

Lessons in Internet Scale Stream Processing @LinkedIn

Kartik Paramasivam

Director of Engineering, Streams Infrastructure, LinkedIn

InfoQ

促进软件开发领域知识与创新的传播



关注InfoQ官方微信
及时获取ArchSummit
大会演讲视频信息

QCon

全球软件开发大会 [北京站]

2017年4月16-18日 北京·国家会议中心

咨询热线: 010-64738142

ArchSummit

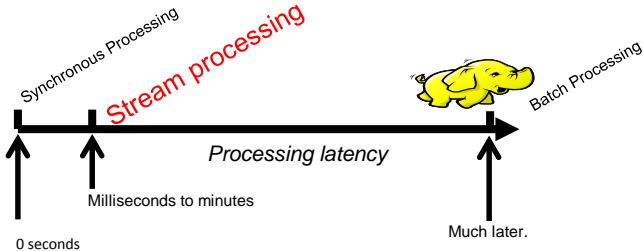
全球架构师峰会 2016 [深圳站]

2017年7月7-8日 深圳·华侨城洲际酒店

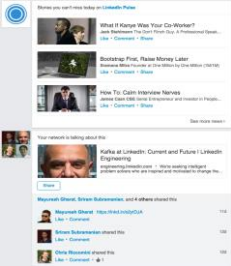
咨询热线: 010-89880682

AGENDA

- **Introduction**
- Typical Architecture for Data Processing
- Data Ingestion
- Stream processing
- Future



FEED RELEVANCE (E.G. NEWSFEED)



Doesn't your car miss today on [LinkedIn Pulse](#)?

What If Kanye Was Your Co-Worker?
Jack Sparrows The Don't Finish Guy, A Professional Opin...
[Like](#) · [Comment](#) · [Share](#)

Bootstrap First, Raise Money Later
Brennan Miller Founder at One Million by One Million (OM1M)
[Like](#) · [Comment](#) · [Share](#)

How To: Calm Interview Nerves
James Dean OSM Senior Entrepreneur and Investor in People...
[Like](#) · [Comment](#) · [Share](#)

[See more news](#)

Your network is talking about this

Kafka at LinkedIn: Current and Future of LinkedIn Engineering
engineering.linkedin.com · We're seeking intelligent problem solvers who are inspired and motivated to change the...
[Share](#)

[Meyurash Ghant](#), [Srikanth Subramanian](#), and 4 others shared this

[Meyurash Ghant](#) [https://lnkd.in/g/...](#) 114
[Like](#) · [Comment](#)

[Srikanth Subramanian](#) shared this 108
[Like](#) · [Comment](#)

[Cheta Bhasani](#) shared this 102
[Like](#) · [Comment](#) · [👍](#)

CYBER-SECURITY



ToonClips.com

#9409

service@toonclips.com

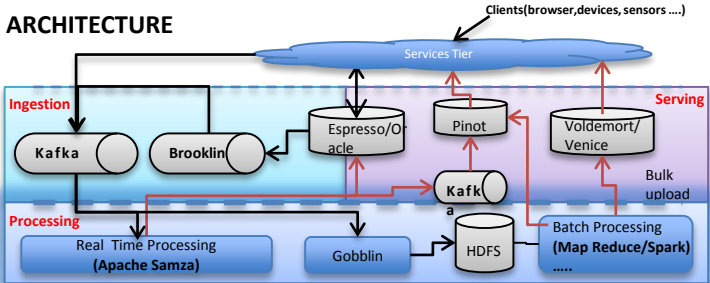
INTERNET OF THINGS



AGENDA

- Introduction
- **Typical Architecture for Data Processing**
- Data Ingestion
- Stream processing
- Future

ARCHITECTURE



AGENDA

- Introduction
- Typical Architecture for Data Processing
- **Data Ingestion**
- Stream processing
- Future

DATA INGESTION OPTIONS



	APACHE KAFKA	AZURE EVENTHUB	AWS KINESIS	GOOGLE PUB-SUB
Partition Aware	Yes	Yes	Yes	No
Ability to Replay	Yes	Yes	Yes	No
Portable across Clouds	Yes	No	No	No

DATA INGESTION :
APPROX.
COST BASED
COMPARISON

- **Running at Small Scale (< 1TB a day)**
 - Pay as you go Cloud managed services (Kinesis/EventHub) are cheaper
- **Running at Medium Scale**
 - < 100 Billion messages/day, < 5000 partitions**
 - Pay as you go Cloud managed services(Kinesis/EventHub) are cheaper (incl. operational cost)
- **Running at Large Scale**
 - > 100 Billion messages/day, > 5000 partitions**
 - Managing your own custom Apache Kafka Cluster is cheaper

KAFKA @ LINKEDIN

– Ingested Data/Day

- ~ 1.2 Trillion Messages
- 350 TB
- Peak load of 16 Million Messages/second

– Data consumed/day : 1.3 PB

– ~1800+ Kafka Broker Machines

LESSONS FROM RUNNING KAFKA @ SCALE

- **Machines/disks will die almost daily**
 - Need auto-healing(Kafka Cruise Control)
- **Take good care of Zookeeper**
 - 5 node Zookeeper clusters on SSD.
- **Monitoring will always reveal problems.**
 - Kafka Monitor ([link](#)), Burrow ([link](#))
- **Dealing with Multi-tenancy**
 - Quotas/Rate Limiting is critical to avoid availability dips

KAFKA : COST CONSIDERATIONS

- **Bigger machines can be cheaper**
 - But.. Machine failures take very long to recover
- **More Data Storage = Bigger Clusters**
 - Better support for regular disks instead of using RAID-10
 - Optimized Data Retention capabilities (future)
 - Background recompression of old data (future)

AGENDA

- Introduction
- Typical Architecture for Data Processing
- Data Ingestion
- **Stream processing**
- Future

TYPES OF STREAM PROCESSING

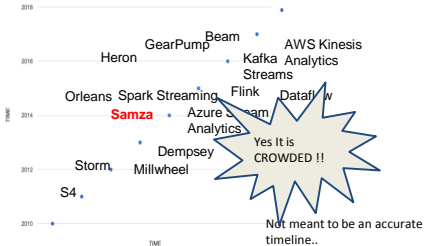
– Stateless Processing

- Transformation etc.
- Lookup adjunct data (lookup databases/call services)
- Producing results for every event

– Stateful Processing

- Triggering/Producing results periodically (time-windows)
 - Maintain intermediate state
- E.g. Joining across multiple streams of events.

STREAM PROCESSING LANDSCAPE



APACHE SAMZA

- Top level Apache project since Dec 2014
- 5 big Releases (0.7, 0.8, 0.9, 0.10, 0.11)
- 62 Contributors
- 14 Committers
- Companies using : LinkedIn, Uber, MetaMarkets, Netflix, Intuit, TripAdvisor, MobileAware, Optimizely
- <https://cwiki.apache.org/confluence/display/SAMZA/Powered+By>
- Applications at LinkedIn : from ~20 to ~200 in 2 years.

HARD PROBLEMS IN STREAM PROCESSING

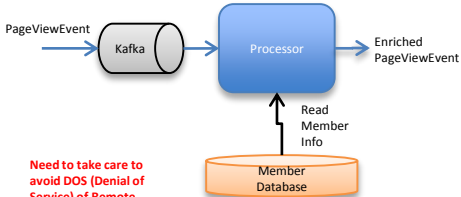
- **Performance !!**
- Stability
- Support for a variety of input sources
- Generating Accurate Results
- Reprocessing

PERFORMANCE

- **I/O for accessing state is the biggest bottleneck !**
 - Reading a Database
 - Maintaining Temporary State
 - Writing Results

PERFORMANCE : READING ADJUNCT DATA

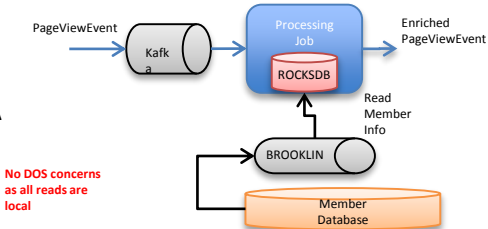
Using a Regular Database



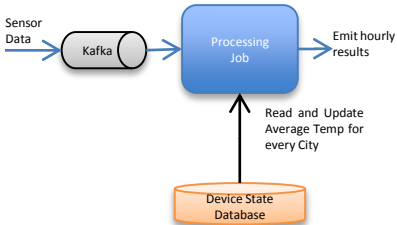
Need to take care to avoid DOS (Denial of Service) of Remote Database access

Using an Embedded Database

PERFORMANCE : READING ADJUNCT DATA



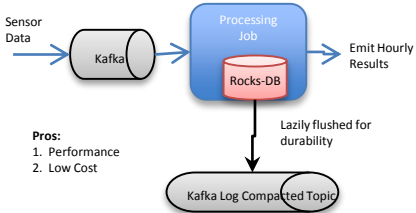
Using a Regular Database



PERFORMANCE :
MAINTAINING TEMPORARY STATE

PERFORMANCE : MAINTAINING TEMPORARY STATE

Using an Embedded Database



Pros:

1. Performance
2. Low Cost

LOCAL VS REMOTE DATA ACCESS

– **100x** difference in Performance

– Local Data access :

- **1.1 Million TPS** on a single processing machine (SSD)
- Used a 3 node Kafka cluster for storing the durable changelog

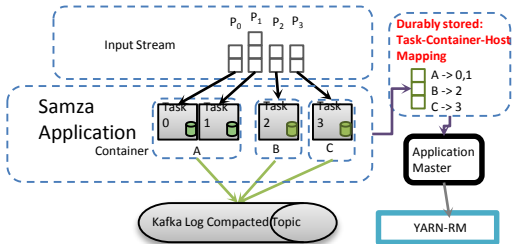
– Remote Data Access:

- 8500 TPS when the Samza job was changed to accessing a remote No-Sql store
- No-Sql Store was also on a 3 node (ssd) cluster

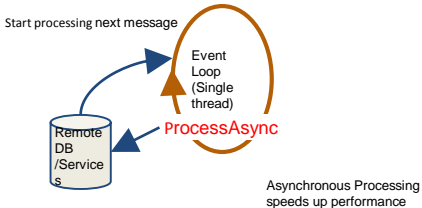
AN EMBEDDED DATABASE DOES NOT ALWAYS WORK WELL

- If a database is too large and its access is not 'partitioned'
- Input stream doesn't support partitioned access (e.g. Google Pub-Sub)
- Number of partitions of the input stream is changing very often
- Aggressive auto-scaling the processor is required
- You anyways have to store the results of the stream processor into a serving Database (e.g. Espresso, Cassandra)

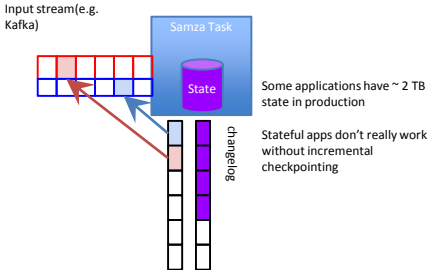
HOW SAMZA KEEPS LOCAL STATE STABLE



SUPPORTING ASYNC I/O



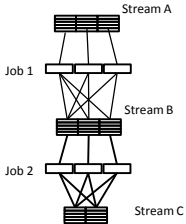
INCREMENTAL CHECKPOINTS



HARD PROBLEMS IN STREAM PROCESSING

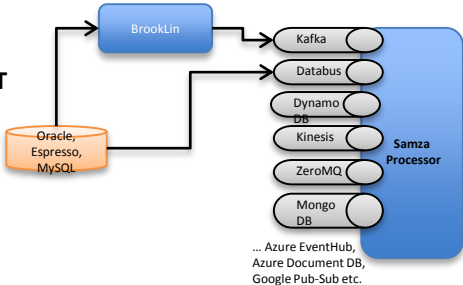
- Performance !!
- **Stability**
- Support for a variety of input sources
- Generating Accurate Results
- Reprocessing

BACKPRESSURE IN A PROCESSING PIPELINE



- Kafka or durable intermediate queues are leveraged to avoid backpressure issues in a pipeline.
- Allows each stage to be independent of the next stage

SUPPORT MANY STREAMING EVENT SOURCES



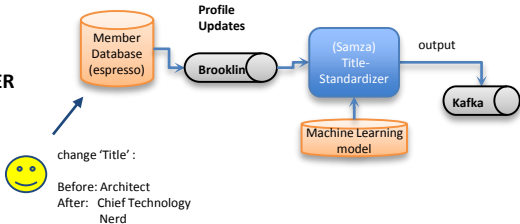
HARD PROBLEMS IN STREAM PROCESSING

- Performance !!
- Stability
- Support for a variety of input sources
- **Reprocessing**
- Generating Accurate Results

REPROCESSING

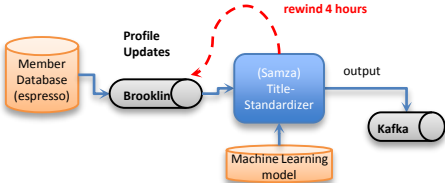
- **What is reprocessing ?**
 - Process events that happened in the past.
- **Why do we have to do Reprocessing ?**
 - Changes in business logic
 - Software Bugs

CASE STUDY : TITLE STANDARDIZER

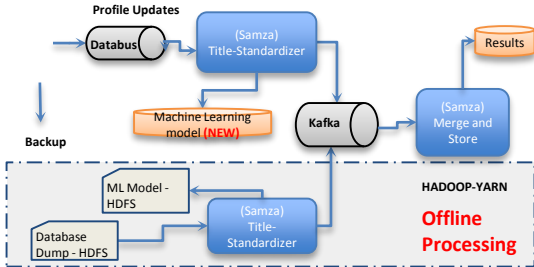


REPROCESSING

- DEALING WITH BUGS



REPROCESSING - ENTIRE DATASET



BATCH PROCESSING IN SAMZA!!

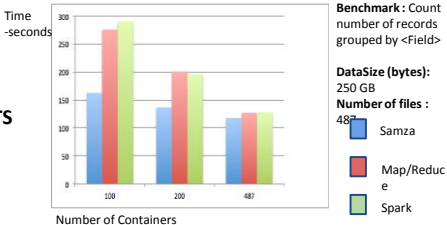
(NEW)

- HDFS system consumer for Samza.
- Same Samza processor can be used for processing events from Kafka and HDFS with no code changes.

SAMZA- HDFS

EARLY PERFORMANCE RESULTS

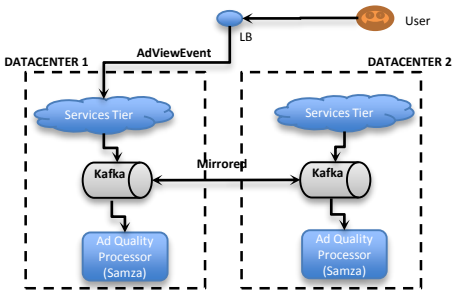
!!



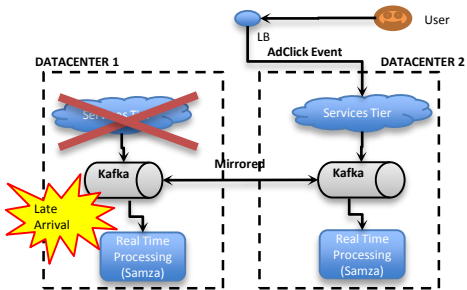
HARD PROBLEMS IN STREAM PROCESSING

- Performance !!
- Stability
- Support for a variety of input sources
- Reprocessing (dealing with business logic changes)
- **Generating Accurate Results**

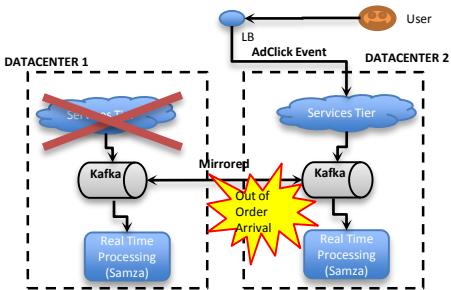
DELAYS IN EVENT STREAM



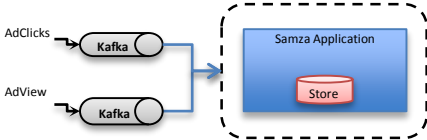
DELAYS IN EVENT STREAM



DELAYS IN EVENT STREAM



DEALING WITH ACCURACY



Influenced by Google
Millwheel

1. All events are stored locally
2. Find impacted 'window/s' for late arrivals
3. Re-compute result
4. Choose strategy for emitting results (absolute or relative value)

AGENDA

- Introduction
- Typical Architecture for Data Processing
- Data Ingestion
- Stream processing
- **Future**

FUTURES

– Data Ingestion

- More cheaper and flexible offerings
- Kafka will become easier to operate

– Stream Processing

- Accuracy (event time support) will be the default
- Convergence of nearline and offline processing technologies
 - Seamless SQL over streams/batch
- Local State will become more prevalent

Thank you !

APPENDIX

KAFKA : PERFORMANCE CONSIDERATIONS

- Bottleneck in the Broker:
 - Network for 1 Gbps NICs
 - CPU for 10 Gbps
- SSL adds CPU overhead on Brokers
- Avoiding Recompression in Brokers saves CPU (new)
- Client side batching improves compression, but increases latency

BROOKLIN – INGESTION FROM DATABASES

