Efficient PDE solving in Option Pricing on Multicore and Manycore Platforms

**Option Pricing**

- An **Option** is a tradable financial security whose value depends on the value of the underlying asset.
- Task of **Option Pricing** is to find the price of the option given the underlying asset and market conditions.
- The Black-Scholes equation for option value \( V \) is:

\[
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + r S \frac{\partial V}{\partial S} - rV = 0
\]

Where \( S \) is underlying stock price, \( \mu \) – its expected rate of return, \( \sigma \) – its volatility (deviation) and \( r \) – riskless rate of return.

- To solve the Black-Scholes equation:
  - Transform to heat diffusion equation
  - Compute using finite difference methods
  - Discretization & Stencil Computation

- Explored parallelization strategies & results for Intel Core i7 (Nehalem), Larrabee and Nvidia GTX280 GPU.

**Crank-Nicolson Method**

- An average of explicit and implicit finite difference methods
- Take one half-step using explicit method, then another half-step using implicit method

\[
(1 + \alpha) u_{j}^{n+1/2} - \alpha (u_{j+1}^{n} + u_{j-1}^{n}) = (1 - \alpha) u_{j}^{n} + \frac{1}{2} \alpha (u_{j+1}^{n} + u_{j-1}^{n})
\]

- Stable for any discretization step size
- Fast convergence

- Implicit method – **Gauss-Seidel** iterative solver
  - **Explicit data dependency** between updates

**Parallelization**

- **Pricing one option:**
  - **Unroll** the iteration loop in Gauss-Seidel solver to expose SIMD parallelism

**Results**

**Overall Breakdown of Computation**

<table>
<thead>
<tr>
<th>Platform</th>
<th>Core i7</th>
<th>Larrabee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speedup</td>
<td>2.13</td>
<td>6.08</td>
</tr>
</tbody>
</table>

**SIMD Scaling (Intel Platforms)**

- Problem size is too small to yield good performance on the GPU
- Poor platform utilization (max 16-wide SIMD)
- Algorithmic innovation needed to get good performance on GTX280

**Core Scaling**

- Crank-Nicolson Scaling on Nehalem
- Crank-Nicolson Scaling on Larrabee

**Future Work**

- Algorithmic innovation for the GPU implementation
- Data structure transformation optimization
- Variation on the iterative solver (Jacobi)