

Fixstars White Paper Series



Cell/B.E.を利用したモンテカルロ・シミュレーションの高速化

2008年6月
株式会社フィックスターズ

1. イントロダクション

デリバティブ評価やリスク管理の場で、モンテカルロ・シミュレーションによる分析を行う機会が増えている。しかし、モンテカルロ・シミュレーションは計算負荷が高いため、実際の業務で活用するには、試行回数や実行頻度を制限せざるを得ない。モンテカルロ・シミュレーションの高速化手法として解析的近似法があるが、ポートフォリオの構造・特性によっては近似精度が悪化することが知られている。一方、ハードウェア業界ではプロセッサのマルチコア化が進んでおり、1チップでスーパーコンピュータ並みの演算性能を持つプロセッサが開発されている。このようなマルチコアプロセッサを使用することで、ポートフォリオの構造・特性や近似精度を考慮する必要の無い、汎用的な高速化を実現することができる。本稿では、モンテカルロ・シミュレーションによるリスク (Value at Risk、VaR) 計量を、マルチコアプロセッサであるCell Broadband Engine™¹(Cell/B.E.)を用いて高速化する手法を紹介する。

2. プロファイリングと移植方針

肥後[1]で使用されたプログラムをgprofツール²でプロファイリングした結果 (表 1参照)、実行時間の90%近くが正規乱数の生成に費やされていることがわかった。したがって、Cell/B.E.で高速化するには、正規乱数生成を並列化することが必要である。乱数列は本質的にデータの依存性が無いため、複数のプロセッサコアでの並列処理に向いているといえる。

一方、金融機関におけるデータ分析では検証可能性が重要なため、Cell/B.E.での処理結果とIntelでの処理結果とが同じ値になることが要求される。そのためには、Cell/B.E.とIntelとで乱数列の使用順序を揃える必要がある。

上記の検討を踏まえ、本開発では以下の3点を移植方針とした。

- A) 検証可能性：Intelの結果と同じ結果を生成する「互換モード」と、実行速度を重視した「高速モード」とを実装する。
- B) 精度と速度：精度に留意しながら、Cell/B.E.が得意とする単精度浮動小数点数を用いる。
- C) 規模拡張性：債務者数、試行回数に制約を設けない。

関数名	実行時間全体に占める消費時間の割合 ³	関数の処理内容
main	93.4% ⁴	データの読み込みから結果の書出しまでの全ての処理
pcNrand	88.2%	標準正規乱数の生成。Mainが呼び出す。
genrand_real3	34.0%	(0, 1)に含まれる一様乱数の生成。 pcNrandが呼び出す。
genrand_int32	21.6%	[0, 約43億]に含まれる一様乱数の生成。 genrand_real3が呼び出す。

表 1：モンテカルロ・シミュレーションにおける消費時間の多い関数

¹ Sony、TOSHIBA、IBMの3社によって共同開発されたマルチコアプロセッサ。1チップで256Gflopsという非常に高い浮動小数点演算能力を持つ。

² プログラム内で実行される関数ごとの消費時間を計算するツール。

³ その関数が呼び出す関数の消費時間も含む。

⁴ main関数の外で、プロセスの初期化や後処理が6.6%の時間を消費していると考えられる。

3. 実装の詳細

Cell/B.E.の特徴として、①並列演算を得意とする8つのプロセッサコア（SPU）、②4つの浮動小数点小数を並列に処理するSIMD演算、③SPUと独立に動作するメモリコントローラ（MFC）、の3点が挙げられる。本実装では、①処理を試行回数で分割して8つのSPUで並列処理、②SIMD命令を用いて4件の債務者データを並列処理、③MFCを活用して演算とデータ転送とを並列処理、の3種類の並列処理を併用することで高速化をはかった。

なお、互換モードでは乱数列の使用順序をIntelと揃える必要があるため、試行ごとに必要となる乱数は事前に一括に生成し、必要に応じてSPUが参照するように実装した。また、各試行において債務者ごとに必要になる乱数は各SPUが独立に生成するが、適切な数の乱数を捨てる処理を追加した。

4. 精度に関する検討

精度に影響を与える要素として、単精度と倍精度の違いと丸めモードの違い⁵との2つが考えられる。単精度の最小精度は 1.2×10^{-7} なので、与信額には同程度の丸め誤差を含むことになる。ただし、与信額が小さい債務者から順に処理することで、丸め誤差の影響を単精度の誤差内に抑えることができる。

丸めモードの違いによってデフォルト判定が逆転する確率を単精度の最小精度で近似すると、「債務者数×試行回数×最小精度」回発生することになる。たとえば、債務者数10,000、試行回数300,000で99.9% VaRを求める場合、デフォルト判定の逆転は約360回発生すると考えられるが、これが、期待損失の上位300（試行回数の0.1%）に現れると、VaRの値に影響を与えることになる。

5. 結果

同じデータ（債務者数10,000、試行回数300,000）を用いて、Cell/B.E.（2.8GHz）の互換モード、高速モードでモンテカルロ・シミュレーションを行った結果を図1に示す。Xeon® 3GHz⁶、Core™2 Duo 1.86GHz⁷との比較では10～30倍程度速度が向上した。

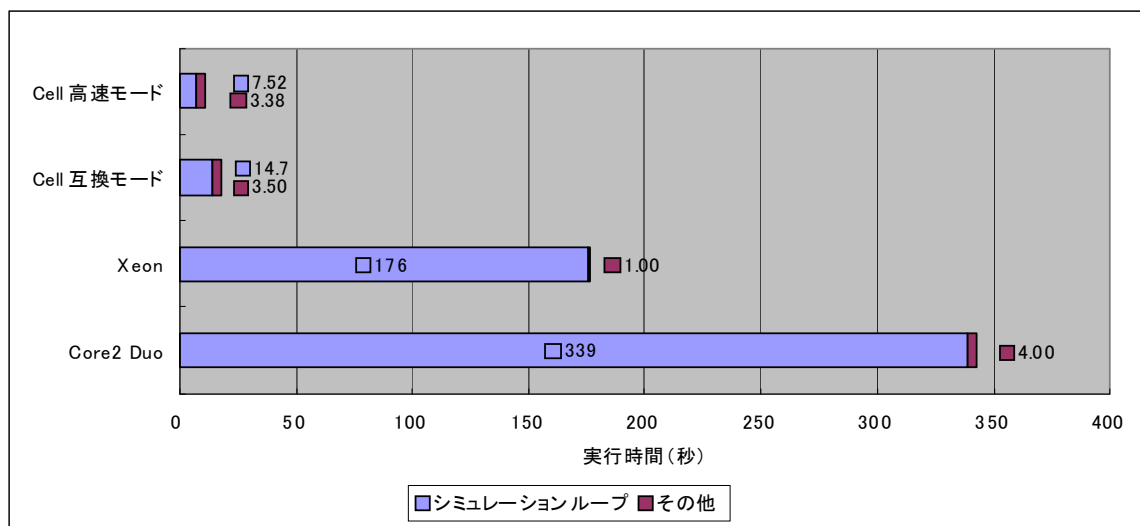


図1：シミュレーションの実行時間（債務者数10,000、試行回数300,000）

債務者数、試行回数が大きくなると、シミュレーションループ部の実行時間が増大するため、他の部分の実行時

⁵ 現在のCell/B.E.の実装では、SPUの浮動小数点演算の丸めモードはゼロへの丸めであり、Intelなどが採用している最近偶数丸めと異なる。

⁶ IBM IntelliStation® Z Pro 9228AC1 に搭載した Dual Intel® Xeon® Processors 5160、3.0 GHz。

⁷ Dell Dimension 9200C に搭載した Intel® Core™2 Duo E6300、1.86GHz。

間は無視できる。シミュレーションループ部のみの比較では、12~45 倍の向上率となる。

比較対象	高速モード		互換モード	
	全体	ループのみ	全体	ループのみ
Xeon	16.2 倍	23.4 倍	9.73 倍	12.0 倍
Core2 Duo	31.5 倍	45.1 倍	18.8 倍	23.1 倍

表 2 : 実行時間の比較

また、このモンテカルロ・シミュレーションの結果をXeonと比較したところ、高速モードでは1~2%の差が発生したが、互換モードでは 10^{-7} 程度の差に抑えられた(表3)。高速モードと互換モードとの違いは乱数列の使用順序だけなので、上記の1~2%の差は、このシミュレーションでVaRが十分に収束していないことを意味している。従って、試行回数を増やして値を十分に収束させれば、高速モードにおいてもXeonとの差を小さくすることが可能だと思われる。

	Xeon	互換モード		高速モード	
		数値	誤差	数値	誤差
99% VaR	314.66987	314.66977	-3.2×10^{-7}	310.54929	1.3×10^{-2}
99.9% VaR	502.93106	502.93100	-1.2×10^{-7}	492.52649	2.1×10^{-2}

表 3 : シミュレーション結果の Xeon との比較 (債務者数 10,000、試行回数 300,000)

6. まとめ

モンテカルロ・シミュレーションによる VaR の計量を Cell/B.E.を用いて高速化した結果、債務者数 10,000、試行回数 300,000 のシミュレーションで 10~45 倍の速度向上を達成し、誤差は $10^{-2} \sim 10^{-7}$ となった。実務での利用を検討するにあたっては、債務者数や試行回数の大きさ、必要な精度などに応じて最適な実装は変わるが、Cell/B.E.を用いる利点は十分にあると考えられる。なお、倍精度浮動小数点演算能力を向上した Cell/B.E.チップが発売されており、倍精度を用いることも検討すべきだろう。

7. 参考文献

- [1]肥後秀明「不均一な与信ポートフォリオのリスク計量におけるモンテカルロ・シミュレーションの効率化」日本銀行ワーキングペーパーシリーズ 2006 年 (<http://www.boj.or.jp/type/ronbun/ron/wps/wp06j18.htm>)