

StreamBase White Paper

StreamBase versus Native Threading

By Richard Tibbetts
Co-Founder and Chief Technology Officer, StreamBase Systems, Inc.

Motivation for Benchmarking StreamBase

Many StreamBase customers are interested on the relative performance of StreamBase and traditional implementation technologies such as C++ or Java. Some have concerns that the graphical development environment or use of Java will have a negative performance impact. The following benchmark demonstrates how StreamBase's flexible and powerful approach to threading outperforms native threading in both C++ and Java on the kind of analytic workloads found in real-time capital markets applications.

Problem Statement

The benchmark is an implementation of parallelized options pricing in C++, Java, and StreamBase. Given an algorithm specified in Visual Basic, implementations are to be produced in the three systems, and executed over one million data points. The threading and state management is as follows:

- Thread 1 runs the analytical version of Black Scholes and produces output
- Thread 2 runs the Quasi Monte Carlo version of Black Scholes and produces output
- Thread 3 calculates the average of the two, and also calculates a weighted average of this run, and the previous run (so we have a "State")

The result of the benchmark is the elapsed time to process one million calculations. The system is executed on a 64-bit quad core Linux server.

Implementation Details

All implementations derive from the Visual Basic code delivered in the Excel sheet.

■ Pure C++

The C++ implementation of Black Scholes and QMC Black Scholes is a mechanical translation of the Visual Basic code.

The threading is implemented using the boost::thread library from <http://www.boost.org>, a standard portable C++ threading library. Threads 1 and 2 run independently, synchronizing their work via mutex, while thread 3 is the main thread responsible for calculations as well as overall timing.

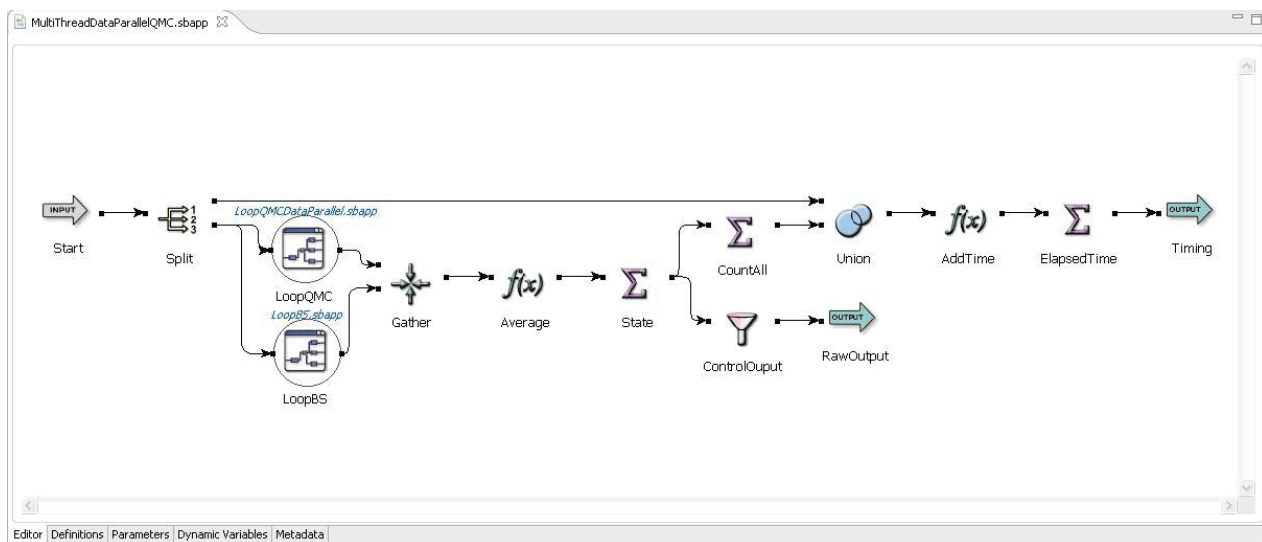
■ Pure Java

The Java implementation of Black Scholes and QMC Black Scholes is a nearly mechanical translation of the Visual Basic code. The only change was to remove the array allocation in the QMC implementation, because the array was unused.

Threading is implemented using standard Java threads, in a similar fashion to the C++ implementation.

The StreamBase Implementation

StreamBase provides a built-in implementation of Black Scholes, but not Quasi Monte Carlo Black Scholes. Best practice in StreamBase development is to use plugins for this kind of purely mathematical operation, and to reserve StreamBase for the threading and state management. Plugins are either custom developed, or drawn from standard libraries like JQuantlib. The StreamBase implementation uses the Java or C++ implementation from above. The built in Black Scholes is not used, for simplicity of results analysis.



Execution Platform

The execution platform is a Dell SC1430 with 4 Xeon 5110 cores running at 1.6GHz with 4GB of RAM. The operating system is 64 bit RHEL 5.

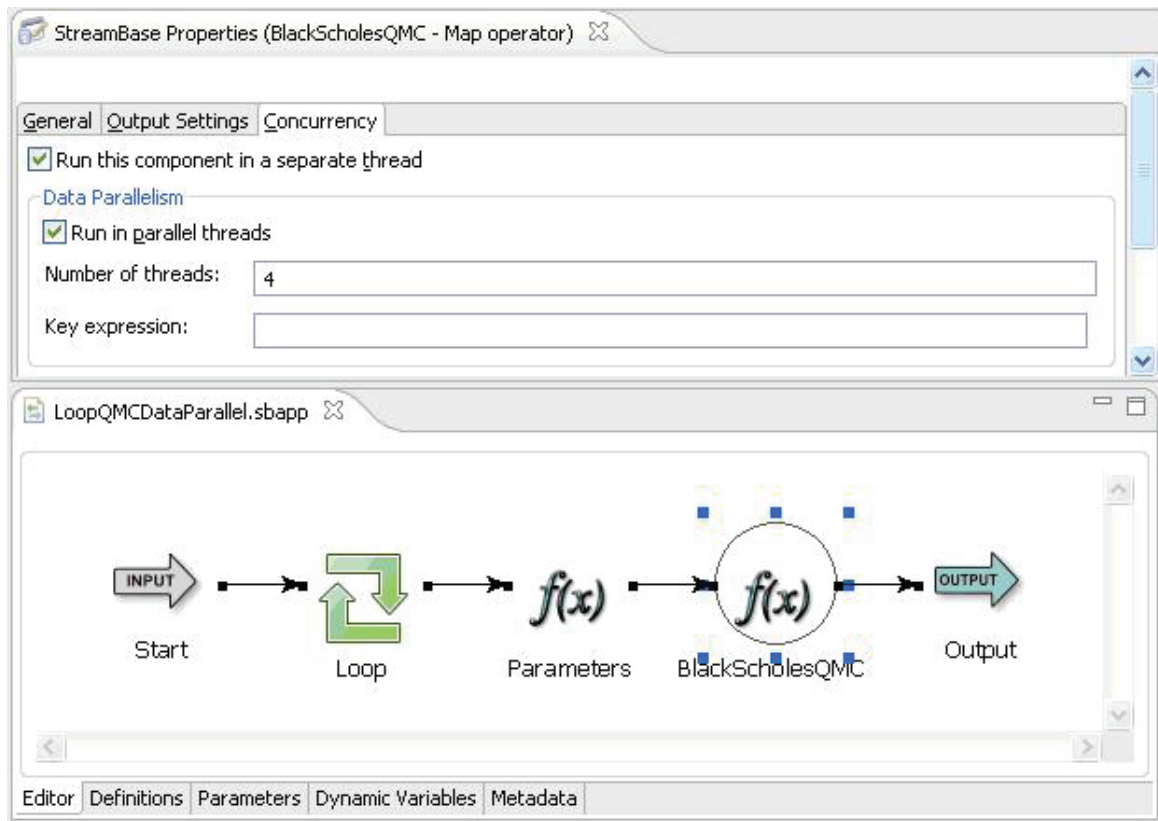
Benchmark Results

The raw results are reflected in the following table. We show the results for full native C++ and Java implementations, as well as StreamBase implementations using either the C++ or the Java plugin functions. All implementations implement the parallelism as described in the specification.

Threading	Functions	Evaluations	Elapsed Time	Evals/sec
Native C++	C++	1,000,000	32 seconds	31,000
Native Java	Java	1,000,000	38 seconds	26,000
StreamBase	C++	1,000,000	41 seconds	25,000
StreamBase	Java	1,000,000	27 seconds	37,000

Problem Statement

Based on analysis of the application performance, the Quasi Monte Carlo (QMC) function is far and away the hotspot in all of these applications. We used the StreamBase functionality for Data Parallelism to dedicate 4 concurrent threads to the QMC function.



This resulted in a substantial speedup:

Treading	Functions	Evaluations	Elapsed Time	Evals/sec
Native C++	C++	1,000,000	32 seconds	31,000
Native Java	Java	1,000,000	38 seconds	26,000
StreamBase Data Parallel	Java	1,000,000	11 seconds	91,000
StreamBase Data Parallel	C++	1,000,000	24 seconds	42,000

Conclusion

This micro-benchmark measures the efficiency of numerical function implementations as well as the efficiency of multi-threaded work sharing. By comparing StreamBase to native threading implementations, we can draw conclusions about both the performance of the platform and the ease of implementation.

The first StreamBase implementation is 20% faster than the Native C++ implementation, and 40% faster than the Native Java implementation. This is most likely due to the superior efficiency of stream-oriented inter-thread communication, and the resulting improved concurrency.

StreamBase stands out not only for the raw performance demonstrated here, but also for the ease with which alternative parallelism structures can be developed and profiled. Having examined the application with a profiler, we discover and demonstrate that the optimal threading structure includes four data-parallel threads dedicated to Quasi Monte Carlo. This implementation is nearly 200% faster than the fastest non-StreamBase threaded implementation.

Five major features explain the performance of StreamBase:

- StreamSQL more naturally expresses event-based applications
- StreamBase compiled executor optimizes whole programs
- Direct bytecode generation delivers superior single-core performance
- Dataflow optimized multithreading drives multi-core performance
- Development tools support identification and optimization of bottlenecks

While this test effectively illustrates one style of event processing application (a highly parallelizable and partitionable computation), it sheds light on only one aspect of CEP product performance. In fact, for sheer bulk computation, single threaded systems are strong contenders, as are other parallel processing systems such as compute clusters. Other use cases in FX aggregation and trading applications more accurately illustrate the performance superiority of a multi-threaded CEP engine, with support for a variety of message formats, management of large state tables, long running aggregations, and multiple kinds of parallelism. For these use cases, the choice of the most flexible platform is going to deliver the best performance on the widest range of applications, in their first deployment and as scale up is required.

About Richard Tibbetts

Richard Tibbetts is co-founder and Chief Technology Officer at StreamBase Systems. Richard provides technical leadership for the company and leads architecture design for StreamBase's Event Processing Platform. Richard is also responsible for furthering new StreamBase capabilities such as StreamBase's 'white-box' application frameworks, for example, the Smart Order Routing Framework. As CTO, Richard directs the next-generation of StreamSQL, the event programming language developed by Richard and the StreamBase team, which applies the benefits of SQL for stored data to real-time transitory data.

About StreamBase

StreamBase Systems, Inc, a leader in high-performance Complex Event Processing (CEP), provides software for rapidly building systems that analyze and act on real-time streaming data for instantaneous decision-making. StreamBase's Event Processing Platform™ combines a rapid application development environment, an ultra low-latency high-throughput event server, and the broadest connectivity to real-time and historical data. Leading investment banks, hedge funds, and government agencies use StreamBase to power mission-critical applications that increase revenue, lower costs, and reduce risk. Applications in Capital Markets include FX Aggregation and Pricing, Smart Order Routing, Market Data Management and Algorithmic Trading. StreamBase customers include CME Group, ConvergEx Group, RBC Capital Markets, PhaseCapital and BlueCrest Capital Management. The company is headquartered in Lexington, Massachusetts with offices in New York, Washington D.C. and London. For more information, visit www.streambase.com.

Corporate Headquarters

181 Spring Street
Lexington, MA 02421

+1 (866) 787-6227

New York Office

845 Third Avenue, 6th Floor
New York, NY 10022

+1 (866) 787-6227

Virginia Office

11921 Freedom Drive, Suite 550
Reston, VA 20190

+1 (443) 472-0767

European Headquarters

60 Cannon Street
London, EC4N 6NP

+44 (0) 20 7002 1095

www.streambase.com (Website) | [@streambase](https://twitter.com/streambase) (Twitter) | streambase.typepad.com (Blog)