High Performance Counterparty Risk and CVA Calculations in Risk Management

Dominique Delarue
Azim Siddiqi
20th March 2013
Overview

- Counterparty Risk Primer
- The Problem
  - Scale
  - Scale Explosion
- The Solution
  - Challenges
  - Pricing Ecosystem
  - Breaking the Cube
- Moving to Kepler
- Summary
Counterparty Risk for Dummies

(1) Market Parameter Simulation
- FX rates, interest rates, equities, commodities, credit spreads and ratings

(2) Revaluation of deals
- mostly full reval
- ‘Norms’ for exotics

(3) Aggregation & Collateral Simulation
- by Global Credit Line

(4) Front-End Analysis Tools
- risk monitoring
- credit adjustments
- stress analysis

Deal data

Market Scenarios

Market data & Calibration parameters

Netting, Collateral & Referential data

BNP PARIBAS Group Risk Management
March 2013
Monte Carlo Process

- Simulation process (‘Monte Carlo’)...
  - Paths (values for different future dates) are simulated for all the market parameters that influence the value of the transaction.
  - Correlation against other market parameters (IR and FX) is taken into account during the simulation process.

- Re-pricing of the transaction...
  - The interest rate swap value is recalculated for the future dates of all the scenarios.
  - In some scenarios, the value will be positive (typically when the interest rates increase), in others, the value will be negative.
For all counterparties and for each scenario

\[ 1,000,000 \times 5,000 \times 400 = 2 \text{ trillion calculations} \]

(8TB of floats)

…and 500 TB of simulations were consumed in this process

But that’s ONE scenario. We need HUNDREDS !!!
Problem – Scale Explosion

- Regulatory Requirements
  - More stress testing
  - Detection of Wrong Way Risk
    (e.g. stressing Turkish Lira to assess impact on Turkish banks)

- Risk Management
  - Exposure Explained
    Incremental Stress Scenarios for Complex Stress Scenario
  - Add/Remove deals from counterparties
Problem – Scale Explosion

- CVA Calculation is simple in itself

\[ CVA_{\text{Regulatory}} = LGD_{\text{MKT}} \sum_{i=1}^{T} \max \left( 0, e^{\frac{S_{i-1}^{\text{MC}} - E_{i-1}^{\text{MC}}}{LGD_{\text{MKT}}}} - e^{\frac{S_{i}^{\text{MC}}}{LGD_{\text{MKT}}}} \right) \times \frac{EE_{i-1} \cdot D_{i-1} + EE_{i} \cdot D_{i}}{2} \]

- But CVA needs to be hedged

10 Shocks Model \times 40 currencies = 400 Stress Scenarios
you're gonna need a bigger boat
Solution - Challenges

- Current Counterparty Risk System
  - CPU Compute Farm (a few thousand cores)
  - Low level of Parallelization
  - C++ Code

- Objective
  - Do at least 50 x more for the same cost

- Re-think of the whole solution
  - Parallelization on multiple levels (counterparty, deal, instrument, market data, time point, path)
  - Re examining data, calculation and latency problems
Solution - Overview

- REST Services
- Compute Engine
  - CPU Servers
  - GPU Servers
- Persistent Data
- In-Memory Data

(BNP PARIBAS Group Risk Management)
Solution - Calculation Engine

- Java
  - Negligible performance penalty compared to C++
  - Development Productivity
  - Maturity of Java for large applications

- JNA (Java Native Access)
  - Integration of native libraries and java
  - Stubs generated by the C# module
  - No measurable footprint

- NIO (New IO)
  - Market Data is managed outside the JVM (native)
  - Manage dozens of GB outside the JVM heap
Solution - Beyond Hybrid

- Single File Source Code
- Two Operating Systems
- Multiple Runtime Platforms

Debugging Environment

Production Environment

Intel® AVX

Microsoft .net

Designed for NVIDIA CUDA
Solution - Pricing Ecosystem

- Microsoft Visual Studio
- Pricer File (.Net)
- Microsoft CL
- .Net assemblies
- Hybridizer
- CUDA source
- OMP source
- NVCC

DSL Tool
- C++ Wrapper functions
- XLL Wrapper functions
- Java Wrapper
- Java Test file
- Test data

- Data Recorder
- Microsoft Excel
- Pricing Example

- GCC / Microsoft CL / Intel ICC
- Javac

- Shared Object (linux)
- Native DLL (Windows)
- XLL
- Jar
- Test Jar
Solution - Pricing Ecosystem

- High Level Language for pricing and simulation
  - C# used

- Rationale
  - Higher Productivity (Managed language)
  - Learning Curve (Different level of abstractions)
  - Integration (Java, Excel, Python)

- Hybridizer™ Technology
  - Product from Altimesh (Florent Duguet)
  - Generate C++/OpenMP and Cuda/C code from C#
Example: Fixed Cash Flow

\[ PV(i,j) = \sum_{k}^{nb\ of\ CF} \text{Notional}_k \times \text{YearFraction} \times \text{Rate}_k \times D_j(t_i,T_k) \]

```java
public class FixedCashFlowsPricer {
    // Pricer Data
    public int cashFlowCount;
    public float[] paymentDate;
    public float[] value;

    // Market Data
    public YieldCurve yieldCurve;

    public void GridPV(PricerResult result, int simFrom, int simTo, int timePointFrom, int timePointTo) {
        int timePoint, simId;
        float price, simulDate;

        for (timePoint = timePointFrom + blockIdx.x; timePoint < timePointTo; timePoint += gridDim.x) {
            simulDate = result.getTimePointDate(timePoint);
            for (simId = simFrom + threadIdx.x; simId < simTo; simId += blockDim.x) {
                price = 0.0f;
                for (int k = 0; k < this.cashFlowCount; k++) {
                    if (simulDate >= paymentDate[k]) continue;
                    price += value[k] * yieldCurve.GetDiscountFactor(simId, timePoint, paymentDate[k] - simulDate);
                }
                result[simId, timePoint] = price;
            }
        }
    }
}
```
Solution - Pricing Ecosystem

- Work distribution is done using cuda concepts
  - Blocks : maps to OMP threads or cuda blocks
  - Thread : maps to FPU register unit and/or cuda-threads
- C# to CUDA
  - Use of different types of memory (shared, global, registers)
  - Access to library functions and existing code snippets
- C# to OMP
  - Basic C/C++ code generation
  - Code Vectorisation (SSE and AVX)
Back to the Cube Problem

Don’t slice the cube, break it up.
Breaking the Cube up

Netted Set

- In-Memory Aggregation
- Higher Accuracy

Pricer

- Maximize Kernel Work
- Higher Accuracy

Simulation

- Bandwidth CPU and GPU Memory
- Bandwidth GPU Memory and Cuda Cores

Huge Gain in Compute and Bandwidth

Maximize parallelization

Huge Gain in Compute and Bandwidth
Calculating Big Counterparties

- LCH Clearing House
  - Mainly IR Swaps
    - 161,025 deals, 5.5 M. Cash Flows
  - Cube
    - $161,025 \times 1,000 \times 400 = 64,4$ Billion calculations

- Calculation
  - 161,025 deals
  - 1,000 simulations
  - 400 time points
  - 1 M2090

- Performance
  - Overall (262s)
  - T0 Pricing (150s)
    - 1,075 deals/s
  - MC Pricing (49s)
    - 3,300 deals/s
    - 113,000 CF/s
Moving to Kepler

■ Current Hardware
  ■ Dual NVIDIA M2090
  ■ Running large calculations
  ■ Production like environment

■ Continuous Approach
  ■ Can constantly test and verify
  ■ Not afraid of changing code/trying things out
  ■ High level of confidence in our approach

➢ We have been able to try out Kepler
  BUT our initial tests did not match expected performance

What was the limiting factor?
How bound are you?

Problems are split into two categories:

- **Compute Bound**
  - Limited by the compute power

- **Memory Bound**
  - Limited by the cuda cores to gpu memory bandwidth
How bound are you?

Problems are split into two categories:

- **Compute Bound**
  - Limited by the compute power

- **Memory Bound**
  - Limited by the cuda cores to gpu memory bandwidth

And a more subtle one…

- **Latency Bound**
  - Limited by the latency between cuda cores and GPU memory
Collaboration with Nvidia
- John Ashley and Thomas Bradley (and his team)
- Nvidia identified our problem was latency bound

LDG
- Simple use of Texture Memory (from memory address)
- Keyword __ldg
- Impact on pricing code

Vectorization on GPU
- Use of float4 instead of float
- Allow to pack memory reads into one request (128 bits)
- Impact on pricing code (much heavier, on going work)
LDG, float4 on Fermi and Kepler

Fermi 1.75x 1.16x Kepler 3x 1.87x

16 LOADS – 8 STORES 4 LOADS – 2 STORES

ALU MEMORY READ
SFU MEMORY WRITE
### Calculation Results

\[ PV(i, j) = \sum_{k}^{\text{nb of CF}} \text{Notional}_k \times \text{YearFraction} \times \text{Rate}_k \times D_j(t_i, T_k) \]

- **Fixed Cash Flow Pricing**
  - 500 cash flows
  - 1,000 path

<table>
<thead>
<tr>
<th></th>
<th>time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVX (dual E5-2640)</td>
<td>40.3</td>
</tr>
<tr>
<td>CUDA (M2090)</td>
<td>7.5</td>
</tr>
<tr>
<td>CUDA (K20)</td>
<td>4.0</td>
</tr>
</tbody>
</table>

x5.3
Is the boat bigger?

- Nightly runs
  - 12x more runs than current system
  - Cost is 5x less than current setup
  - $60x$ more calculations for the same cost

- And the boat can get even bigger…
  - Less sensitive to the number of deals
  - Scale linearly with number of instruments
  - Reuse of data for different instruments
Summary

- Counterparty Risk and Stress Scenario is A Big Problem

- Had to rethink the solution

- Use a hybrid technology stack

- The Cube is broken, not sliced

- Continuous approach
Summary

- Kepler can allow us to run 2x as many scenarios as Fermi
- Kepler investigation has helped us gain new insights
- Can now do massive parallel computations
- Well positioned to leverage future improvements in processor technology
Thank You

QA
No.17 European group
all sectors⁽¹⁾

190,000 employees
representing more than
150 nationalities

Present in nearly
80 countries

€94.4bn
of consolidated equity
as at 31/12/2012

23 million clients
and more than 7,000 branches
across its retail network

A leading global bank

⁽¹⁾ Based on the Group’s FY11 net income, a European ranking in the Stoxx 600 index universe
Group Risk Management

Risk Coverage

- Credit risk
  - Corporate
  - Retail
- Market & Counterparty risk
- Liquidity risk
- Insurance risk

Operational Risk and Reputation risk are placed under the responsibility of Conformity

Main Missions

- To advise Bank Management on risk policy definition
- To contribute as a “second pair of eyes” that risks taken on by the Bank align with its policies
- To report and alert Bank Management of the status of risks to which the Bank is exposed

Risk Cycle

- Risk Policy
- Risk analytics and modeling
- Counterparty & transaction analysis
- Anticipation
- Approval
- Portfolio Analysis: Capital, Concentrations, Stress-testing
- Monitoring & Control
- Reporting

BNP PARIBAS
March 2013