Introduction of OpenMM

Vijay Pande

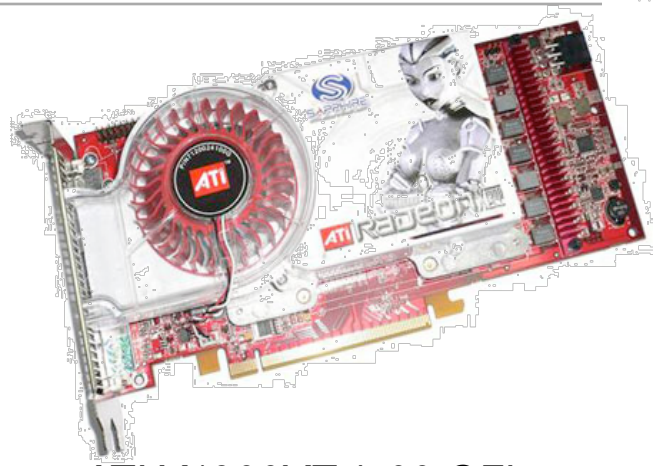
OpenMM Workshop, February 12, 2009

Folding@home on your desktop: GPU's

- **Graphics Processing Units (GPUs) are very powerful**
  - Folding@home calculation circa 2003 = 10,000 PC's @ 1 GFLOP/PC = 10,000 GFLOPS
  - Fast GPU today = 1,000 GFLOP
  - Fast GPU cluster today = ~50,000 GFLOPS

- **GPU's are getting faster, faster than CPU's**
  - Moore's law is dead for traditional CPU's
  - we now see more cores per chip, but each core isn't any faster
  - GPU's figured out this trick a long time ago
  - typical GPU's now have 100's of cores
  - GPU's use their cores more efficiently

- **BUT, GPU's are horrible to program**
  - can't just recompile
  - must rethink algorithms
  - must understand the nature of the hardware
  - work closely with vendors (we collaborate close with AMD/ATI, NVIDIA, and Intel)



ATI X1900XT (500 GFlops peak, ~$100 + of a cost computer)

Sony PS3 (Cell processor: 220 GFlops peak, ~$400 total)

Unique aspects of comp biology on GPUs

- **Design algorithms that are GPU friendly**
  - FLOPS are free, memory is expensive
  - low lying fruit: algorithms which map well to GPUs

- **Code everything on the GPU**
  - If the original bottleneck is 90% of the calc, that's still only a 10x speed up at best
  - to get 100x to 1000x, one needs to have the whole calculation on the GPU (in our experience)

- **Centralized libraries, open source (eg OpenMM)**
  - avoid reinventing the wheel
  - build on others' work

- **Next steps**
  - not just speeding existing algorithms, but new methods
  - code methods which we wouldn't even dare to try now

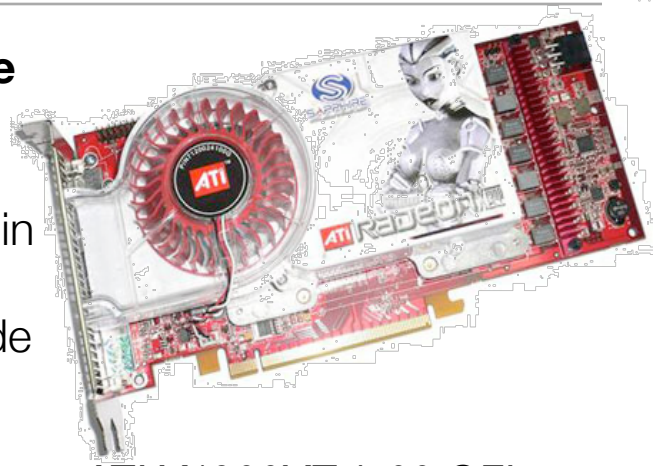
Large speed increases seen using GPU

Molecule	# atoms	ns/day	speedup*	GFLOPS (GPU)	GFLOPS (x86)
fip35	544	576	128	311	657
villin	582	529	136	328	692
lambda	1254	202	255	547	1153
a-spectrin	5078	17	735	805	1702

(*comparing a GTX280 to a single core of a 3GHz core 2 duo using the AMBER code)

The OpenMM opportunity

- **The molecular mechanics community has become fragmented**
  - tens of different MD codes with overlapping functionality, and this has greatly reduced synergy in the community
  - new advances need to be ported to these multitude of codes in order to have a broad impact
  - hardware acceleration (via multi-core, SSE, MPI, GPU's, and math coprocessors) is a critical element
  - much like the graphics community in the 1980's



ATI X1900XT (500 GFlops peak, ~$100 + of a cost computer)

- **Hardware acceleration is a great unifying facto**
  - we propose OpenMM, an extensible API for molecular mechanics
  - unifying API the way OpenGL unified graphics
  - incorporates hardware acceleration in its base design
  - this API would be used as the backend to exist codes, allowing for all to benefit from hardware acceleration

Sony PS3 (Cell processor: 220 GFlops peak, ~$400 total)

Long term goals

- **A complete library for molecular mechanics**
  - complete = what would need to do to do the most common calculations
  - complete != does everything conceivable
  - “Steve Jobs approach”

- **Fast and general**
  - Don’t exposure hardware specifics
  - but optimize for speed underneath
  - long term: broad support for GPUs, multicore, etc

- **Two level API**
  - OpenMM for high level: read like text
  - Low level API: for developers (mainly in-house & accelerator devs)
  - OpenMM can be a nexus for application and low level programmers to meet

What sorts of capabilities will we support?

- **Simulation protocols: the standards**
  - now: Langevin dynamics, implicit solvent
  - “soon”: Explicit solvent, reaction-field, constant temperature
  - future: Explicit solvent, PME, constant temperature, constant pressure

- **Force fields (everything that Gromacs supports)**
  - AMBER94, AMBER96, AMBER99, AMBER2003
  - OPLS-AA, OPLS-AA/L
  - CHARMM19
  - GROMOS, GROMACS

- **Sampling methods**
  - constant temperature MD, constant energy MD
  - Simulated tempering (like replica exchange)

- **Will you support my exotic method XXX?**
  - maybe, but you can contribute plug ins and reference code
  - complete open source code (BSD)

Development Roadmap

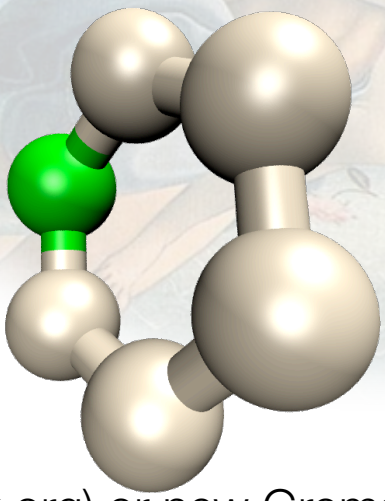
- **Preview Release 1 (Sept. 2008)**
  - First draft of the public API
  - Included reference implementation only

- **Preview Release 2 (Jan. 2009)**
  - Included implementations for ATI and Nvidia GPUs
  - No support for explicit solvent

- **Preview Release 3... to 1.0**
  - Support explicit solvent: Cutoffs, periodic boundary conditions, SETTLE (Prototype implementation is complete)
  - Particle Mesh Ewald (Under development)
  - AMOEBA (Under development)

- **Later releases**
  - Port GPU code to OpenCL
  - Other forces, integration methods, barostats, etc.
  - Support other platforms (e.g. clusters, multicore)

New Application: OpenMM Zephyr



- **Goals**
  - make MD easy to run
  - easy but correct setup (not just PDB -> MD, but think about protonation, missing residues, etc)
  - easy to run on GPU's
  - visual feedback
- **Under the hood**
  - Wrap GPU enabled MD code
  - use MMtools (Pande group, SimTk.org) or new Gromacs set up tools
  - Use VMD IMD interface for visualization (leverage a standard in molecular visualization)
- **Use of real time visualization**
  - immediate feedback is not just fun, but can be useful
  - key to correct setup, etc

Licensing and distribution

- **BSD license**
  - so do whatever you want!
  - we're looking for collaborations for new features

- **But, please cite us**
  - Any work that uses OpenMM should cite the following paper: M. S. Friedrichs, P. Eastman, V. Vaidyanathan, M. Houston, S. LeGrand, A. L. Beberg, D. L. Ensign, C. M. Bruns, V. S. Pande. "Accelerating Molecular Dynamic Simulation on Graphics Processing Units." J. Comp. Chem., (2009)
  - early access: http://www3.interscience.wiley.com/journal/121677402/abstract

Summary

- **What is it**
  - API & library for core molecular dynamics / molecular mechanics applications
  - emphasis on speed (eg hardware acceleration) and generality
  - dual APIs (one for applications and one for low level hardware)
  - demo application: OpenMM Zephyr
  - open source (BSD) software

- **What is it not**
  - a general solution for all possible molecular mechanics tasks
  - a compiler which can turn an MD code into accelerated code