Theme:

Parallelizing Monte Carlo Simulation of Financial Derivatives using SYCL

Team Name:

Breakfast Banana

Team Members:

Surya Prakash S

Sakthi Vel T

intel

TECHGIG

1 oneAPI

oneAPI CodeMaven

Theme Chosen and Motivation

The theme we have chosen is Parallelizing Monte Carlo Simulation of Financial Derivatives using SYCL.

- Monte Carlo simulation is a computationally expensive process used to model complex systems with a lot of uncertainties, like business risk analysis.
- SYCL is a framework for optimizing computations through the use of specialized hardware and parallel execution.
- Using SYCL we can optimize the Monte Carlo simulation to perform calculations much faster and accurate.
- The Intel One API's SYCL implementation enables simulation to be run on a variety of hardware platforms, including CPUs, GPUs, and FPGAs from different vendors, making it more scalable.

Proposed Solution

The Goal of this project is to predict stock prices. The condition provided to us is to use Monte Carlo simulation.

- We used geometric Brownian motion to simulate the stock price. This method uses two factors
  - Drift : Moves the new price path based on the previous trends of stock returns
  - Shock: The factor that shocks the drift curve to simulate random ups and downs in the price path.

$$\frac{\Delta S}{S} = \mu \Delta t + \sigma \epsilon \sqrt{\Delta t}$$

The first part of the right-hand side of the equation is drift and second part is shock

Tech Stack Used:

- Geometric Brownian motion for stock prediction.
- SYCL for parallel calculation of samples.
- Sciplot for displaying result (based on gnuplot)
    - Sciplot requires gnuplot to be installed on the system

Sciplot

Output from our programs is a graph of price paths. There are not many header only library to perform plotting in C++. Sciplot is our best plotting library for our task. It is simple, fast and the plots generated are very easy to understand. Writing output to a PDF file using sciplot is much faster than any format from any other tool.

It requires a system with gnuplot installed to work.

Proposed Tech Architecture

In this project, we use an implementation of SYCL provided by Intel’s OneAPI HPC toolkit.

- SYCL is a framework that provides a consistent programming language across that can be used to program CPU, GPU, FPGA, and AI accelerators like TPUs in a heterogeneous framework where each architecture can be programmed and used either in isolation or together.
- The drift and shock values are calculated parallelly for each and every day of the sample.
- The price path is also calculated parallelly for each price path.
- This Drastically improved the performance.

BenchMarks

CPU : AMD Ryzen 5600H (laptop cpu)
GPU: Nvidia RTX 3060 Laptop GPU

Sample	Days	SYCL-GPU(s)	SYCL-CPU(s)	CPU(s)
1000	252	1.2	1.38	3.55
10000	252	1.3	1.43	40
50000	252	1.46	4.19	177.19
100000	252	1.67	8.44	355

The number of sample and days calculation done in the program is 4 times the above samples Because In our program we are calculating for four different stocks with the above sample and days value.

Impact of inclusion of oneAPI

- Good amount of samples is required for better accuracy. Calculating large amount of samples is computationally expensive if it calculated one by one.
- So, we harnessed the power GPU to calculate samples in parallel, since each sample calculation is independent of each other.
- From the benchmark, we can understand that even when running on CPU, SYCL provides better performance by using integrated GPU.
- The maximum I reached with Nvidia GPU is
    - 700000 samples with 1000 days steps
    - which is (700000*1000)*4 calculations (for four different stocks)
    - completed in 10.59 seconds!
    - There is memory bottleneck for going above the numbers.

Innovation Quotient and Scalability

- The algorithms we have chosen can be executed in a parallel fashion. So usage of SYCL will improve the performance.
- We use the compiler provided by intel HPC toolkit.
- This compiler compiles C or C++ code that's compatible with SYCL, which can be run on other accelerator languages or frameworks specific to a particular hardware.
- This allows the compiled program can run on hardware from different vendors, making it both horizontally and vertically scalable. We can always add more GPUs, using Intel's SYCL we can scale the app to distribute work between devices.

Further Improvements

- We can improve this project by implementing following:
  - Fetch stock price data using a third party API
  - Nice user interface for visualization
  - Real-time GBM calculations
  - Calculating more stats with metrics like CAPM returns, Sharpe factor etc.,

Quick Summary

- Our project simulates stock prices by Monte Carlo simulation technique using Geometric Brownian movement.
- We optimize Simulation using SYCL by taking samples parallelly

A Quick thanks for taking a look into our project :)

intel®