A Performance Study on Operator-based Stream Processing Systems

Miyuru Dayarathna∗, Souhei Takeno∗, Toyotaro Suzumura†

∗Department of Computer Science, Tokyo Institute of Technology, Japan
†IBM Research - Tokyo, Japan
dayarathna.m.aa@m.titech.ac.jp, allcoveredac@gmail.com, suzumura@cs.titech.ac.jp

Abstract—This short paper compares and contrasts performance characteristics of System S and S4, two stream processing systems which use operator-based programming model. Our aim is to investigate and characterize which architecture is better for handling which type of stream processing workloads and observe the reasons for such characteristics.

Stream processing [1] has emerged as an exciting new field to support online information processing activities. Currently there is a growing attention towards operator-based stream programming models. We conducted this study on System S [2] and S4 [3] which are currently two of the most prominent systems developed following this model.

The contribution of this work is the identification and characterization of performance and trade-offs of the two different stream processing systems with respect to their underlying architectural design principles. Such result will be useful for system designers and developers to identify what type of stream programming models would suit for their needs.

We use three different stream programs (Application-Specific Benchmarks) which are are used for different purposes such as Call Detail Record Processing (CDR), Volume Weighted Average Price calculation (VWAP) and Twitter trend detection. Then we use a micro-benchmark to get basic characterization of the two systems’ performance.

The results we obtained by running ten different applications (totaling 50 applications considering 1,2,4,8,12 node scenarios) on System S and S4 gave us sufficient insight to what kind of processing happen in the both the systems. It became clear from the throughput comparisons shown on Figure 1 that these two stream programming models should be used carefully to maximize the throughput. E.g. A SPADE program written with single source operator might not scale well in different hardware configurations. Also a S4 application that generates huge numbers of PEs for incoming events cannot scale well. Yet introduction of multiple source operators resulted in a 3.2 times speedup for CDR application on System S and a 1.34 times speed up for reducing 0.1 million Aggregator PEs of S4 to 100 PEs which indicated possible avenues for performance improvements.

While Java based stream processing system architectures are gaining considerable attention due to their portability, system designers and stream programmers have to think carefully before choosing the correct solution. E.g. A light weight input event rate job could be easily processed using S4 with few amount of PEs (E.g. Twitter application run on S4). However a large scale application with high commercial importance such as the VWAP and CDR might produce millions of PEs since S4 dynamically generates PEs for each new data events its receives.

By analyzing the throughput and profiling results we observed that properly designed stream applications result in high throughput. Another conclusion we arrived at is that choice of a stream processing system need to be made carefully considering factors such as performance, platform independence, size of the jobs. Furthermore we understood the importance of key role played by operating system kernel in stream processing system’s performance. While we observed heavy use of network bandwidth by S4, by using optimized protocols and techniques such as Java New I/O, InfiniBand Remote Direct Memory Access the conditions could be improved.

Creating a stream processing system architecture that scales in terms of the number of PEs is a further work that was inspired by this work. In future we hope to extend this work to a micro level performance study on S4, specially to identify which components, code segments are more resource intensive. Moreover we hope to implement Linear Road Benchmark on System S and S4 and observe and characterize the systems’ performance.

REFERENCES

