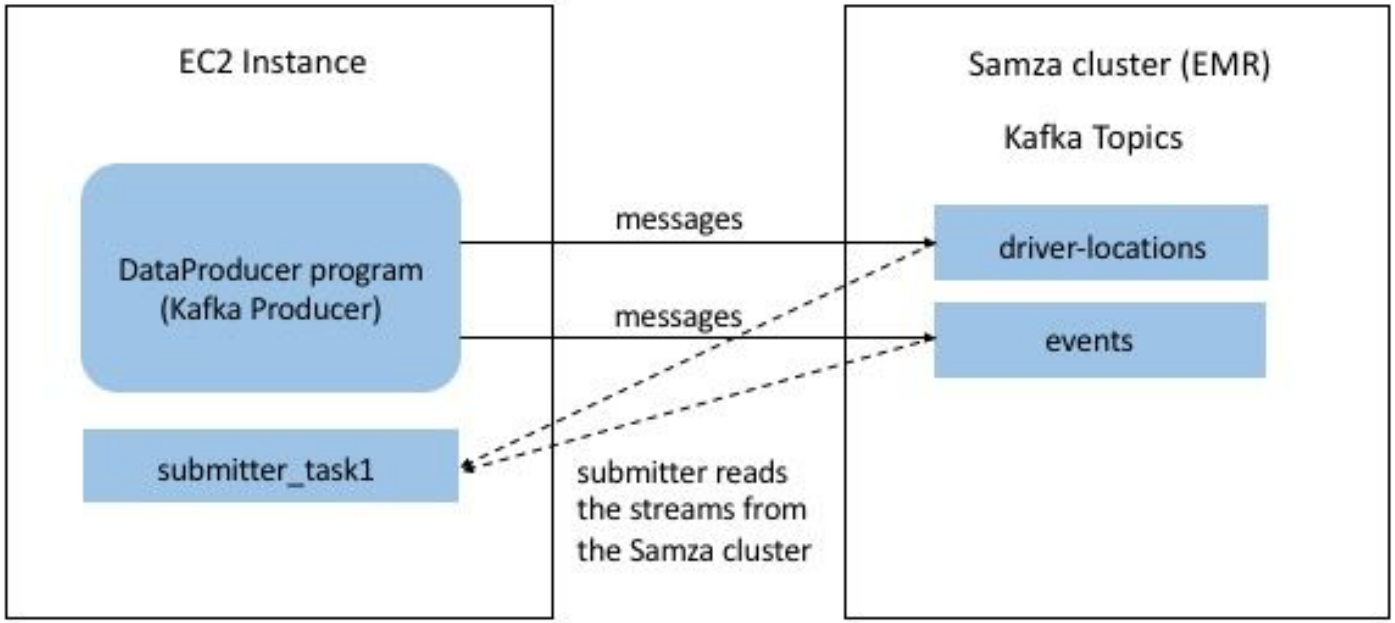
Task	Description	APIs	Points	Language
1	Generate Kafka streams from a given tracefile	Kafka Producer API	20	Java
2	Join streams and output best client-driver pairing	Samza	55	Java
Reflection and discussion	Reflection due Friday 5/4 Discussion due **5 days** after	-	5	-
Bonus	Build a visualization to display updated driver locations on a map in real-time	Node.js + React	10	Javascript

Task 1

- Each line in the tracefile represents a new message in JSON format
- Can assume the data is clean
- Send them to Kafka topics based on the `type` field

Type	Stream
DRIVER_LOCATION	driver_locations
LEAVING_BLOCK	events
ENTERING_BLOCK	events
RIDE_REQUEST	events
RIDE_COMPLETE	events

Task 1 Architecture



Task 2

- Find the best match of a ride request with a driver located on the same block as the rider
- Use the following formula to calculate the score:

match_score = distance_score * 0.4

+ gender_score * 0.1

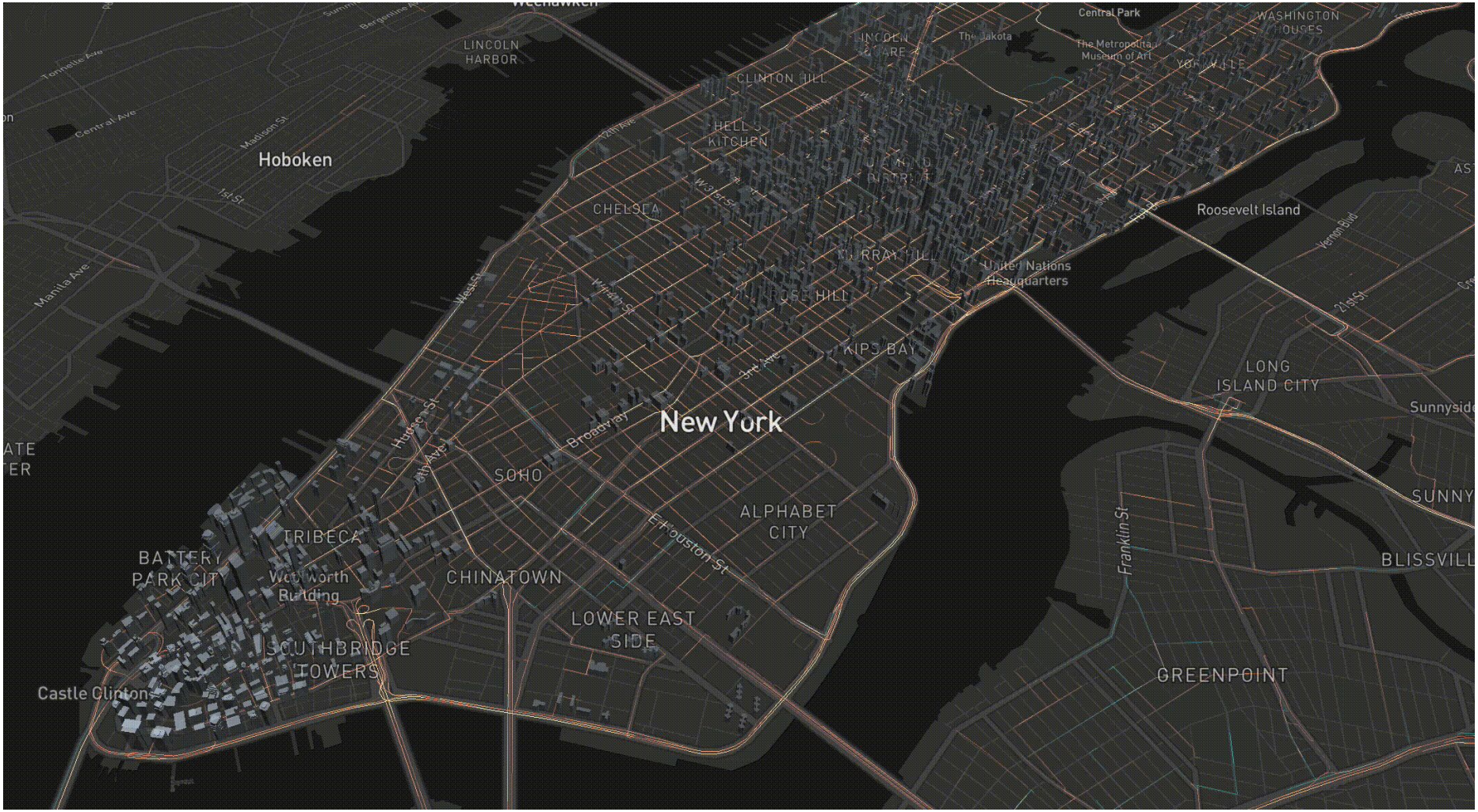
+ rating_score * 0.3

+ salary_score * 0.2

Task 2

- Debugging (IMPORTANT!)
  - Use the YARN UI
  - Output a Kafka stream for debugging
  - Yarn application commands
    - `yarn application -list`
  - YARN container logs
    - On the machine where the YARN container is running
- Read the debugging section in the writeup carefully!

Bonus Task


Central Park
WASHINGTON HOUSES
LINCOLN HARBOR
CLINTON HILL
The Metropolitan Museum of Art
HELL'S KITCHEN
Hoboken
CHELSEA
Roosevelt Island
United Nations Headquarters
KIPS BAY
LONG ISLAND CITY
New York
SOHO
ALPHABET CITY
TRIBECA
BATTERY PARK CITY
CHINATOWN
LOWER EAST SIDE
SOUTHBRIDGE TOWERS
Castle Clinton
GREENPOINT
BLISSVILL
SUNNY

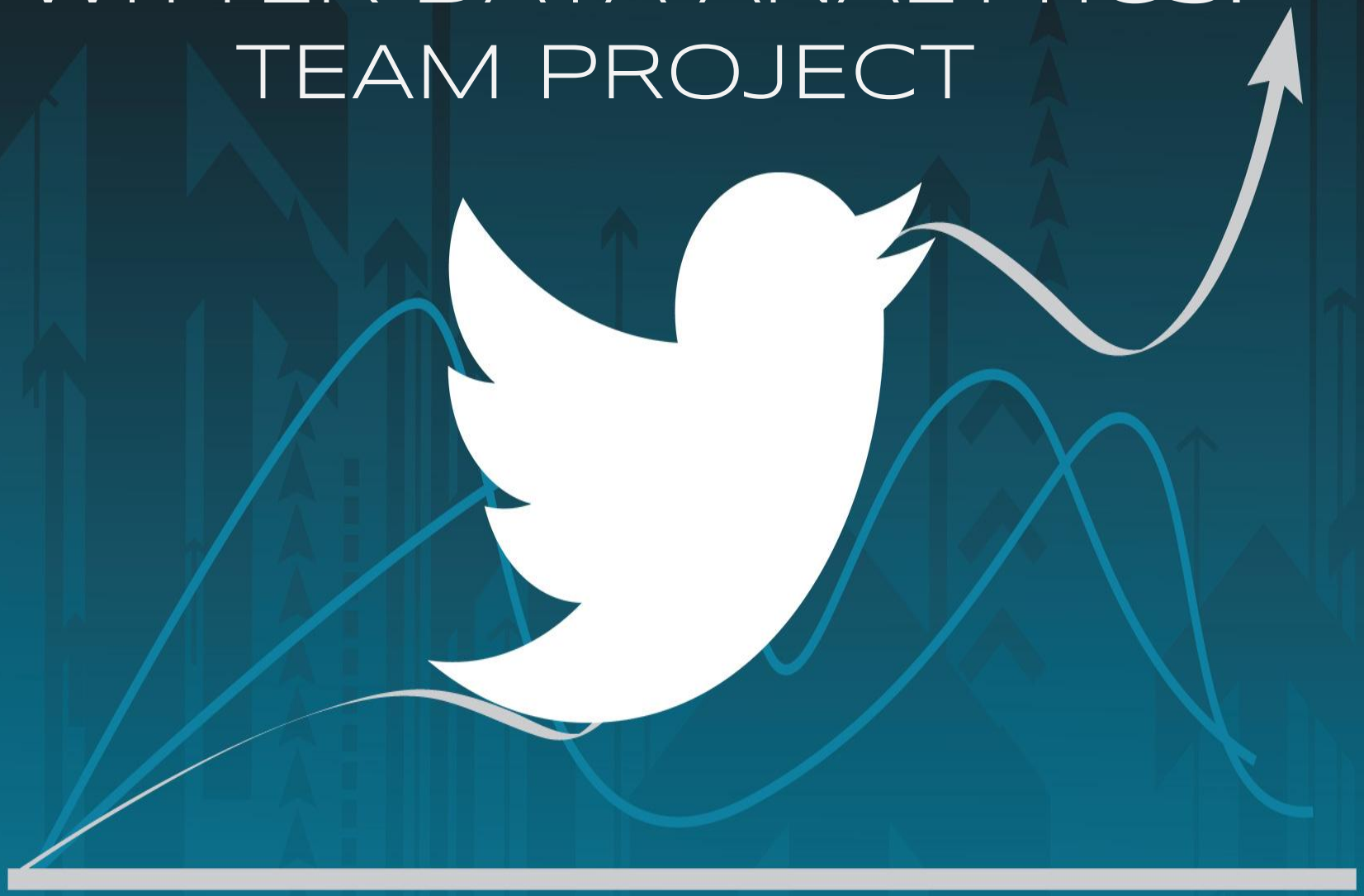
Bonus Task

- Stream Visualization
- Play with our current visualization as described in the writeup
- **Task**: Replace the current trip data - which is populated from a flat file - with data that is consumed from the Kafka stream
- Additionally, you are free to read the events and match-stream to visualize rider events

P4.3 Grading

- Submitters can be found in the skeleton code package
- Please run the `deploy` script only AFTER EMR's status is 'Running'
- Follow the instructions in the submitter
  - Prompts for starting the Kafka Producer and Samza job
- We will look for usage of KV stores and reasonably efficient code
  - Do not iterate through ALL drivers to find the best match!

TWITTER DATA ANALYTICS: TEAM PROJECT



Please come to Thursday's Recitation to learn more!

Team Project

Phase 3 Live Test Scoreboard

Submitter	Score	Q1 Tput	Q2 Tput	Q3 Tput	Q4 Tput
Pyrocumulus	80	30389.2	18977.81	5853.74	16885.5
DDLKiller	80	31276.1	12099.88	4542.12	7194.78
monoid	75	30463	23562.24	11409.92	7991.3
liulaliula	75	29187.58	14038.89	12628.64	2637.07
Hong Kong Journalists	74	29637.1	11276.69	8036.03	2880.6
wen as dog	73	29677.9	24347.2	8686.85	3720.43
TaZoRo	73	30274.69	11580	2518.82	2303.26
aaa	71	30182.9	16086.49	3647.39	3183.93
team6412	70	29445.3	11781.2	1889.15	1950.8
Boat	70	29421.61	10987.66	1932.18	4513.02
TakemezhuangbTakemefly	68	30194	21864.39	2001.64	4976.7
NoInsomnia	68	29377.8	17443.3	5381.94	1995.29
30 centimeter	68	28335	22017	4457	15536.3
FatTiger	65	30397.2	14460.1	4465	3952
return no_sleep	64	29302.6	15790.6	5191.46	3625.2

Team Project Overall Winners

- Attend the Thursday (5/3) recitation
  - Pittsburgh: 4:30pm ET in GHC 4307
  - SV: 1:30pm PT in SV 109/110
- See the winners of the Team Project
- Hear the top three teams discuss their implementation
- Eat cupcakes
- Have fun!

Upcoming Deadlines

- Phase 3 report
  - **Due: TODAY 5/1/2018 11:59 PM Pittsburgh**
- Project 4.3 : Stream Processing with Kafka/Samza
  - **Due: FRIDAY 5/4/2018 11:59 PM Pittsburgh**
- Apply to TA in F18
  - https://goo.gl/forms/FX2x0t04zbrZBStf2
- Complete the course survey (announced on Piazza)
  - 2% bonus for the overall course grade (Don't miss it!!!)
  - **Due: Saturday 5/4/2018 11:59 PM Pittsburgh**
- Cupcake Party (GHC 4307 Pittsburgh and SV 109/110)
  - Thursday 5/3/2018 4:30 PM ET Pittsburgh, 1:30 PM PT SV

Questions?