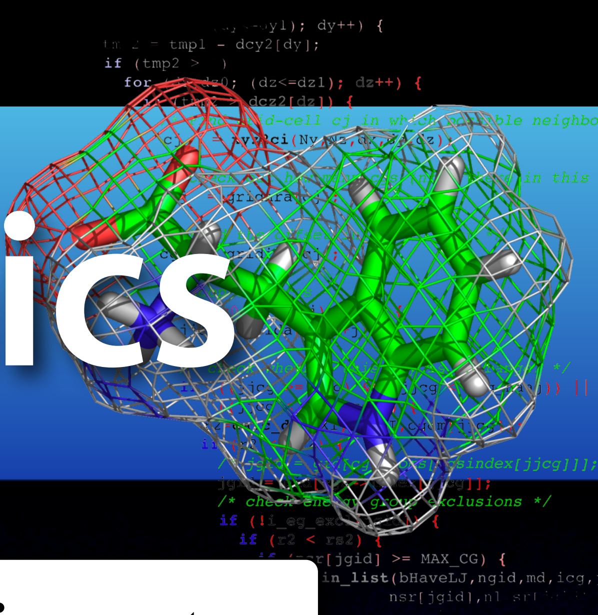
LCSC 2007-10-18

Scaling and Other Bad Ideas in High Performance Computing

Erik Lindahl

lindahl@cbr.su.se
Center for Biomembrane Research
Stockholm University, Sweden

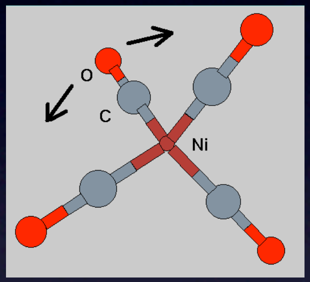
Timescales of Dynamics

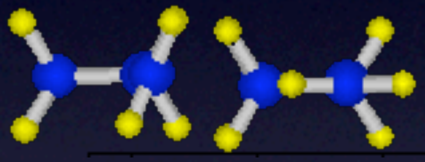


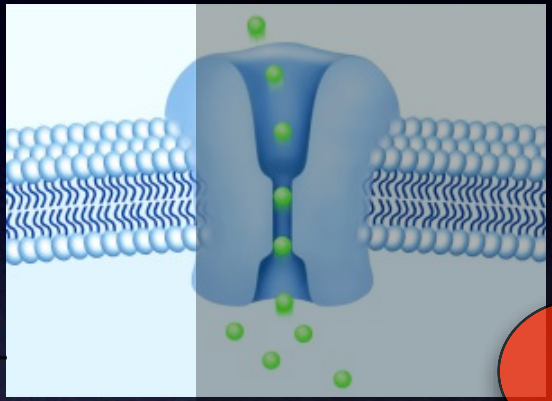
Experiments

Efficient averaging

Less detail







Where we need to be

Physics | Chemistry | Biology

10^{-15}s | 10^{-12}s | 10^{-9}s | 10^{-6}s | 10^{-3}s | 10^{0}s | 10^{3}s

Simulations

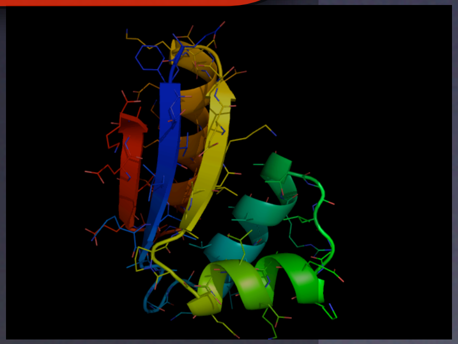
Where we are

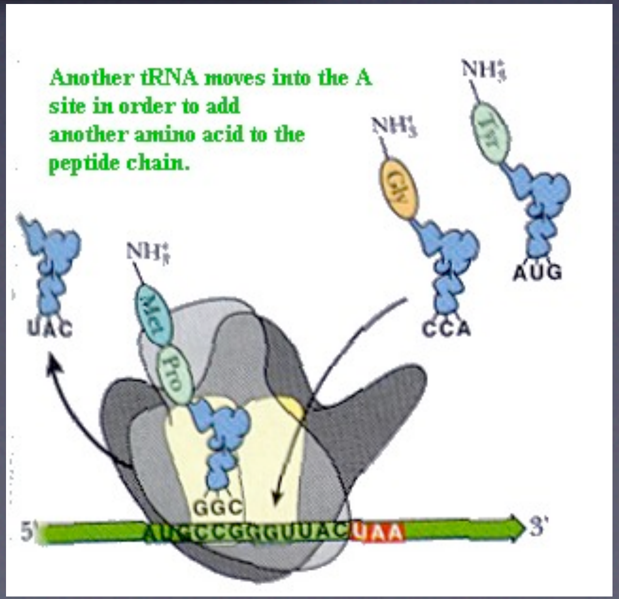
Where we want to be

Extreme detail

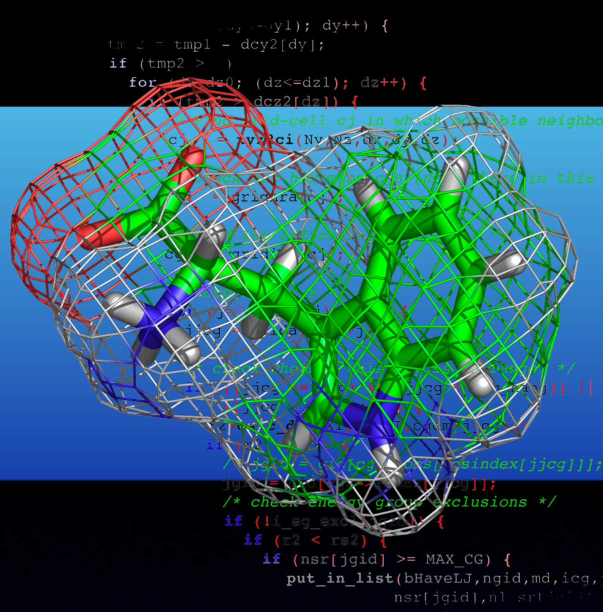
Sampling issues?

Parameter quality?




Another tRNA moves into the A site in order to add another amino acid to the peptide chain.
NH3+
Tyr
Gly
AUG
UAC
Met
Pro
CCA
GGC
5'
AUGCCGGGUUACUAA
3'

Molecular Dynamics



- Solve Newton's equations of motion:

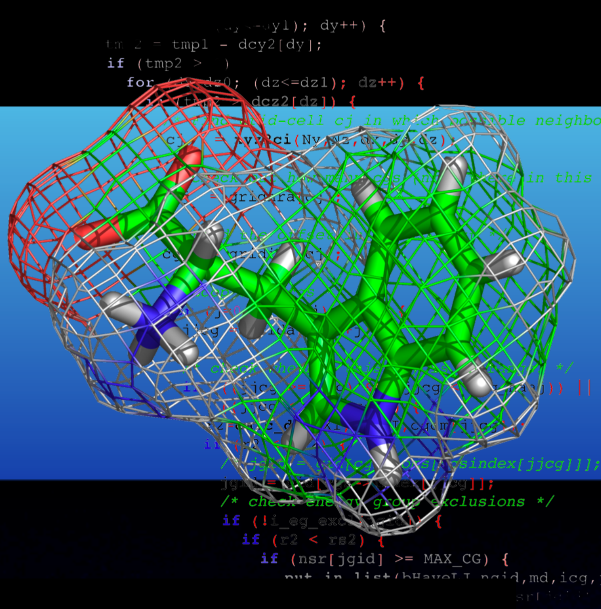
$$m_i \frac{\partial^2 r_i}{\partial t^2} = F_i \qquad i = 1..N$$

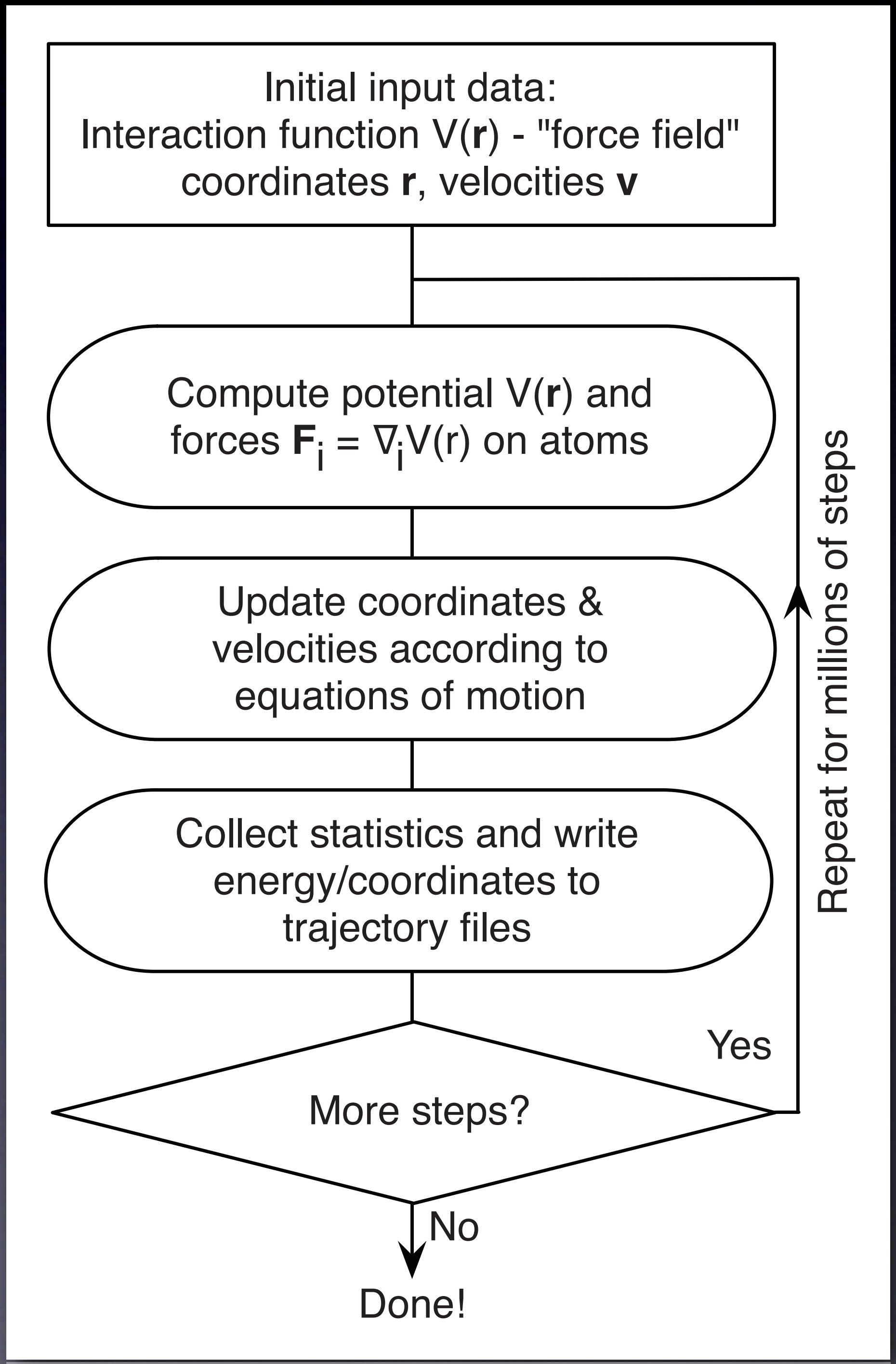
$$F_i = -\frac{\partial V(r)}{\partial r_i}$$

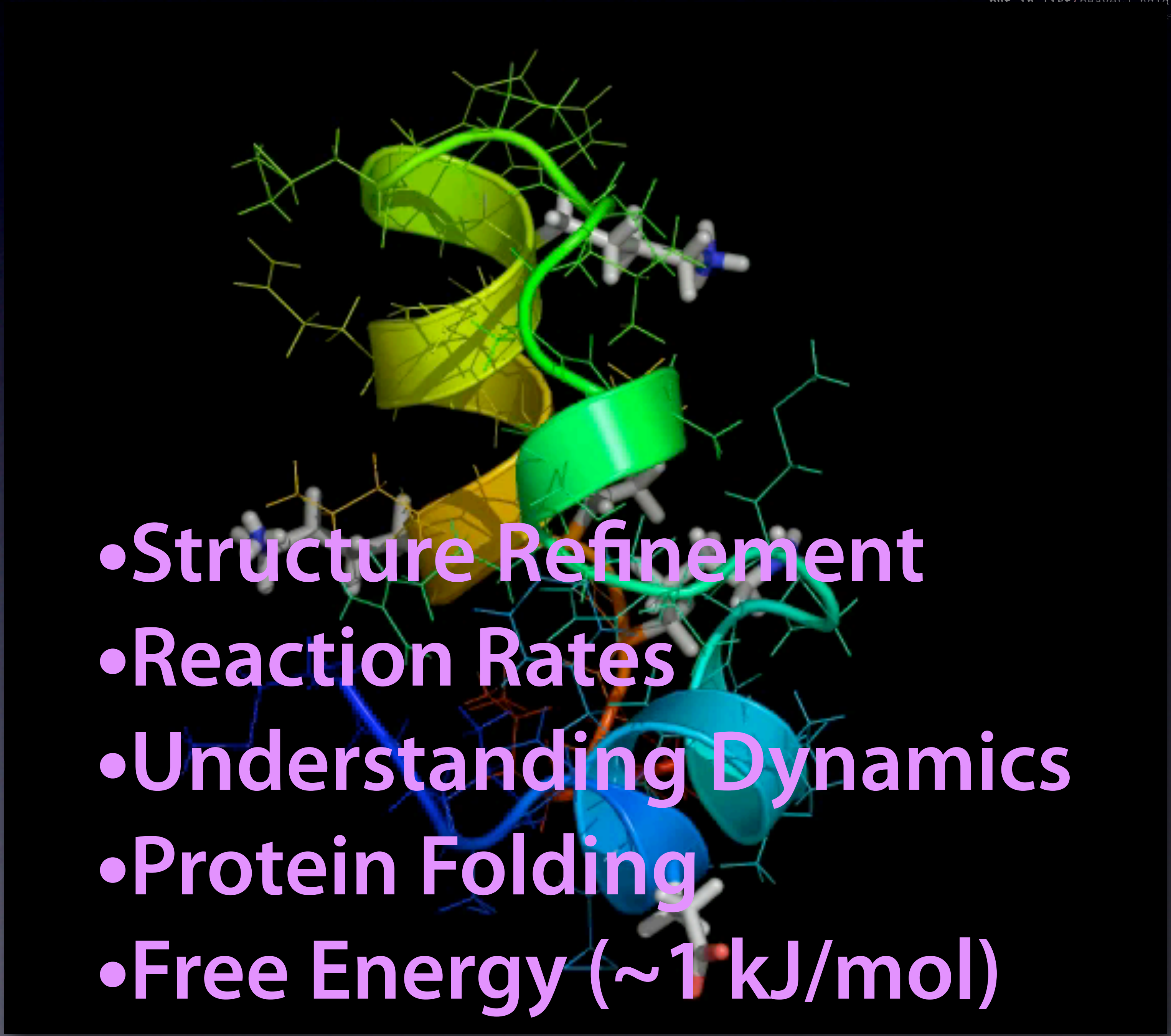
$$\begin{aligned} V(r) = & \sum_{bonds} \frac{1}{2} k_{ij}^b \left(r_{ij} - r_{ij}^0\right)^2 \\ & + \sum_{angles} \frac{1}{2} k_{ijk}^\theta \left(\theta_{ijk} - \theta_{ijk}^0\right)^2 \\ & + \sum_{torsions} \left\{ \sum_n k_\phi \left[1 + \cos\left(n\phi - \phi_0\right)\right] \right\} \\ & + \sum_{impropers} k_\xi \left(\xi_{ijkl} - \xi_{ijkl}^0\right) \\ & + \sum_{i,j} \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}} \\ & + \sum_{i,j} \left[\frac{C_{12}}{r_{ij}^{12}} - \frac{C_6}{r_{ij}^6} \right] \end{aligned}$$

- Timestep has to be small (fs)
- Forces depend on all particle coordinates in the system
- Generates a system *trajectory* over time

MD in Practice

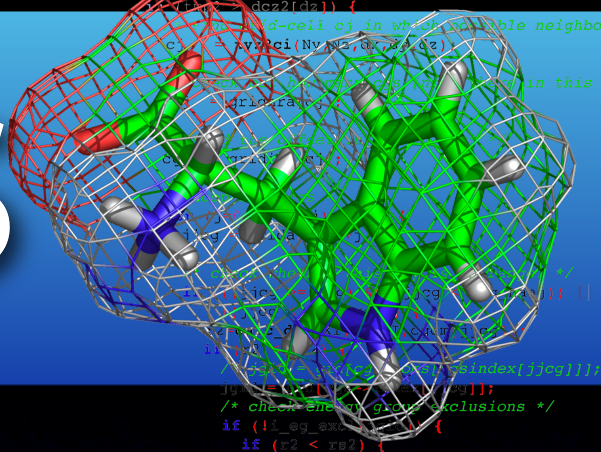


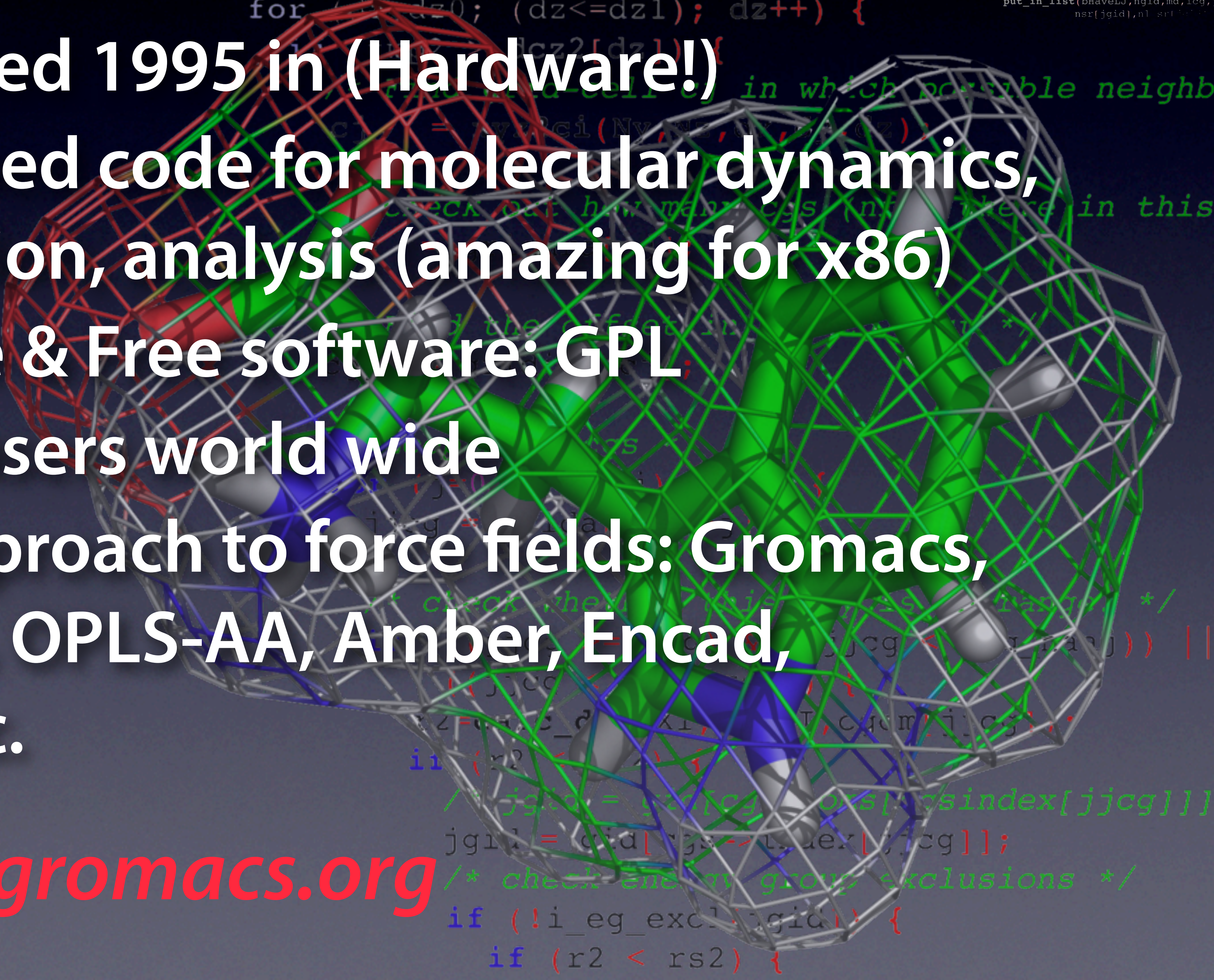

Initial input data:
Interaction function V(**r**) - "force field"
coordinates **r**, velocities **v**
Compute potential V(**r**) and forces $\mathbf{F}_i = \nabla_i V(r)$ on atoms
Update coordinates & velocities according to equations of motion
Collect statistics and write energy/coordinates to trajectory files
More steps?
Yes
No
Done!
Repeat for millions of steps



- Structure Refinement
- Reaction Rates
- Understanding Dynamics
- Protein Folding
- Free Energy (~1 kJ/mol)

Our baby: GROMACS





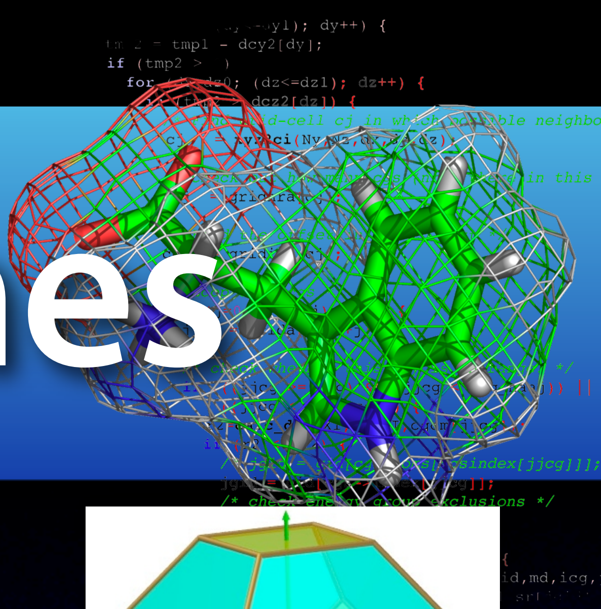
- Project started 1995 in (Hardware!)
  - Highly tuned code for molecular dynamics, minimization, analysis (amazing for x86)
- Open source & Free software: GPL
- 3000-5000 users world wide
- Agnostic approach to force fields: Gromacs, GROMOS96, OPLS-AA, Amber, Encad, Charmm, etc.

http://www.gromacs.org

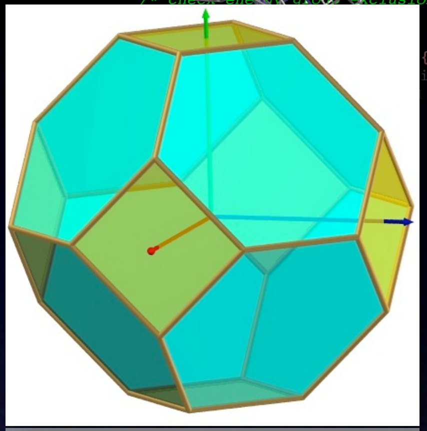
How can we push simulation performance into the “Biology” realm?

Requires a factor 100-1000 improvement

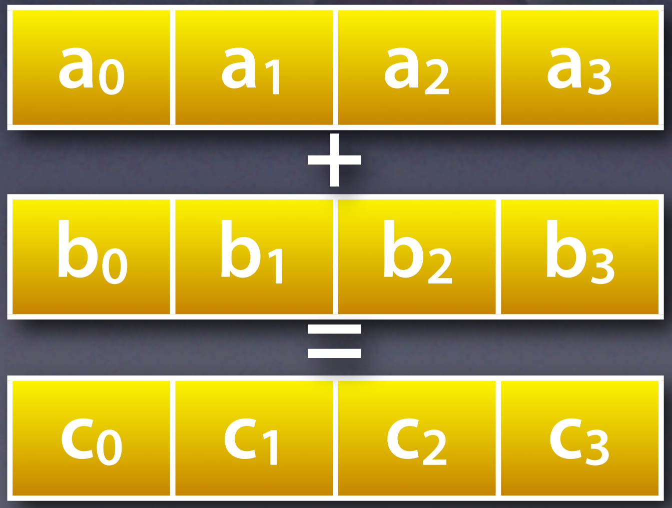
GROMACS Approaches



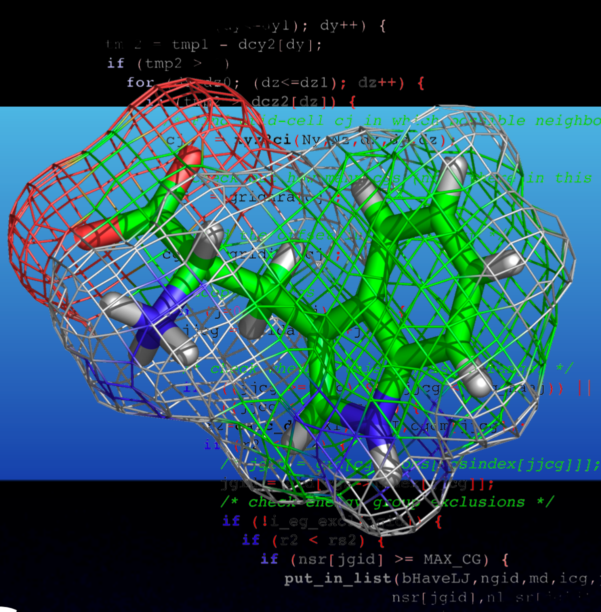
- Algorithmic optimization:
  - No virial in nonbonded kernels
  - Single precision by default (cache, BW usage)
  - Tuning to avoid conditional statements such as PBC checks
  - Triclinic cells everywhere: can save 15-20% on system size



- Optimized 1/sqrt(x)
  - Used ~150,000,000 times/sec
  - Handcoded asm for ia32, x86-64, ia64, Altivec, VMX, BlueGene (SIMD instructions)



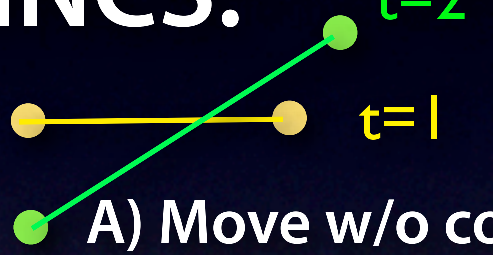
Constraints



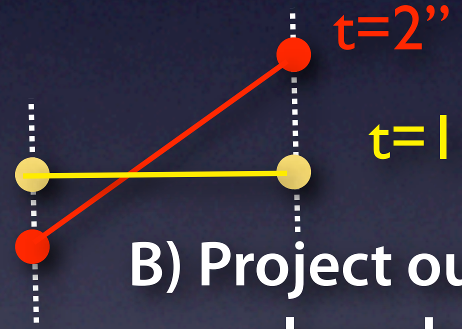
- Δt limited by fast motions - 1fs
  - Remove bond vibrations
- SHAKE (iterative, slow) - 2fs
  - Problematic in parallel (won't work)
  - Compromise: constrain h-bonds only - 1.4fs
- GROMACS (LINCS):
  - LINear Constraint Solver
  - *Approximate* matrix inversion expansion
  - Fast & stable - much better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

Nobody has yet implemented efficient parallel constraints!

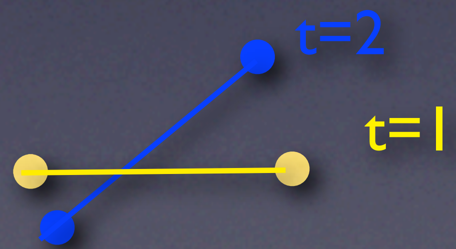
LINCS:



A) Move w/o constraint

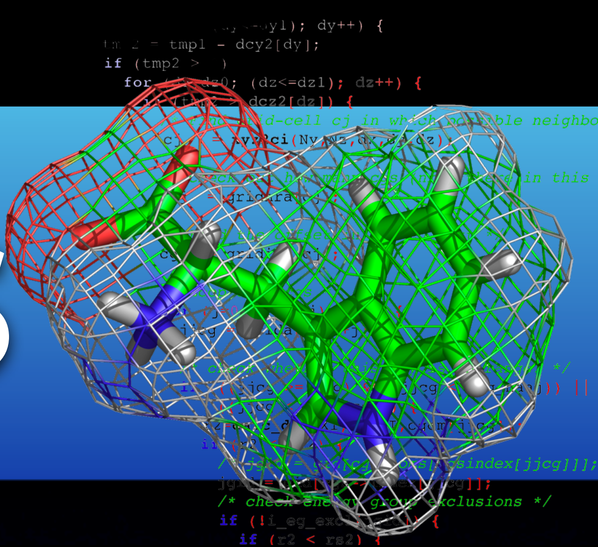


B) Project out motion along bonds

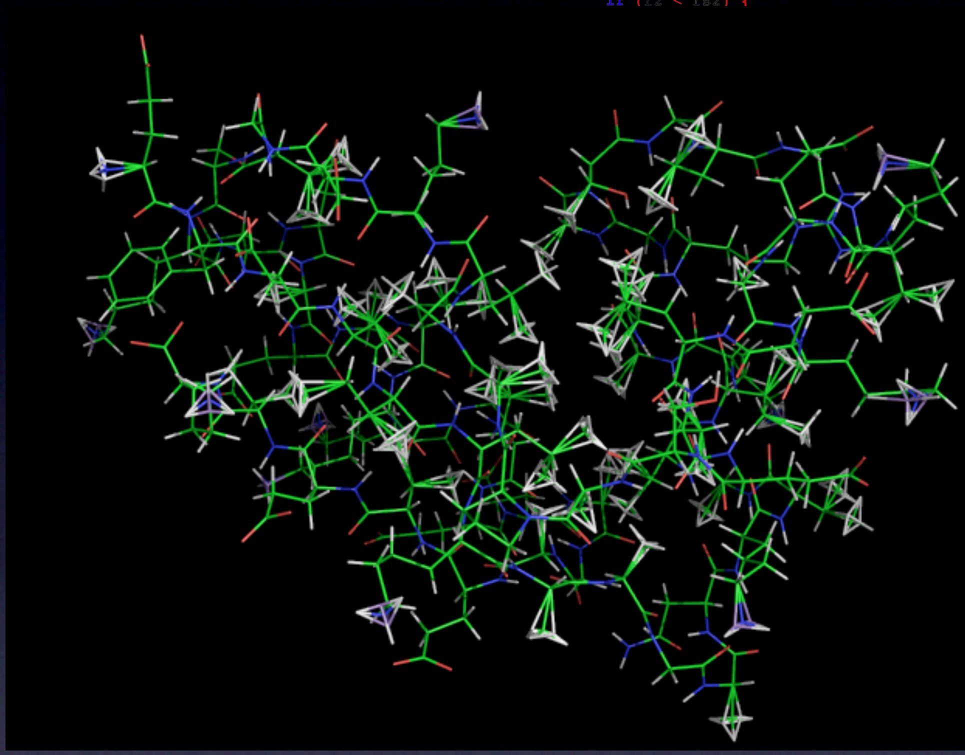


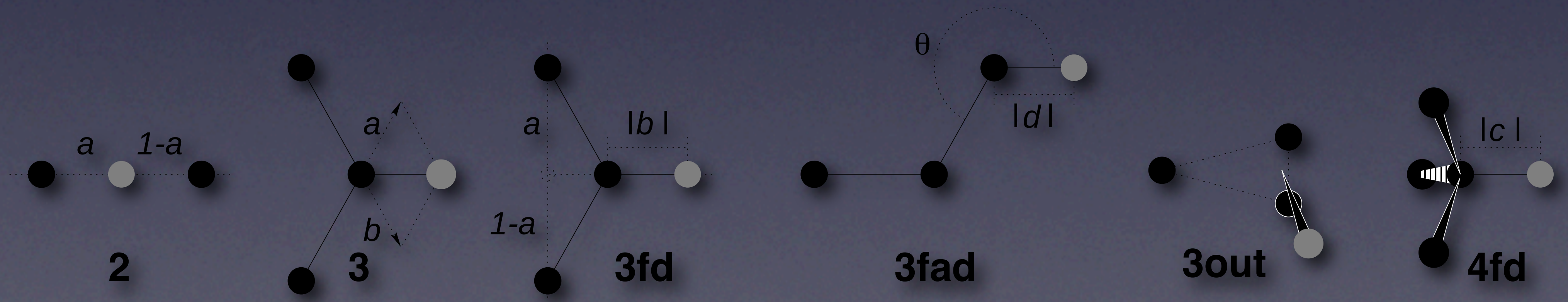
C) Correct for rotational extension of bond

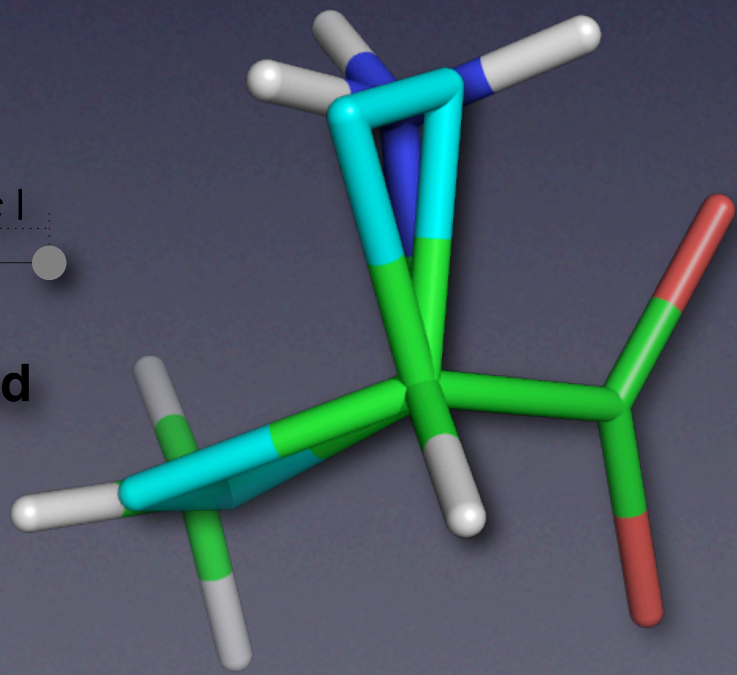
Better: Virtual sites



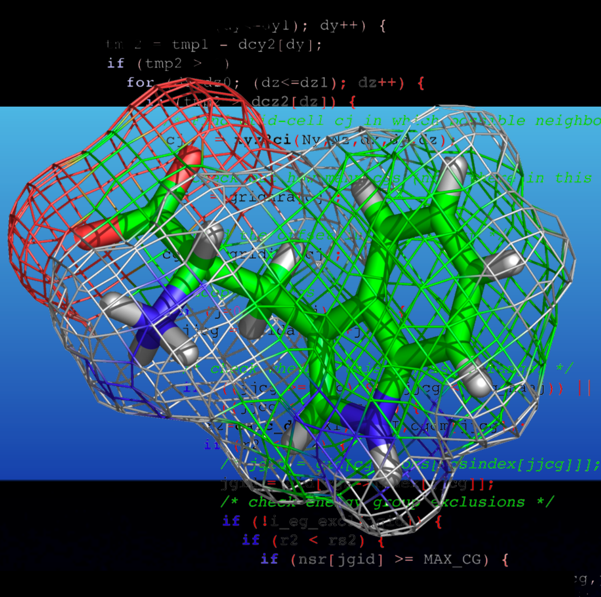
- Next fastest motions is H-angle and rotations of CH_3/NH_2 groups
- Try to remove them:
  - Ideal H position from heavy atoms.
  - CH_3/NH_2 groups are made rigid
  - Calculate forces, then project back onto heavy atoms
  - Integrate only heavy atom positions, reconstruct H's
- Enables 5fs timesteps!

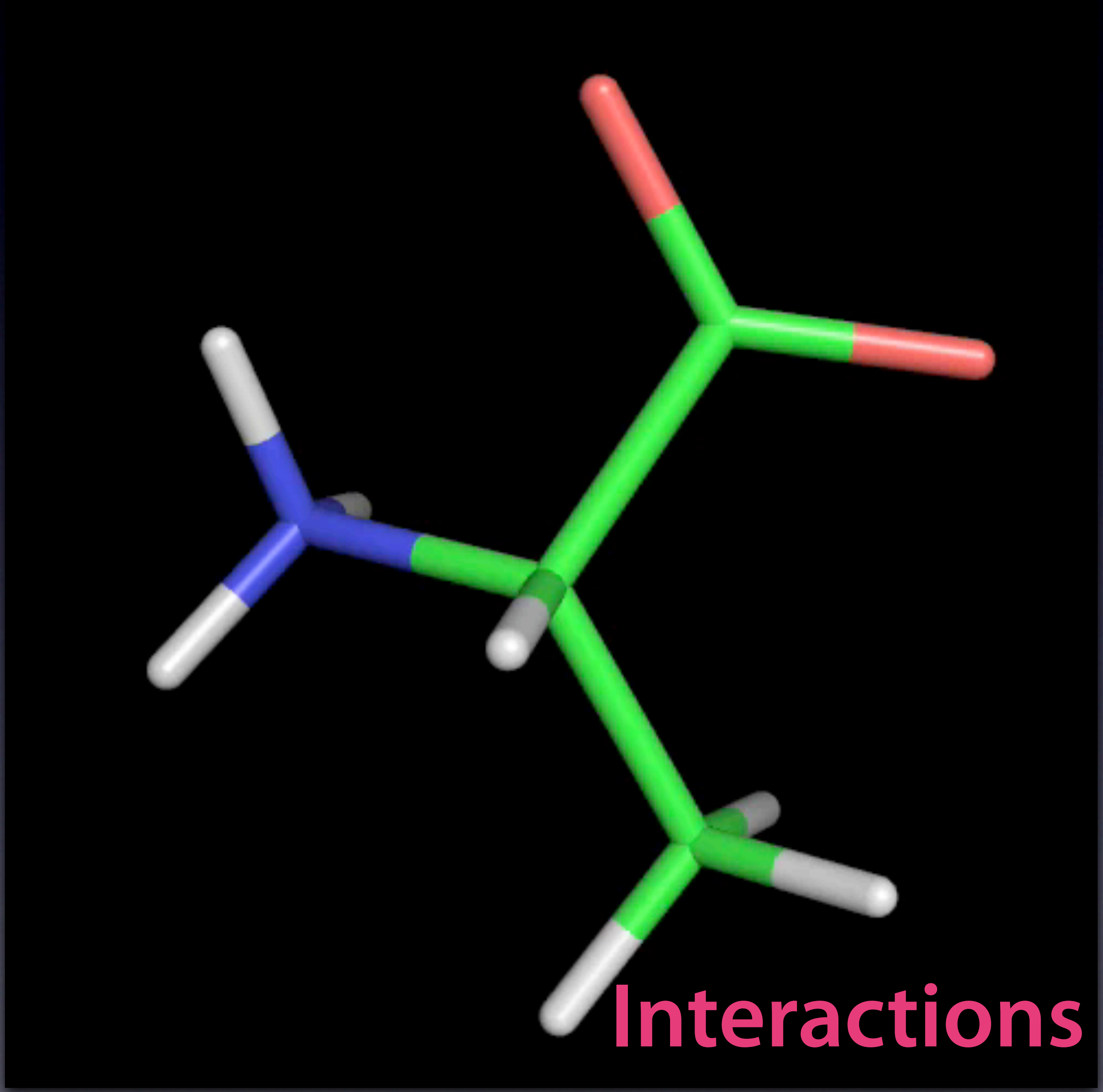




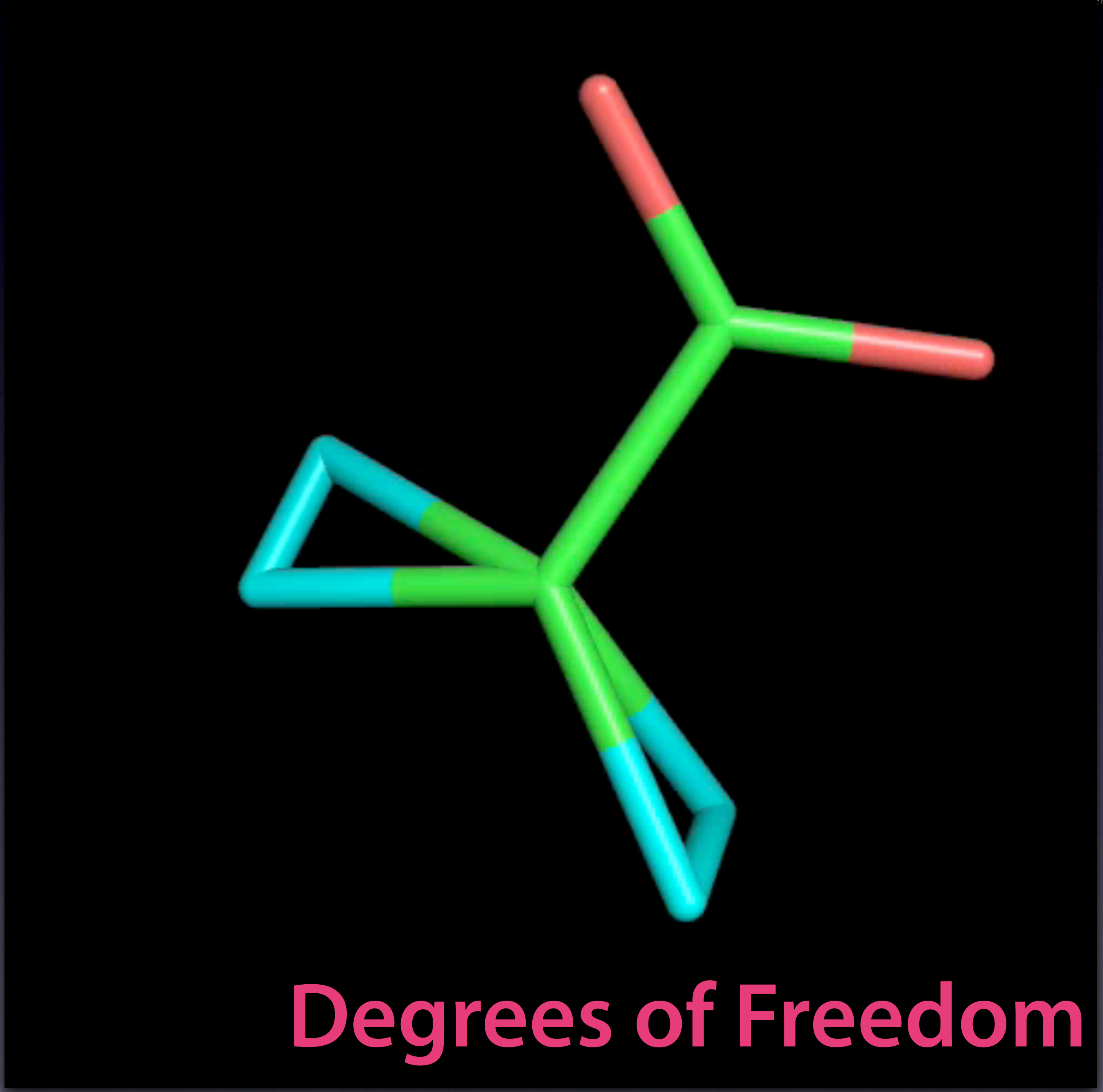


Virtual Hydrogens





Interactions

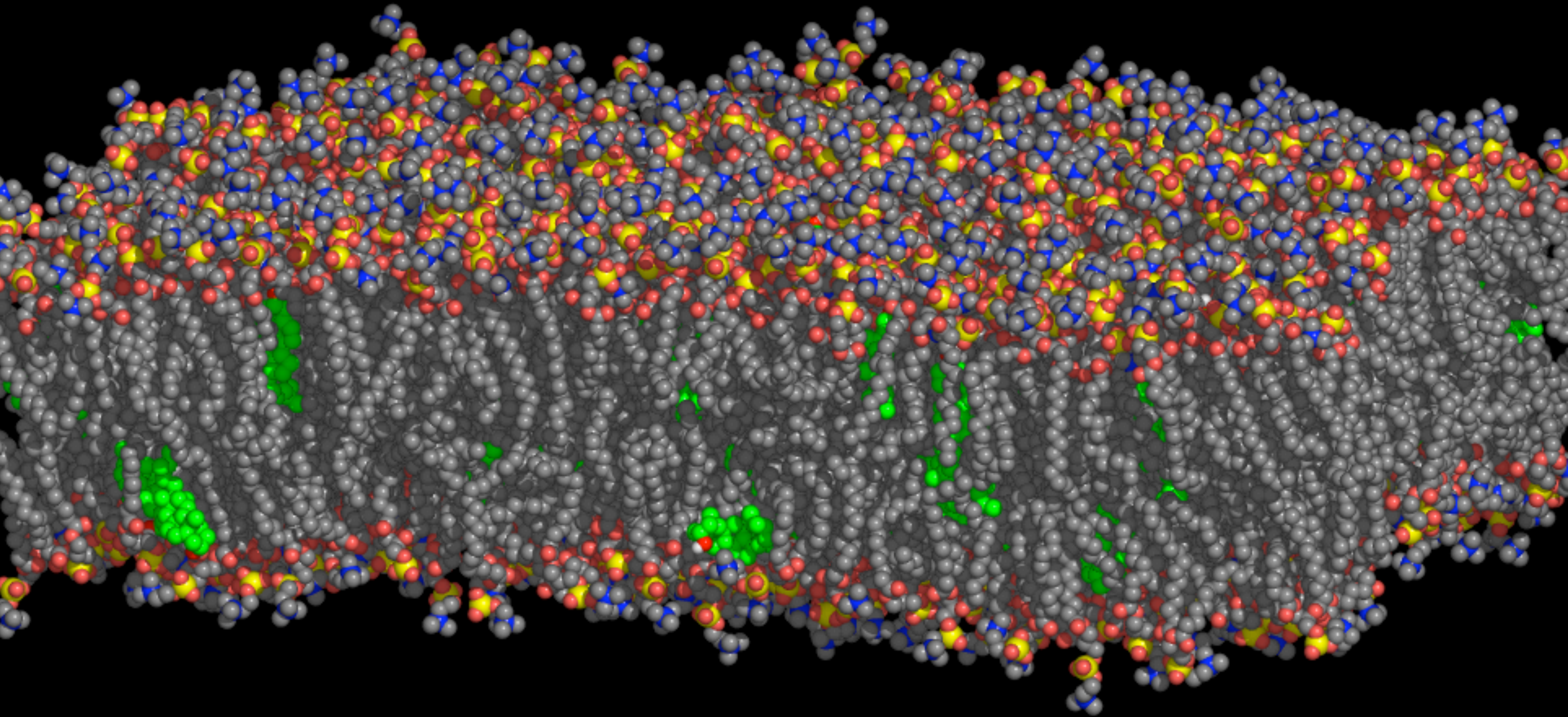


Degrees of Freedom

Scaling?

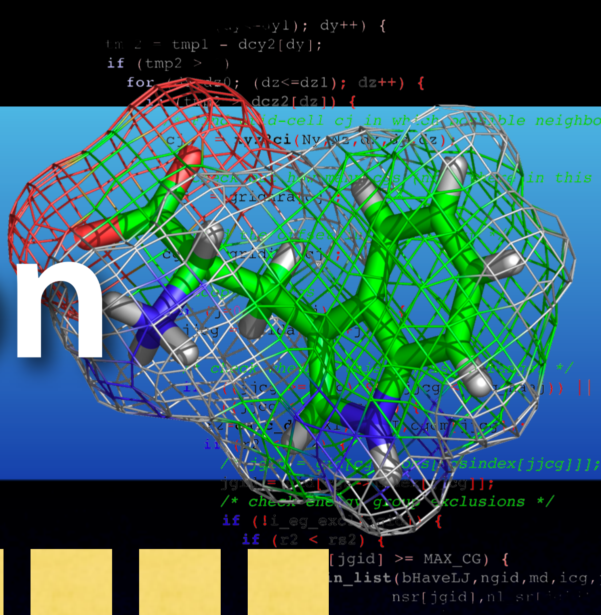
DPPC & Cholesterol
130k atoms

Blue Gene/L & Blue Matter:
scales to 3 atoms/CPU
~10ns/day on 8192 CPUs



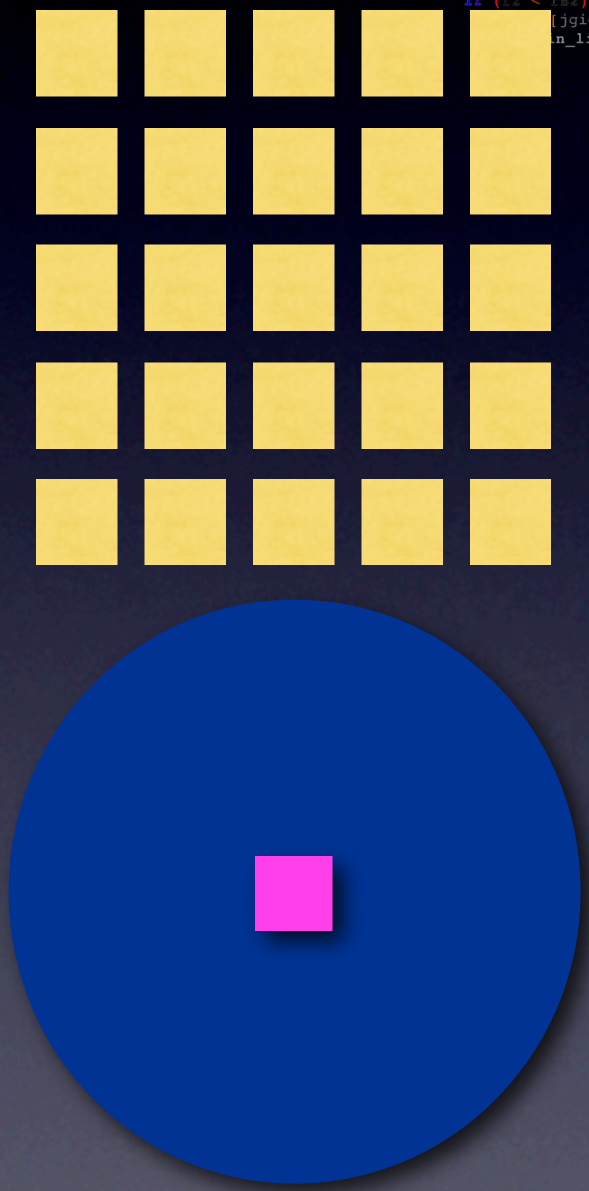
GROMACS: 2ns/day

...on a single dual
dual-core Opteron!

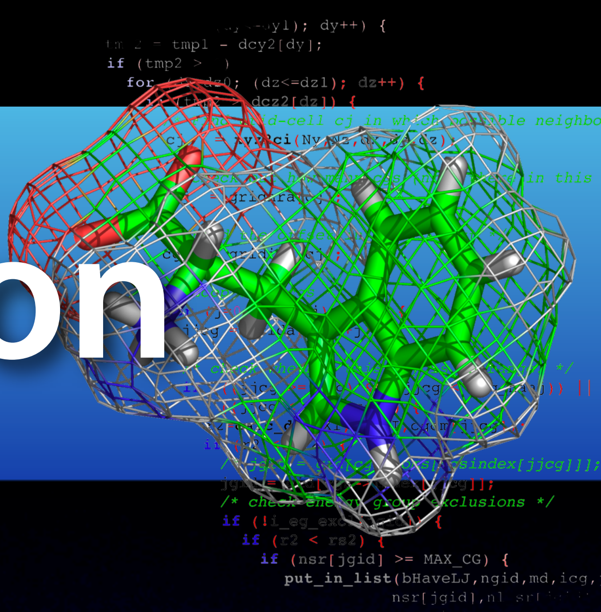


Classical Decomposition

- Partition space, instead of atoms, over nodes
- Previously used in Gromacs (v 3.3)
- Good for load balancing
- Bad for communication bandwidth
- Each node ‘imports’ coordinate and exports forces from neighbors within a sphere with radius=cutoff (expensive)

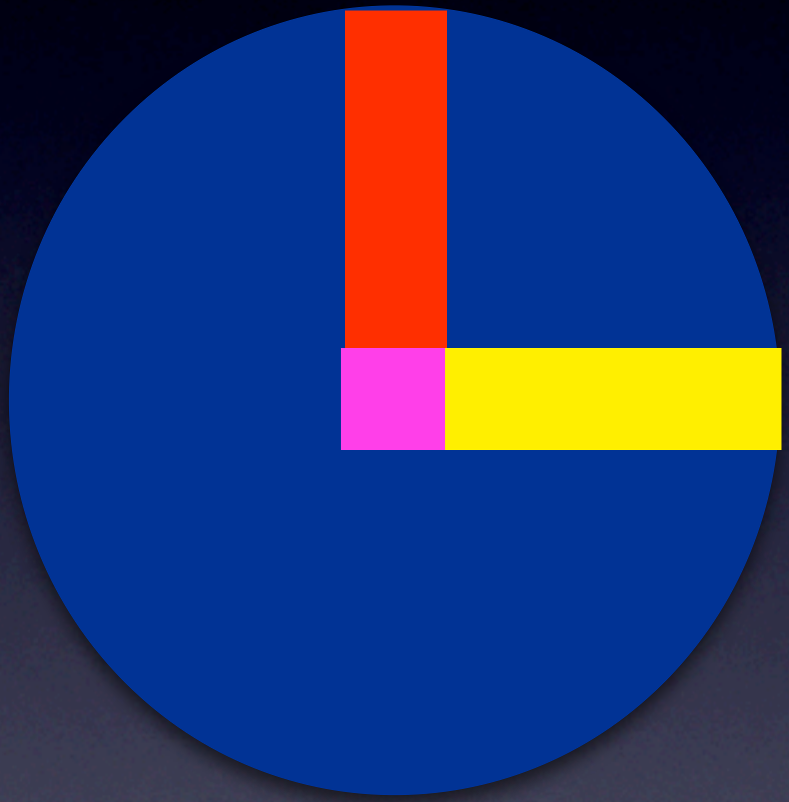


Data must be imported from whole sphere, although it can be optimized to half



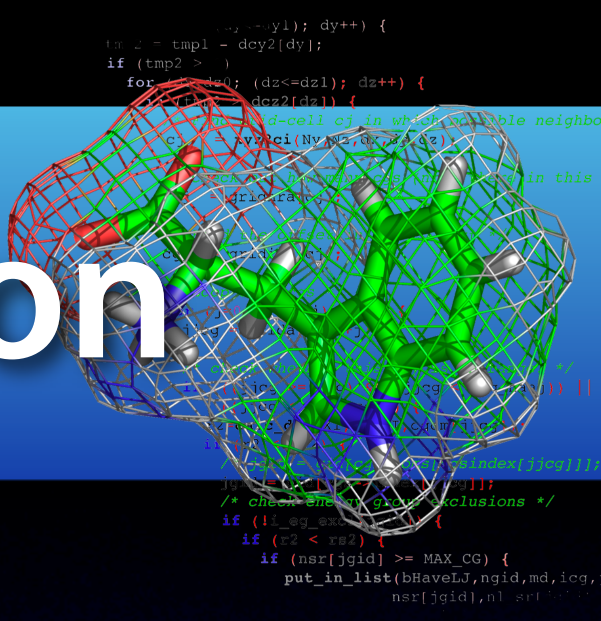
8th-sphere decomposition

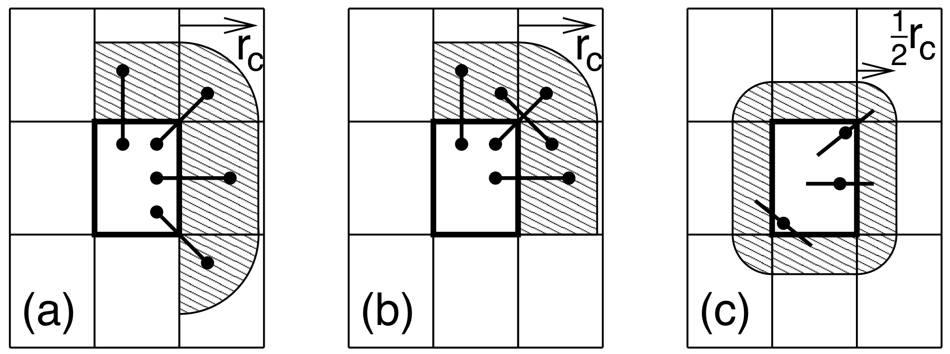
- Smarter: Don’t calculate interactions on a home node, but on “neutral territory”
- Drastically reduces communication bandwidth needs - see 2D example
- In 3D, we need to import data from 1/8 sphere to the central cell
- Working in Gromacs CVS version



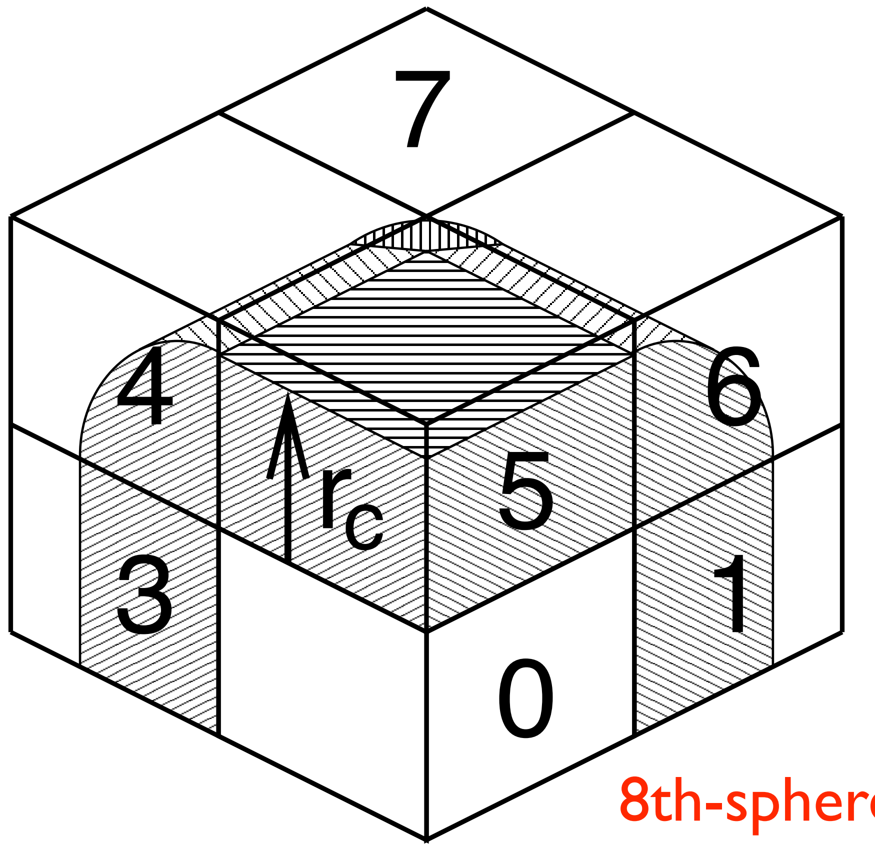
Red/Yellow cells send coordinates to central (purple) cell, where interactions are calculated, and then forces are sent back

8th-sphere decomposition

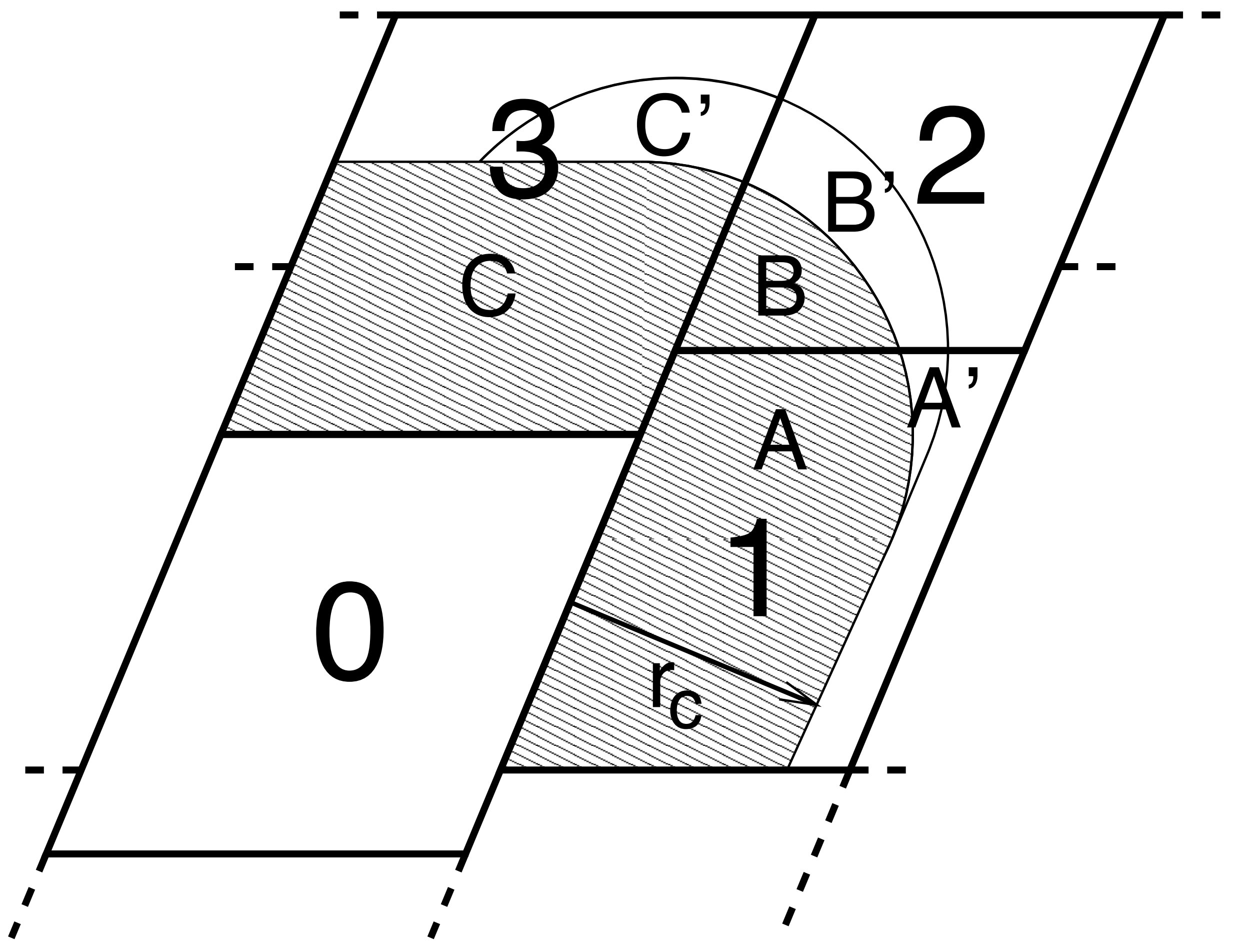




half-shell "8th-sphere" midpoint



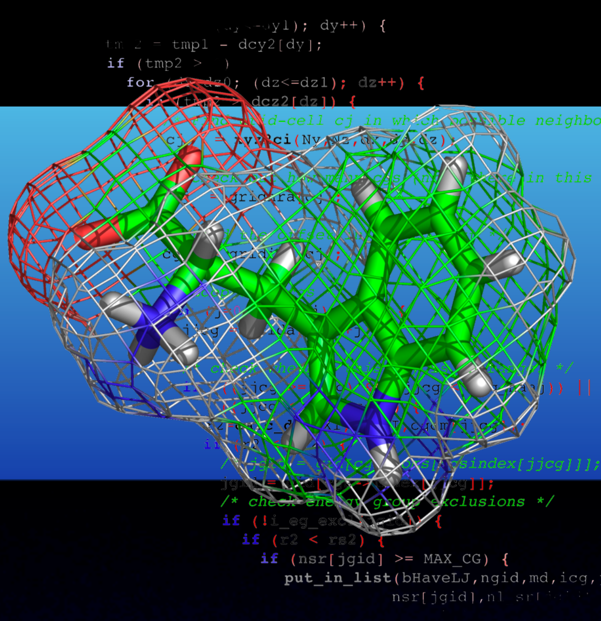
8th-sphere



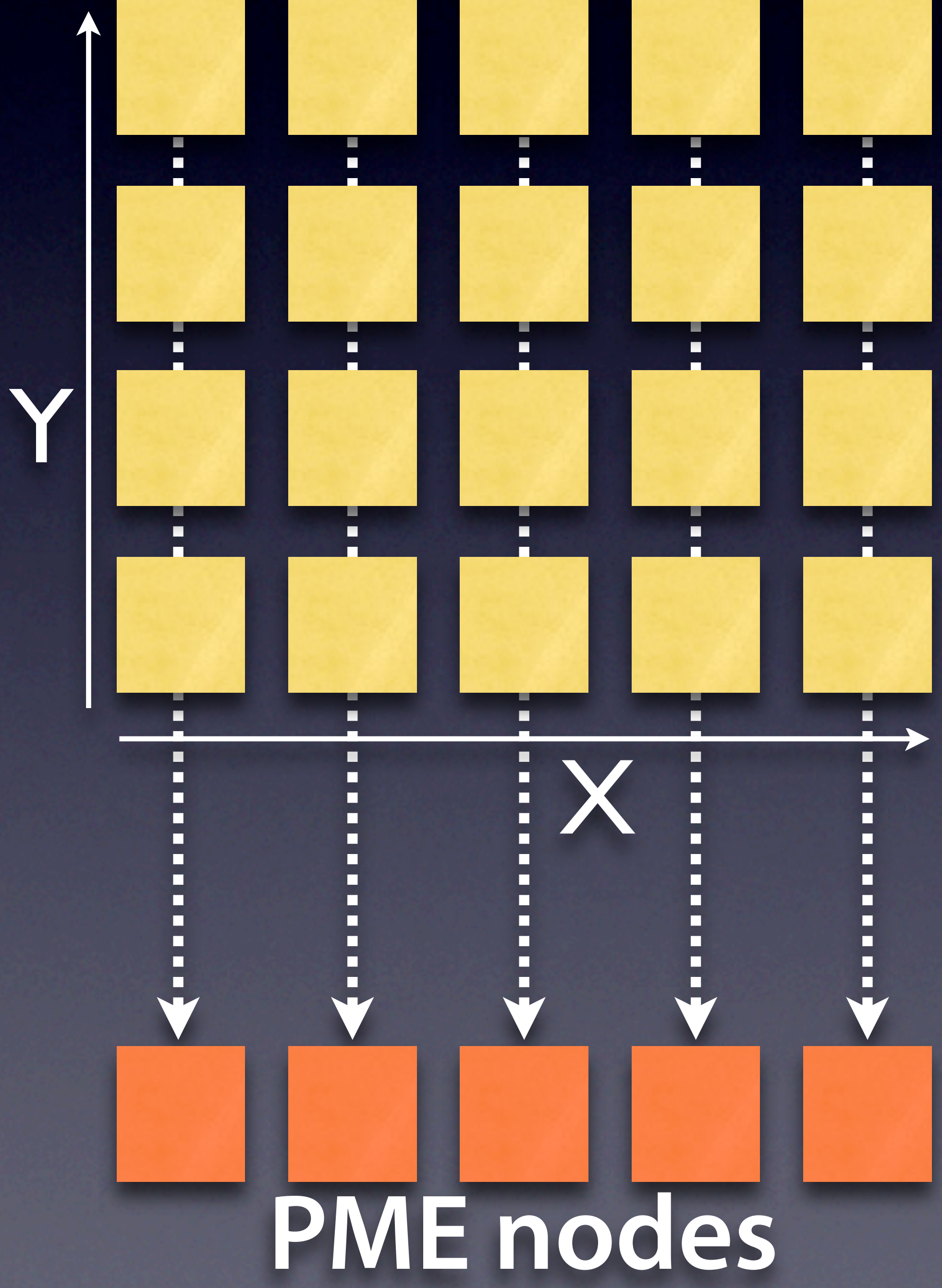
Dynamic load balancing in 2D

Complicated (but it's working!) in 3D

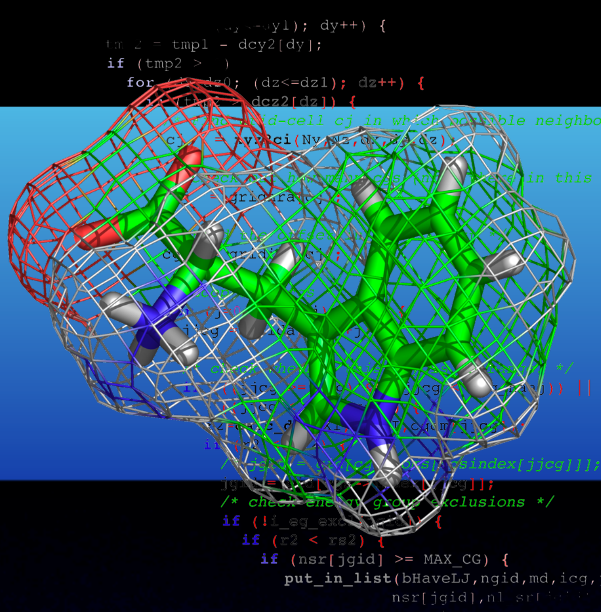
MPMD Revisited



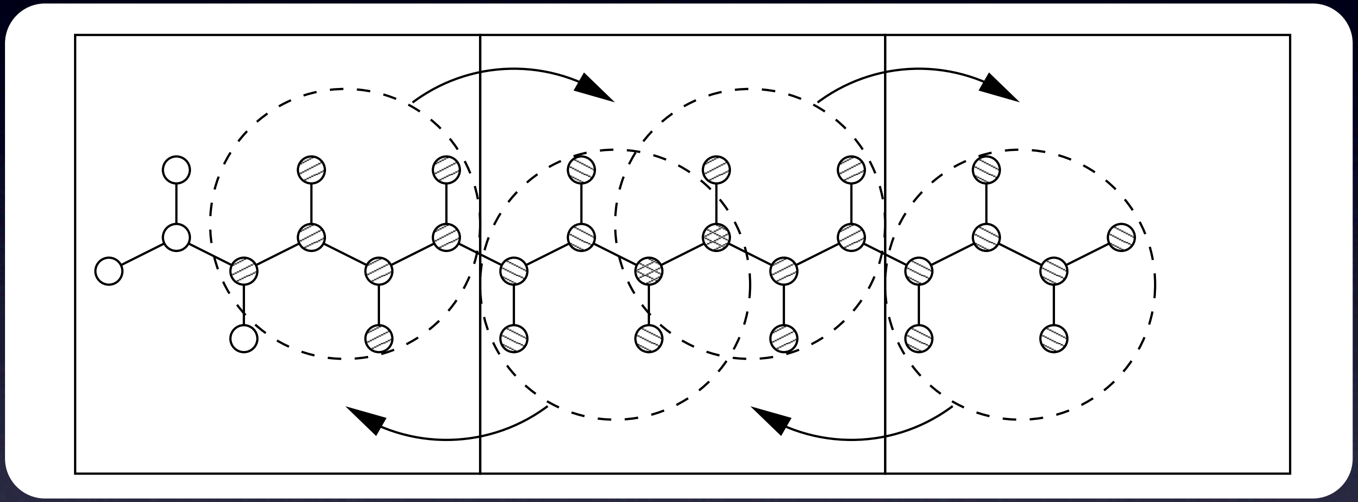
- PME = rapid Ewald summation
- Ubiquitous in simulations today
- Small 3D Fourier Transforms scale bad - all-to-all communication
- Real space & PME are independent
- Dedicate a subset of nodes to run a separate PME-only version of the program to improve scaling
- FFT over 5 instead of 25 nodes!

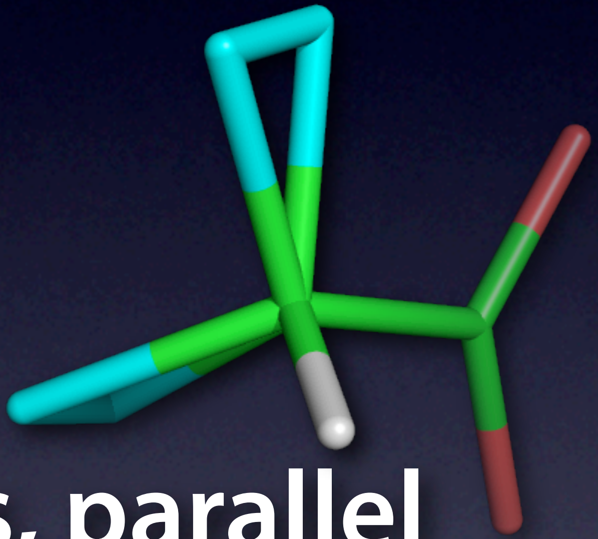


GROMACS 4



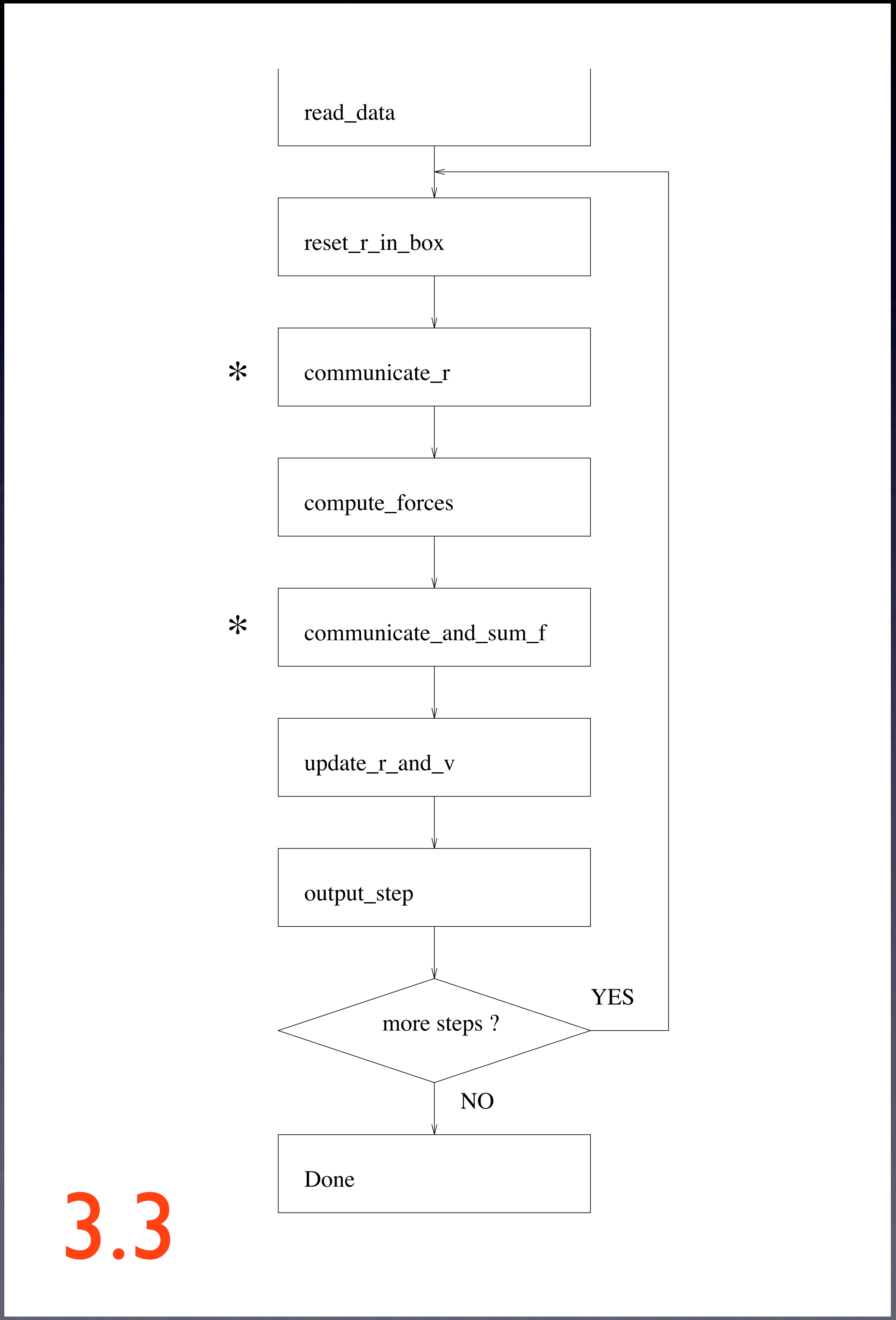
- Holonomic parallel constraints: P-LINCS



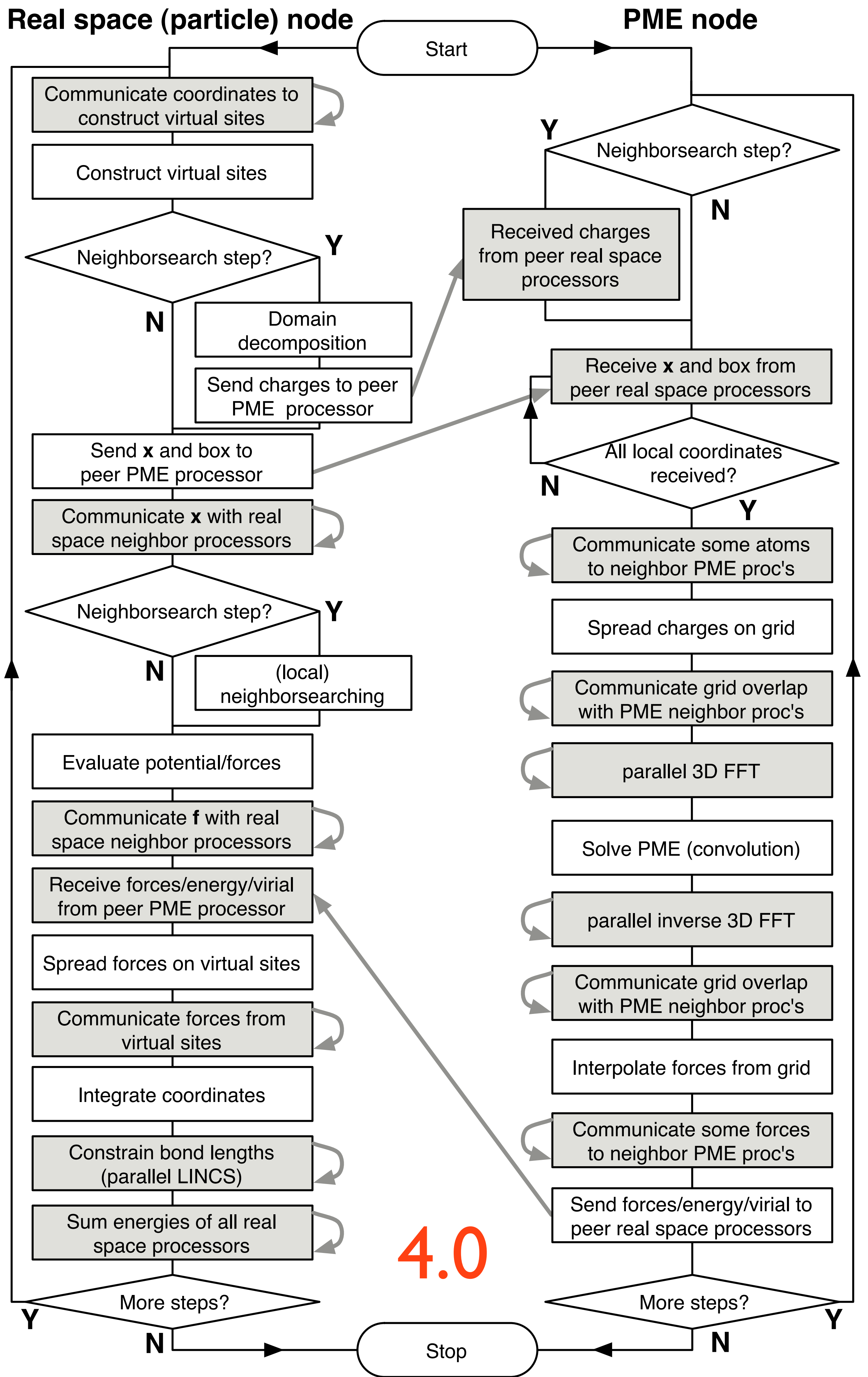


- Virtual site hydrogens & 5fs timesteps, parallel
- Automatic sorting for better caching
- Timestep counters on ~10 architectures
- Pulsed communication for Cray XT4 & IBM BG
- SIMD assembly for BlueGene double hummers

Flowcharts

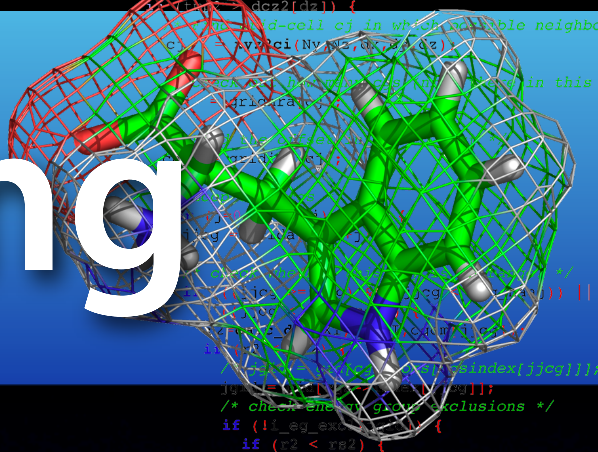


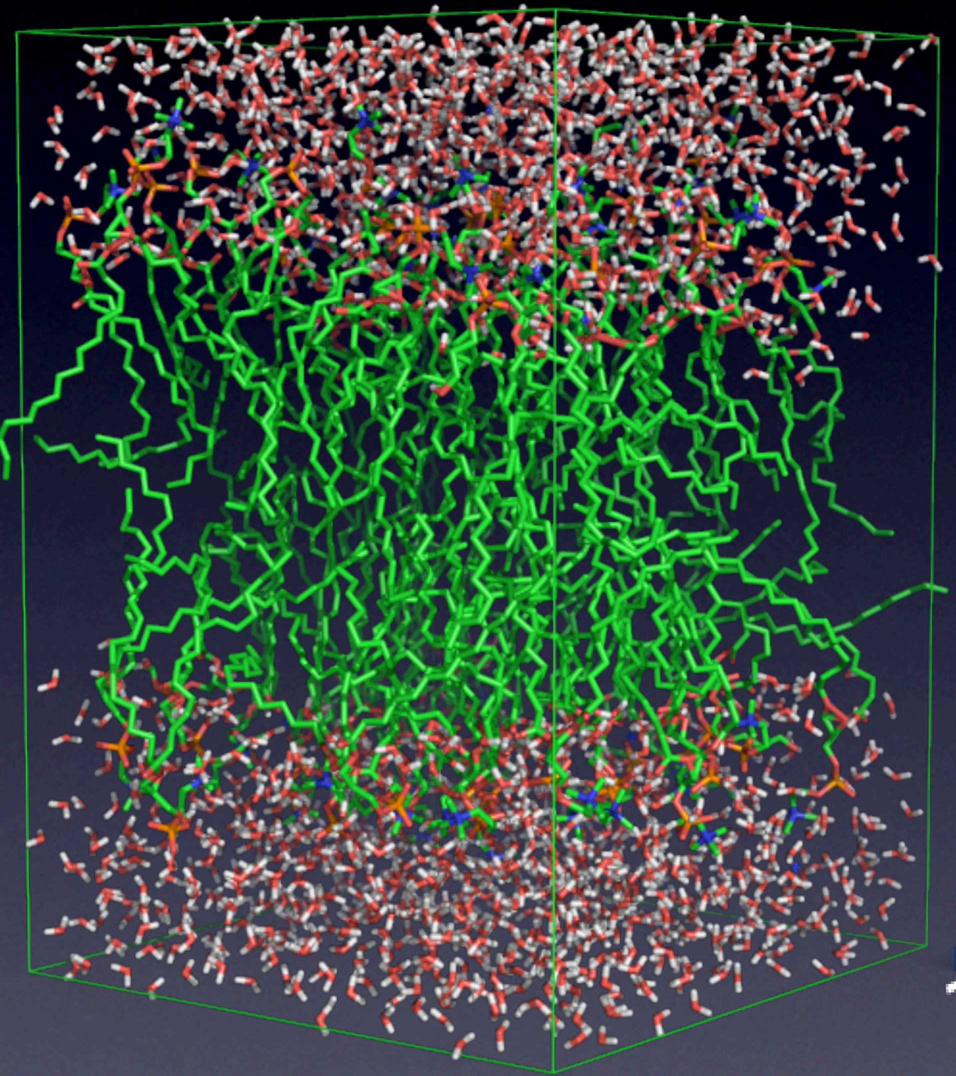
read_data
reset_r_in_box
*
communicate_r
compute_forces
*
communicate_and_sum_f
update_r_and_v
output_step
more steps ?
YES
NO
Done
3.3



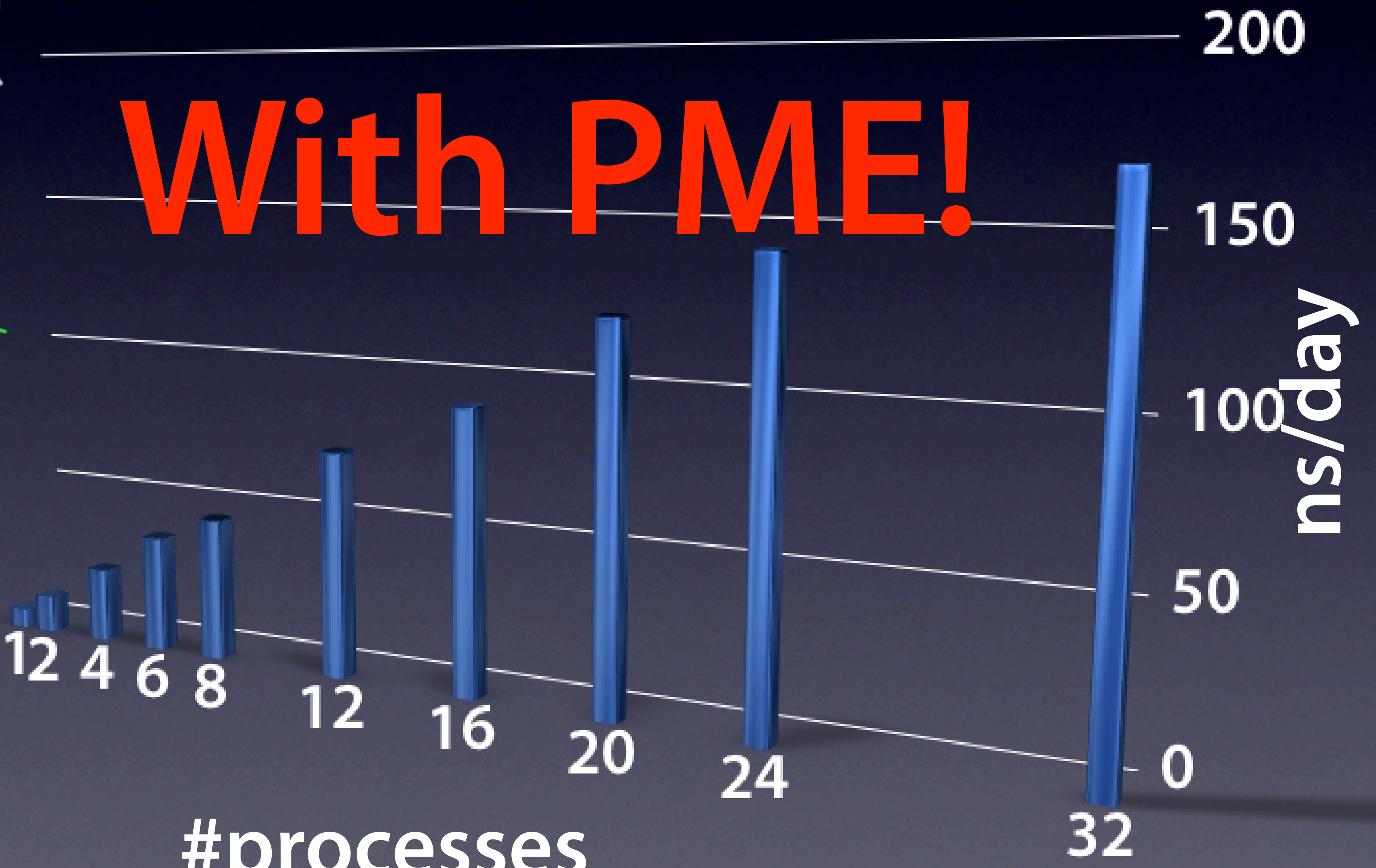
Real space (particle) node
Start
PME node
Communicate coordinates to construct virtual sites
Construct virtual sites
Neighborsearch step?
Y
N
Domain decomposition
Send charges to peer PME processor
Send **x** and box to peer PME processor
Communicate **x** with real space neighbor processors
Neighborsearch step?
Y
N
(local) neighborsearching
Evaluate potential/forces
Communicate **f** with real space neighbor processors
Receive forces/energy/virial from peer PME processor
Spread forces on virtual sites
Communicate forces from virtual sites
Integrate coordinates
Constrain bond lengths (parallel LINCS)
Sum energies of all real space processors
More steps?
Y
N
Stop
Neighborsearch step?
Y
N
Received charges from peer real space processors
Receive **x** and box from peer real space processors
All local coordinates received?
N
Y
Communicate some atoms to neighbor PME proc's
Spread charges on grid
Communicate grid overlap with PME neighbor proc's
parallel 3D FFT
Solve PME (convolution)
parallel inverse 3D FFT
Communicate grid overlap with PME neighbor proc's
Interpolate forces from grid
Communicate some forces to neighbor PME proc's
Send forces/energy/virial to peer real space processors
More steps?
Y
N
4.0

GROMACS 4 Scaling



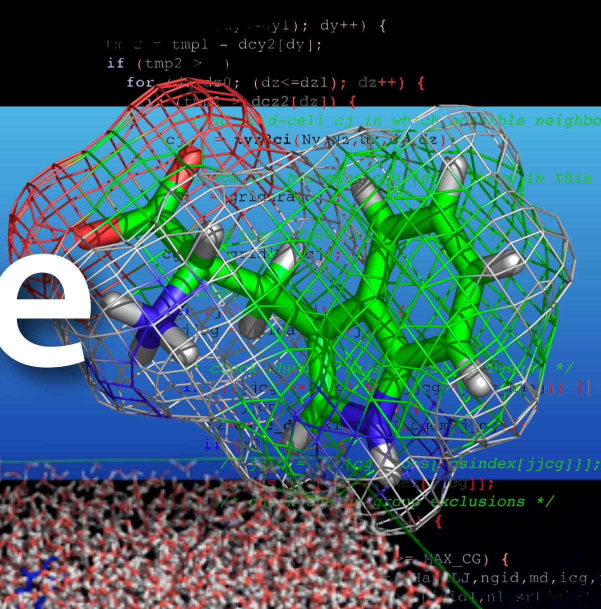


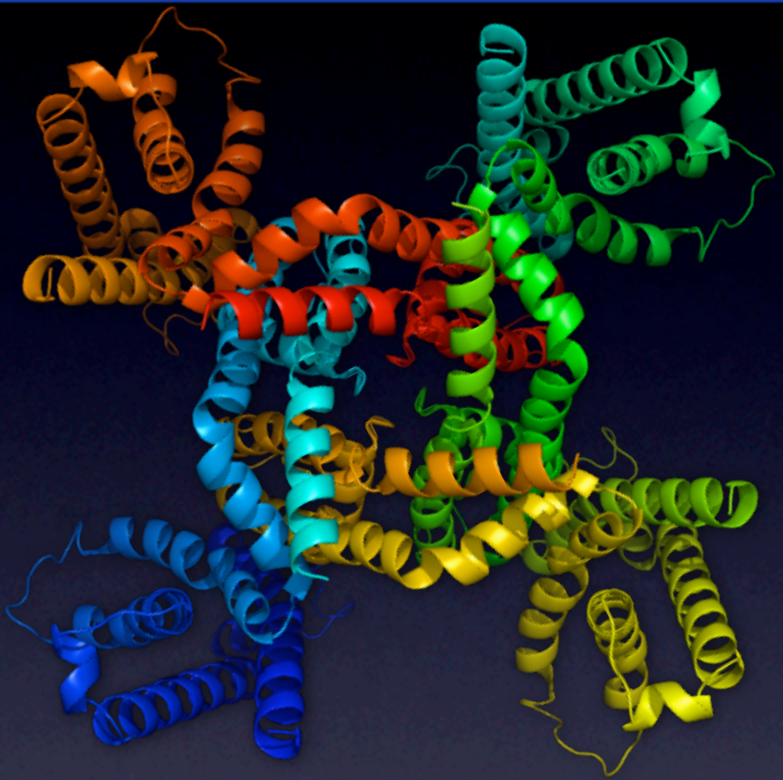
With PME!

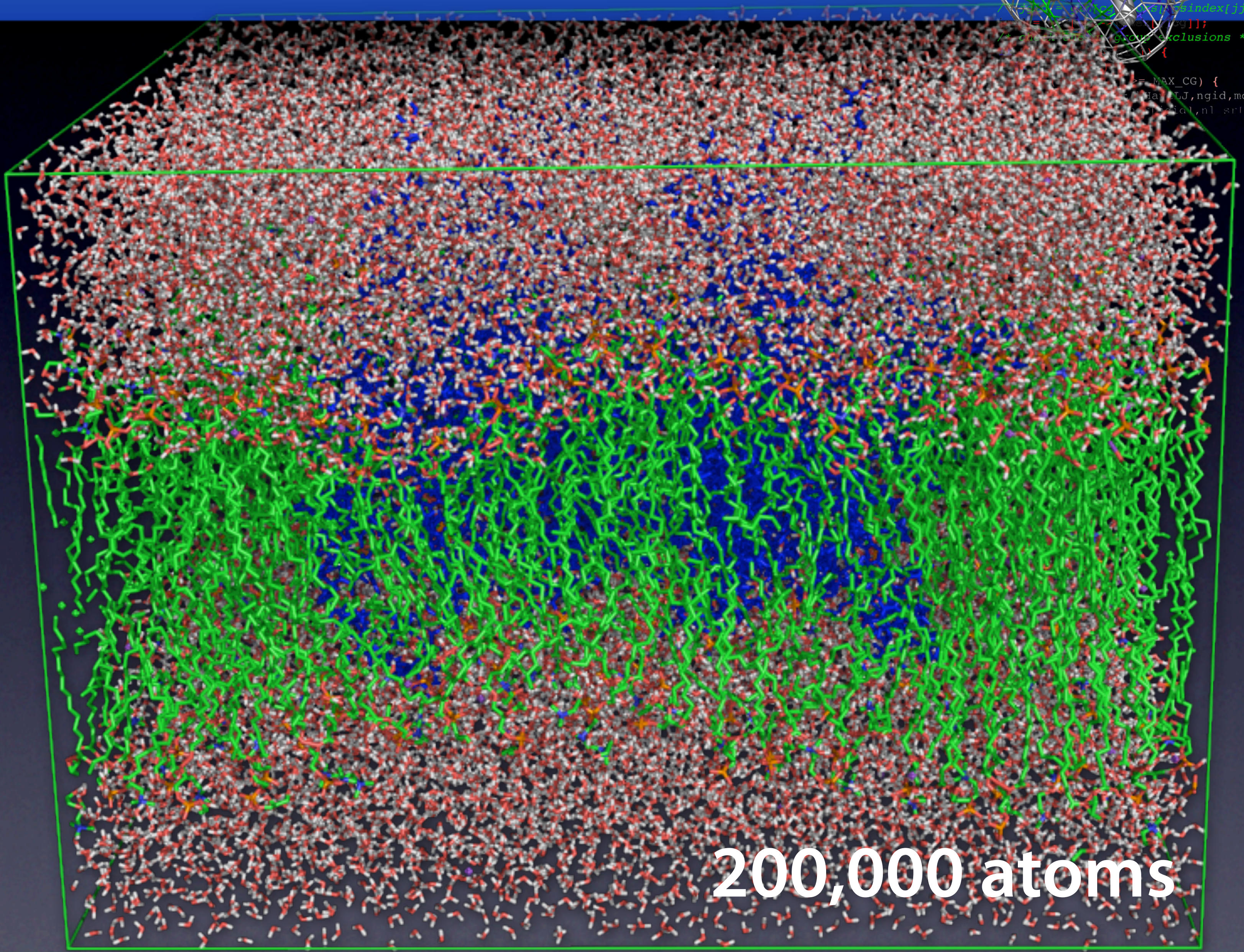


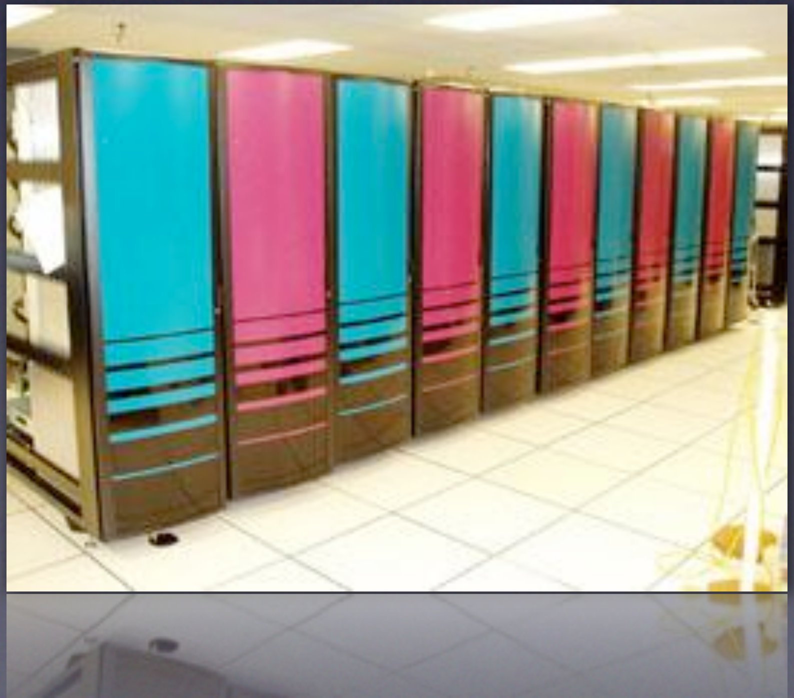
~280 atoms per core

Practical performance





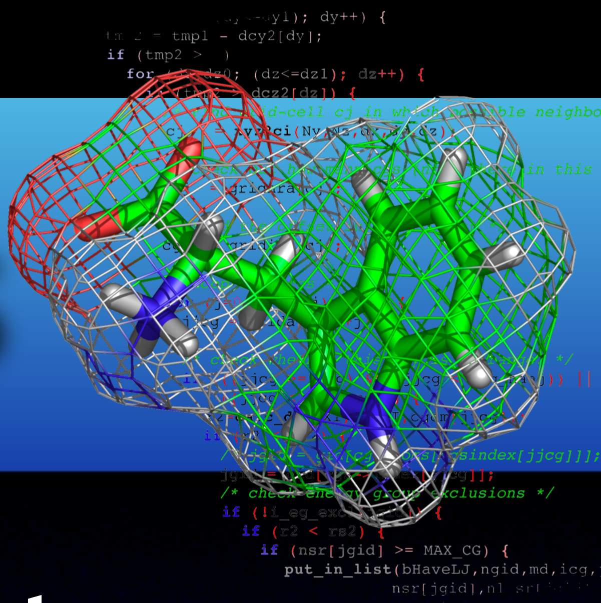




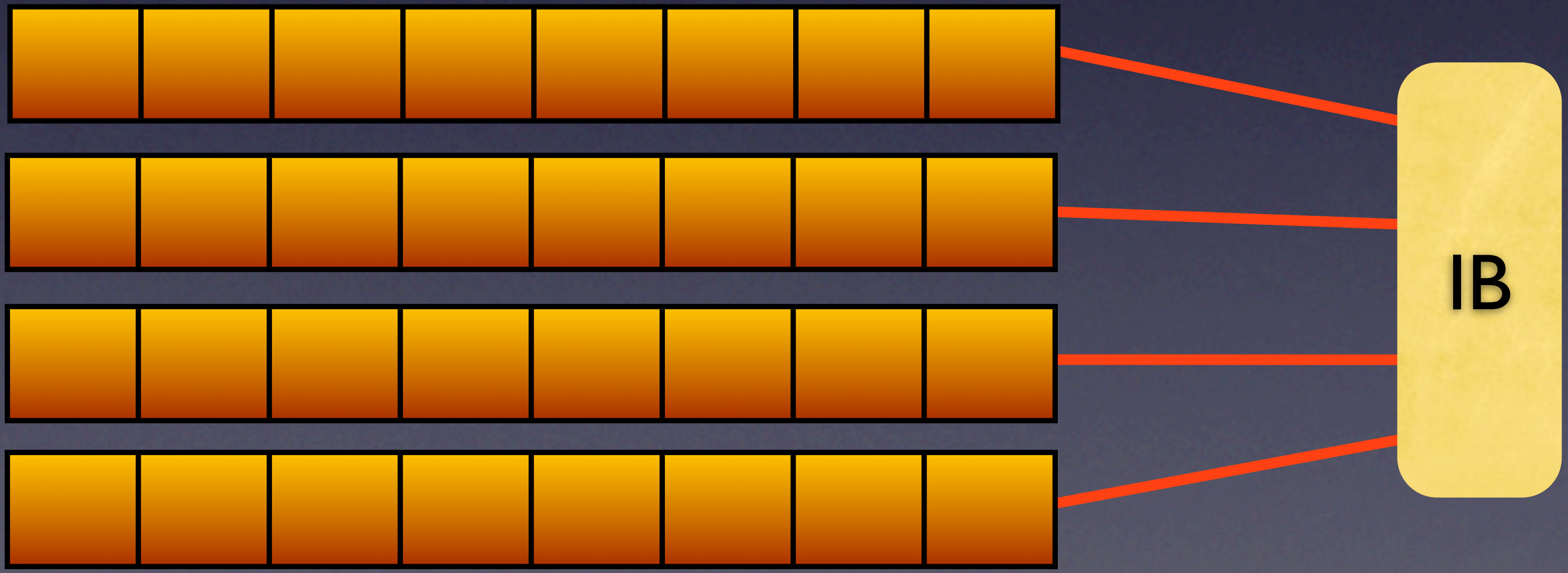
Cray XT4 @ CSC

1 µs in 3-4 weeks using 170 CPUs:
100X longer than state-of-the-art

SMP Node allocations

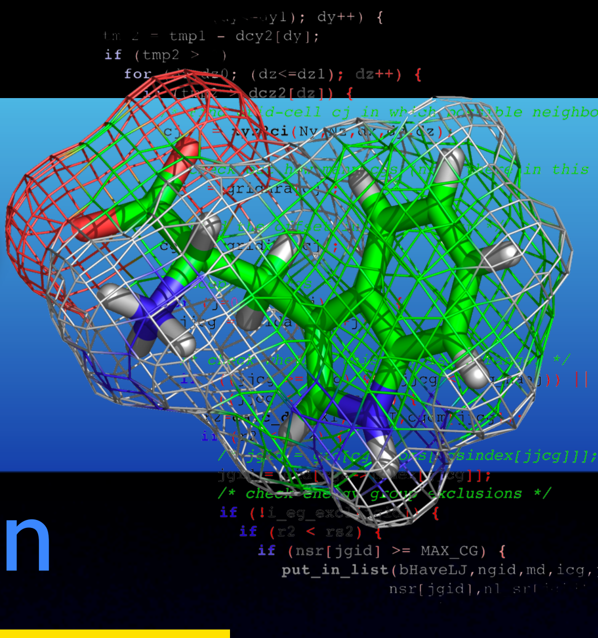


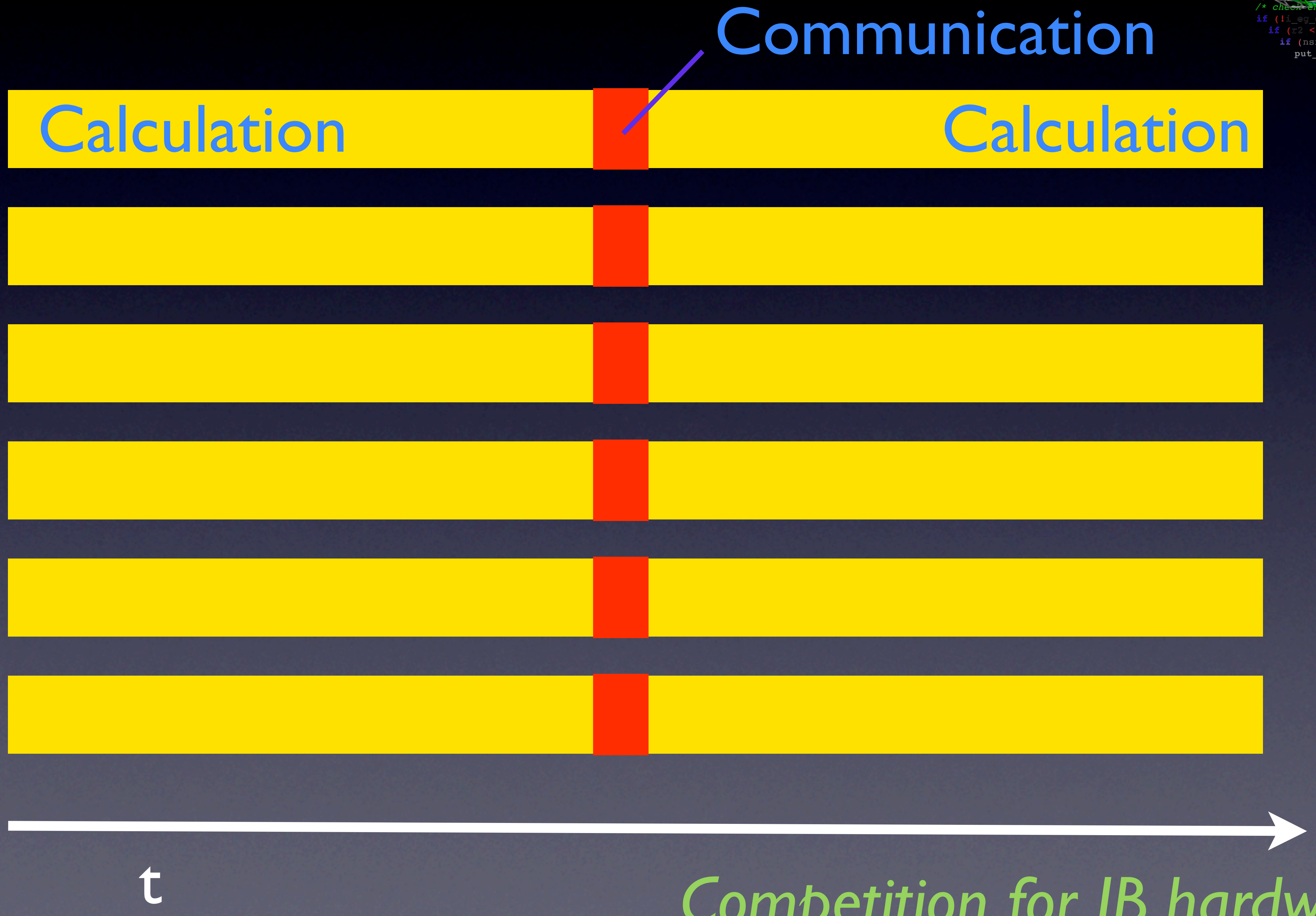
- MPI parallelization on Neolith-like systems
- Intra-node SMP bandwidth higher than IB
- Latency: Pack onto as few nodes as possible?



4 nodes * 8 cores = 32 processes: 120ns/day on test system

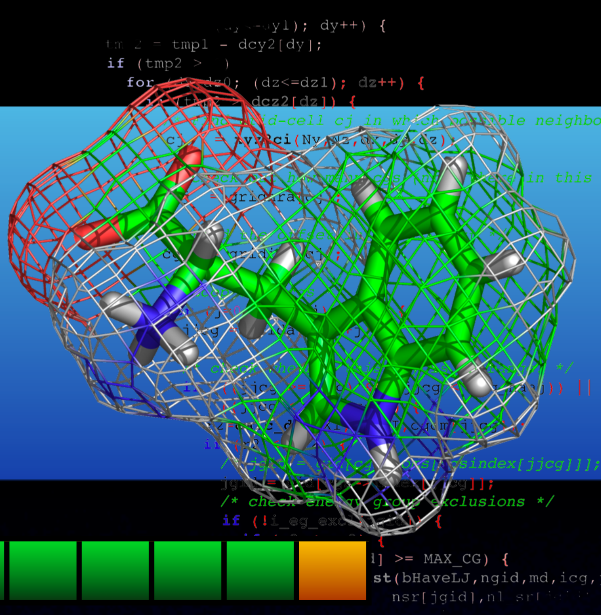
Typical job timeline?

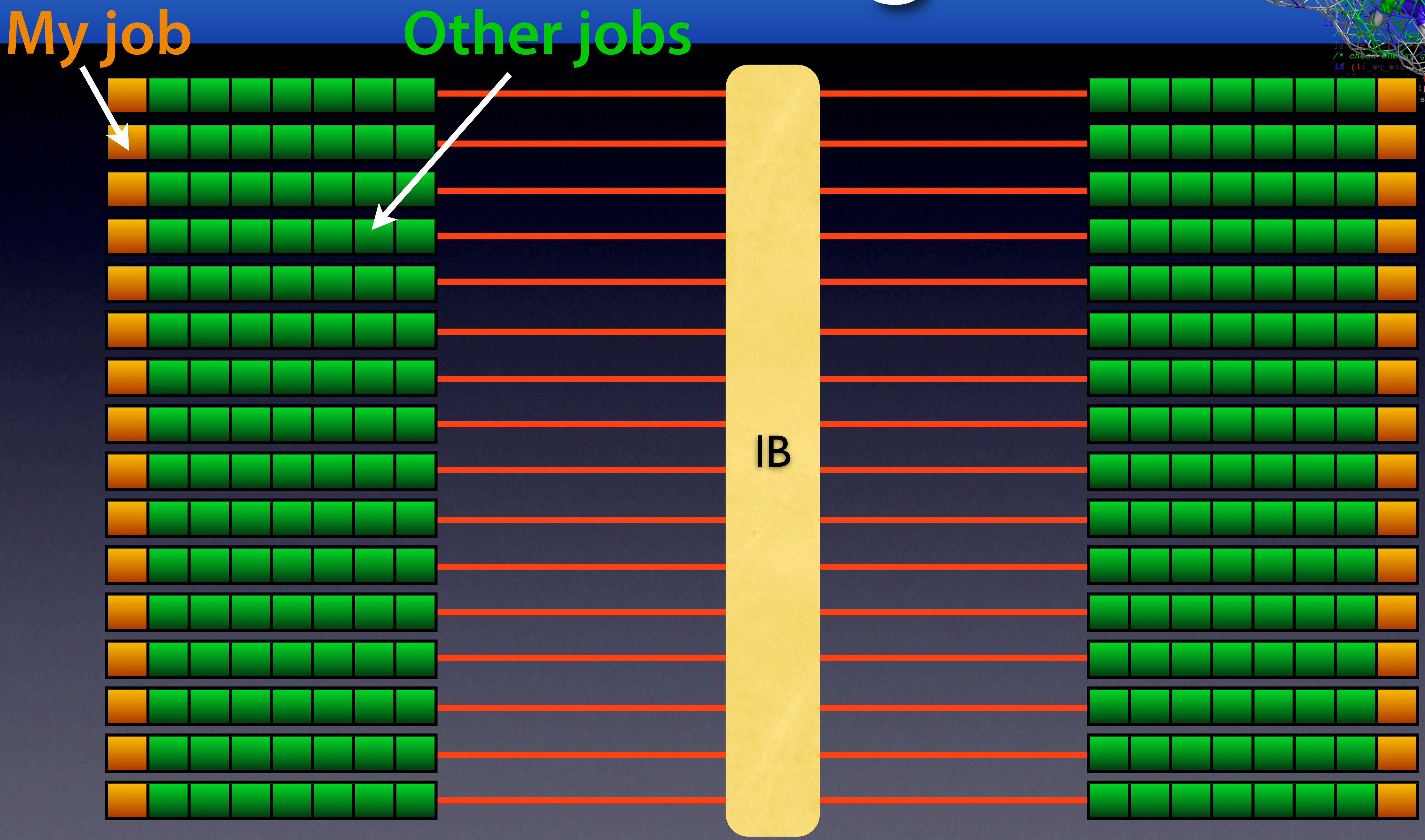




Competition for IB hardware!

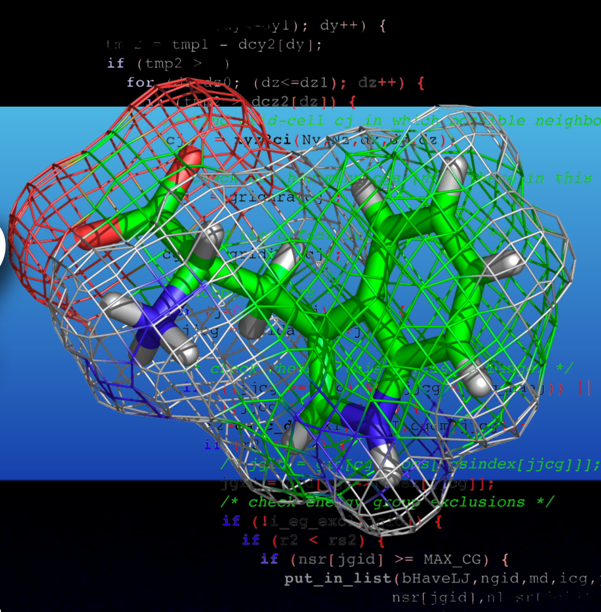
Better IB usage





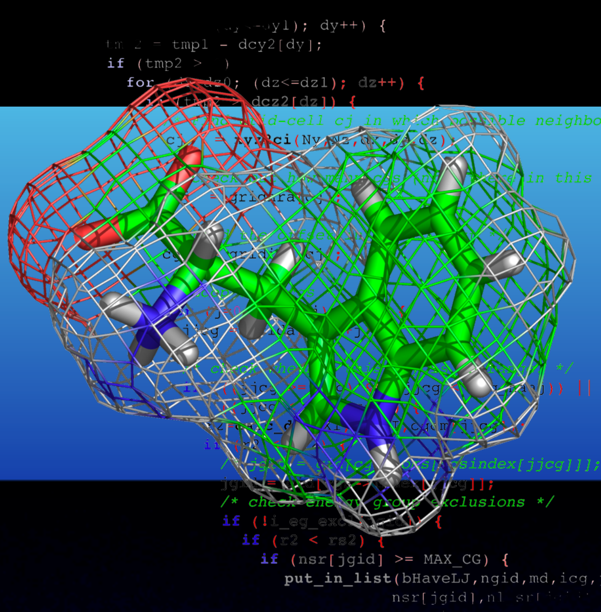
32 nodes * 1 core = 32 processes: 166ns/day on test system

Allocation policies?

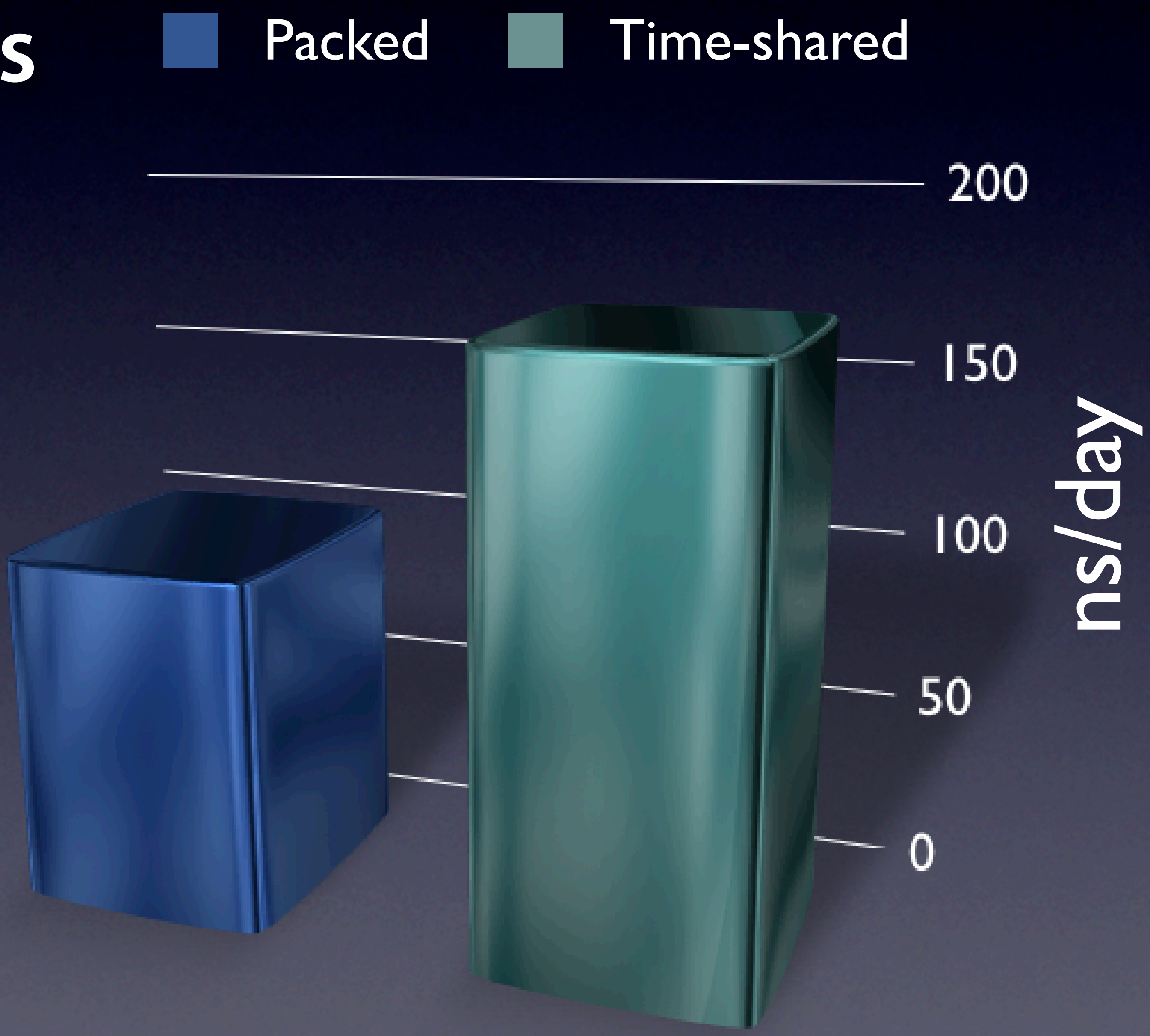


- Not only do we get better total throughput, but we even get 38% better single job performance by sharing!
- Share with everybody, not only yourself
  - Bad idea if your colleague is running STREAM
  - Very little problems in practice on a life science cluster (mix of MD, Bioinformatics, QM)
- 2 processes per node seem to be optimal for Gromacs
- Interleave direct and reciprocal space nodes in Gromacs
- The effect will depend on latency/bandwidth needs
- Haakon: This is best handled by the queue system :-)

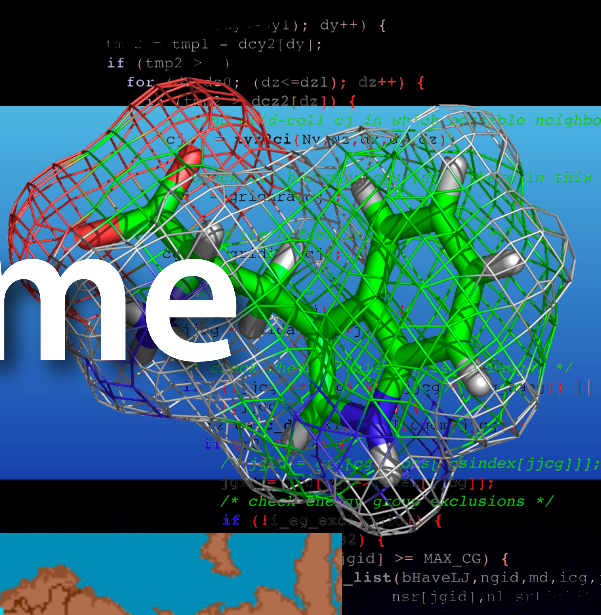
IB time-sharing

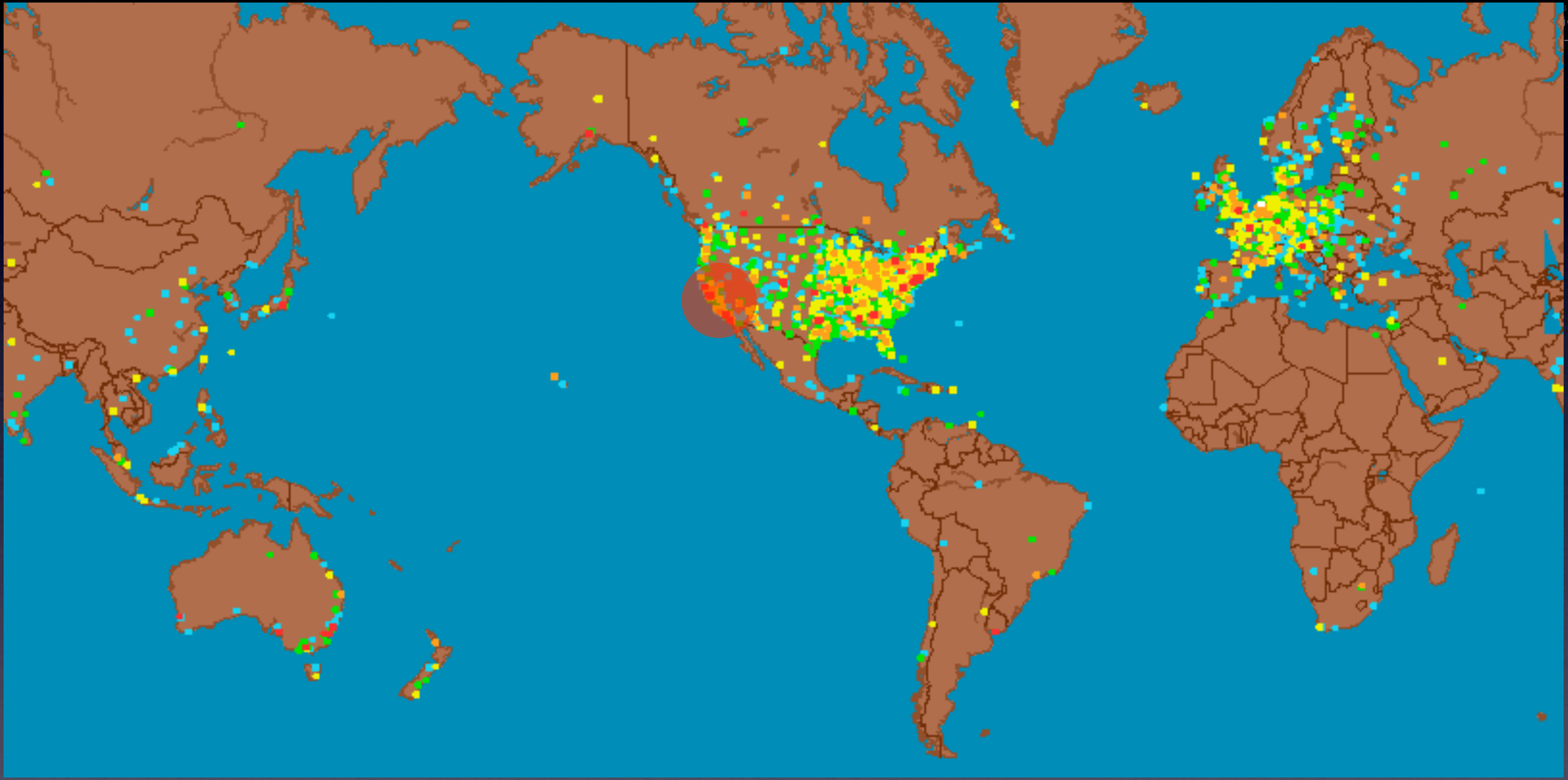


- There are compromises with dual quad-core
- But time-sharing IB is an almost free lunch!
- Might require queue system changes?
- Alternative is a mixed thread/MPI approach



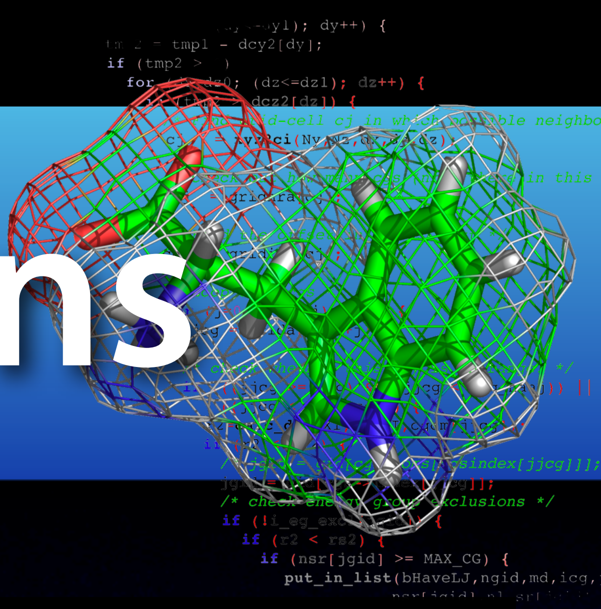
Gromacs & Folding@Home



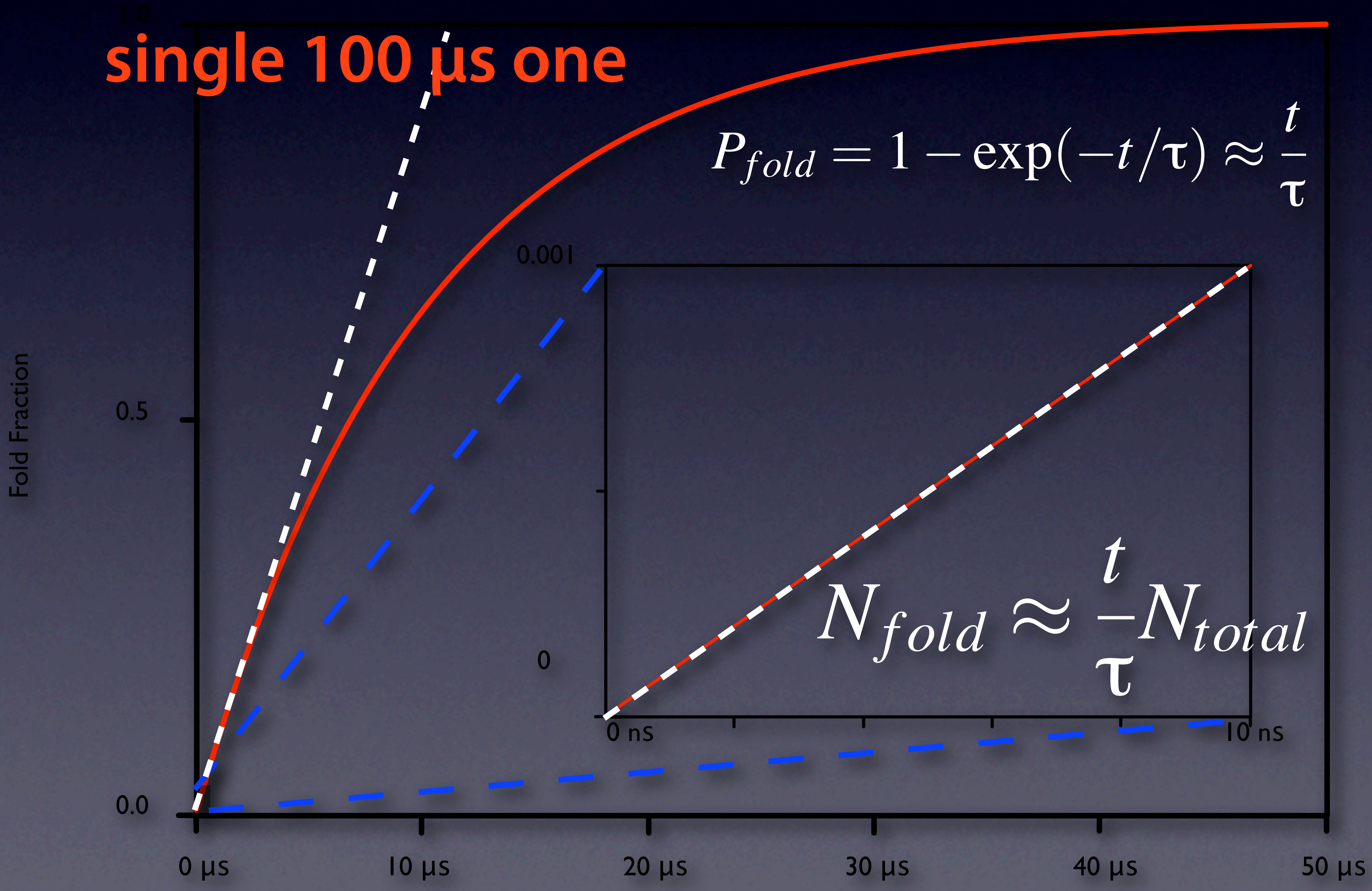


- Running as a screensaver all over the world
- >200,000 active voluntary clients
- 1.5 Petaflops - working today

Decoupled Simulations

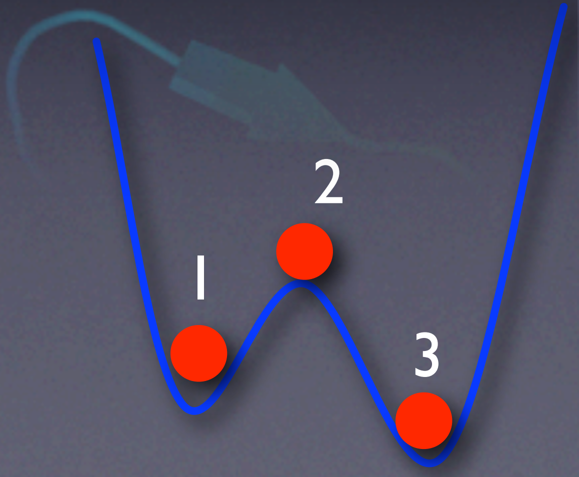


Perform 10,000 independent 10ns simulations instead of a single 100 µs one

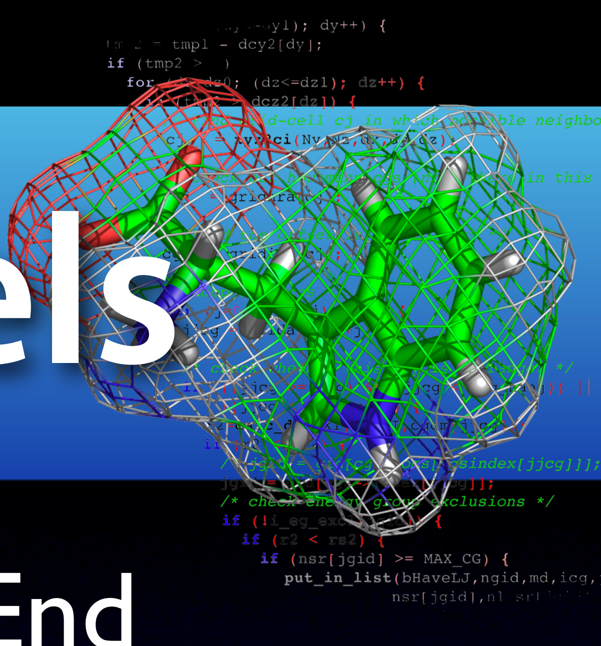


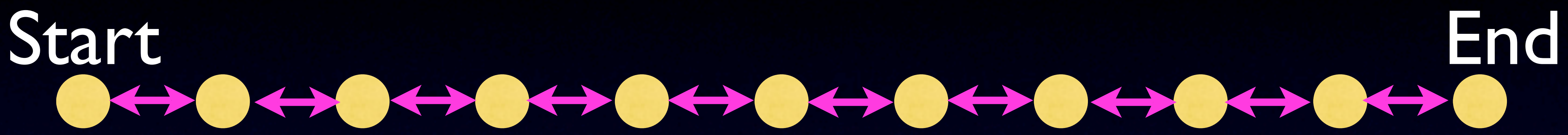
$$P_{fold} = 1 - \exp(-t/\tau) \approx \frac{t}{\tau}$$

$$N_{fold} \approx \frac{t}{\tau} N_{total}$$

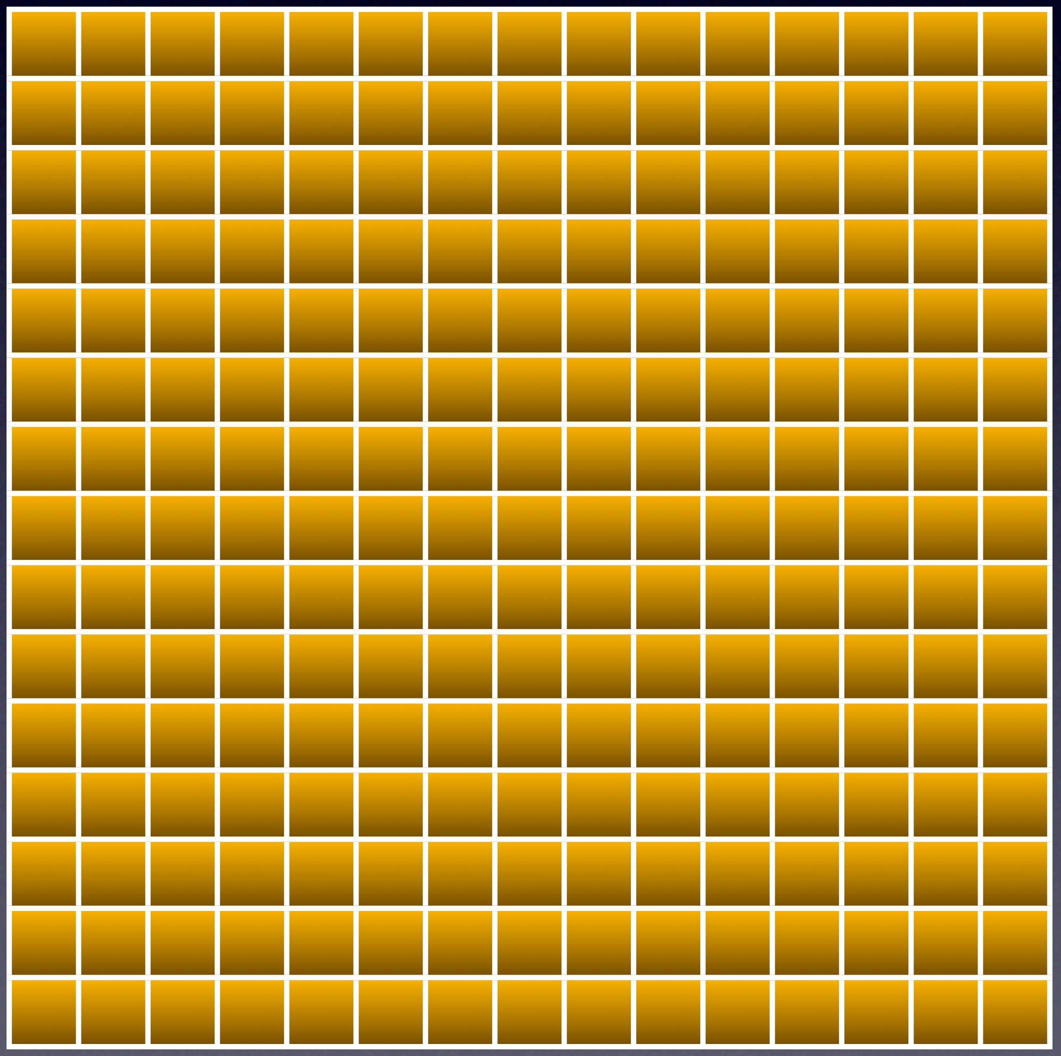


Markovian state models





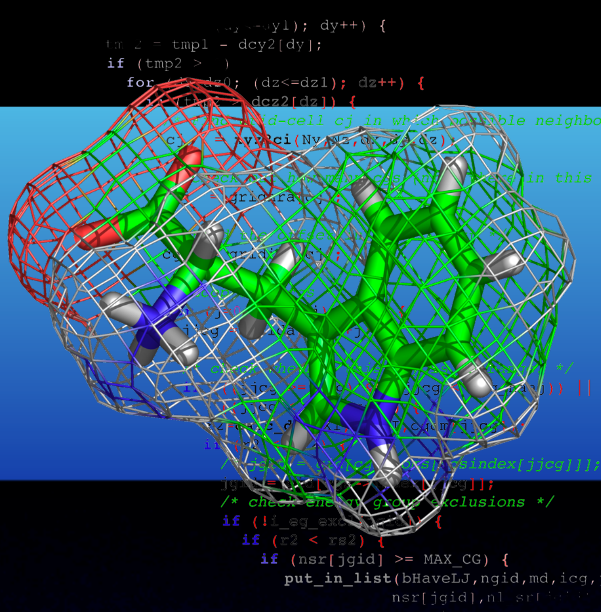
- Start from (short) simulation trajectories
- Cluster them into states
- Calculate transition probabilities (matrix)
- For markov processes:



$$\lambda_i(P(n\Delta t)) = \lambda_i(P(\Delta t))^n$$

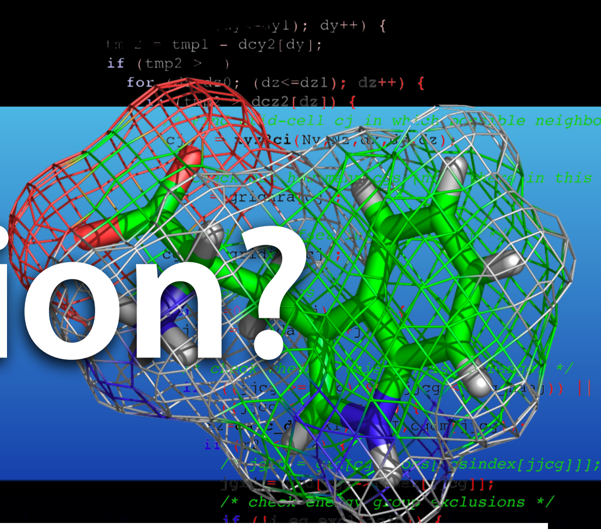
Markovian Properties can be checked - usually seems to hold!

A sneak peak of...



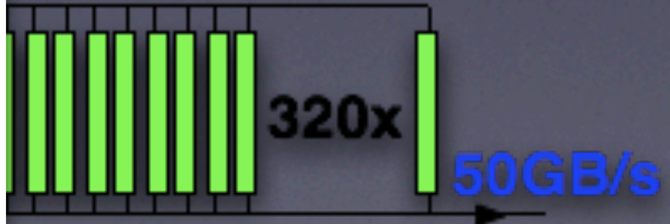
- We can often "work around" communication
- Raw computational power is the bottleneck
- Are there faster computers out there?
- We've spent tons of time on x86 optimization
  - Using assembly, game instructions for $1/\sqrt{x}$
  - Tried FPGA, special FP cards (too expensive)
- New ASIC hardware from DE Shaw in 2008
  - Expensive doesn't even begin to describe it...

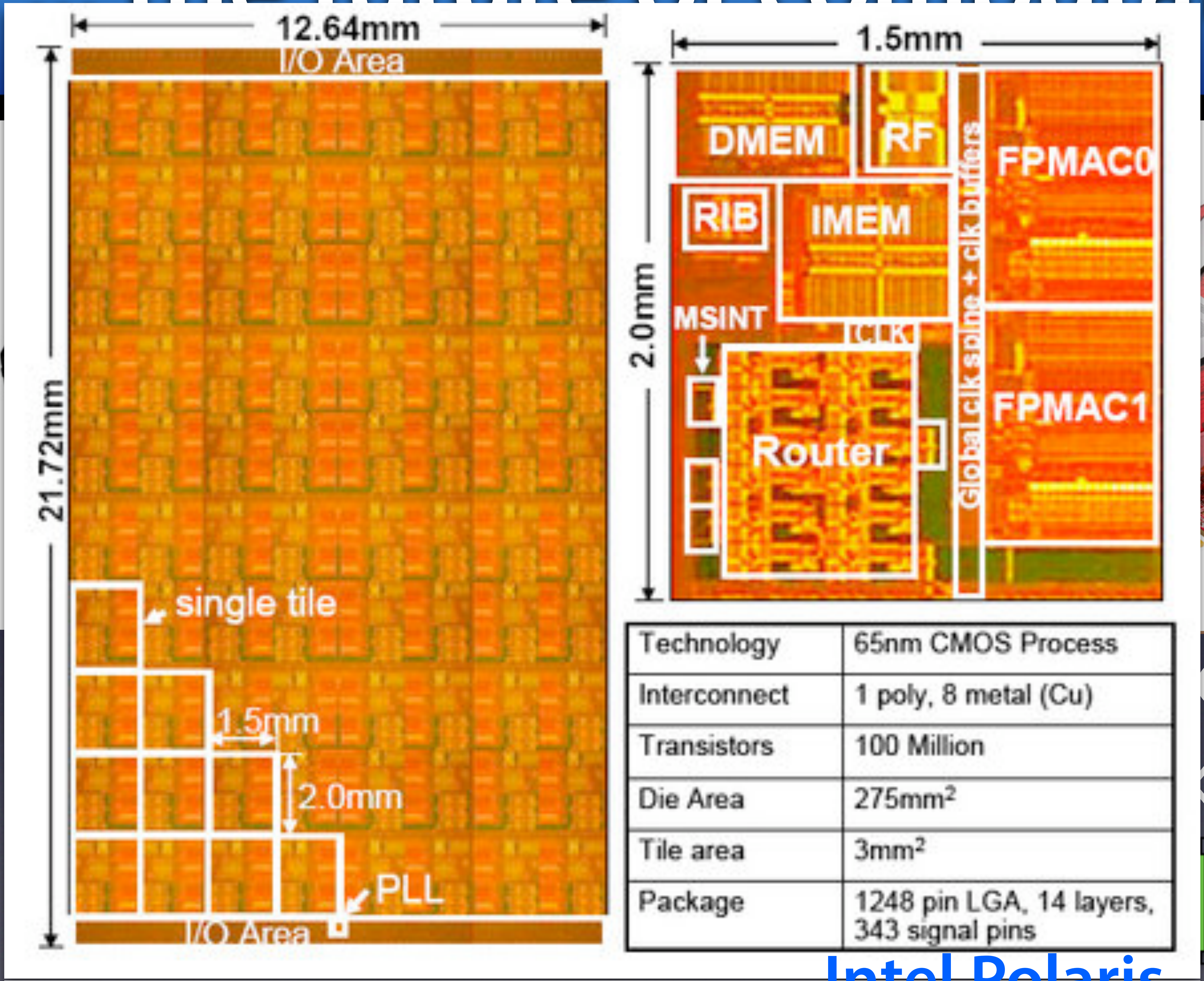
the future of computation?



~~Gflop~~

XT: 600 Gflop

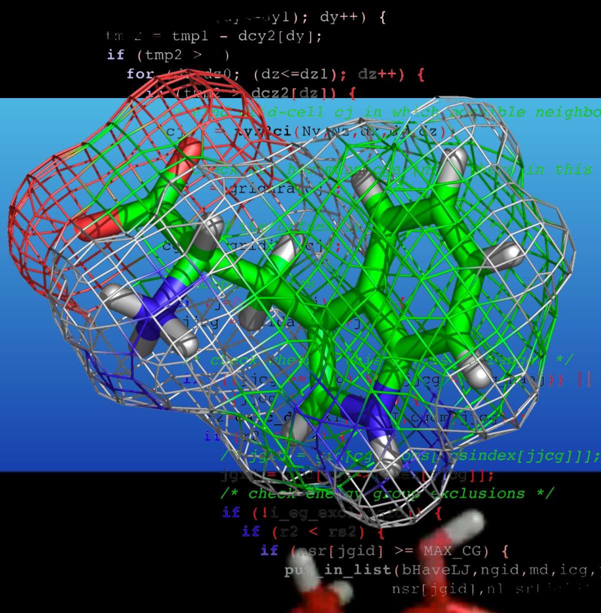




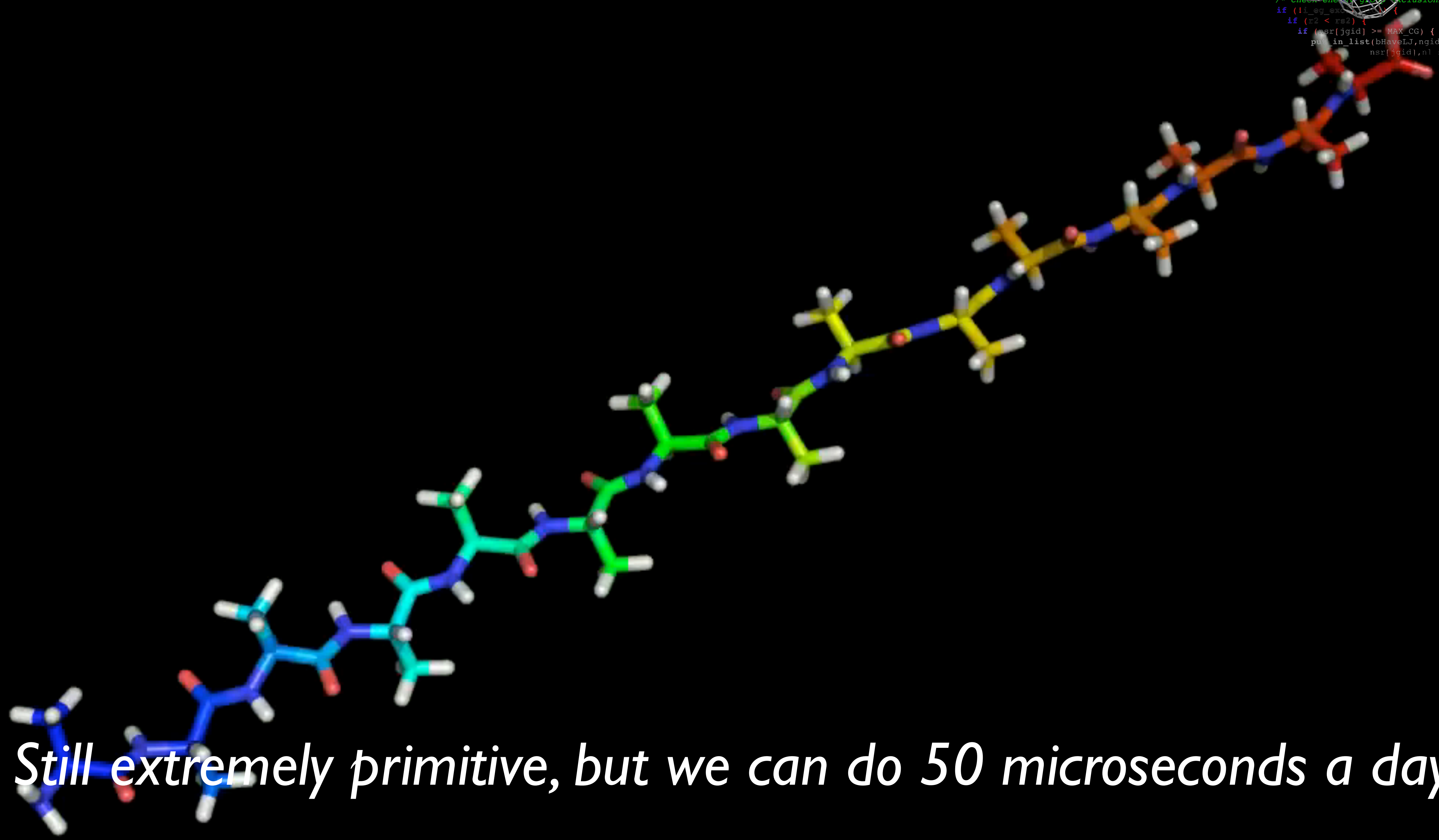
Technology	65nm CMOS Process
Interconnect	1 poly, 8 metal (Cu)
Transistors	100 Million
Die Area	275mm^2
Tile area	3mm^2
Package	1248 pin LGA, 14 layers, 343 signal pins

Intel Polaris

GPU peptide folding

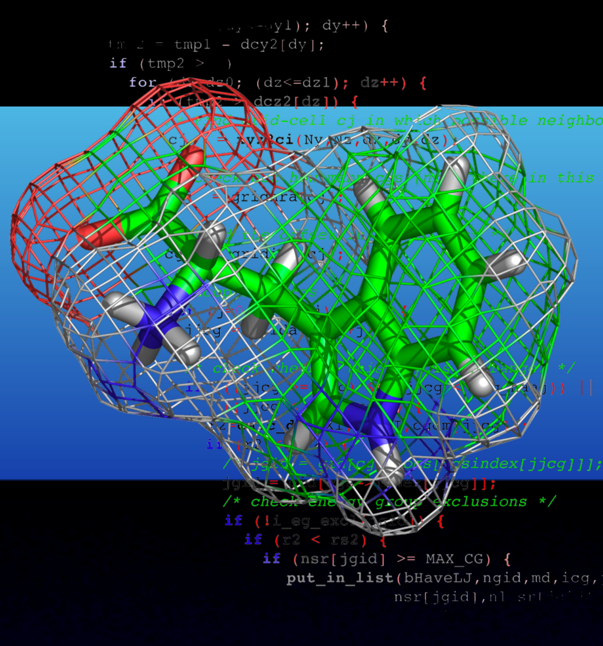


```
        [illegible]yl); dy++) {
      tm 2 = tmp1 - dcy2[dy];
      if (tmp2 > [illegible])
        for ([illegible]dz0; (dz<=dz1); dz++) {
          [illegible]dcz2[dz]) {
            /* [illegible]id-cell cj in which [illegible]ble neighbo
            cj [illegible]ci(Ny,Nz,[illegible]dz);
            /* check [illegible] in this
            [illegible] = gridra[illegible];
            [illegible] the offset [illegible] */
            cg[illegible]gridi[illegible];
            [illegible]
            /* check whe[illegible] */
            if ([illegible]jjcg[illegible]) ||
            [illegible]
            if [illegible]
              // jgid = [illegible]sindex[jjcg]]];
              jgid = [illegible][jjcg]];
              /* check energy group exclusions */
              if (!i_eg_exc[illegible]) {
                if (r2 < rs2) {
                  if (nsr[jgid] >= MAX_CG) {
                    put_in_list(bHaveLJ,ngid,md,icg,
                                nsr[jgid],nl sr[illegible]
```



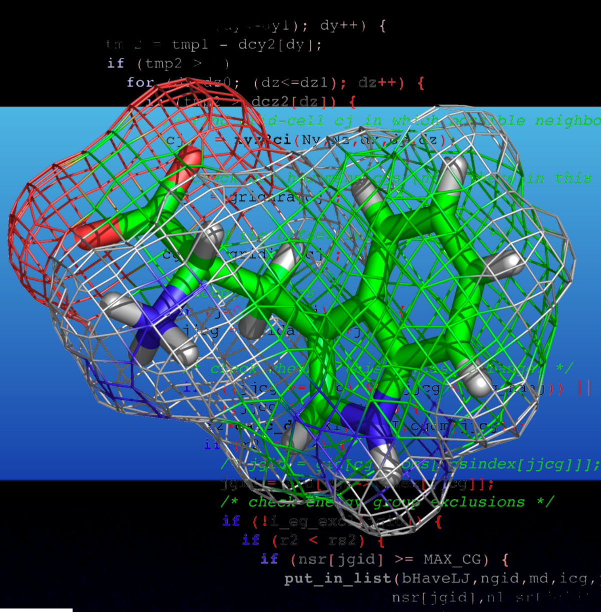
Still extremely primitive, but we can do 50 microseconds a day!

Summary



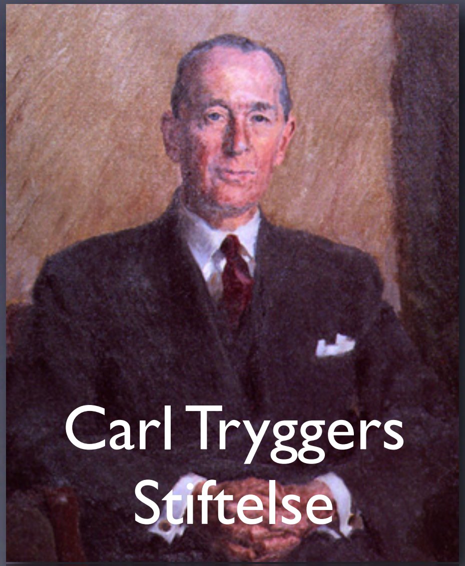
- Multi-level parallelism necessary
    - SIMD -> Threads -> MPI -> Distributed Computing
- Neutral Territory Decomposition is counter-intuitive, but extremely efficient
- Performance matters. Relative scaling doesn't.
- For Neolith and similar systems, it often works better to interleave communicating processes
- Too late to start optimizing for 4-8 cores!
- Streaming architectures are coming
- But you WILL need to adapt your algorithms
    - Not optional - single cores won't scale

Thanks for the fish...



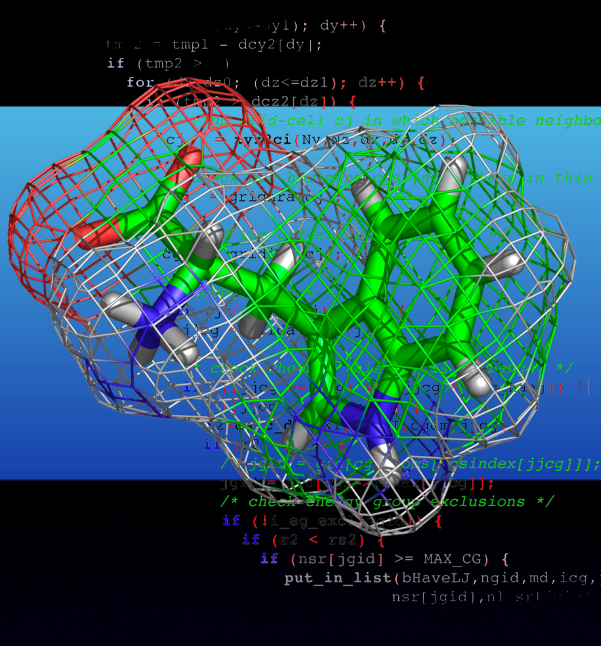
NATURVETENSKAP • TEKNIK • MEDICIN
Stiftelsen för Strategisk Forskning

HFSP


Carl Tryggers Stiftelse

Vetenskapsrådet

Acknowledgments



- Gromacs work:
  David van der Spoel, Uppsala
  Berk Hess, MPI Mainz
- Membrane & ion channel simulations:
  Anna Johansson, Stockholm
  Pär Bjelkmar, Stockholm
- Distributed Simulation in Folding@Home:
  Young-Min Rhee, Stanford
  Vijay Pande, Stanford