IOCMA 2026 Conference
The 2nd International Online Conference on Mathematics and Applications
10–12 June 2026 | Online
MDPI

An Optimized Accelerator for Option Pricing Using Monte Carlo Simulation on a GPU

Sani Abba, Gloria Nuhu Magit
Department of Computer Science, Faculty of Computing, Abubakar Tafawa Balewa University, Gubi campus, P.M.B. 0248, Bauchi state, Nigeria.

INTRODUCTION & AIM

The Monte Carlo simulation is a numerical method that produces its results through repeated random sampling. It's a technique used to model the different types of uncertainty that impact the value of the investment or underlying instrument under consideration [1-2]. This study used the Monte Carlo method on a GPU to investigate and assess the Arithmetic Asian Option. By utilizing the GPU's built-in parallelization capability, the study was able to accelerate pricing options more effectively and decisively when compared to the CPU implementation. It also evaluated the GPU's power consumption and optimized it for greater efficiency. The effectiveness of the method was examined by utilizing the various paths that were constructed using a randomized function to test this assertion. Ten paths (samples) with ranges between 100,000 and 1,000,000 were taken into consideration. The GPU was utilized to enhance performance in terms of speedup, computation time, power consumption, and time complexity. The GPU's power utilization is then further optimized to improve performance. A detailed experiment using GPU and quad-core multiprocessor systems was conducted. The optimized accelerator for the GPU was coded using the CUDA programming language, libraries, and Application Programming Interface (API).

The contributions of the research work are as follows:

1. The use of Monte Carlo simulation as a numerical computing algorithm that computes its outcome through repeated random sampling. It's a technique used to model the different types of uncertainty that impact the value of the investment or underlying instrument under consideration.
2. Applied the GPU's natural parallelization capabilities to price Asian options and investigated the computing efficiencies of GPU systems compared to a multi-threaded CPU implementation.
3. Investigated the speedup in the pricing of the Asian option, which lowers the impact of volatility and can be immediately implemented by a financial organization.
4. Evaluate power consumption used in comparison with the CPU and optimize the power for more efficiency.
5. To optimize and evaluate the GPU in pricing options and improve performance.
6. This study would provide financial institutions with a practical, efficient, and affordable technique for pricing the Asian option based on the Monte Carlo approach that they may utilize to help with decision-making and increase profitability in stock market price forecasting.

METHOD

Analyzing mathematical model-based trading models requires the computation of a particular integral, which can be valued analytically or computed using a partial differential equation (PDE). However, when the number of dimensions in the problem is large, PDEs and numerical integrals become complex. Due to its slow rate of convergence and high computational intensity, the Monte Carlo simulation is to be executed on a GPU to accelerate this computational task [2].

This system, using the Monte Carlo simulation, builds models of possible stock prices by substituting a range of values (a probability distribution) for the option. It then calculates results over and over, each time using a set of random values from the probability functions. In this case, the size ranges from 100,000 to 1,000,000. The Monte Carlo simulations involve tens of thousands of recalculations before each iteration is complete to generate the stock price, which in turn has a significant effect on the execution time, time complexity, and cost in terms of power consumption.

The arithmetic Asian option

This option requires that the payoff be calculated using the arithmetic averages of the prices over the life span of the option [1].

$$\text{Arithmetic }(0,T) = \frac{1}{n}\sum_{i=1}^{n} S(ti) \quad (1)$$

Where:
S (*ti*) = Spot price at time t
n = Number of equally distributed sample points
t = Time of maturity

Simulating the underlying Asset

The option is modelled as follows:

$$dS_t = \mu S_t dt + \sigma S_t dW_t \quad (2)$$

To stimulate the underlying asset, we must discretize the underlying changes, i.e., transform the continuous variables into a discrete form. Using the Euler method,
This is a finite difference,

$$Y^{n+1} - Y^{n} = \mu Y^{n} \Delta t + \sigma \Delta W_t \quad (3)$$

Where:
μ: is the expected return per year
σ: is the expected volatility per year (i.e., the degree of variation of a trading price series over time)
T: is the time to maturity
S_t and S_{t-1} are the current and previous prices.
dW: is a random number distributed according to a normal distribution with mean 0 and variance.
dt: (This is the Brownian motion component)
Y: is the price at the time step *n*, where $Y_0 = S_0$

RESULTS & DISCUSSION

This research work computes option prices using a Monte Carlo simulation on a GPU, compared with a CPU. The proposed algorithm's effectiveness and efficiency are assessed using scenarios with small and large sample paths (data points) by analyzing the outcomes across several metrics. The study, in turn, reports performance Speedups, Computation time, Power consumption, and Time complexity. In addition, the GPU is further optimized for power consumption to improve performance.

Experimental setup procedure

The number of paths in a set of ten, ranging from 100,000 to 1,000,000, was to be considered. For each of these paths, the implemented algorithm (program) is run. An arithmetic Asian call option was computed with the following parameter settings:

✓Initial price = 35
✓The strike price is 27
✓Time to maturity is one year (365 days)
✓Risk-free interest rate 0.05% (i.e., less inflation rate from yield)
✓Volatility is 0.4% (i.e., fluctuations in the market price of the underlying asset)
✓ The steps are 365 (i.e., one year) and were considered and implemented.

Scenario 1: Sample Paths (100,000 and 200,000 Datapoints)

A relatively modest sample path of 100,000 was used to demonstrate the technique's capability to handle smaller problems Figure 1. Another relatively modest sample path of 200,000 was used to demonstrate the technique's capability to handle smaller problems Figure 2.

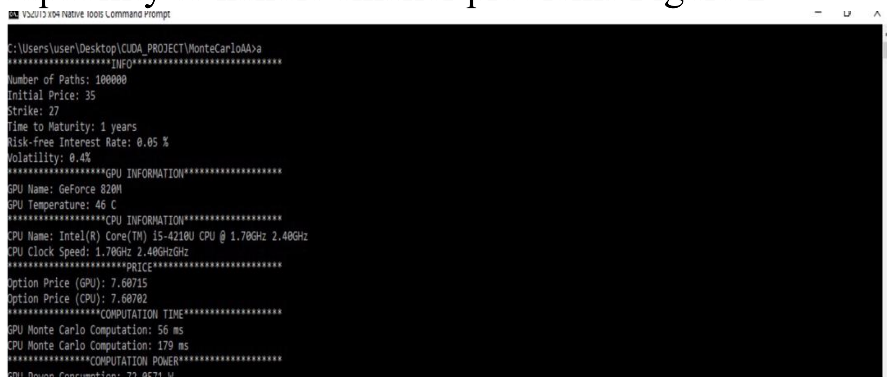

Figure 1: Experimental results of the option price of 100,000 sample paths (datapoints)

Figure 2: Experimental results of the option price of 200,000 sample paths (datapoints)

Scenario 2: Sample Paths (900,000 and 1, 000, 000 Datapoints)

In this scenario, a sample path of 900,000 and 1,000,000 was used to conduct this experiment, as demonstrated in Figures 3 and 4, respectively.

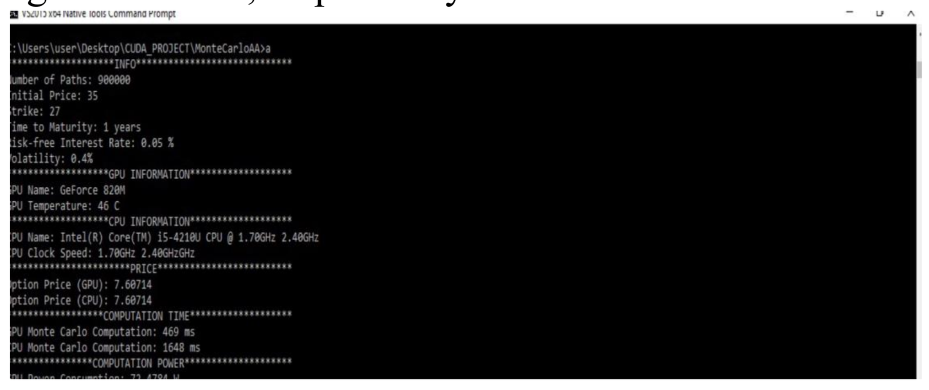

Figure 3: Experimental results of the option price of 900,000 sample paths (datapoints)

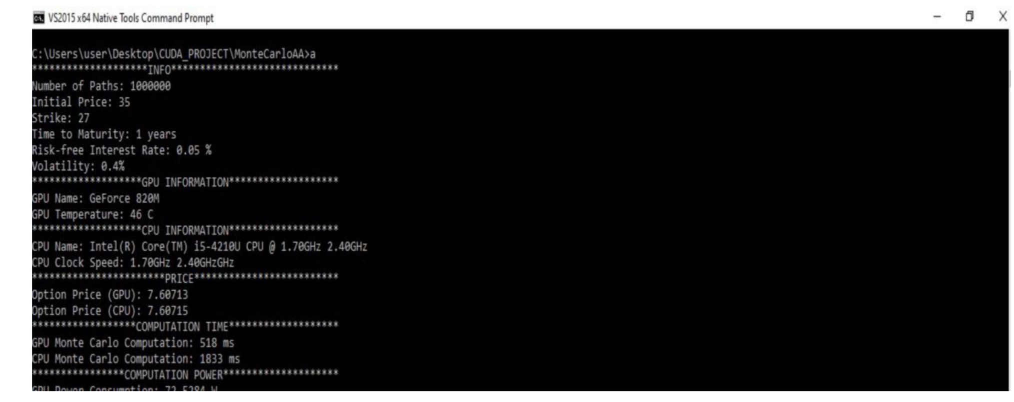

Figure 4: Experimental results of the option price of 1,000,000 sample paths (datapoints)

Computation time

Figure 5 demonstrates that the GPU typically takes less time to compute the option price than the CPU. This is anticipated given that the GPU's capacity to do concurrent computations, indicating that the GPU has advanced computation abilities that speed up the data computation process beyond what the CPU can handle at a given time.

No of Paths(datasets)'000'	100	200	300	400	500	600	700	800	900	1000
CPU 'ms'	196	375	551	743	935	1099	1278	1458	1673	1850
GPU '	57	112	167	224	279	325	374	422	470	517

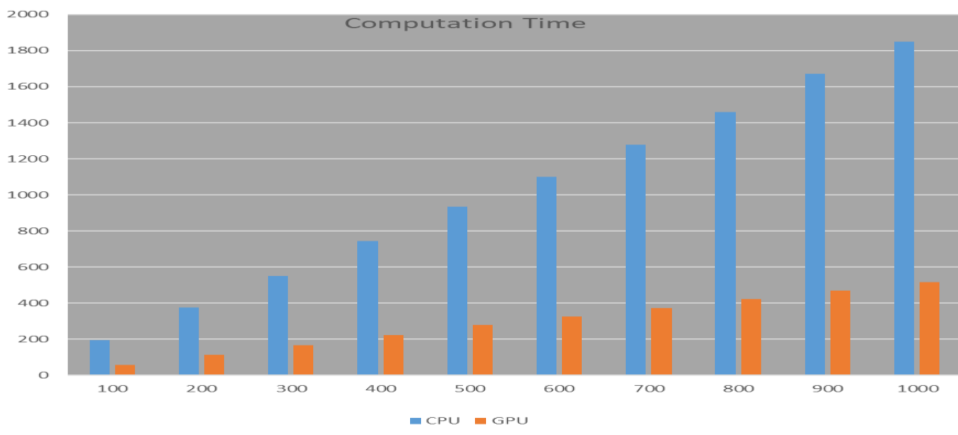

Figure 5: Experimental results of the computational time between the CPU and GPU on different sample paths from 100 to 1,000.

CONCLUSION

The experimental findings showed that computing the option price on the GPU generally requires less time than on the CPU. Given the GPU's ability to perform concurrent computations, this is expected. The results of the power consumption analysis also demonstrated that during computing, the CPU often consumes more power than the GPU. Additionally, when comparing the GPU to the CPU, the GPU lowered the algorithm's runtime as a function of input length when the algorithm's time complexity was measured. Investigated the speedup in pricing the Asian option, which lowers the impact of volatility and can be immediately implemented by a financial organization. This study would provide financial institutions with a practical, efficient, and affordable technique for pricing the Asian option based on the Monte Carlo approach that they may utilize to help with decision-making and increase profitability.

FUTURE WORK / REFERENCES

1. Hsu C-C, Lin C-G, Kuo T-J. Pricing of Arithmetic Asian Options under Stochastic Volatility Dynamics: Overcoming the Risks of High-Frequency Trading. *Mathematics*. 2020; 8(12):2251. https://doi.org/10.3390/math8122251
2. Claus Jespersen "Monte Carlo Evaluation of Financial Options using a GPU" *A thesis presented for the degree of Master of Science, Computer Science Department, Aarhus University, Denmark 02-02-2015.*
3. Simon Suo, Ryan Attridge, Ruiming Zhu, Justin Wan "GPU Option Pricing", *Proceedings in the 8th workshop on High-Performance Computational Finance Conference, Austin, Texas, USA, 15 November 2015.*