Objective

• Develop understanding of what “Computational Finance” means

• Learn who uses Computational Finance
  – Why do the users need it?
  – What do they do with the information?

• Take a look at some typical algorithms

• Consider the challenges and benefits of adoption
AGENDA

- Financial Instruments
  - Who’s Who
  - Modeling
  - Algorithms
  - Adoption
  - Summary
Financial Instruments

Cash Instruments
- Equities
- Commodities
- Fixed Income
- Foreign Exchange

Derivatives
- Exchange-traded
- Over-the-counter
Equities

- Share ownership
- Value determined by market
- Dividends
Commodities

• Raw resource
  – Agriculture: corn, rice
  – Livestock: pork bellies
  – Energy: oil, gas
  – Metals: precious, industrial

• Supply and demand
Fixed Income

- Also: Credit
- Loans and bonds
- Different rates according to duration
Foreign Exchange

• Also: Forex or FX
• Take advantage of changes in rates
Derivatives

• Based on one or more underlying assets
  – Equities, FX, credit
• Many types of contract
  – Forwards and futures
  – Options
  – Swaps
• Exchange-traded or Over-the-Counter (OTC)
Example: Options

- Holder has the right to buy (*call*) or sell (*put*) the underlying asset
  - By a certain date
  - At a certain price

\[
Payoff_{put} = \max(K - S, 0)
\]

where \( K = \) strike price

\( S = \) spot price
AGENDA

- Financial Instruments
- Who’s Who
- Modeling
- Algorithms
- Adoption
- Summary
Traders

• Trading:
  – Standardised instruments
  – New (often complex) instruments

• Requires models:
  – Pricing
  – Prediction
  – Risk analysis
Backoffice

• Monitoring the banks exposure
• Model all trades
  – Value
  – Risk
• Value-at-Risk (VaR) required for regulation
Quants

- Develop and implement models for traders
- Develop independent models for validation
- Research
- Modelling exposure and capital
Developers

- Implement models from quants
- Integrate into larger applications
  - Interface to other models
  - Interface to database
  - Interface to user
AGENDA

- Financial Instruments
- Who’s Who
- Modeling
- Algorithms
- Adoption
- Summary
Inputs/Outputs

Instrument parameters

Market data

Value
Confidence
Sensitivities
Traders

• Determine price for trade
  – Negotiate over the phone, require results fast
  – Minimize out-trades (errors)

• Run positions
  – Sensitivities allow trader to predict response to changes in underlying assets
  – Run often to allow trader to react quickly
Backoffice

• Manage risk and capital reserves for regulation
  – Large runs can take days to complete
  – Accurate results allow greater control, and hence more trades

• Monitor traders’ exposure
  – Run intra-day

• Greater accuracy requires longer run times
AGENDA

- Financial Instruments
- Who’s Who
- Modeling
- Algorithms
- Adoption
- Summary
Analytic

• Some derivatives have an analytic solution

\[
\frac{\partial f}{\partial t} + r S \frac{\partial f}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} = rf
\]
Black - Scholes Formula

\[
f = Ke^{-rT}\Phi(-d_2) - S_0\Phi(-d_1)
\]
\[
d_1 = \frac{\ln(S_0/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}
\]
\[
d_2 = \frac{\ln(S_0/K) + (r - \sigma^2/2)T}{\sigma\sqrt{T}} = d_1 - \sigma\sqrt{T}
\]
Price for European Put Option

• Compare analytical result with numerical result
  – Provides a Control Variate
Binomial Trees

• Represent possible paths of stock

• Assumptions:
  – Stock has a probability $p$ of moving up by a certain percentage $u$
  – Stock has a probability $(1-p)$ of moving down by a certain percentage $d$
Binomial Trees

• Construct the tree
  – Create a branch for each time step
  – At each node the stock can either go up u% or down d%
Binomial Trees

<table>
<thead>
<tr>
<th>Stock price</th>
<th>Derivative payoff</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u^5 S$</td>
<td>0</td>
</tr>
<tr>
<td>$u^4 dS$</td>
<td>0</td>
</tr>
<tr>
<td>$u^3 d^2 S$</td>
<td>0</td>
</tr>
<tr>
<td>$u^2 d^3 S$</td>
<td>$K - u^2 d^3 S$</td>
</tr>
<tr>
<td>$u d^4 S$</td>
<td>$K - u d^4 S$</td>
</tr>
<tr>
<td>$d^5 S$</td>
<td>$K - d^5 S$</td>
</tr>
</tbody>
</table>
Binomial Trees on the GPU

• Work backwards in time
  – Compute value at each node

• Partition the work across the SMs
  – Overlap input data
  – Fit input partition in smem
Finite Differences

• Solve the Partial Differential Equation iteratively
  – Divide the life of the derivative into equal intervals of length $\Delta t$
  – Divide the range of stock prices $[0, S_{\text{max}}]$ into equal intervals of size $\Delta S$
• Work backwards in time, compute the value at each node
Finite Differences - Implicit

- Relationship:
  - Three values at $t$ and one value at $t + \Delta t$
- Always converges to solution
- Requires solving simultaneous equations
Finite Differences - Explicit

- Relationship:
  - One value at $t$ and three values at $t + \Delta t$
- Compute nodes in parallel
- Can diverge from solution
Explicit Finite Differences on the GPU

• Partition the grid across the SMs
Explicit Finite Differences on the GPU

- Partition the grid across the SMs
- Each SM requires a “halo”
  - Halo size determines how many time steps in batch
Explicit Finite Differences on the GPU

- Partition the grid across the SMs
- Each SM requires a “halo”
  - Halo size determines how many time steps in batch
  - After each batch, distribute new halos
Monte Carlo

- Sample a random walk for the asset(s)
- Calculate the payoff of the derivative
- Repeat to get many sample payoff values
- Calculate the mean payoff
Monte Carlo

![Graph showing asset price and payoff over time.](image-url)
Monte Carlo on the GPU

Generate Random Numbers
- Distribution
- Covariance

Compute payoff for each path
- Scenarios are independent

Compute statistics
- Reduction
Monte Carlo - Multiple Kernels

- RNG: parfor [0..T) for [0..N) genrand()
- Paths: parfor [0..N) for [0..T) genpath()
- Payoffs: parfor [0..N) payoff()
- Reduce: for [0..lg_2N) reduce()
Monte Carlo - Single Kernel

parfor [0..N)
for [0..T)
genrand()
adddtopath()
payoff()
reducestep()

for [0..\log_2 N)
reduce()
AGENDA

- Financial Instruments
- Who’s Who
- Modeling
- Algorithms
- Adoption
- Summary
Software legacy

• Millions of lines of code
• Complex relationships between code blocks
  – E.g. Primary algorithm generates paths which are reused in multiple payoff models
• Hundreds of man-years of work
• Significant refactoring of application required
  – Support/feed the parallelized algorithms
Education

• Parallel programming is paradigm shift
  – Quants are starting to rethink algorithms
  – Designing or reusing different algorithms/strategies

• Reuse of libraries and frameworks
  – Concentrate on the core algorithm
  – Increasing number of libraries for random numbers, linear algebra, reduction etc.
## Case Study: Equity Derivatives

<table>
<thead>
<tr>
<th></th>
<th>15x Faster</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Tesla S1070</td>
<td>16x Less Space</td>
<td>500 CPU Cores</td>
</tr>
<tr>
<td>$24 K</td>
<td>10x Lower Cost</td>
<td>$250 K</td>
</tr>
<tr>
<td>2.8 KWatts</td>
<td>13x Lower Power</td>
<td>37.5 KWatts</td>
</tr>
</tbody>
</table>

Source: BNP Paribas, March 4, 2009
Case Study: Equity Derivatives

- 15x Faster
- 16x Less Space
- 10x Lower Cost
- 13x Lower Power
- 190x Lower Power in Total
- 15
- 2 Tesla S1070
- $24 K

No need to compromise

Source: BNP Paribas, March 4, 2009
## Case Study: Real-time Options

<table>
<thead>
<tr>
<th></th>
<th>Same Performance</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3 Tesla S870</td>
<td>9x Less Space</td>
<td>600 CPU Cores</td>
<td></td>
</tr>
<tr>
<td>$42 K</td>
<td>6x Lower Cost</td>
<td>$262 K</td>
<td></td>
</tr>
<tr>
<td>$140 K</td>
<td>9x Lower Annual Cost</td>
<td>$1,200 K</td>
<td></td>
</tr>
</tbody>
</table>

Figures assume:
- NVIDIA Tesla S870s with one 8-core host server per unit
- CPUs are 8-core blade servers; 10 blades per 7U
- $1,800/U/month rack and power charges, 5-year depreciation
Case Study: Security Pricing

<table>
<thead>
<tr>
<th>2 hours</th>
<th>8x Faster</th>
<th>16 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>48 Tesla S1070</td>
<td>10x Less Space</td>
<td>8000 CPU Cores</td>
</tr>
</tbody>
</table>

Bloomberg

Source: Wall Street & Technology, September 24, 2009
Conclusion

• Computers are used to model financial instruments for price, sensitivity and risk
• Algorithms include Finite Differences and Monte Carlo
• Parallelising algorithms requires structural support from the application
  – Benefit is substantial on all measures
  – GPUs are transforming the industry
• Opportunities for algorithmic development
Resources

• GTC presentations
  – Finance presentations, Thursday from 2pm, Atherton Room
  – 3D Finite Differences on GPU, Friday 10.30am, Empire Room
  – Tridiagonal solvers on GPU, Friday 2pm, Atherton Room

• SDK examples
  – binomialOptions, 3DFD, MonteCarlo/MonteCarloMultiGPU

• NVIDIA finance page (links to online resources)