



# MONTE CARLO BLACK-SCHOLES ALGORITHM BENCHMARK

AUGUST 2007

## Executive Summary

The Gedae Development Environment (Gedae DE) for the Cell Broadband Engine (Cell/B.E.) processor is a platform that allows programmers to leverage the exponential power of a new generation of multi-core processors, including the Cell Broadband Engine. In this technical note, we compare and contrast the performance of the Monte Carlo Black-Scholes (MCBS) algorithm running on a CPU, a dual-core CPU and the Cell Broadband Engine. We demonstrate a 300x speed-up using the Gedae DE. The hardware platforms compared are an Intel Core 2 Duo Processor and an IBM QS20 Blade Server.

## Monte Carlo Black-Scholes

The Monte Carlo Black-Scholes model is used in quantitative financial analysis to value options. The Black-Scholes formula is an equation of the market for an equity, in which the equity's price is determined via a stochastic or random process. A Monte Carlo model is a method of estimating a value by the random generation of numbers. When used in conjunction random numbers are generated and fed into the Black-Scholes model. This model is executed repeatedly and the aggregate results are used to price the option.

## Benchmark Description

In this technical note, we present results on an implementation of an MCBS algorithm. The implementation used for the bench mark is based on a European option.

Both the standard Monte Carlo and Quasi-Monte Carlo versions of the Black-Scholes were implemented. The difference between the two is in the random number generator. Quasi-Monte Carlo uses a deterministic low-discrepancy sequence such as the Hammersley function, while a regular Monte Carlo method is based on sequences of pseudorandom numbers. As a basis for our discussion we are referring to the standard Monte Carlo version.

We also compare a traditional serially programmed CPU implementation of an MCBS with a Gedae implemented

version on a multi-core processor.

MCBS is composed of 4 stages: random number generation, stochastic formula, summing of results and computation of the call value.

The following systems were used for this test:

System	Processor	Cores	Compiler	Library
A	Intel Core 2 Duo T7400 <sup>€</sup>	1	Generic C++	None
B		2	Intel C++ 9.1	MKL*
C	Cell/B.E.	1 PPE/ 16 SPE	SPU-XLC <sup>‡</sup>	SDK <sup>‡</sup> / SAL <sup>†</sup>

<sup>€</sup> 2.16 GHz

\* Intel Math Kernel Library v.9.1

<sup>‡</sup> IBM Cell SDK 2.1

<sup>†</sup> Mercury Computers SAL from MultiCore Plus 1.1

## Benchmark Results

The results achieved for the MCBS on the Cell Broadband Engine are significant. The table below lists the results for the QMCBS and the MCBS running on the Cell/B.E.

System	MC	Speedup	Quasi-MC	Speedup
A	4.6 M/s	1 x	5.0 M/s	1 x
B	92.8 M/s	20 x	110.6 M/s	22 x
C	1410 M/s	361 x	2156 M/s	471 x

M/s : million experiments per second

The table above compares the Gedae execution times for the baseline, system A, to the dual-core on system B and the Cell Broadband Engine on system C. Both systems B and C use optimized libraries and compilers.

While the performance is very good on system B, the Gedae/Cell/B.E. combination is 18x faster and 361x faster than the baseline implementation.

## Programming Effort

Stage	Task	Ph	M/s
0	Initial implementation	3.0	-
1	Fully optimize vector library	2.0	1219
2	Collopase into one for-loop	5.0	1410

Ph : programmer hours

### Stage 0

In stage 0 the initial implementation of the algorithm was constructed. The algorithm was then tested on a workstation before being moved to a the Cell Broadband Engine.

### Stage 1

Using iteration parameters and auto-subscheduling to reduce memory footprint the algorithm was mapped to 16 SPU's. The Trace Table was used to identify inefficiencies in the original specification, and two key improvements were made to the implementation to enhance performance. For the majority of the operations Mercury Computers SAL functions were automatically used while the Cell SDK random number generator was used for the standard Monte Carlo simulation. Utilizing optimized functions for the algorithm the performance on the Cell/B.E. was 1219 M/s.

### Stage 2

In this final stage effort was made to approach theoretical maximum performance for this algorithm. In order to achieve maximum efficiency custom functions were created to chain or fuse calls to vector routines from the Cell SDK in one for-loop. Intermediate buffering was removed by storing all intermediate results in registers and loop unrolling was used to minimize pipeline dependency stalls. With these final optimizations the theoretical maximum performance of 1410 M/s was achieved.

## Analysis of Effort and Results

Given the small amount of effort needed to acheive these outstanding performance levels Gedae is clearly capable of performing financial algorithms on the Cell Broadband Engine processor. Gedae allows developers with no experience programming the Cell/B.E. architecture to quickly and easily implement applications on the processor and still achieve spectacular speedup.

As evidenced by these results Gedae provides high performance and reduced time to market for applications like the MCBS and QMCBS. Gedae excels at providing those same benefits for more complex applications with heavy interprocessor communication and behavior too rich to be encapsulated in one for-loop.

## Key Features of Gedae Utilized

The key Gedae features utilized in order to acheive these outstanding results include:

- use of E Library for performance
- language features to break a large # of experiments into 8 K batches
- partitioning and mapping of SPUs, easy scalability through the constants table
- use of Trace Table to measure performance and identify items to optimize

## Get Started Today

Gedae and IBM have collaborated on a quick start package that includes a PS3 pre-loaded not only with the Gedae software, but with the IBM SDK and Fedora Core Linux OS. The package includes an accelerated training program and sample programs.

Contact us today to start realizing the benefits of this powerful combination.

---

© 2007 Gedae Inc. All rights reserved. Patents pending. Gedae is a trademark of Gedae Inc. in the United States and/or other jurisdictions. All other markets and names mentioned herein may be trademarks of their respective companies.