GPU Monte Carlo Simulation for VaR Calculation

Xuefei He
Manchester Business School

26 November, 2010

Monte Carlo simulation in finance

- The most general, powerful, and important numerical method for derivative pricing (simple option example)
  - Underlying stock: $d \log S = (\mu - \frac{1}{2}\sigma^2)dt + \sigma dX$
  - Option payoff at T is equal to $\max(S_T - K, 0)$
  - One path: $\log S_{\Delta t,i} = \log S_0 + (\mu - \frac{1}{2}\sigma^2)\Delta t + \sigma\sqrt{\Delta t}\epsilon_i(0,1)$
  - Option price at now is $P = e^{-rT}\frac{1}{N}\sum_{i=0}^{N} \max(S_{T,i} - K, 0)$
- Time consuming — 10,000 to 100,000 paths due to slow convergence rate $(\frac{1}{\sqrt{N}})$
- Suitable for GPU — independent paths and no frequent data transfer between host and device

Value at risk (VaR)

- VaR — maximum loss value in next M days with $X\%$ confidence level
- VaR of the example option
  1. Volatility σ is the only risk factor.
  2. How much will σ change in next M days? — distribution of σ $(\sigma_1, \sigma_2, ..., \sigma_n)$
  3. Distribution of price P $(P_1, P_2, ..., P_n)$
  4. The X percentile is VaR.

GPU implementation steps

- Start on device
- For every Δt, generate random number
- Update stock price
- At T, calculate payoff
- Calculate the average of discounted prices
- Retrieve the result P from the device

Any easy-use library?

- Random number generator and summation
- Easy to install without sudo password