Aalto University
School of Science
Degree Programme in Computer Science and Engineering

Yangjun Wang

Stream Processing Systems Benchmark: StreamBench

Master's Thesis
Espoo, Nov 20, 2015

DRAFT! — December 31, 2015 — DRAFT!

Supervisors: Assoc. Prof. Aristides Gionis
Advisor: D.Sc. Gianmarco De Francisci Morales

Aalto University
School of Science
Degree Programme in Computer Science and Engineering

ABSTRACT OF MASTER'S THESIS

Author:	Yangjun Wang		
Title: Stream Processing Systems Benchmark: StreamBench			
Date:	Nov 20, 2015	**Pages:**	12
Major:	Foundations of Advanced Computing	**Code:**	T-110
Supervisors:	Assoc. Prof. Aristides Gionis		
Advisor:	D.Sc. Gianmarco De Francisci Morales		

Batch processing technologies(Such as MapReduce, Hive, Pig) have matured and been widely used in the industry. These systems solved the issue processing big volumes of data successfully. However, first big data need to be collected and stored in a database or file system. Then it takes time to finish batch processing analysis job before get any results. While there are many cases that need analysed results from streaming data in seconds or sub-seconds. To satisfy the increasing demand of processing stream data, several streaming processing systems are implemented and widely adopted, such as Apache Storm, Apache Spark, IBM InfoSphere Streams and Apache Flink. They all support online stream processing, high scalability, and tasks monitoring. While how to evaluate a stream processing system before choosing it to use in production development is a open question.

In this thesis, we introduce StreamBench, a benchmark framework to facilitate performance comparisons of stream processing systems. A common API component and a core set of workloads are defined. We implement the common API and run benchmarks for three widely used open source stream processing systems: Apache Storm, Spark Streaming and Flink Streaming. Therefore, a key feature of the StreamBench framework is that it is extensible – it supports easy definition of new workloads, in addition to making it easy to benchmark new stream processing systems.

Keywords:	Big Data, Stream, Benchmark, Storm, Flink, Spark
Language:	English

Acknowledgements

I want to thank Professor Aristides Gionis and my advisor Gianmarco De Francisci Morales for their good guidance.

Espoo, Nov 20, 2015

Yangjun Wang

Abbreviations and Acronyms

Symbols

$\mathbf{B}$	magnetic flux density
c	speed of light in vacuum $\approx 3 \times 10^8$ [m/s]
ω_{D}	Debye frequency
ω_{latt}	average phonon frequency of lattice
$\uparrow$	electron spin direction up
$\downarrow$	electron spin direction down

Operators

$\nabla \times \mathbf{A}$	curl of vectorin $\mathbf{A}$
$\frac{\mathrm{d}}{\mathrm{d}t}$	derivative with respect to variable t
$\frac{\partial}{\partial t}$	partial derivative with respect to variable t
$\sum_i$	sum over index i
$\mathbf{A} \cdot \mathbf{B}$	dot product of vectors $\mathbf{A}$ and $\mathbf{B}$

Abbreviations

AC	alternating current
APLAC	an object-oriented analog circuit simulator and design tool (originally Analysis Program for Linear Active Circuits)
BCS	Bardeen-Cooper-Schrieffer
DC	direct current
TEM	transverse eletromagnetic

Contents

List of Figures

Chapter 1

Introduction

Along with the rapid development of information technology, the speed of data generation increases dramatically. To process and analysis such large amount of data, the so-called Big Data, cloud computing technologies get a quick development, especially after these two papers related to MapReduce and BigTable published by Google [2, 5].

1.1 Stream Processing and Evaluation

In theory, Big Data don't only mean "big" **v**olume. Besides volume, Big Data still have two other properties: **v**elocity and **v**ariety [7]. Velocity means the amount of data is growing at high velocity. Variety refers to the various data formats. They are called three ***V***s of Big Data. When deal with Big Data, there are two types of processing model, batch processing and stream processing. A big data architecture contains several parts. For batch processing, masses of structured and semi-structured historical data are stored in HDFS (**V**olume + **V**ariety). On the other side, stream processing is used for fast data requirements (**V**elocity + **V**ariety)[9].

Batch processing is generally more concerned with throughput than latency of individual components of the computation. In batch processing, data is collected and stored in file system. When the size of data reaches a tradeoff, batch jobs could be configured to run without manual intervention, trained against entire dataset at scale in order to produce output in the form of computational analyses and data files. Because of time consume in data collection and processing stage, depending on the size of the data being processed and the computational power of the system, output can be delayed significantly. Generally, latency could be range from minutes to hours.

Streaming processing is required for many practical cases which need analysed results from streaming data in a very short latency. For example, a online shopping website would want give a customer accurate recommendations as soon as possible after he/she scan the website for a while. Several streaming processing systems are implemented and widely adopted, such as Apache Storm, Apache Spark, IBM InfoSphere Streams and Apache Flink. They all support real-time stream processing, high scalability, and awesome monitoring.

How to evaluate a real time stream processing system before choosing it to use in production development is a open question. Before these real time stream processing systems are implemented, Michael demonstrated the 8 requirements[8] of real-time stream processing, which gives us a standard to evaluate whether a real time stream processing system satisfies these requirements. A very common and traditional approach to verify whether the performance of a system meets the requirements is benchmarking. Published benchmarking results from industry standard benchmark systems could help users compare products and understand features of a system easily. In this thesis, we introduce a benchmark framework called StreamBench to facilitate performance comparisons of stream processing systems.

1.2 Structure of the Thesis

The main topic of this thesis is stream processing systems benchmark. First, big data and cloud computing technology background is introduced in Chapter 2. Chapter 3 presents architecture and main features of three widely used stream processing systems. In Chapter 4, we demonstrate the design of our benchmark framework – StreamBench, including the whole architecture, test data generator and extensibility of StreamBench. Experiments and results are discussed in Chapter 5. At last, conclusions are given in Chapter 6.

Chapter 2

Background

Background knowledge of StreamBench which includes Big Data, Cloud Computing and widely accepted benchmark systems.

2.1 Cloud Computing

Concept of Cloud Computing and how Cloud Computing solves Big Data issues

2.1.1 Parallel Computing

2.1.2 Computing Cluster

2.1.3 Batch Processing and Stream Processing

2.2 Apache Hadoop

Introduce Apache Hadoop and several important modules

2.2.1 MapReduce

2.2.2 Hadoop Distribution File Systems

2.2.3 YARN

2.2.4 Zookeeper

2.3 Benchmark

Describe benchmark systems of traditional database and cloud service systems. Demonstrate design and components of benchmark system Traditional database management systems were evaluated with industry standard benchmarks like TPC-C[3], TPC-H[4]. These have focused on simulating complete business computing environment where plenty of users execute business oriented ad-hoc queries that involve transactions, big table scan, join and aggregation. The queries and the data populating the database have been chosen to have broad industry-wide relevance. This benchmark illustrates decision support systems that examine large volumes of data, execute queries with a high degree of complexity, and give answers to critical business questions[4]. The integrity of the data is verified during the process of the execution of the benchmark to check whether the DBMS corrupt the data. If the data is corrupted, the benchmark measurement is rejected entirely[6]. Benchmark systems for DBMS mature, with data and workloads simulating real common business use cases, they could evaluate performance of DBMS very well. Some other works were done related to specific business model. Linkbench[1] benchmarks database systems which store "social network" data specifically. The workload of database operations are based on Facebook's production workload and the data is also generated in such a way that key properties of the data match the production social graph data in Facebook.

2.3.1 Traditional Database Benchmark

2.3.2 Cloud Service Benchmark

Chapter 3

Stream Processing Platforms

Introduce three widely used stream processing platforms, point out core concepts and key features

3.1 Apache Storm

3.1.1 Storm Architecture

3.1.2 Computing Model

3.2 Apache Flink

3.2.1 Flink Architecture

3.2.2 Memory Management

3.2.3 Flink Streaming

3.3 Apache Spark

3.3.1 Resilient Distributed Datasets(RDDs)

3.3.2 Spark Streaming

Chapter 4

Benchmark Design

4.1 Architecture

4.2 Experiment Environment Setup

4.3 Data Source

4.3.1 Test Data Generation

4.3.2 Kafka

4.4 Experiment Log and Statistic

4.5 Extensibility

Chapter 5

Experiment

5.1 Experiment Environment

5.2 Classic Workload

5.2.1 WordCount

5.2.2 Data Source

5.2.3 Algorithm Description

5.2.4 Results and Discussion

5.3 Multi-Streams Join Workload

5.3.1 Advertisements Click

5.3.2 Data Source

5.3.3 Algorithm Description

5.3.4 Results and Discussion

5.4 Iterate Workload

5.4.1 WordCount

5.4.2 Data Source

5.4.3 Algorithm Description

5.4.4 Results and Discussion

Chapter 6

Conclusions

Summary of experiment results

6.1 Selection in Practice

Summarize several factors which affect selection of stream processing systems in practic

6.1.1 Performance Summary

6.1.2 Issues

6.2 Future Work

Future works

6.2.1 Scale-out and Elasticity Evaluation

6.2.2 Evaluation of Other Platforms

Bibliography

[1] Timothy G. Armstrong, Vamsi Ponnekanti, Dhruba Borthakur, and Mark Callaghan. Linkbench: A database benchmark based on the facebook social graph. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 1185–1196, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2037-5. doi: 10.1145/2463676.2465296. URL `http://doi.acm.org/10.1145/2463676.2465296`.

[2] F Chang, J Dean, S Ghemawat, WC Hsieh, DA Wallach, M Burrows, T Chandra, A Fikes, and R Gruber. Bigtable: A distributed structured data storage system. In *7th OSDI*, pages 305–314, 2006.

[3] Transaction Processing Performance Council. Tpc-c benchmark specification, . URL `http://www.tpc.org/tpcc/`.

[4] Transaction Processing Performance Council. Tpc-h benchmark specification, . URL `http://www.tpc.org/tpch/`.

[5] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[6] Anamika Dey, Alan Fekete, Raghunath Nambiar, and Uwe Rohm. Ycsb+t: Benchmarking web-scale transactional databases. In *Data Engineering Workshops (ICDEW), 2014 IEEE 30th International Conference on*, pages 223–230. IEEE, 2014.

[7] L Doug. Data management: Controlling data volume, velocity, and variety, 2001.

[8] Michael Stonebraker, U?ur Çetintemel, and Stan Zdonik. The 8 requirements of real-time stream processing. *ACM SIGMOD Record*, 34(4): 42–47, 2005.

[9] Kai Wähner. Real-time stream processing as game changer in a big data world with hadoop and data warehouse, 2014. URL `http://www.infoq.com/articles/stream-processing-hadoop`.

Appendix A

Source Code

.1 WordCount

.2 Advertisements Click