A Performance Evaluation of Apache Kafka in Support of Big Data Streaming Applications



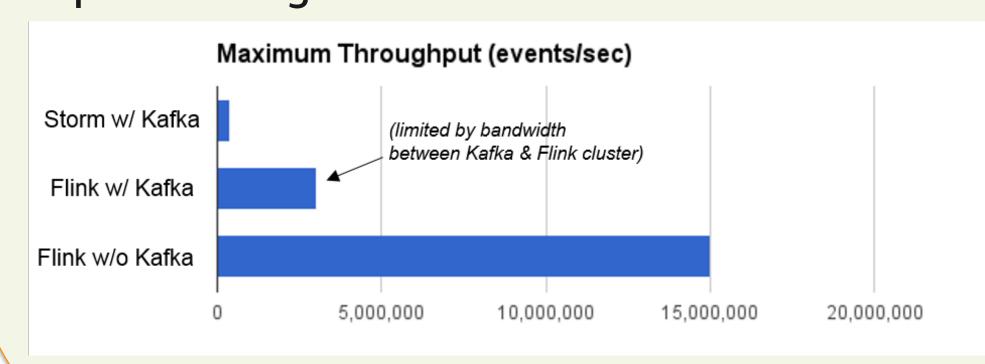
Paul LE NOAC'H¹, Alexandru COSTAN², Luc BOUGE³

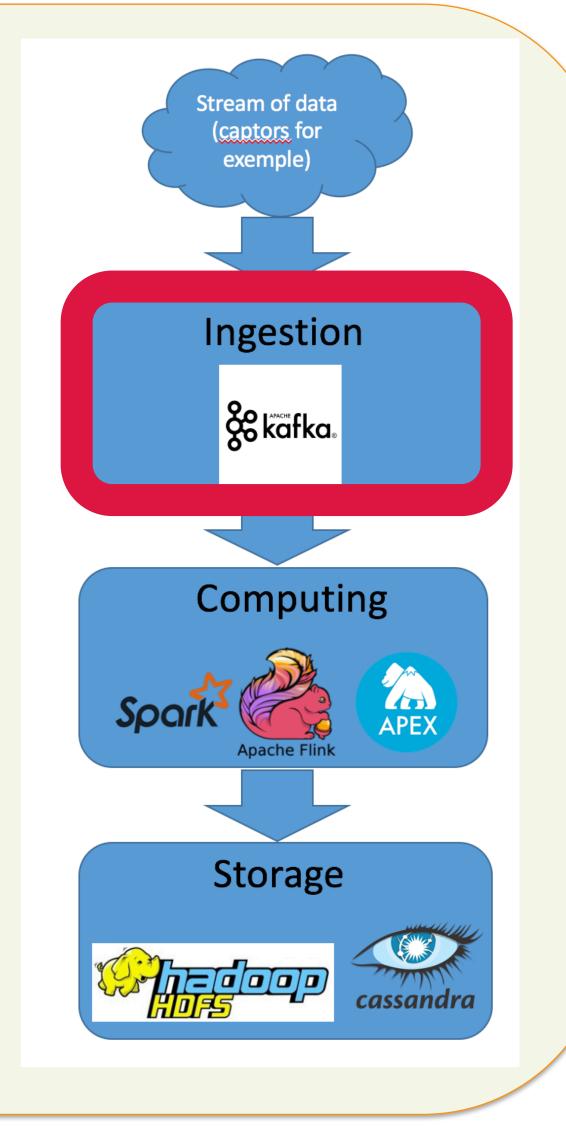
¹ INSA Rennes, ² INRIA / INSA Rennes, ³ ENS Rennes



1. Context

- Stream computing: a new paradigm enabling real-time Big Data processing through 3 steps
 - Ingestion: Apache Kafka
 - Processing: Apache Spark / Flink
 - Storage: HDFS, Cassandra
- Ingestion can be a bottleneck for stream processing



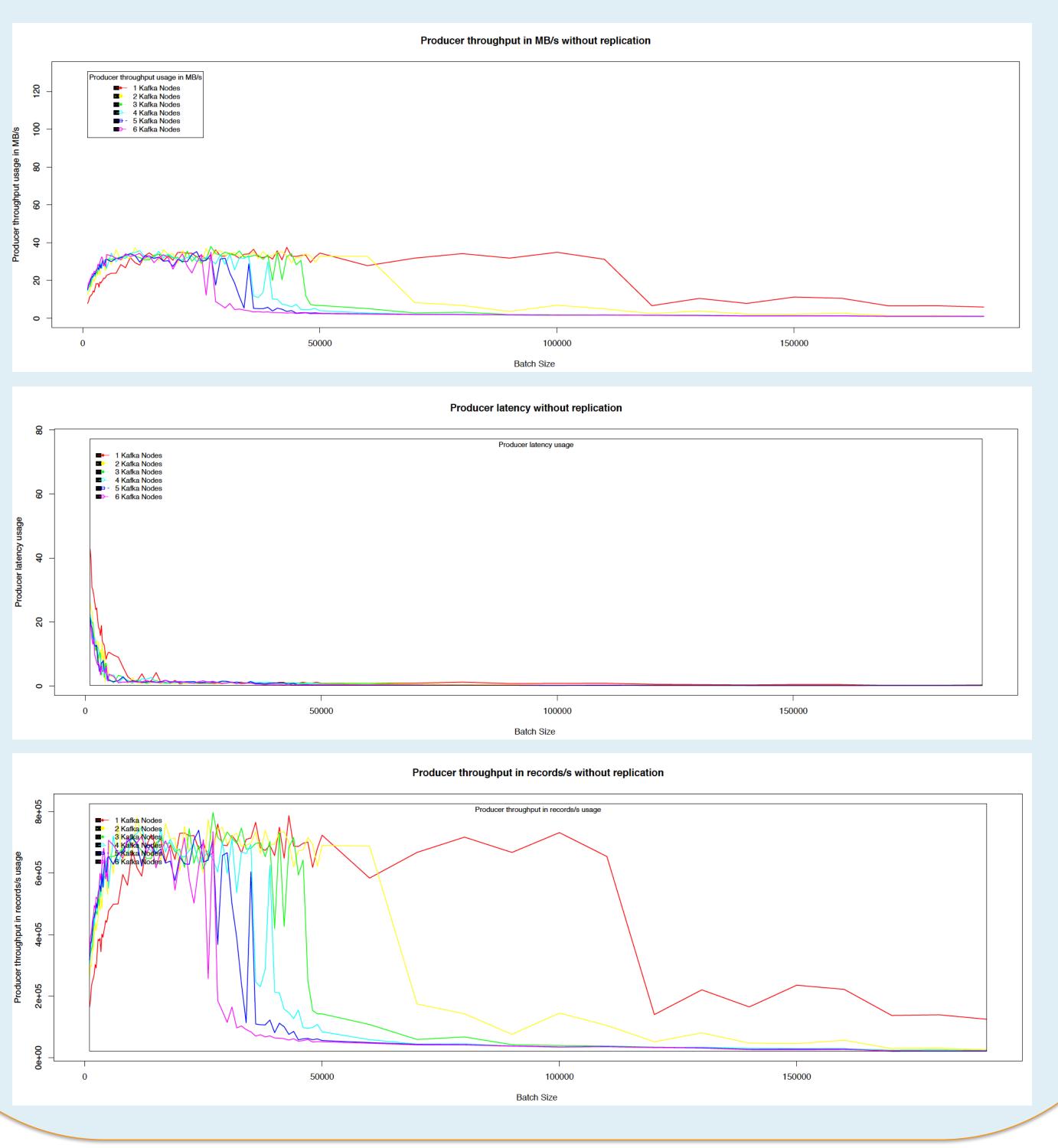


2. Contribution

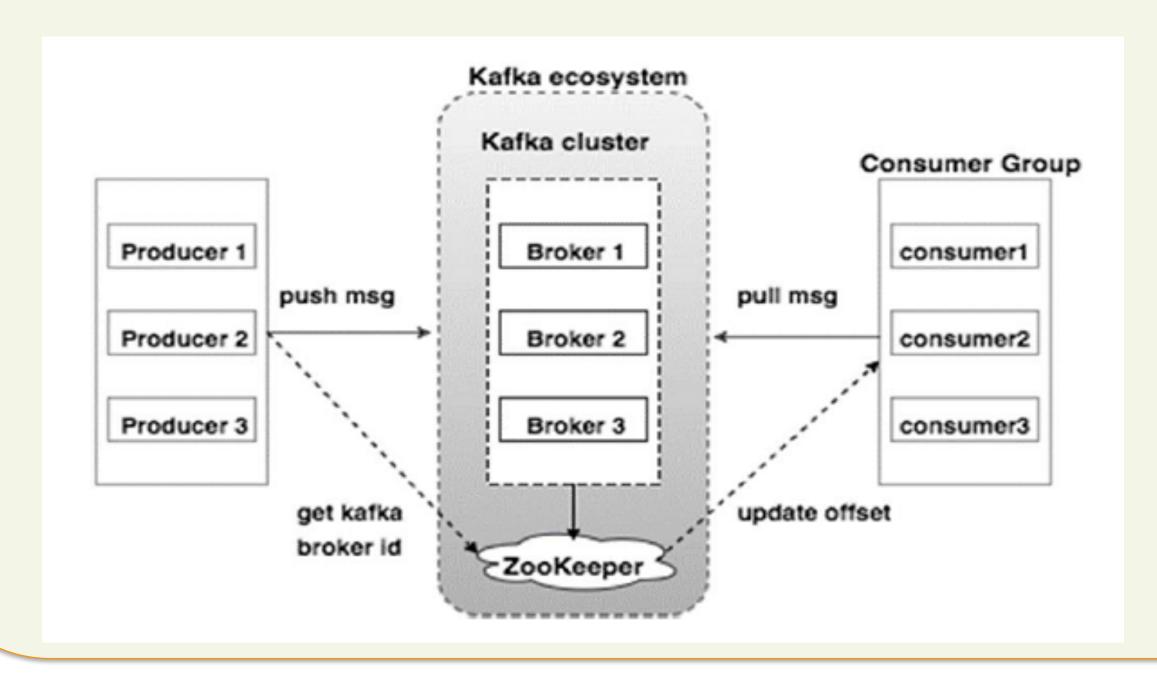
- Identify the **impact of different parameter** settings on Kafka's overall performance
- Experiment evaluation of several configurations and performance metrics of Kafka
- Allow users to avoid bottlenecks and achieve good practice for stream processing

5. Results

Producer performances when modifying batch size for several number of nodes and a message size of 50B



3. Kafka Architecture



4. Methodology

- Isolate the performance of each Kafka component
- Separated tests for Producers and Consumers
- Make correlations between configuration parameters, resource usage and performance metrics
- Experiments executed on **Grid5000**
- Up to **32 nodes** (16 cores per nodes, 28 GB RAM, 10 Gigabit Ethernet)

6. Key metrics

Parameters:

- Message size
- Batch size
- Acquirement strategy
- Network and disk I/O threads
- Message replication
- Hardware

Performance Metrics:

- Throughput (MB/s, items/s)
- Latency
- CPU usage
- Disk usage
- Memory usage
- Network usage

7. Take-aways

- The variation of the batch size shows that there is a range of batches with a better performance.
- When varying the number of nodes in some scenarios: a sudden performance drop (probably due to the internal Kafka synchronizations as well as the underlying network).
- Future work: evaluating reference processing frameworks (Apache Spark and Flink)

Bibliographie / sources

- 1. https://data-artisans.com/blog/extendingthe-yahoo-streaming-benchmark
- 2. http://www.tutorialspoint.com/apache_kafka/

