Nonbonded forces
Force Function (~90% compute time):

\[
F_i (r_{ij}) = \left( \frac{1}{4\pi\varepsilon_0} \frac{q_i q_j}{\epsilon r_{ij}^2} + 12 \frac{C_{12}}{r_{ij}^{12}} - 6 \frac{C_6}{r_{ij}^6} \right) \frac{r_{ij}}{r_{ij}}
\]

Electrostatics  Lennard-Jones
What we do in the inner loop?

For each i atom {
    fetch atom i data
    i_force = 0;
    For each j atom in our neigborlist {
        fetch atom j data
        calculate vectorial distance; \( \mathbf{dr} = \mathbf{r}_i - \mathbf{r}_j \)
        calculate \( r^2 = dx^*dx + dy^*dy + dz^*dz \), and \( 1/r = 1/\sqrt{r^2} \)
        Calculate potential and vectoral force
        Subtract the force from the j atom force
        i_force += force;
    }
    Store i_force;
}
What we do in the inner loop?

\[
d1 = \text{imolecule} - \text{jmolecule1}; \\
d2 = \text{imolecule} - \text{jmolecule2}; \\
d3 = \text{imolecule} - \text{jmolecule3}; \\
d4 = \text{imolecule} - \text{jmolecule4}; \\
\text{rinv.x} = 1.0 / \sqrt{\text{dot} (d1, d1)}; \\
\text{rinv.y} = 1.0 / \sqrt{\text{dot} (d2, d2)}; \\
\text{rinv.z} = 1.0 / \sqrt{\text{dot} (d3, d3)}; \\
\text{rinv.w} = 1.0 / \sqrt{\text{dot} (d4, d4)}; \\
\]

Compute 4 force interactions in parallel

\[
\text{rinvsq} = \text{rinv} \times \text{rinv}; \\
\text{rinvsix} = \text{rinvsq} \times \text{rinvsq} \times \text{rinvsq}; \\
\text{vnb6} = \text{rinvsix} \times \text{c6}; \\
\text{vnb12} = \text{rinvsix} \times \text{rinvsix} \times \text{c12}; \\
\text{vcoul} = \text{qq} \times \text{rinv}; \\
\text{fs} = (\text{twelve} \times \text{vnb12} - \text{six} \times \text{vnb6} + \text{vcoul}) \times \text{rinvsq}; \\
\]

Requires 2x computation

\[
\text{t1} = d1 \times \text{fs.x}; \\
\text{t2} = d2 \times \text{fs.y}; \\
\text{t3} = d3 \times \text{fs.z}; \\
\text{t4} = d3 \times \text{fs.w}; \\
\text{force} = \text{prev} + \text{t1} + \text{t2} + \text{t3} + \text{t4}; \\
\]
Preliminary Results

- Radeon X800 Platinum
Research Goals

• Working with Eric Darve, Vijay Pande
  – Exploring different ways force kernel can be written
    • Is 2x compute with less write bandwidth cheaper than 1x compute with scatter add?
    • Understanding bandwidth vs. compute
Research Goals

• Working with Eric Darve, Vijay Pande
  – Understanding precision in Gromacs
    • Reciprocal Square Root:
      – Accurate to 7 decimal places in 0 to 1 range
      – How does this effect the result?
    • Sin & Cosine required for bonded forces
    • 32-bit, 24-bit, 16-bit floating point math
      – Where can we use lower precision?
Research Goals

• Working with Eric Darve, Vijay Pande
  – Pushing a Protein
    • Entire simulation lives on the GPU
      – Bonded & Non-bonded forces
      – Integrator
    • Understanding where the time is spent
      – 1/3 of time for H2O simulation is unaccounted for.
Protein Docking

“Molecular surface recognition: Determination of geometric fit between proteins and their ligands by correlation techniques.” Katchalski-Katzir, et al.
Molecular representation

• similar to iso-surface representation

\[
\overline{a}_{l,m,n} = \begin{cases} 
1 & \text{on the surface of the molecule} \\
\rho & \text{inside the molecule} \\
0 & \text{outside the molecule,}
\end{cases}
\]

and

\[
\overline{b}_{l,m,n} = \begin{cases} 
1 & \text{on the surface of the molecule} \\
\delta & \text{inside the molecule} \\
0 & \text{outside the molecule,}
\end{cases}
\]

• where \(\rho \ll 0\) and \(1 > \delta > 0\)

June 1, 2004
Computing the fit

• Hold molecule A still, translate B
• Repeat for all orientations

\[
\bar{c}_{\alpha,\beta,\gamma} = \sum_{l=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} \bar{a}_{l,m,n} \cdot \bar{b}_{l+\alpha,m+\beta,n+\gamma},
\]

where \{\alpha, \beta, \gamma\}: shift vector
\(c\): score
Sign difference in \(\rho\) and \(\delta\) penalize overlap
Computing the fit

- Pick the highest score
Using the frequency space

**Convolution Theorem:** Multiplication in the frequency domain is equivalent to convolution in the space domain.

\[ f \otimes g \leftrightarrow F \times G \]

- **Space Domain:**

\[ \bar{c}_{\alpha, \beta, \gamma} = \sum_{l=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} \bar{a}_{l,m,n} \cdot \bar{b}_{l+\alpha, m+\beta, n+\gamma}, \]

- **Frequency Domain:**

\[ C_{o,p,q} = A_{o,p,q}^* \cdot B_{o,p,q}, \]
Using the frequency space

(i) derive $\vec{a}$ from atomic coordinates of molecule $a$ (Eq. 2),
(ii) $A^* = [\text{DFT}(\vec{a})]^*$ (Eq. 4),
(iii) derive $\vec{b}$ from atomic coordinates of molecule $b$ (Eq. 2),
(iv) $B = \text{DFT}(\vec{b})$ (Eq. 4),
(v) $C = A^* \cdot B$ (Eq. 5),
(vi) $\vec{c} = \text{IFT}(C)$ (Eq. 6),
(vii) look for a sharp positive peak of $\vec{c}$,
(viii) rotate molecule $b$ to a new orientation,
(ix) repeat steps (iii–viii) and end when the orientations scan is completed, and
(x) sort all of the peaks by their height.
Performance

- **Direct method:**
  - \[ c_{\alpha, \beta, \gamma} = \sum_{l=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} a_{l,m,n} \cdot b_{l+\alpha, m+\beta, n+\gamma}, \]
  - \( N^3 \) multiply adds for each \( N^3 \) shift: \( O(N^6) \)

- **FFT method:**
  - \( N \times N \times N \) FFT costs \( O( N^3 \ln(N^3) ) \)
  - Much faster!