Gromacs on GPU

Ian Buck
Eric Darve
Vidya Rangasayee
Gurjeet Singh
Vishal Vaidyanathan
Gromacs

• Molecular dynamics code for protein and biomolecule simulation.
• The fastest MD code available.
• SSE, Altivec and 3D-Now! instructions are used for speed-up.
• Several computational inner loops have been rewritten for efficiency:
  – Removal of several ‘if’ statements using separate lists.
  – Use of fast ‘custom’ invsqr
Molecular dynamics simulation

• Algorithm is extremely simple.
• Newton’s equations are integrated using the simplest symplectic integrator: velocity Verlet.

\[ \vec{a} = \frac{\vec{F}}{m} \quad \Rightarrow \]

\[ \vec{r}(t + \Delta t) = 2\vec{r}(t) - \vec{r}(t - \Delta t) + \frac{(\Delta t)^2}{m} \vec{F}(t) \]

• Most of the cost goes into the force calculation.
Force functions in MD

**Non bonded Forces**
- Pair wise additive forces. Scale as $N^2$ unless special technique is used.
  - Electrostatics
    - Short range
    - Long range correction (reaction field, PME, fast multipole method)
  - Van der Waals

**Bonded Forces**
- Approximates complex quantum chemical interactions between atoms. Scales as $N$
  - Bond stretching
  - Bond bending (angles)
  - Bond torsion (dihedrals)
  - Chirality (improper dihedrals)
The different components of the code

- Non bonded forces
  - water water forces using simple cutoff
  - protein protein forces using tabulated forces
  - water water forces using tabulated forces
  - protein water forces using tabulated forces

- Bonded forces
  - Angles
  - Dihedrals

- Periodic boundary conditions

- Verlet integrator

- Constraints
  - SETTLE (water)
  - SHAKE (protein)

- Technically straightforward, but important for real physical simulations
  - Temperature/Pressure Coupling
  - Energy bookkeeping, virial calculation
  - Center of mass motion removal

Complete

In progress
Long range forces

- Electrostatic + Lennard-Jones
  \[
  \frac{q_i q_j}{r_{ij}}, \quad 4\varepsilon_{ij} \left( \left( \frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left( \frac{\sigma_{ij}}{r_{ij}} \right)^6 \right)
  \]

- Computationally very expensive
- System is periodically repeated.
- In general, every atom interacts with all other atoms and all its images.
- In order to simplify this calculation, interactions at a distance larger than a cut-off distance are ignored.
- Neighbor list is used to store list of atoms within the cut-off distance.
Gromacs time stepping loop on GPU

1. Init
2. Do Neighbor Search
   - If done
     - Y: Finish
     - N: Read x from GPU
       - If Neighbor Search
         - Forces on GPU
           - Update on GPU
             - Constraints on GPU
               - Thermostat / Pressure Coupling on GPU

How are the long range forces calculated?

- Pure SIMD execution.
- No scatter-add.
- To minimize cost of reduction of forces for a given atom, each kernel execution calculates interactions between one atom and 16 neighbors.
- Padding with dummy atoms and replication are used.
- Reduction of compute time by using vector capability of GPU: 4 interactions are computed simultaneously.

List of interactions
Pad and expand
Final reduction

• Final reduction is required but no scatter-add.
• Take the maximum number of times an atom has been repeated: $r_{\text{max}}$.
• Loop $r_{\text{max}}$ times over atoms: if partial force is found add it, otherwise skip.

<table>
<thead>
<tr>
<th>Atoms Number</th>
<th>List 0</th>
<th>List 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>No data</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>
Long range force options

• Several options have been implemented:
  – Straightforward force.
  – Smooth transition at cutoff
  – Tabulated forces: interpolation from a table.
    • Allows for a simple implementation of transition region
    • Simplifies implementation of different force fields.
  – Reaction field as first order correction to model an infinite system (vs. finite cutoff).
Implementation of constraints in gromacs

- Constraints are used to freeze certain degrees of freedom such as chemical bonds.
- This allows increasing the time step significantly. Arguably gives a more accurate partition function at room temperature because bond vibrations are quantized.
- Derivation of equation with Lagrangian formalism.
- The trajectory of the particle is that which minimizes:

\[ S = \int_{t_1}^{t_2} L(t) \, dt = \int_{t_1}^{t_2} (E_{kin}(\vec{r}) - E_{pot}(\vec{r})) \, dt \]

- This results in the following equation of motion:

\[ \frac{d}{dt} \frac{\partial L}{\partial \vec{\dot{r}}} = \frac{\partial L}{\partial \vec{r}}, \quad \text{where} \quad L = E_{kin}(\vec{r}) - E_{pot}(\vec{r}) \]
Incorporating constraints

• Example of a constraint: in the water molecule the distance between the 2 OH bonds and the HH bond are constrained.

• The modified Lagrangian is:

\[ L^{\text{cons}} = L - \sum_k \lambda_k \sigma_k, \quad \text{where} \quad \sigma_k = r_{ij}^2 - d_k^2 \]

• The second term is due to the presence of constraints with unknown Lagrangian factors (\( \lambda \)'s)

• The new equation of motion is:

\[ m \ddot{r}_i = \frac{\partial L^{\text{cons}}}{\partial \dot{r}_i} = \ddot{F}_i - \sum_k \lambda_k \frac{\partial \sigma_k}{\partial \dot{r}_i} \]
Verlet Integrator

• Using the Verlet algorithm the position of the atom \( i \) at time \( t + \Delta t \) is

\[
\vec{r}_i(t+\Delta t) = 2\vec{r}_i(t) - \vec{r}_i(t-\Delta t) + \frac{\Delta t^2}{m_i} \left( \vec{F}_i - \sum_k \lambda_k \frac{\partial \sigma_k(t)}{\partial \vec{r}_i} \right)
\]

• We can find the unconstrained trajectory by ignoring the last term.
• To find the constrained positions of the atoms, we need to calculate \( \lambda \).
• That can be done by expanding \( \sigma \) and solving a matrix equation, but that is computationally very intensive and turns out not to be very accurate.
• Two approaches are used in practice:
  – SETTLE (exact – water only)
  – SHAKE (approximate – general).
SETTLE

• Used for 3 atoms and 3 constraints, such as water.

• Produces analytically correct solution
  No iterative process is needed

• Trivial Brook Implementation using kernels:
  – Input: molecule and set of constraints
  – Output: molecule with updated atom positions.
Shake

- Consider a case with two atoms, with initial distance $d_{\text{init}}$
- After one step of the Verlet integration, the unconstrained distance is $d_{\text{mid}}$
- We need to apply a constraint force to reset the distance between the two atoms to the bond length by adding the following displacement to each atom

$$\Delta \vec{r}_i \approx \frac{1}{m_i} \frac{\mu(d_{\text{init}}^2 - d_{\text{mid}}^2)}{2(\vec{d}_{\text{init}} \cdot \vec{d}_{\text{mid}})} \vec{d}_{\text{init}}$$

Here $\frac{1}{\mu} = \left(\frac{1}{m_i} + \frac{1}{m_j}\right)$ is the reduced mass
Shake Algorithm: example

Before SHAKE

Step 1 in SHAKE

Step 2

Step 3

Bond at next step

Initial bond

$d = 1.1894957$

$d = 1.0153361$

$d = 1.0001184$

$d = 1.0000000$
Shake Algorithm

- All atoms in the system are moved using the Verlet algorithm, assuming an absence of rigid bonds.
- Looping over bonds, the deviation from the constrained length is used to calculate the constraint force.
- After the correction has been applied to all bonds in turn, every bond length is checked. If the largest deviation found exceeds the desired tolerance, the correction calculation is repeated.
- Above steps are repeated until all bond lengths satisfy the convergence criterion.
- This is an iterative process because:
  - Each individual equation is non-linear and is solved approximately.
  - Equations are coupled so that a correction applied to a bond will modify the length of all bonds who share one atom.
Shake in Brook – split lists

• Since we operate in parallel on a stream, it is not possible for a given atom to appear more than once in a given bond list.

• Example:

• Therefore, we need to create several lists to go through all the constrained bonds.

• In organic systems, the maximum number of such split lists is 4. This number is obtained by considering the maximum number of bonds for a given atom (valency).
Shake in Brook – split lists

For example, in the following chain, we can process the similarly color coded atoms together.

The numbers above the bonds represent the constraints.
Shake in Brook – split lists

- We need to determine which atom went to which lists so that its position can be updated.
- An inverse map is used for each list.
- A single inverse map is not possible since an atom may belong to several inverse maps (reduction).

<table>
<thead>
<tr>
<th>Atom#</th>
<th>Rlist1</th>
<th>Rlist2</th>
<th>Rlist3</th>
<th>Rlist4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>-1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>-2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>-2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>-2</td>
<td>0</td>
</tr>
</tbody>
</table>
Shake in Brook - update

• For each list
  • Streaming loop: for each atom
    • If reverse[atom] == 0
      out_pos = in_pos
    • If reverse[atom] >0
      out_pos = in_pos + diff[reverse[atom]]
    • If reverse[atom] <0
      out_pos = in_pos – diff[-reverse[atom]]

diff are the constraint wise outputs of SHAKE.
Implementation of bonded forces

- Not the most computationally intensive part of the application.
- Necessary to model proteins and complex molecules.
- Based on a static connectivity between atoms (chemical bonds).
- Categories:
  - Bond stretching
  - Bond angle
  - Dihedral angle
Bond Stretching

- Harmonic (spring like)
  \[ V(r) = \frac{1}{2} k (r - r_0)^2 \]

- Morse (anharmonic, more realistic for large deviation from equilibrium)
  \[ V(r) = \frac{k}{2a^2} (1 - \exp(-a(r - r_0)))^2 \]

- Constrained (fixed bond length)
Bond bending

- Harmonic (spring like)

\[ V(\theta) = \frac{1}{2} k (\theta - \theta_0)^2 \]

- Since frequencies are lower than bond stretches, bond bending modes are important and usually cannot be constrained away.

- In gromacs-brook, the input is a stream of triplets of atoms (i, j, k) and output is a stream of triplets of forces (fi, fj, fk).
  - Computation involves calculation of an inverse cosine and a reciprocal square root for each triplet.
  - Since forces are calculated out of order, they have to be scattered back with an inverse gather.
  - Assuming an atom can have at most 4 bonds, we know beforehand the maximum number of times an atom can appear in the input triplets (4 in our case) and use that to do the inverse gather properly.
Partial forces

<table>
<thead>
<tr>
<th></th>
<th>$f_i$</th>
<th>$f_j$</th>
<th>$f_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 1, 2)</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>(1, 3, 5)</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>(5, 6, 7)</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

- Scatter is transformed into a gather
- More than one inverse map is needed if a given atom appears more than once in a list.

Inverse map

<table>
<thead>
<tr>
<th></th>
<th>$\text{map}_i$</th>
<th>$\text{map}_j$</th>
<th>$\text{map}_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>999</td>
</tr>
<tr>
<td>2</td>
<td>999</td>
<td>999</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>999</td>
<td>1</td>
<td>999</td>
</tr>
<tr>
<td>4</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>999</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>999</td>
<td>2</td>
<td>999</td>
</tr>
<tr>
<td>7</td>
<td>999</td>
<td>999</td>
<td>2</td>
</tr>
</tbody>
</table>

Dummy index 999 stores zero force
Bond torsions (Dihedrals)

Dihedrals
• Multiple maxima and minima

\[ V(\phi) = C_1 \sin(n_1\phi) + C_2 \sin(n_2\phi) \]

• In gromacs-brook, the input is a stream of quartets of atoms \((i, j, k, l)\) and output is a stream of quartets of forces \((f_i, f_j, f_k, f_l)\). The same scheme is used to scatter the forces as in the case of angles.

Chirality (Improper dihedrals)
• Used to prevent umbrella flipping motions.
• Improper dihedral angle is calculated exactly like a dihedral angle, but connectivity of the atoms is different. Quadratic potential is usually used to prevent flipping.
Performance

- No performance yet!
- Code works on linux-CPU backend for box of water molecules. Some issues when running on GPU.
- Protein simulation: still fixing bugs in SHAKE.