Accelerating Rosetta with OpenMM

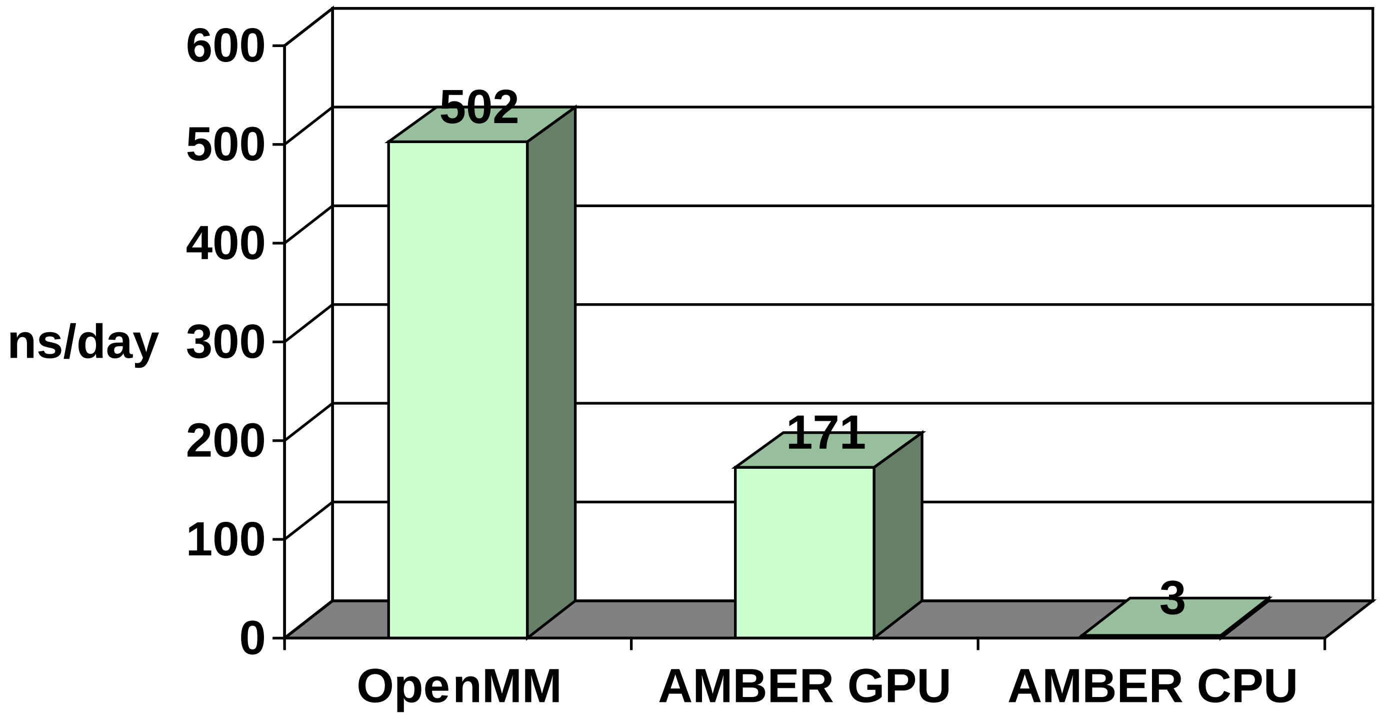
Peter Eastman

RosettaCon, August 5, 2010

What is OpenMM?

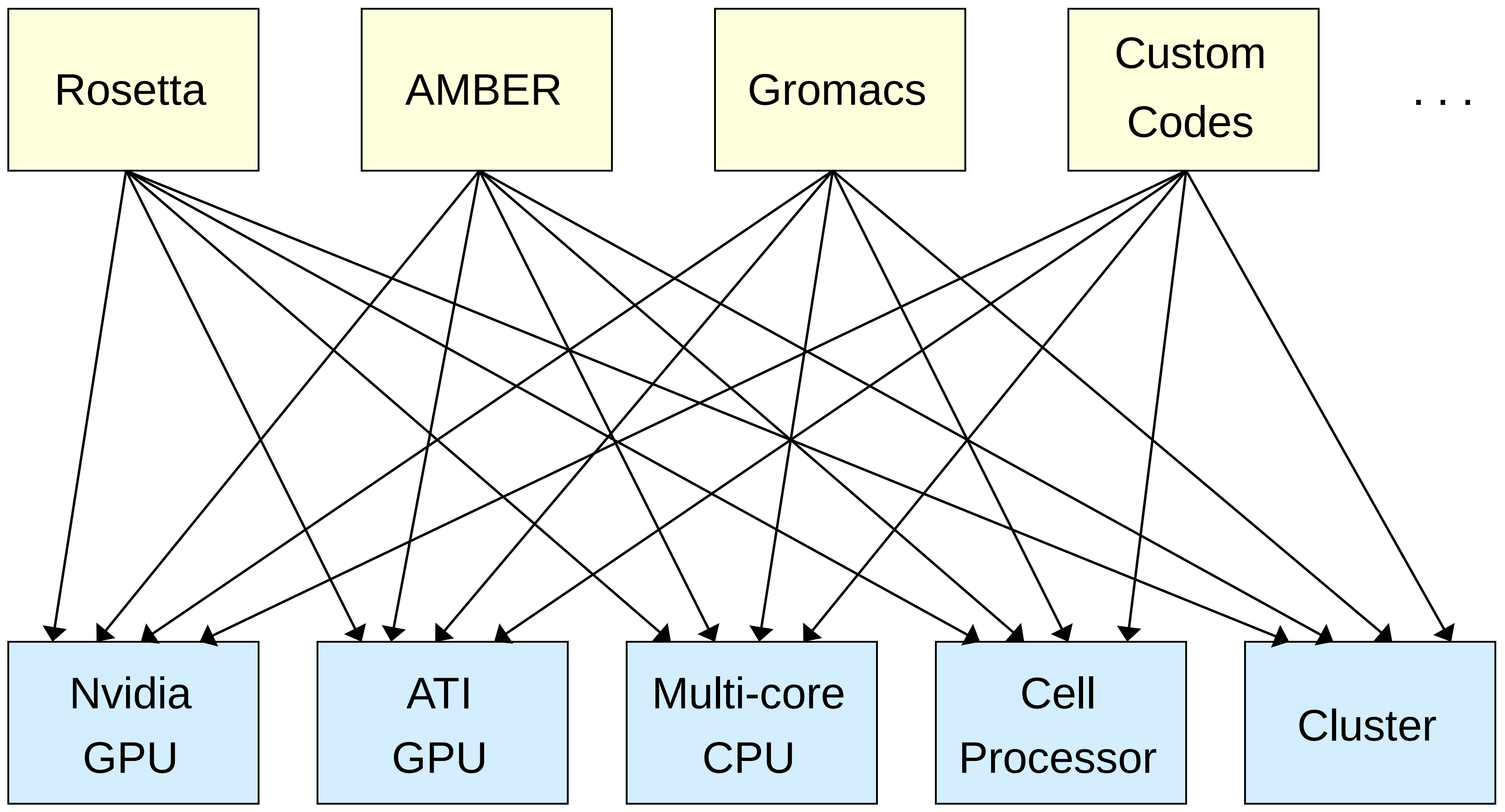
OpenMM is a *library* for molecular modeling on high performance architectures.

Performance

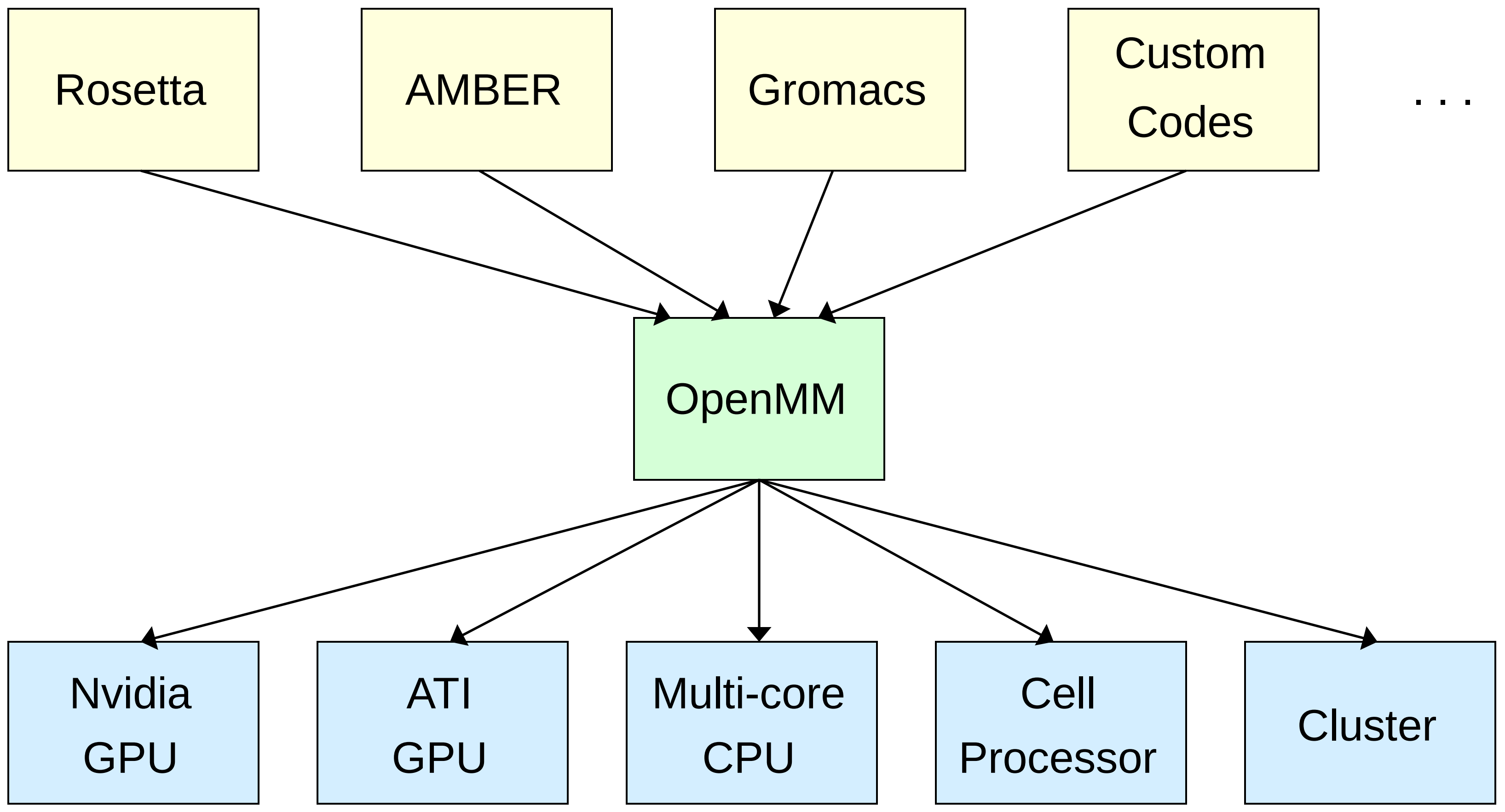


Villin headpiece, 576 atoms, GBSA, 2 fs time step, no cutoff, run on one GPU core of a Tesla S1070 and one CPU core of a 2.5 GHz Xeon E5420

The OpenMM Vision



The OpenMM Vision



Features

- Standard Forces
  - Standard bonded terms (harmonic bonds, angles, etc.)
  - Coulomb and Lennard Jones nonbonded forces
    - Reaction field, Ewald, and PME for long range Coulomb forces
  - GBSA implicit solvent

Features (continued)

- Integrators
  - Verlet (leapfrog)
  - Langevin
  - Brownian
- Temperature/Pressure Coupling
  - Andersen thermostat
  - Monte Carlo barostat
- Energy Minimization
  - L-BFGS

Custom Forces

- Allow arbitrary algebraic expressions for forces
- Example: Morse Potential

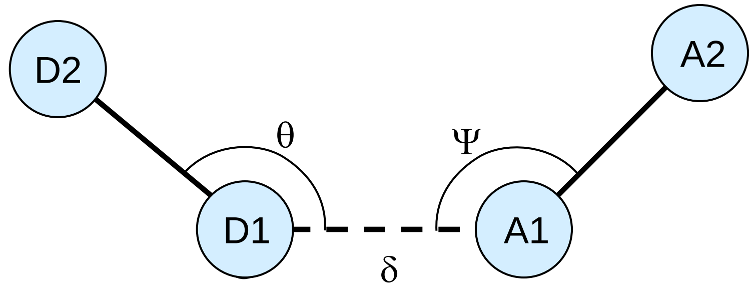
```
CustomBondForce* bonds = new CustomBondForce("D*(1-exp(a*(r0-r)))^2");
bonds->addPerBondParameter("D");
bonds->addPerBondParameter("a");
bonds->addPerBondParameter("r0");
```

Custom Forces (continued)

- Algebraic expressions are converted to OpenCL code and compiled
  - Very little performance cost
  - A powerful tool for research and prototyping
- Available custom forces
  - Bond/angle/torsion bonded terms
  - Nonbonded pairwise forces
  - External forces
  - Hydrogen bond forces
  - Implicit solvent models

Rosetta Hbond Potential

- Hbond connects “donor” and “acceptor” groups



$$E = E_\delta(\delta)F_\theta(\theta)F_\Psi(\Psi) + \\ F_{sr}(\delta)\Big(E_{\theta,sr}(\theta)F_\Psi(\Psi) + E_{\Psi,sr}(\Psi)F_\theta(\theta)\Big) + \\ F_{lr}(\delta)\Big(E_{\theta,lr}(\theta)F_\Psi(\Psi) + E_{\Psi,lr}(\Psi)F_\theta(\theta)\Big)$$

- $E_\delta(\delta)$, $E_{\theta,sr}(\theta)$, $E_{\theta,lr}(\theta)$, $E_{\Psi,sr}(\Psi)$, $E_{\Psi,lr}(\Psi)$ are derived from structure data

- $F_\theta(\theta)$, $F_\Psi(\Psi)$, $F_{sr}(\delta)$, $F_{lr}(\delta)$ are piecewise linear “fading functions”

OpenMM Implementation

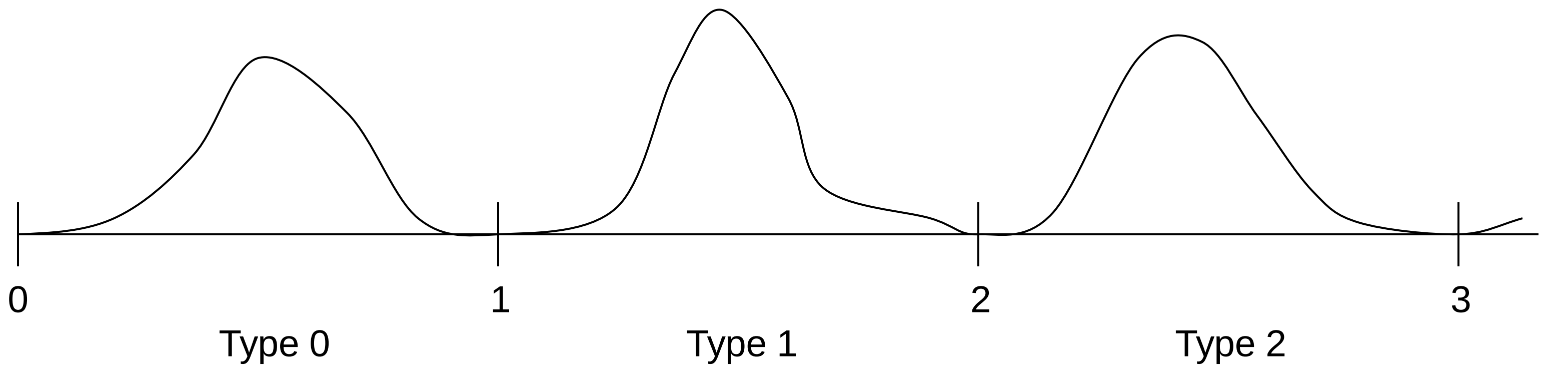
```
string energy = "ed(d)*fadeTheta*fadePsi+"
    "fadeShort*(ethetaShort(ctheta)*fadePsi+epsiShort(cpsi)*fadeTheta)+"
    "fadeLong*(ethetaLong(ctheta)*fadePsi+epsiLong(cpsi)*fadeTheta);"
  "fadeShort=min(min(100*(d-0.14), 1-25*(d-0.19)), 1);"
  "fadeLong=min(25*(d-0.19), 1-(1/0.07)*(d-0.23));"
  "fadeTheta=min(20*ctheta, 1);"
  "fadePsi=min(20*cpsi, 1);"
  "d=distance(a1,d1); ctheta=-cos(angle(a1,d1,d2)); cpsi=-cos(angle(d1,a1,a2))";
CustomHbondForce* force = new CustomHbondForce(energy);
force->addFunction("ed", createDistanceTable(), 0.14, 0.3, true);
force->addFunction("ethetaShort", createShortRangeThetaTable(), 0, 1, true);
force->addFunction("ethetaLong", createLongRangeThetaTable(), 0, 1, true);
force->addFunction("epsiShort", createShortRangePsiTable(), 0, 1, true);
force->addFunction("epsiLong", createLongRangePsiTable(), 0, 1, true);
```

A Complication

- The energy also depends on the type of chemical groups interacting
  - $E_{\delta}(\delta)$, $E_{\theta,sr}(\theta)$, $E_{\theta,lr}(\theta)$, $E_{\Psi,sr}(\Psi)$, $E_{\Psi,lr}(\Psi)$ all have multiple versions for different types of acceptors

The Solution

- Define a per-acceptor parameter to specify the type
- Append the different versions of each function
- Use the parameter to select the correct range



Implementation with Types

```
string energy =
  "ed(d-0.14+type*0.26)*fadeTheta*fadePsi*step(d-0.14)+"
      "fadeShort*(ethetaShort(ctheta+type)*fadePsi*step(ctheta)+"
          "epsiShort(cpsi+type)*fadeTheta*step(cpsi))+"
      "fadeLong*(ethetaLong(ctheta+type)*fadePsi*step(ctheta)+"
          "epsiLong(cpsi+type)*fadeTheta*step(cpsi));"
  "fadeShort=min(min(100*(d-0.14), 1-25*(d-0.19)), 1);"
  "fadeLong=min(25*(d-0.19), 1-(1/0.07)*(d-0.23));"
  "fadeTheta=min(20*ctheta, 1);"
  "fadePsi=min(20*cpsi, 1);"
  "d=distance(a1,d1); ctheta=-cos(angle(a1,d1,d2)); cpsi=-cos(angle(d1,a1,a2))";
```

Acknowledgements

- Vijay Pande
- Rhiju Das
- Kyle Beauchamp
- NIH

https://simtk.org/home/openmm