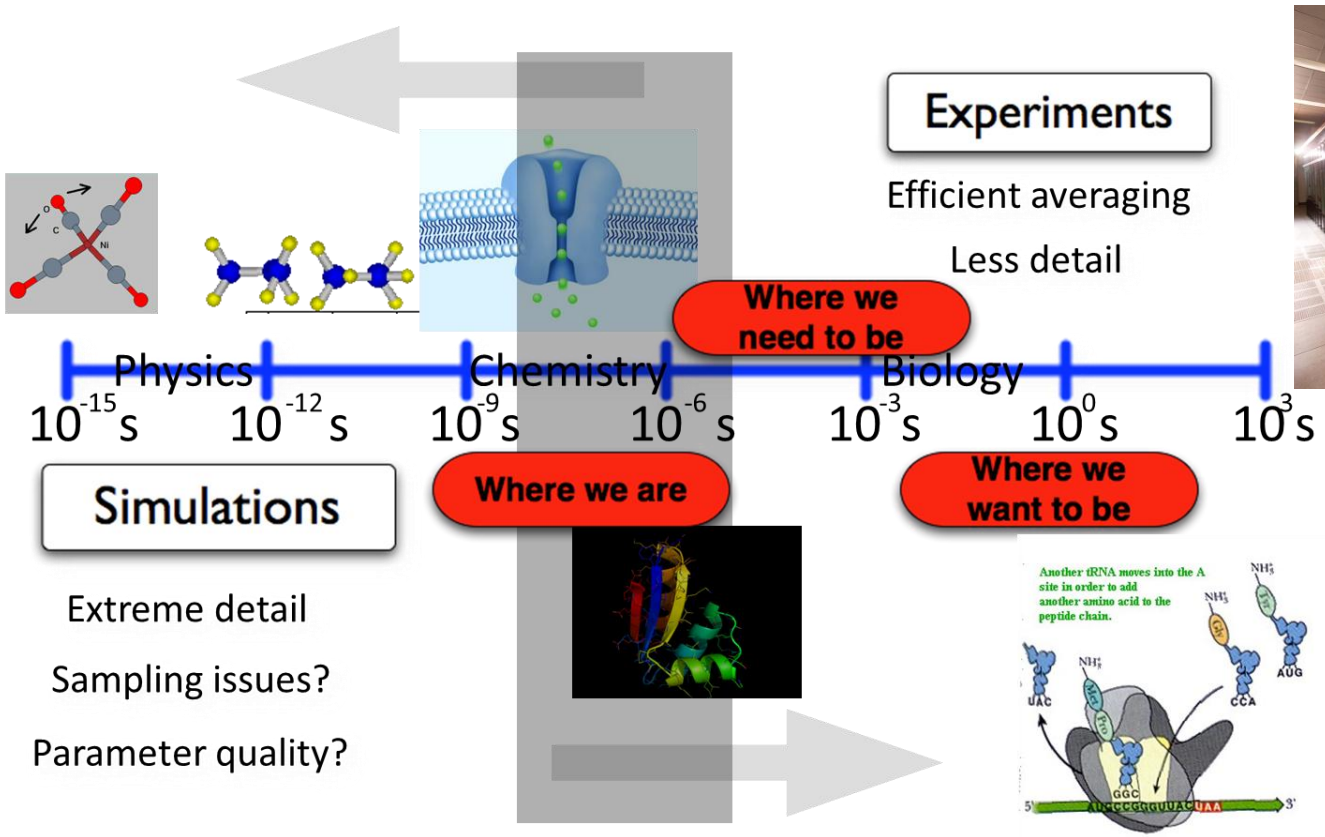
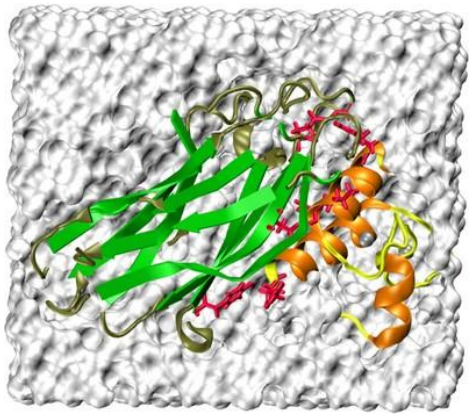
How can molecular simulation reach the exascale? Challenges in performance and parallelism

Roland Schulz & Erik Lindahl



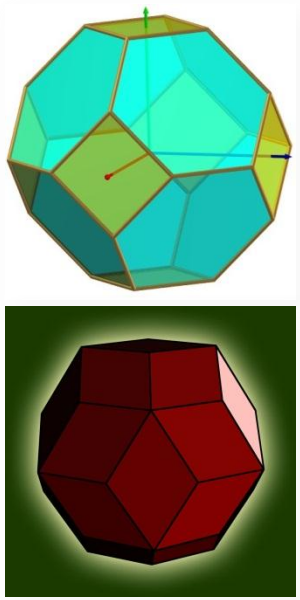


Content

- Single node performance
  - Algorithms, SSE, time-step
- Scaling
  - Load-balancing, Reaction-field, PME
- GPU
- Ensemble

GROMACS Approaches

- GPL
  - ~ 500 citations/yr, 5k-10k users
- Algorithmic optimization:
  - No virial in nonbonded kernels
  - Single precision by default (cache, BW usage)
  - Tuning to avoid conditional statements such as PBC checks
  - Triclinic cells everywhere: can save 15-20% on system size
- Optimized 1/sqrt(x)
  - Used ~150,000,000 times/sec
  - Handcoded asm for ia32, x86-64, ia64, Altivec, VMX, BlueGene (SIMD instructions)

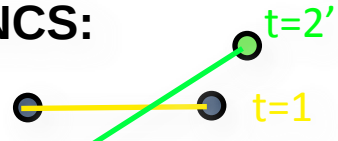


a_0	a_1	a_2	a_3
+			
b_0	b_1	b_2	b_3
=			
c_0	c_1	c_2	c_3

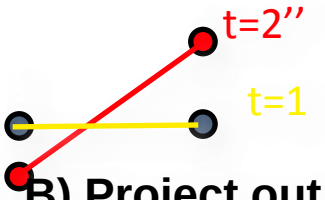
Constraints

- Δt limited by fast motions - 1fs
  - Remove bond vibrations
- SHAKE (iterative, slow) - 2fs
  - Problematic in parallel (won't work)
  - Compromise: constrain h-bonds only - 1.4fs
- GROMACS (LINCS):
  - LINear Constraint Solver
  - Approximate matrix inversion expansion
  - Fast & stable - much better than SHAKE
  - Non-iterative
  - Enables 2-3 fs timesteps
  - Parallelized

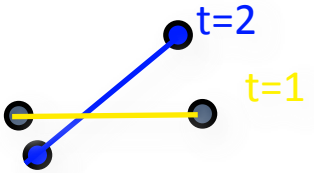
LINCS:



A) Move w/o constraint



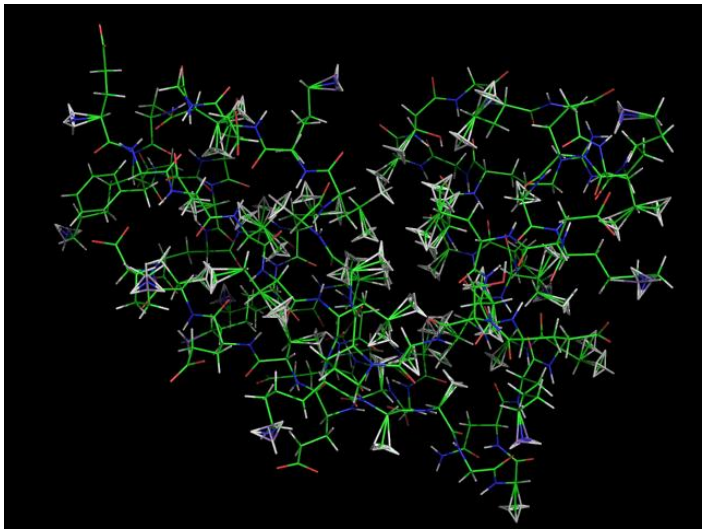
B) Project out motion along bonds

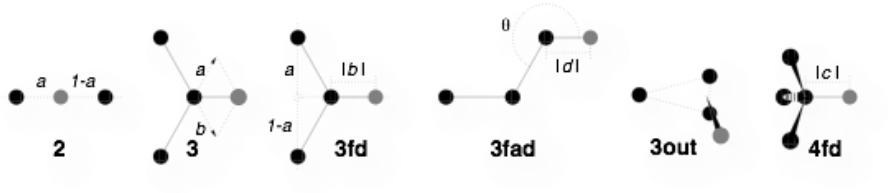


C) Correct for rotational extension of bond

Virtual sites

- Next fastest motions is H-angle and rotations of CH_3/NH_2 groups
- Try to remove them:
  - Ideal H position from heavy atoms.
  - CH_3/NH_2 groups are made rigid
  - Calculate forces, then project back onto heavy atoms
  - Integrate only heavy atom positions, reconstruct H's
- Enables 5fs timesteps!





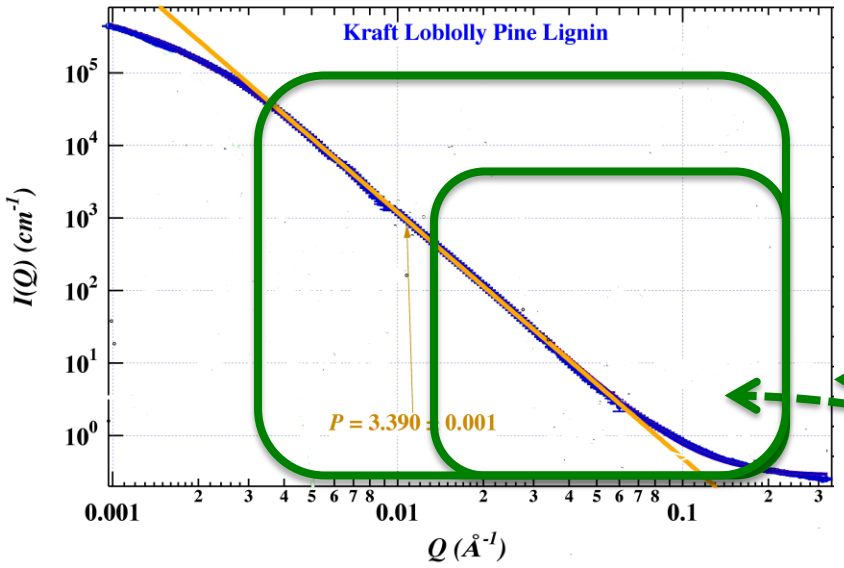
QuickTime™ and a H.264 decompressor are needed to see this picture.

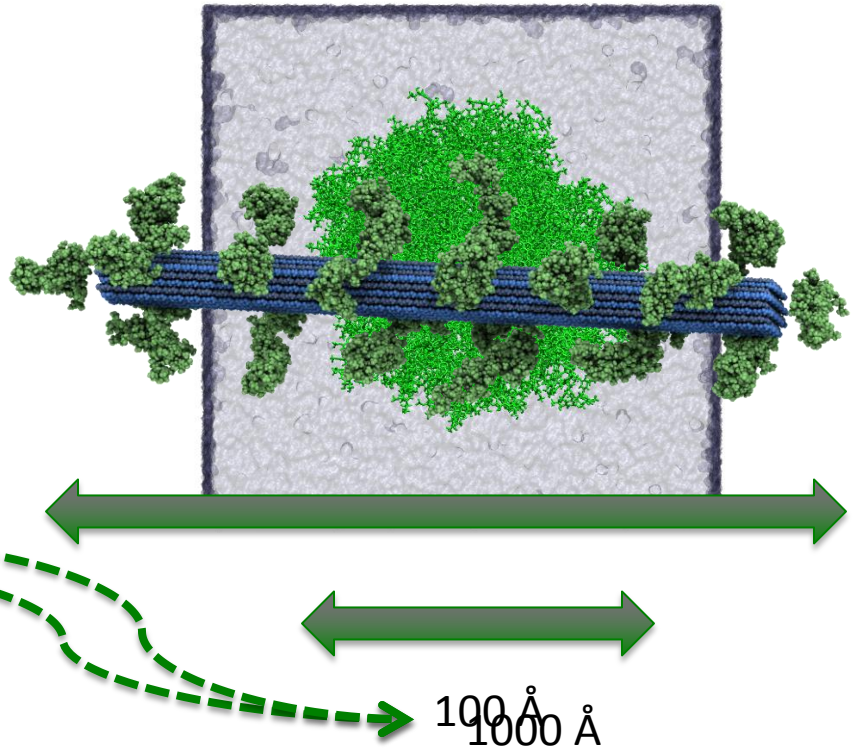
QuickTime™ and a H.264 decompressor are needed to see this picture.

Interactions

Degrees of Freedom

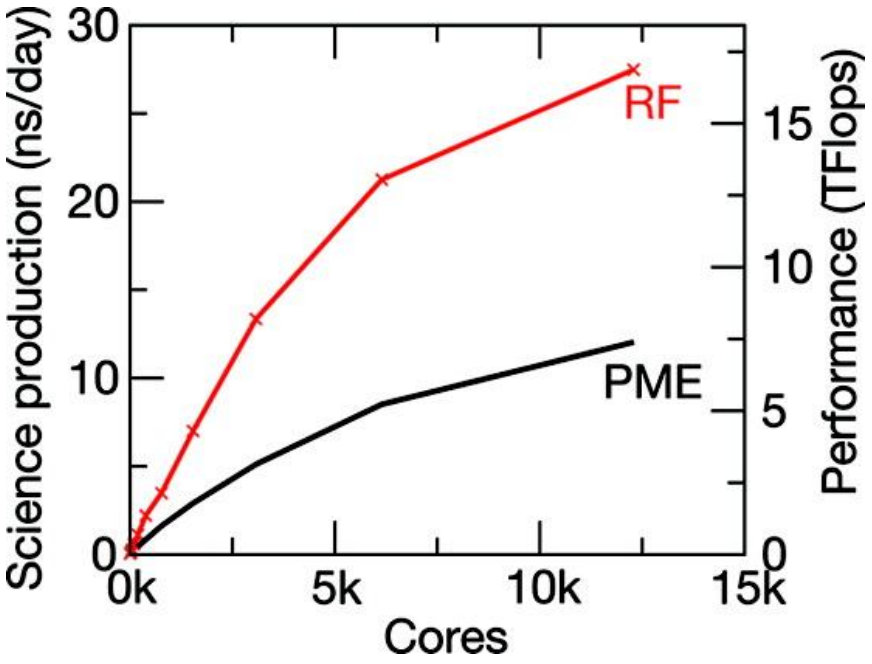
Small Angle Neutron Scattering of Lignin

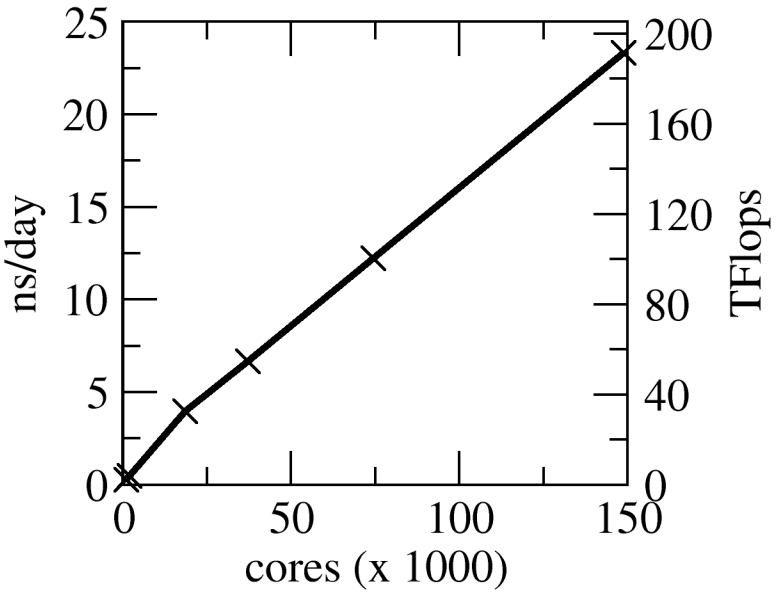




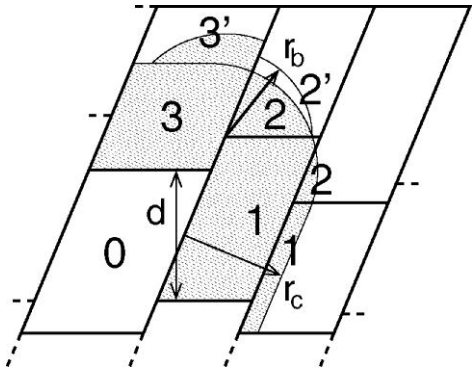
*SV Pingali, V Urban, BR Evans

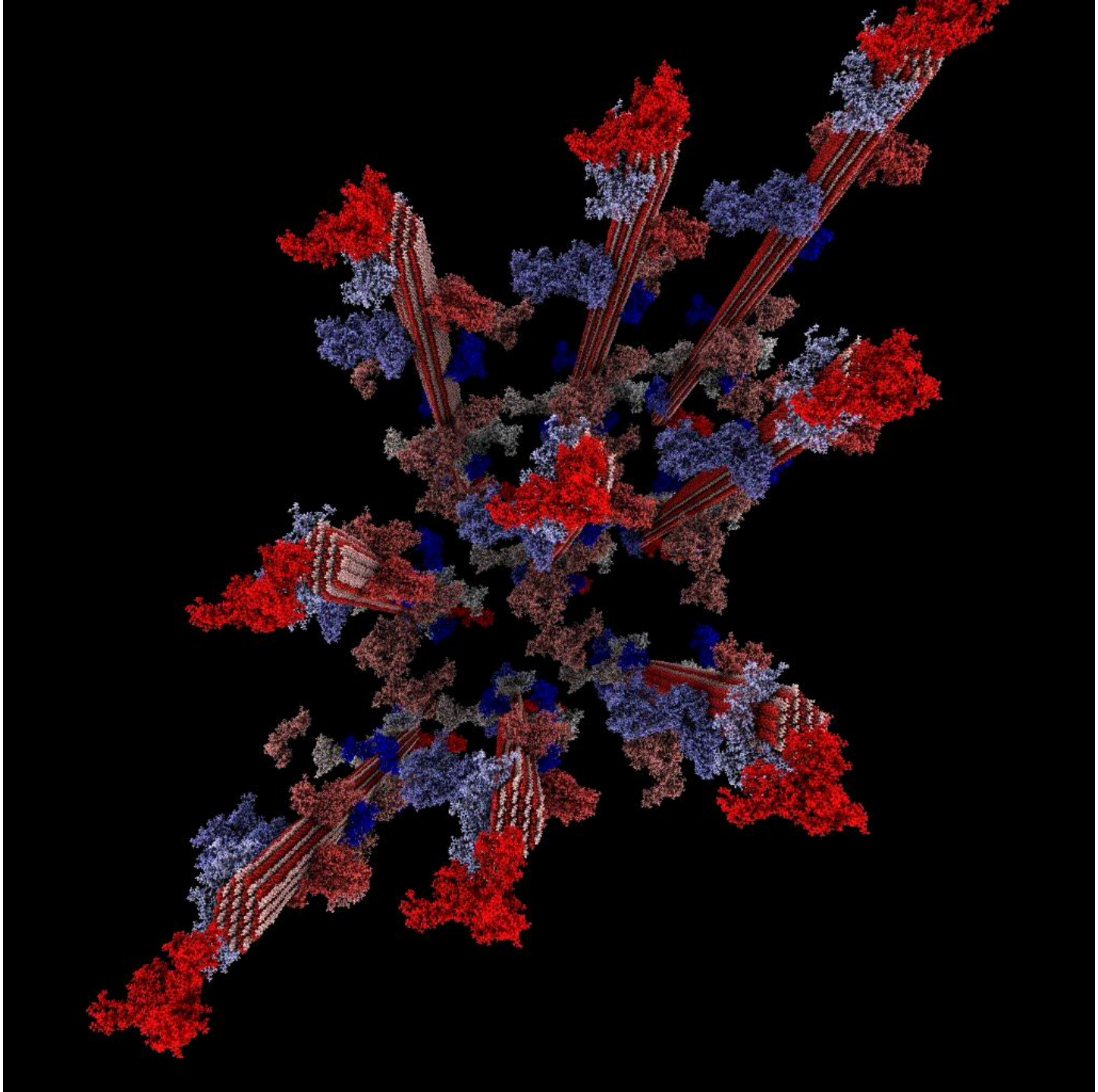
Reaction Field





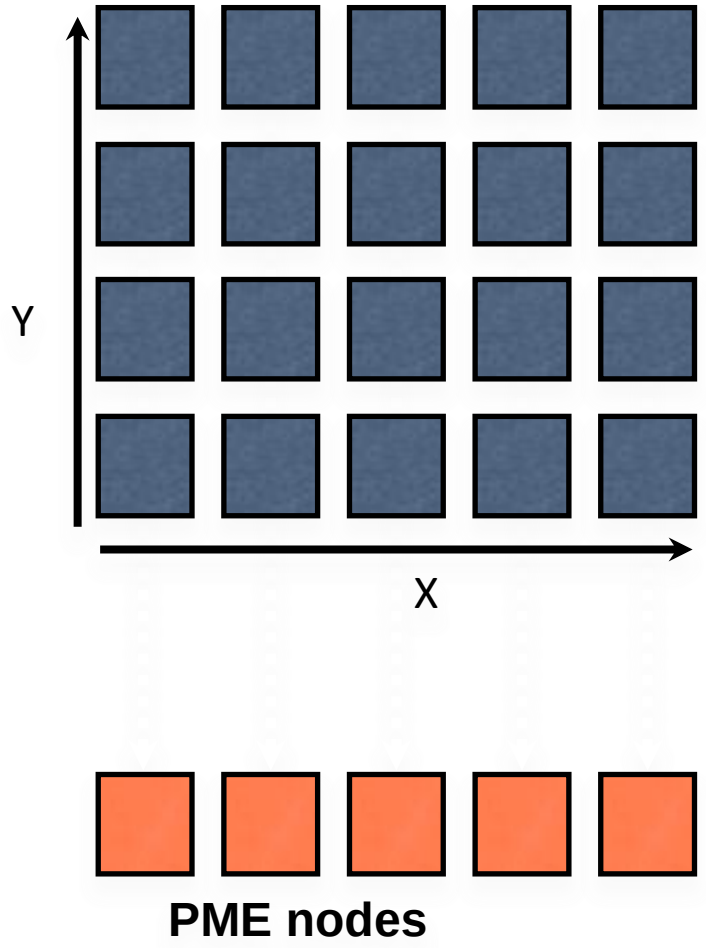
- Load balancing critical
- Water – solute difference
- Imbalance from 200% to 75%
- 44% speed improvement



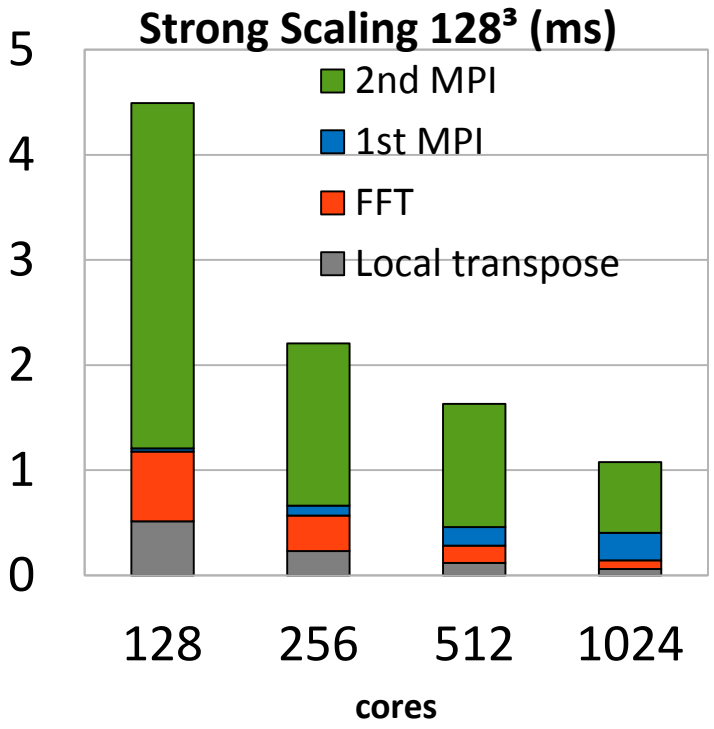


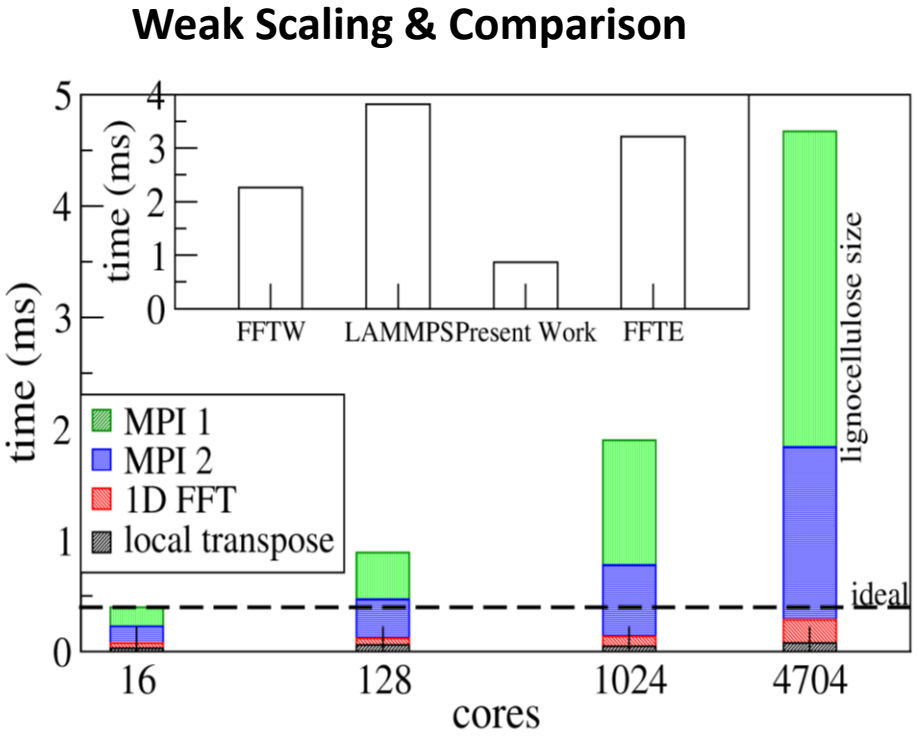
PME

- accurate
- very fast on single CPU
- MPMD
- 2D FFT in Gromacs 4.5



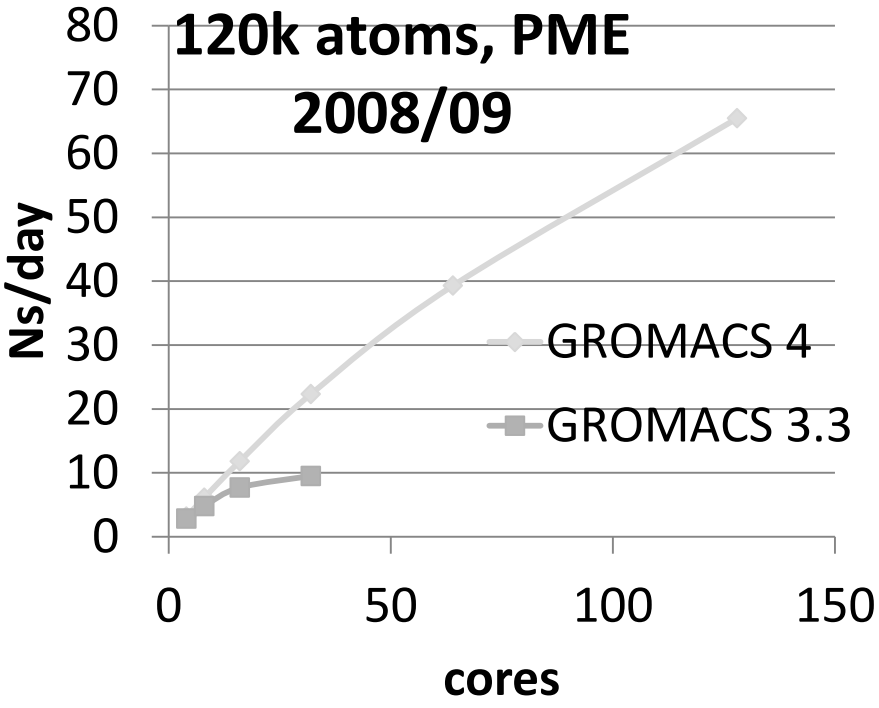
FFT

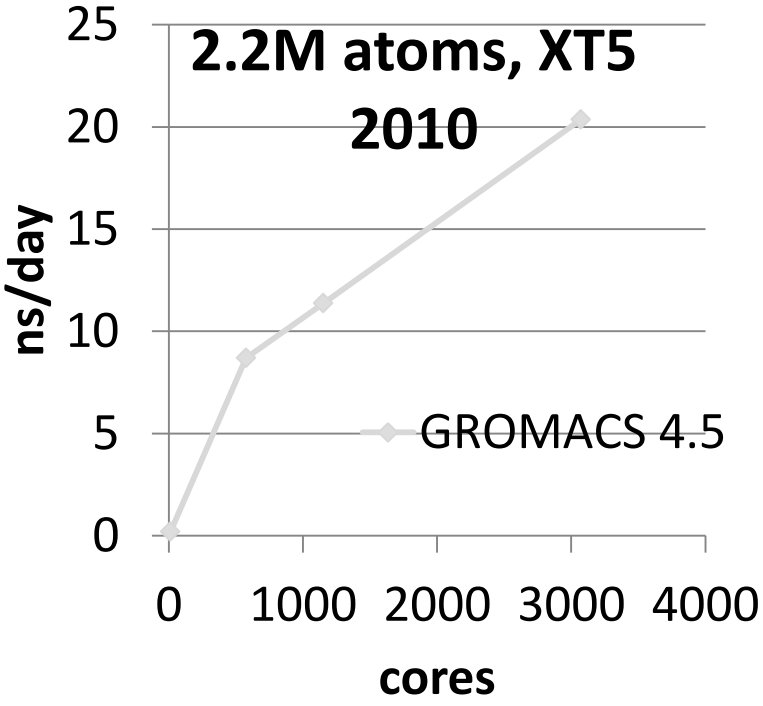




- AllToAllV **much** slower (up to ~10x)
  -> requires padding
- FFTW3 significant faster
- single precision important

Scaling PME


120k atoms, PME
2008/09
Ns/day
80
70
60
50
40
30
20
10
0
0
50
100
150
cores
GROMACS 4
GROMACS 3.3


2.2M atoms, XT5
2010
ns/day
25
20
15
10
5
0
0
1000
2000
3000
4000
cores
GROMACS 4.5

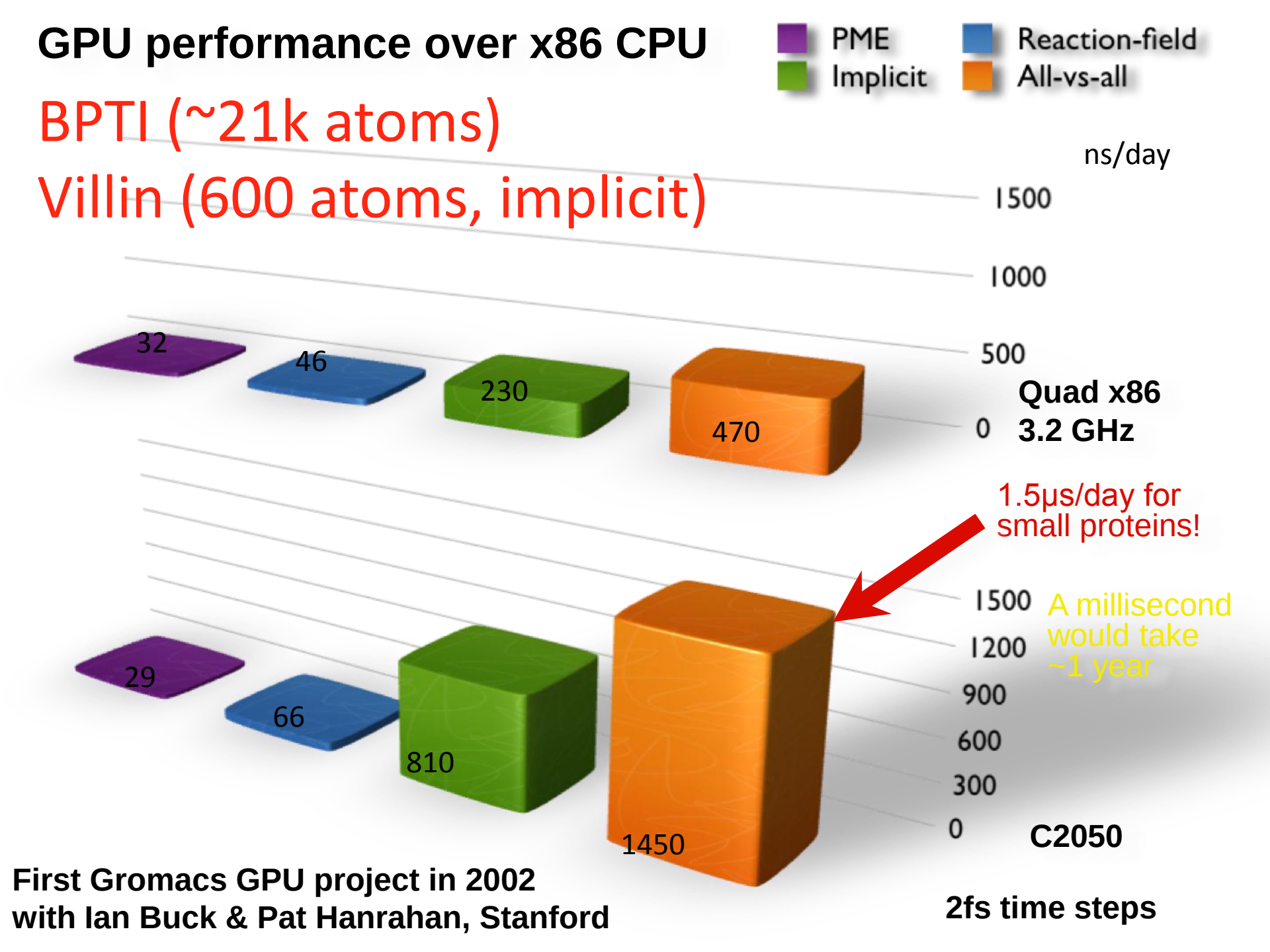
AllToAll performance

- Small sub-communicator important
  - MPMD
  - 2D
- Depends on task placement
  - Has to be done at run-time because of scheduler
  - Up to 40% faster
- MPICH algorithm choice not always optimal
- Wish: Good auto-tuning for MPI

Scaling / Performance

- It is easier to get a simple algorithm to scale!
- Absolute performance not scaling matters

GPU performance over x86 CPU



First Gromacs GPU project in 2002
with Ian Buck & Pat Hanrahan, Stanford

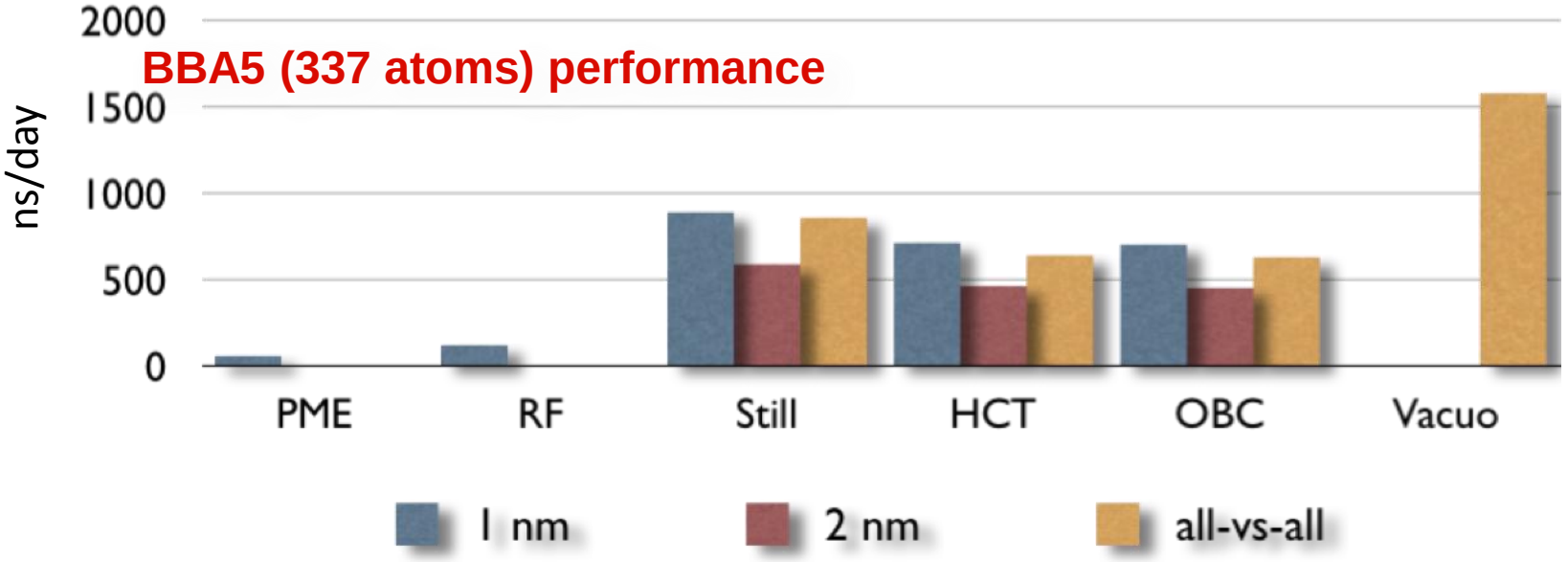
Gromacs & OpenMM in practice

- GPUs supported in Gromacs 4.5
  mdrun ... -device "OpenMM:Cuda"
- Same input files, same output files: "It just works"
- Subset of features work on GPUs for now (checked)
- No shortcuts taken on the GPU:
  - At least same accuracy as on the CPU (<1e-6)
  - Potential energies calculated, free energy works
- Prerelease availability: NOW!
  www.gromacs.org/gpu

Streaming on the CPU

- Lessons from OpenMM applied to CPUs
- New implicit solvent kernels in Gromacs 4.5
- Neighborlist & all-vs-all - and both parallel

Nehalem 2.8 GHz quad-core



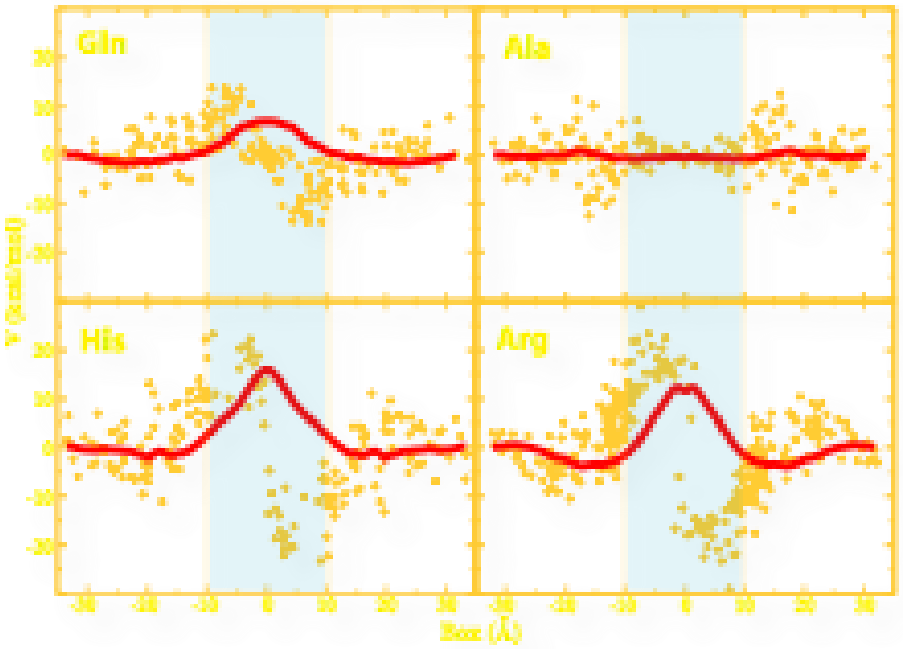
GPU + CPU

- Best performance on 1 GPU if all on GPU
- Plan on cluster:
  - Particle-Particle on GPU
  - CPU
    - PME threaded (slow on GPU)
    - Advanced Algorithms
      - E.g. Constraints + Vsite (allows long time-step)
  - Good parallelization

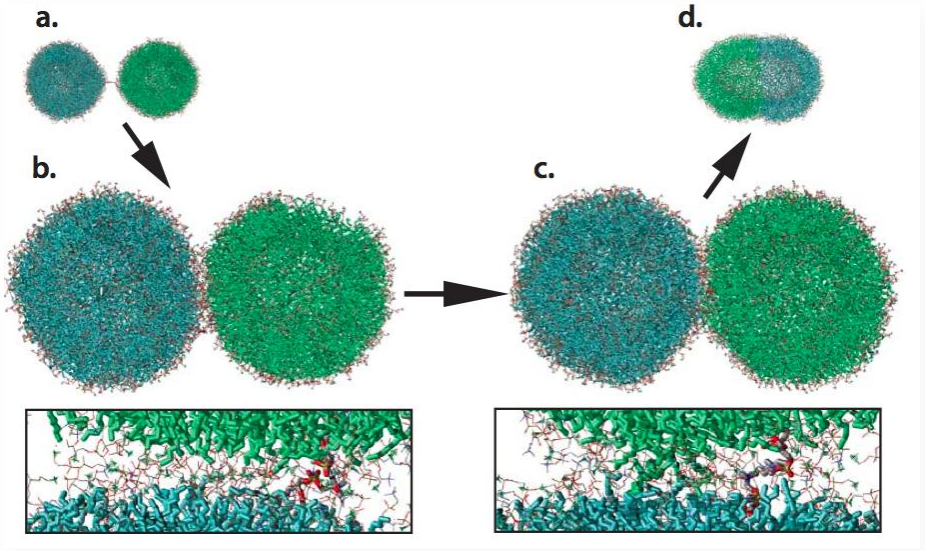
Ensemble Simulation

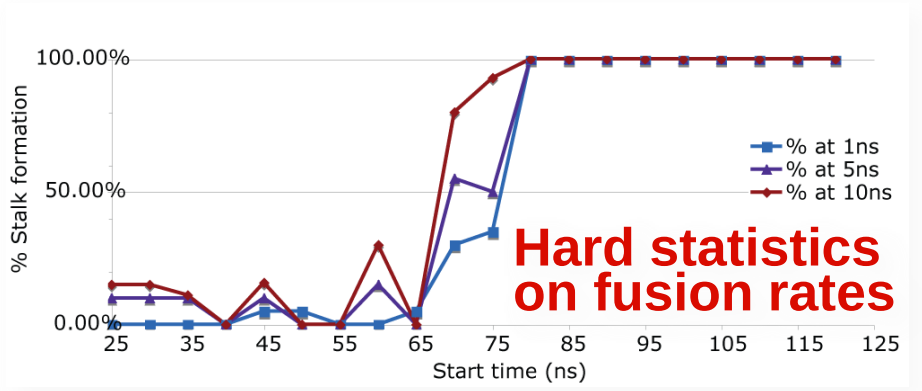
Membrane Protein Insertion Free Energy

4,000 20-ns simulations



Every dot is a simulation!





Anna Johansson, Peter Kasson

Think Massively Parallel - Scale the Problem, not Runs

- Stream Computing is the future
- We're doing *statistical* mechanics!
- No algorithm will parallelize 5000 degrees of freedom over 1 billion processors
- Parallelize in the problem domain instead

The Big Challenge

- **Standard parallelization impossible –**
  We won't get 50ns network latency
  (still standard parallelization will stay important)
- **Explicitly data parallel algorithms needed**

Parallel & ensemble simulations are efficient complementing techniques

Simulation performance is exploding - both on CPUs, GPUs and clusters

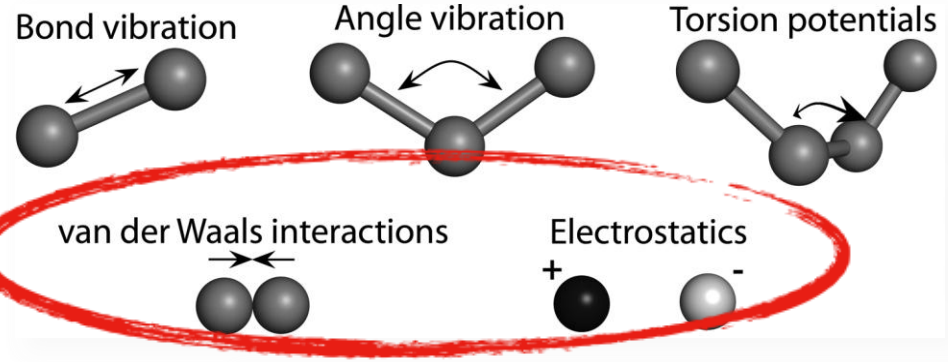
Summary

- Multi-level parallelism necessary
  – SIMD -> Threads -> MPI -> Distributed Computing
- Performance matters.
  Relative scaling doesn't.
- >20ns for 100 million atoms possible
- GROMACS 4.5 good scaling PME
- Streaming architectures are coming
- Currently GPU similar to quad-core for PME
- For scaling best to use both

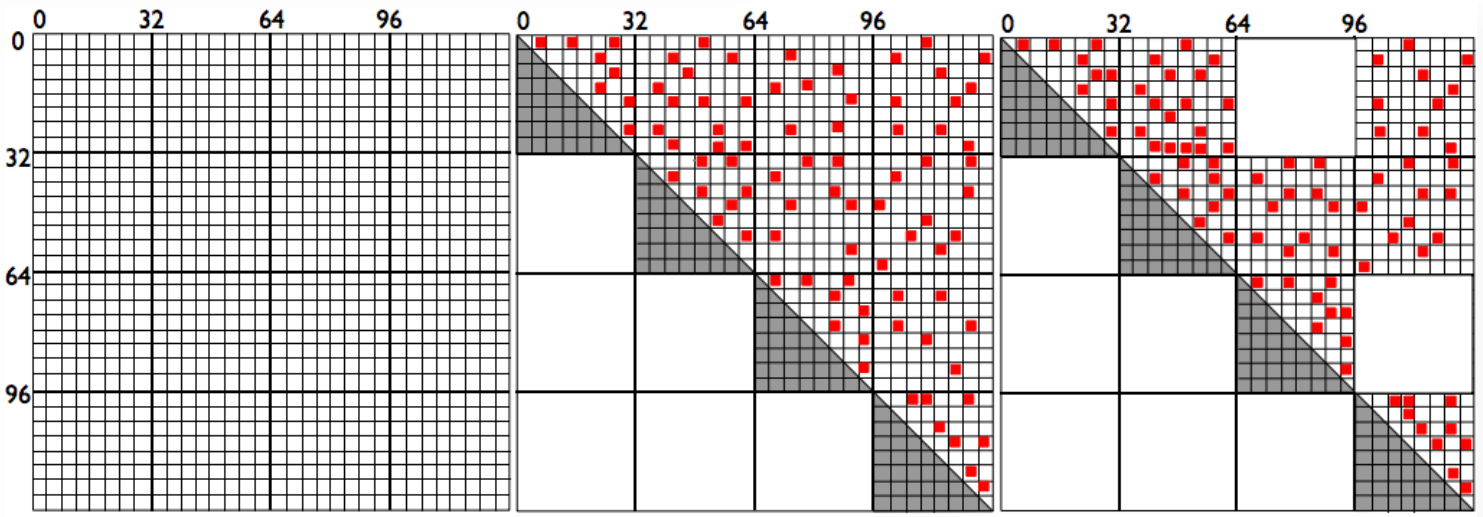
Acknowledgments

- GROMACS: Berk Hess, David van der Spoel, Per Larsson
- OpenMM: Rossen Apostolov, Szilard Pall, Peter Eastman, Vijay Pande
- Nvidia: Scott LeGrand, Duncan Poole, Andrew Walsh, Chris Butler
- Ensemble Simulations: Peter Kasson
- CMB: Jeremy Smith, Loukas Petridis, Benjamin Linder, et al

Molecular Dynamics with CUDA



Absolute performance critical, not speedup relative to a reference implementation!



All-vs-all (CUDA book) Newton's 3rd law Sort atoms in tiles

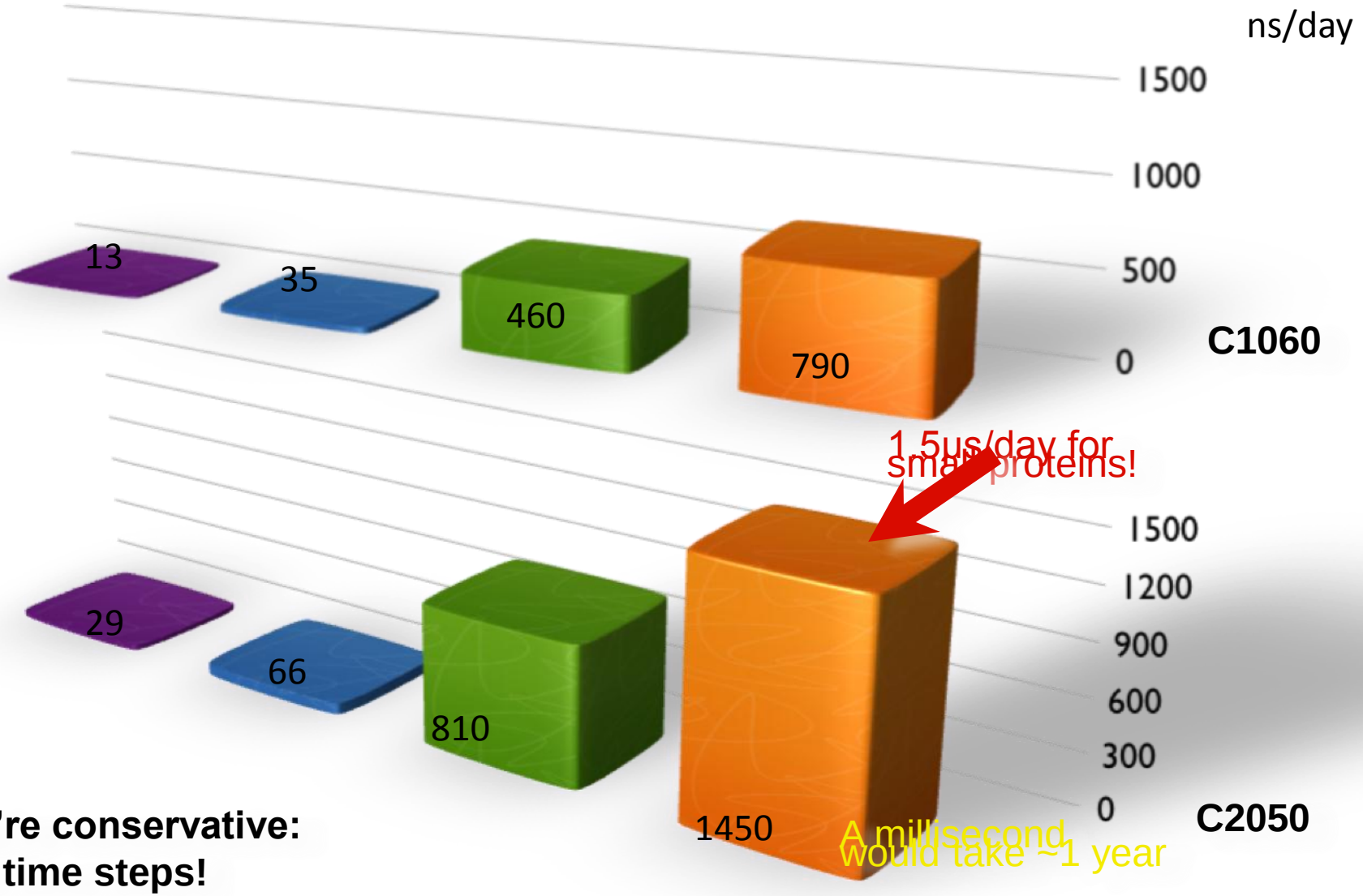
N^2 (N^2)/2 N log N

Scott LeGrand, Peter Eastman

Fermi (C20) performance over C10

BPTI (~21k atoms)

Villin (600 atoms, implicit)



We're conservative:
2fs time steps!