March 18, 2014

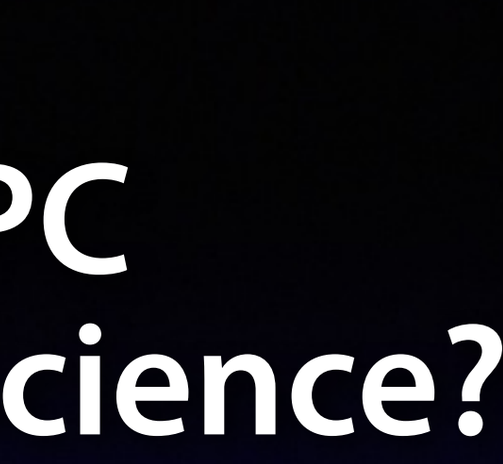
What do we need for HPC to make a revolution in Life Science?

Needs, Possibilities & Challenges in Molecular Dynamics

St Moritz , SOS18

Erik Lindahl

erik.lindahl@scilifelab.se

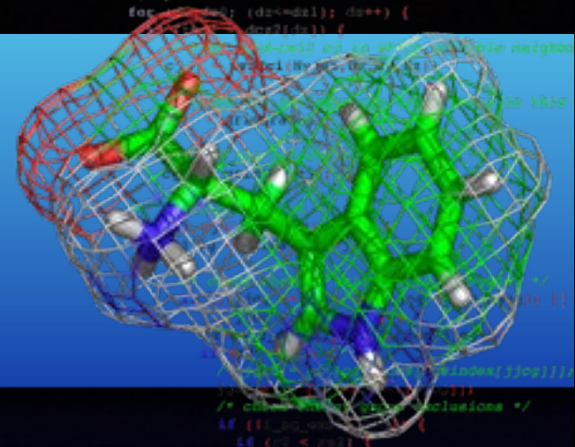


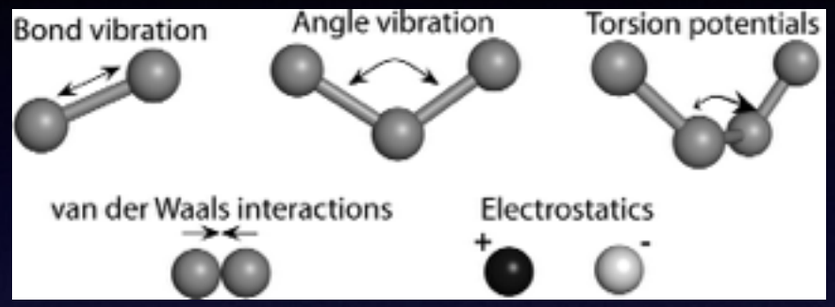
SciLifeLab STOCKHOLM

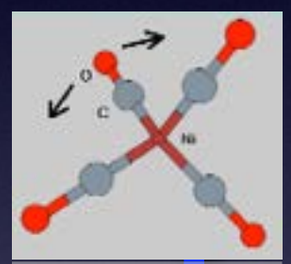
UNIVERSITET STOCKHOLMS

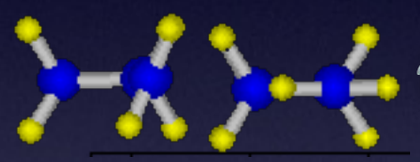
KTH VETENSKAP OCH KONST

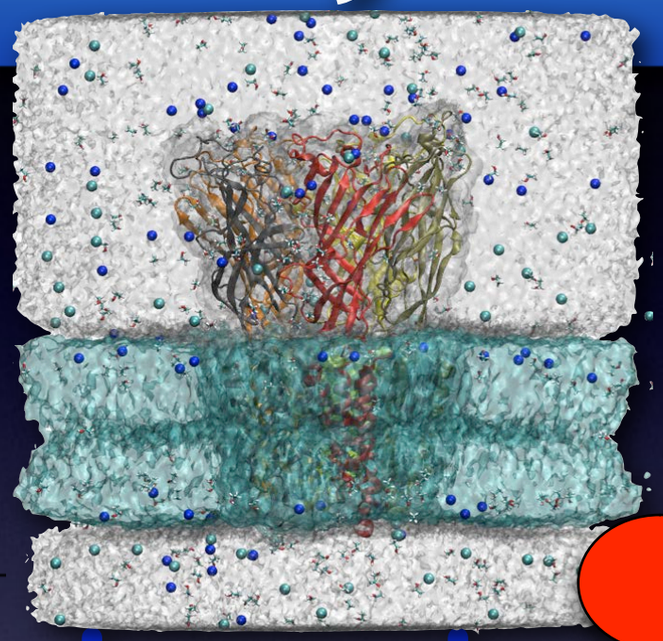
Molecular Dynamics











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

Extreme detail

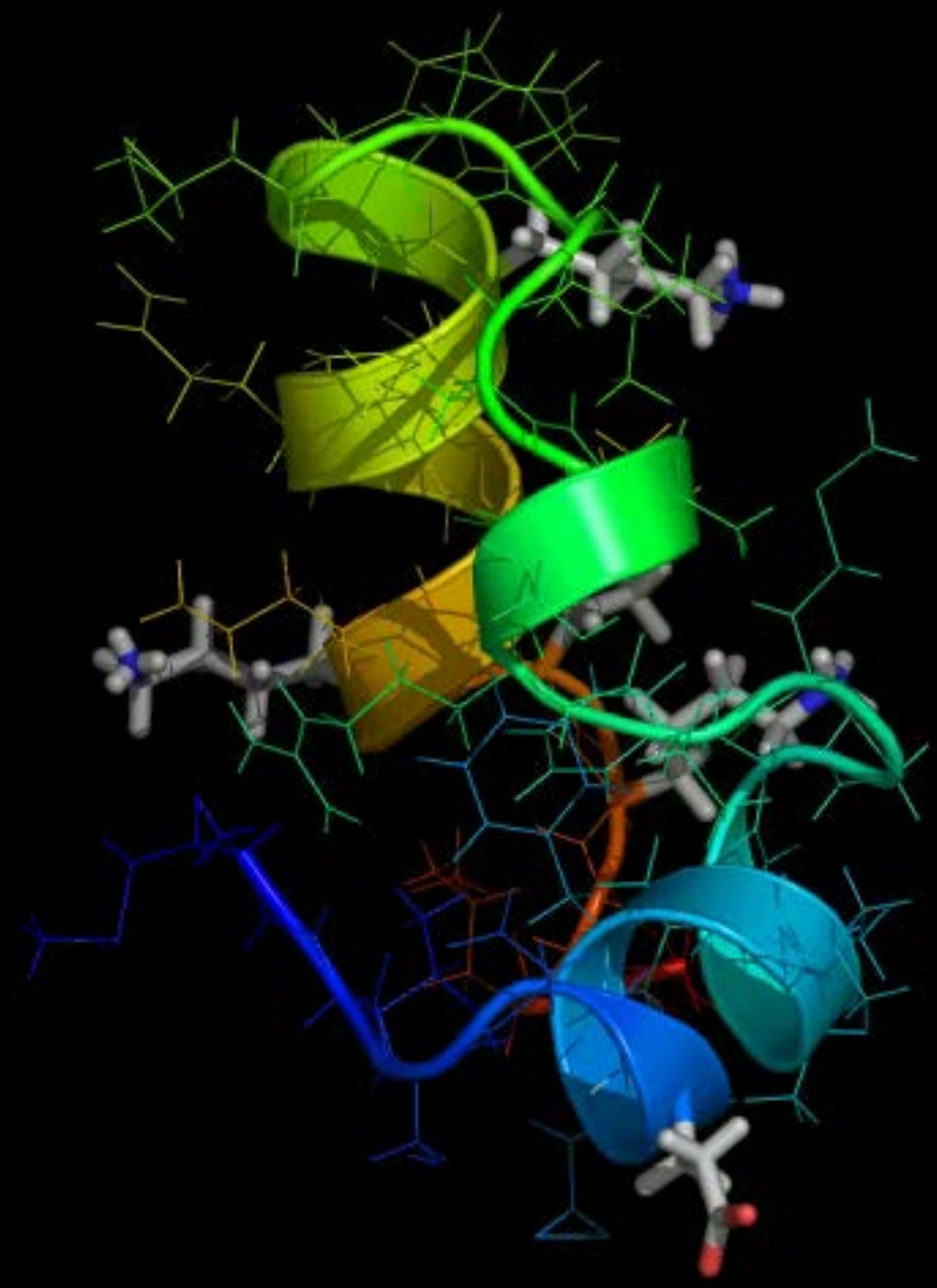
Sampling issues?

Parameter quality?

Where we are

Where we want to be

Molecular Mechanics



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

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

$$V(r) = \sum_{bonds} \frac{1}{2} k_{ij}^{b} \left(r_{ij} - r_{ij}^{0}\right)^2$$

$$+ \sum_{angles} \frac{1}{2} k_{ijk}^{\theta} \left(\theta_{ijk} - \theta_{ijk}^{0}\right)^2$$

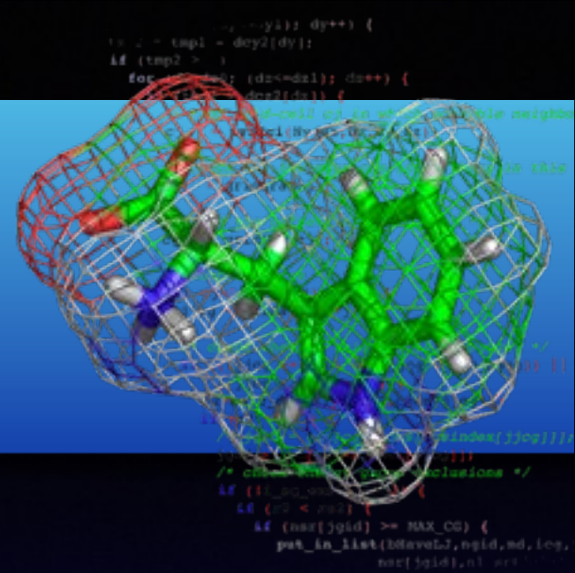
$$+ \sum_{torsions} \left\{ \sum_{n} k_{\theta} \left[1 + \cos\left(n\phi - \phi_0\right)\right] \right\}$$

$$+ \sum_{impropers} k_{\xi} \left(\xi_{ijkl} - \xi_{ijkl}^{0}\right)$$

$$+ \sum_{i,j} \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}}$$

$$+ \sum_{i,j} \left[\frac{C_{12}}{r_{ij}^{12}} - \frac{C_6}{r_{ij}^{6}} \right]$$

How can we achieve longer simulations?



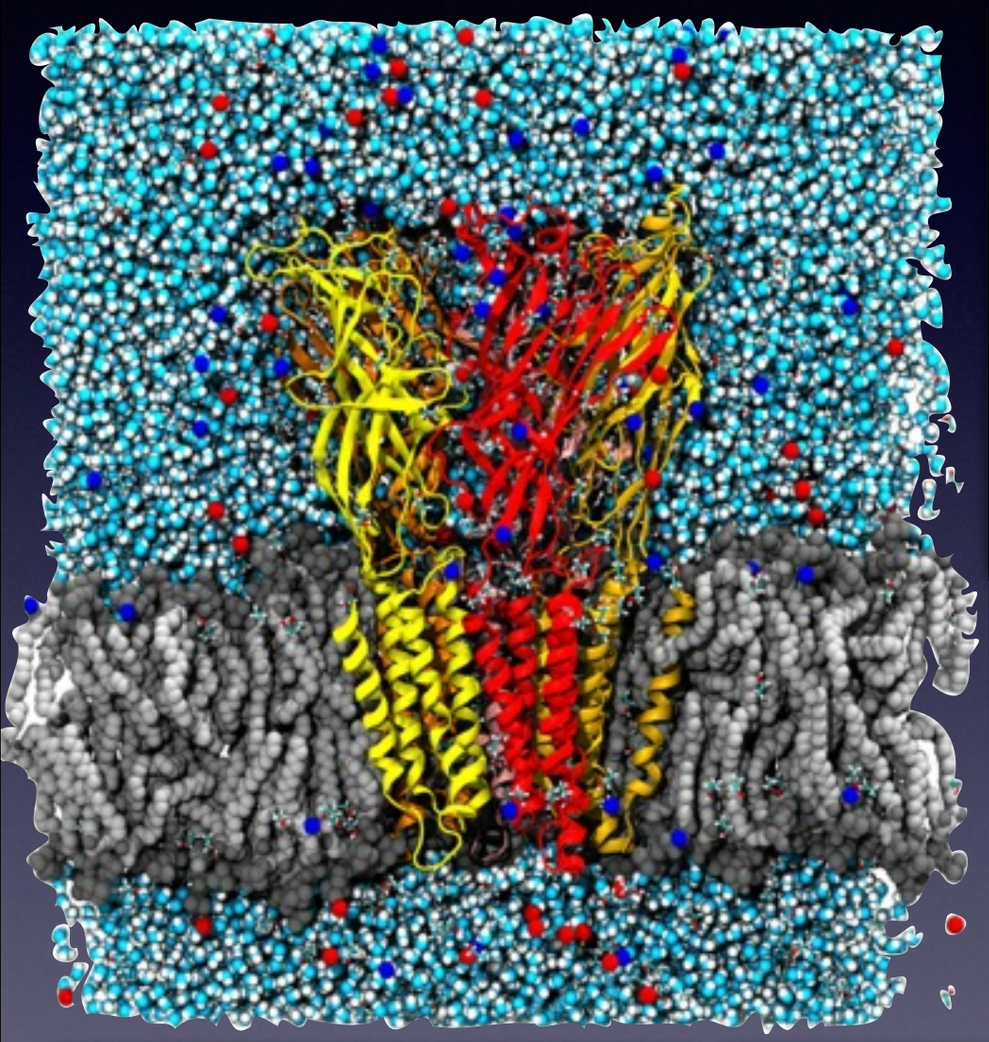
With a time step of 5fs...

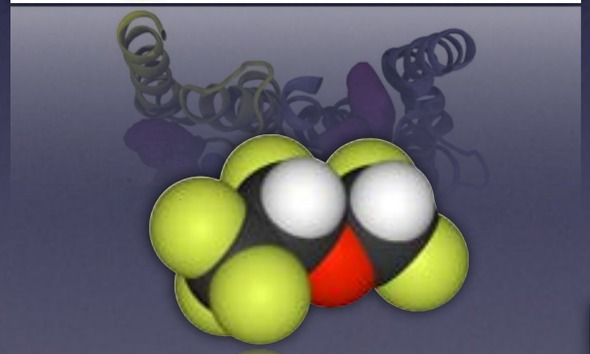
... you need 200 million iterations to reach 1 μs of simulated time

To achieve that in a day (86,400 seconds)...

...each iteration must complete in 432 (wallclock) μs!

Evolutions & Revolutions



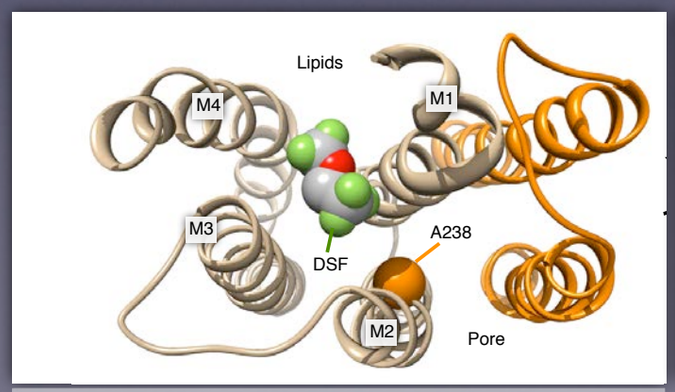


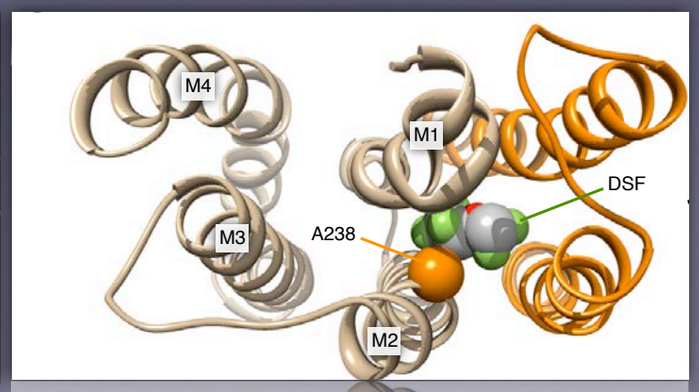
1µs/day:
Understanding motions

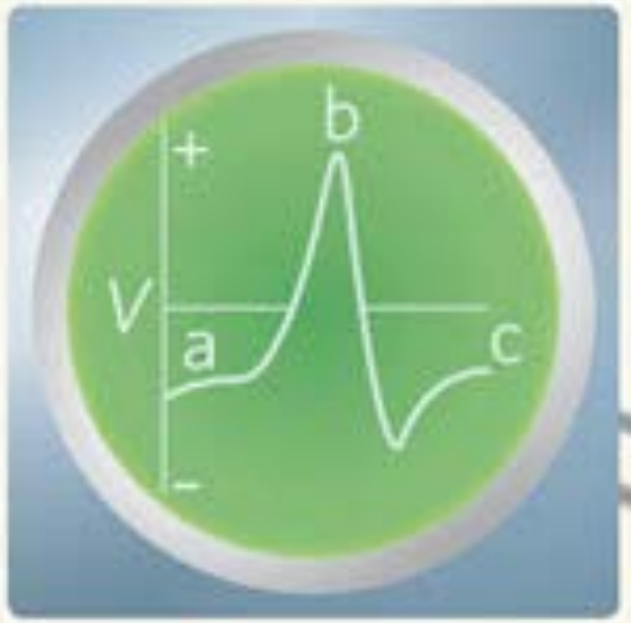
10µs/day:
Predicting motions
Interpreting/improving experiments

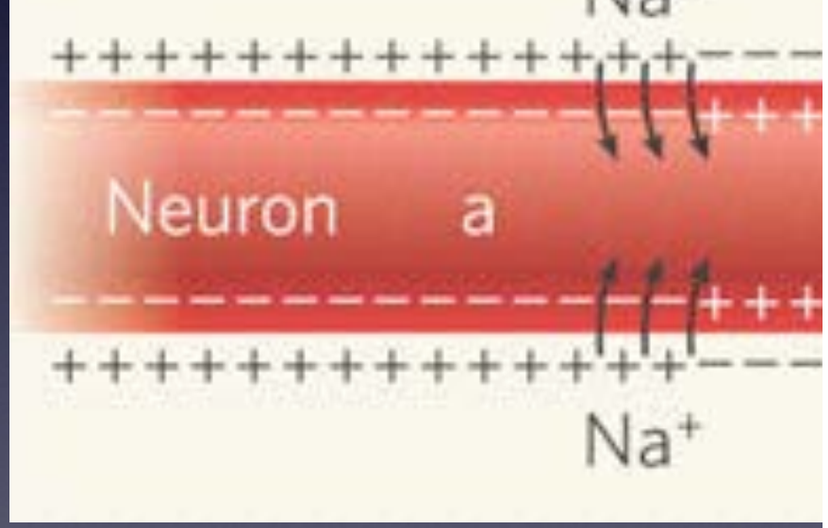
>100µs/day:
Replacing biochemistry experiments

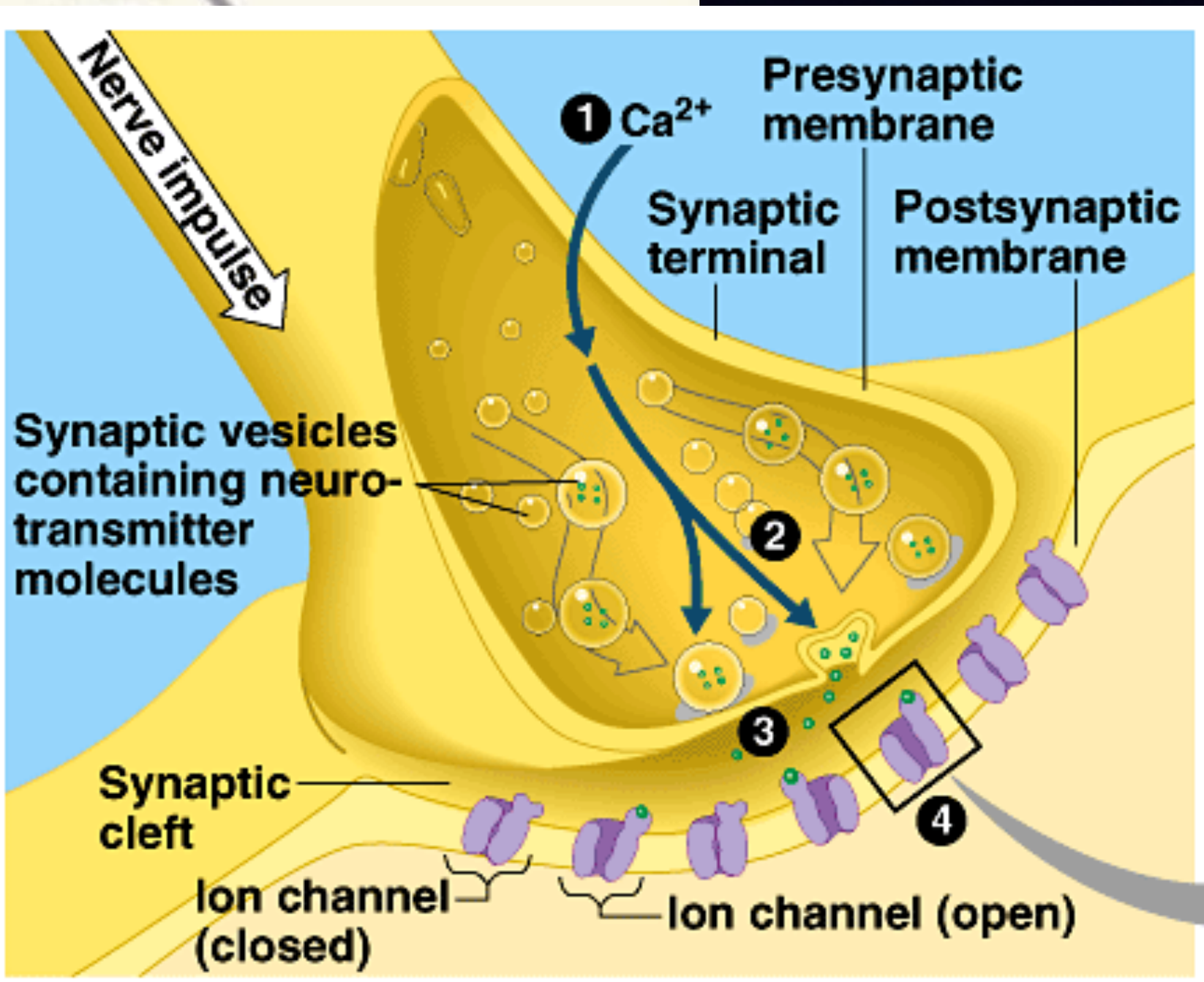
>1ms/day:
Replacing medicine/biology experiments



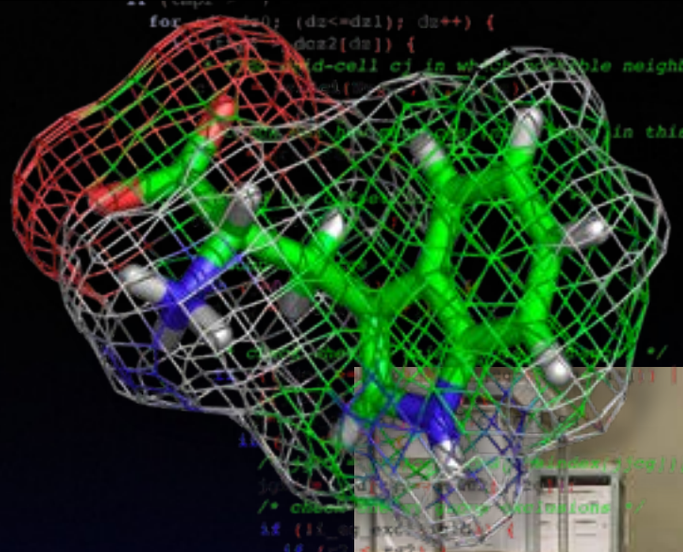



Oscilloscope
+
b
V
a
c
–

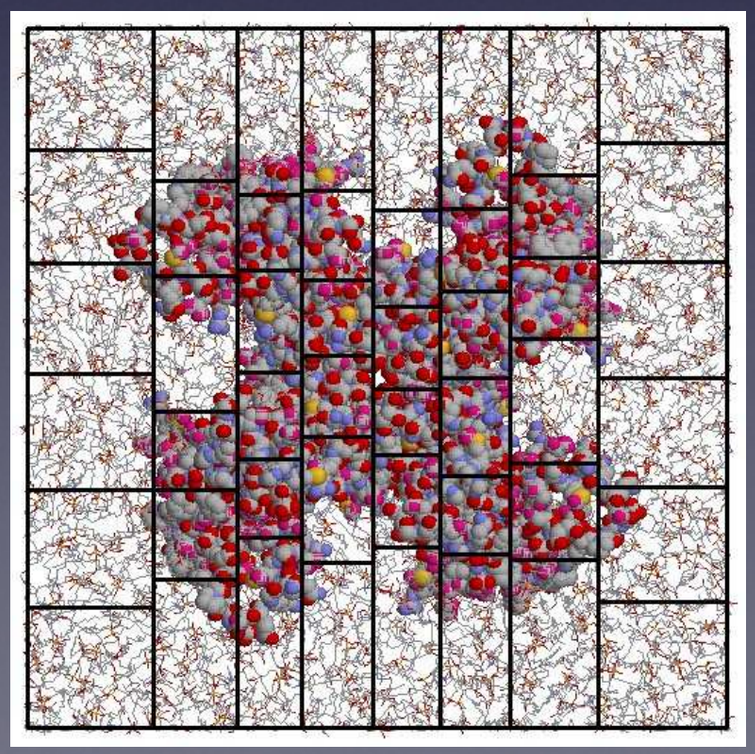

Na+
Neuron
a
Na+


Nerve impulse
1 Ca2+
Presynaptic membrane
Synaptic terminal
Postsynaptic membrane
Synaptic vesicles containing neuro-transmitter molecules
2
3
4
Synaptic cleft
Ion channel (closed)
Ion channel (open)

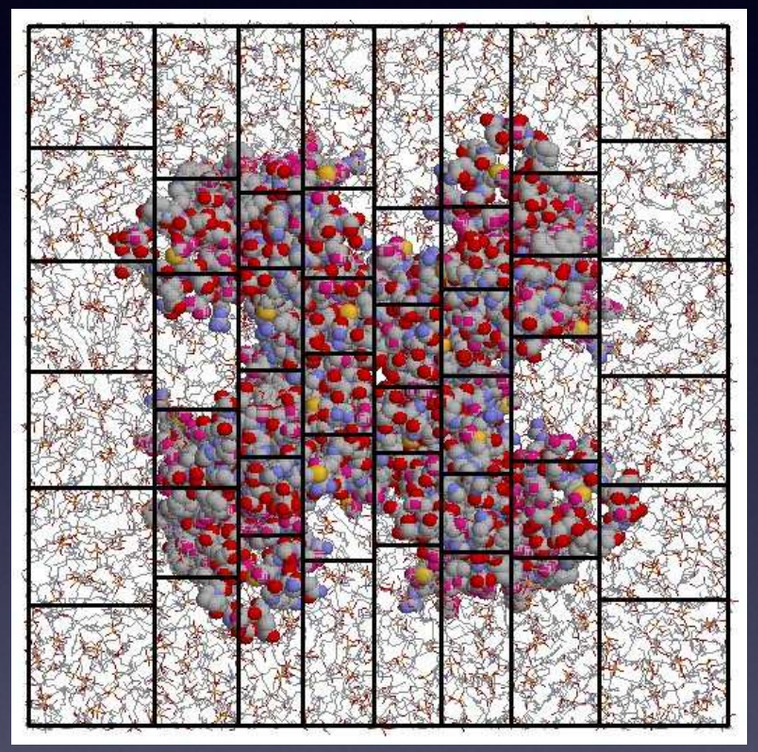
We're on the single-µs scale today (for small systems)



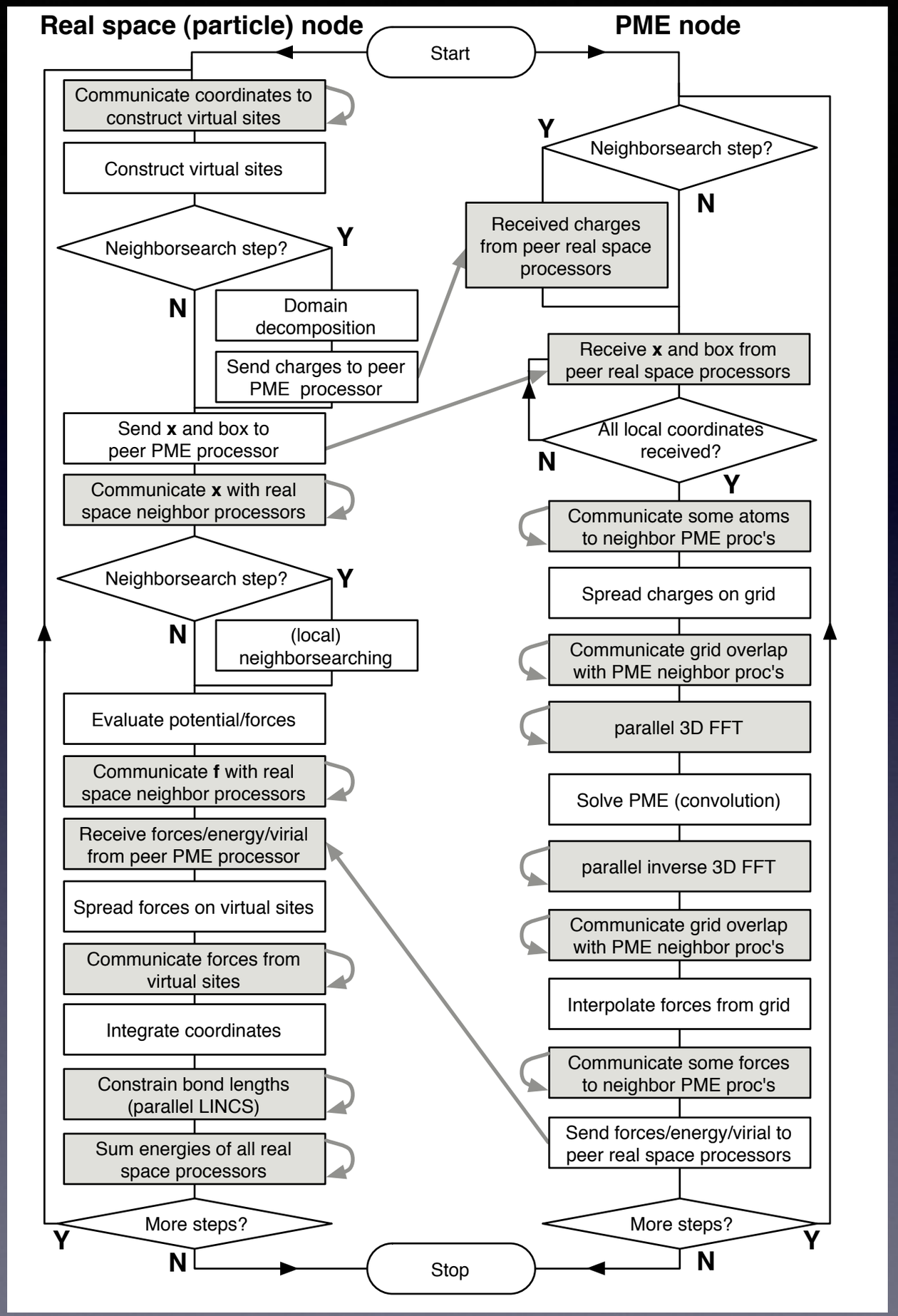
Larger machines have enabled larger systems, not longer simulations



Inherent limits to parallelism



Larger supercomputer do not address this



~2024: 1B 'cores'

2022: ~300M cores

2020: ~100M cores

2018: ~30M cores

2016: ~10M cores

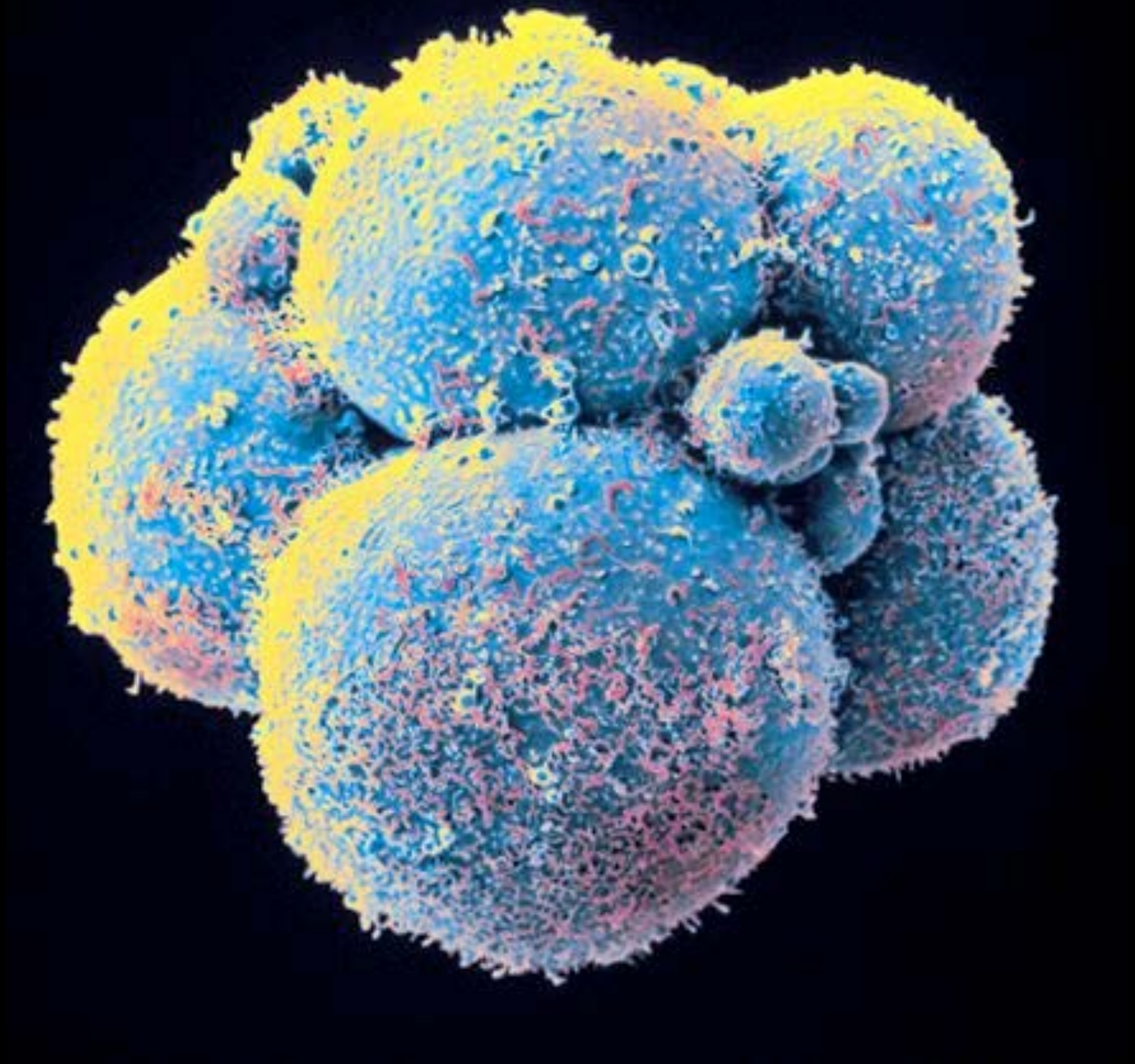
2014: ~3M cores

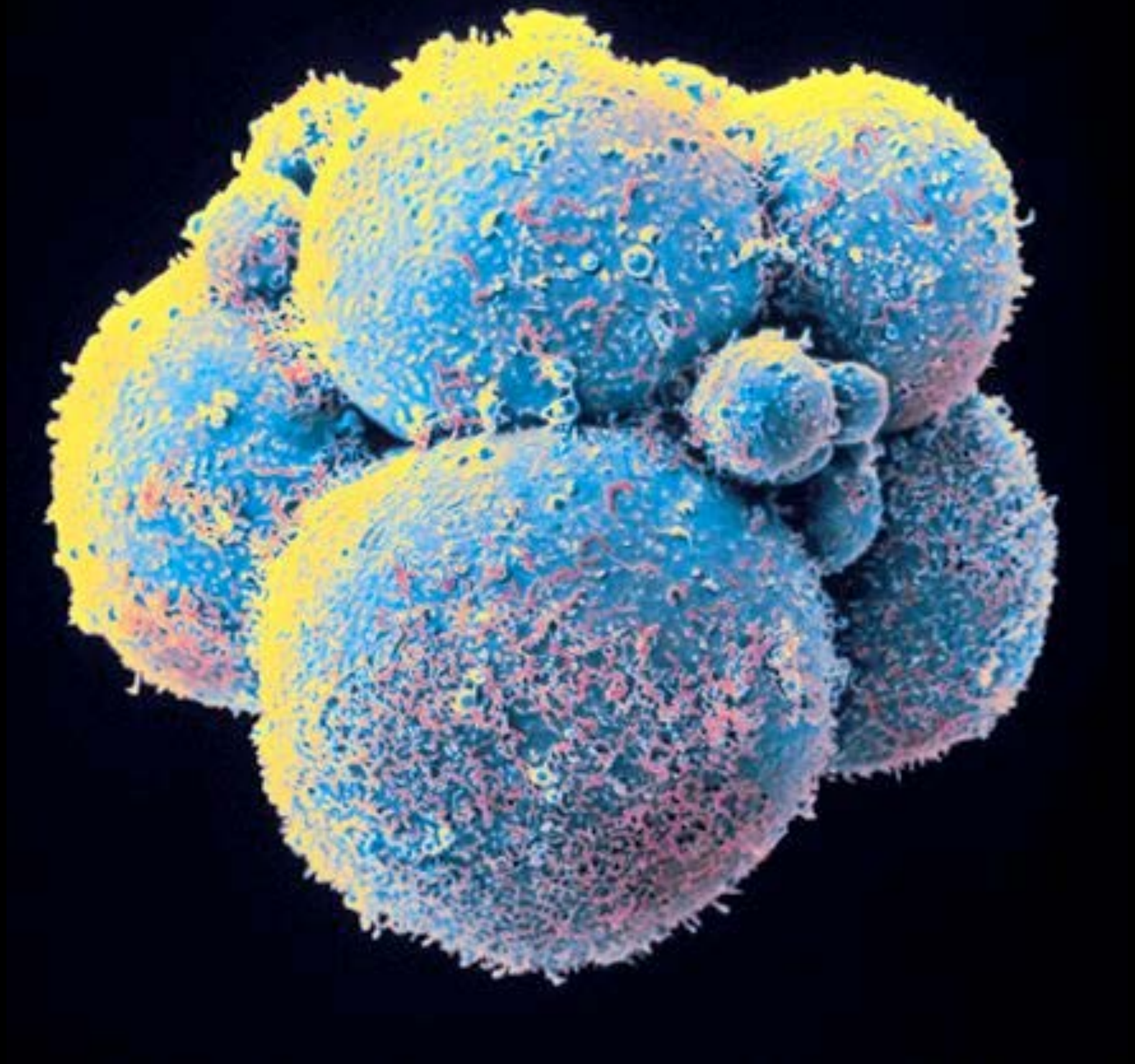
2012: ~1M cores

2010: ~300,000 cores

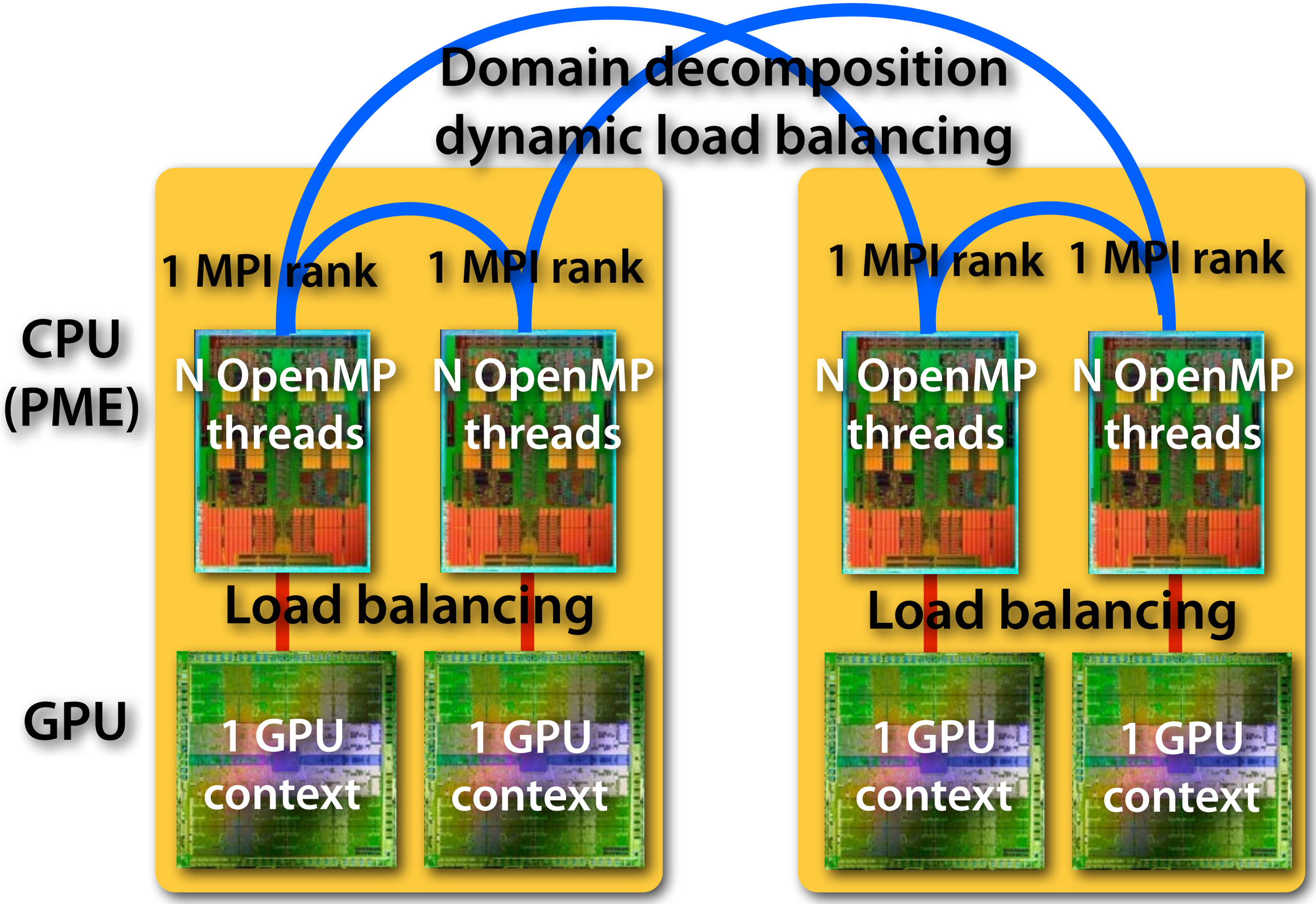
How will YOU use a billion cores?

We keep scaling "up" (larger simulations) where we should scale "down" (more fine-grained parallelism)!

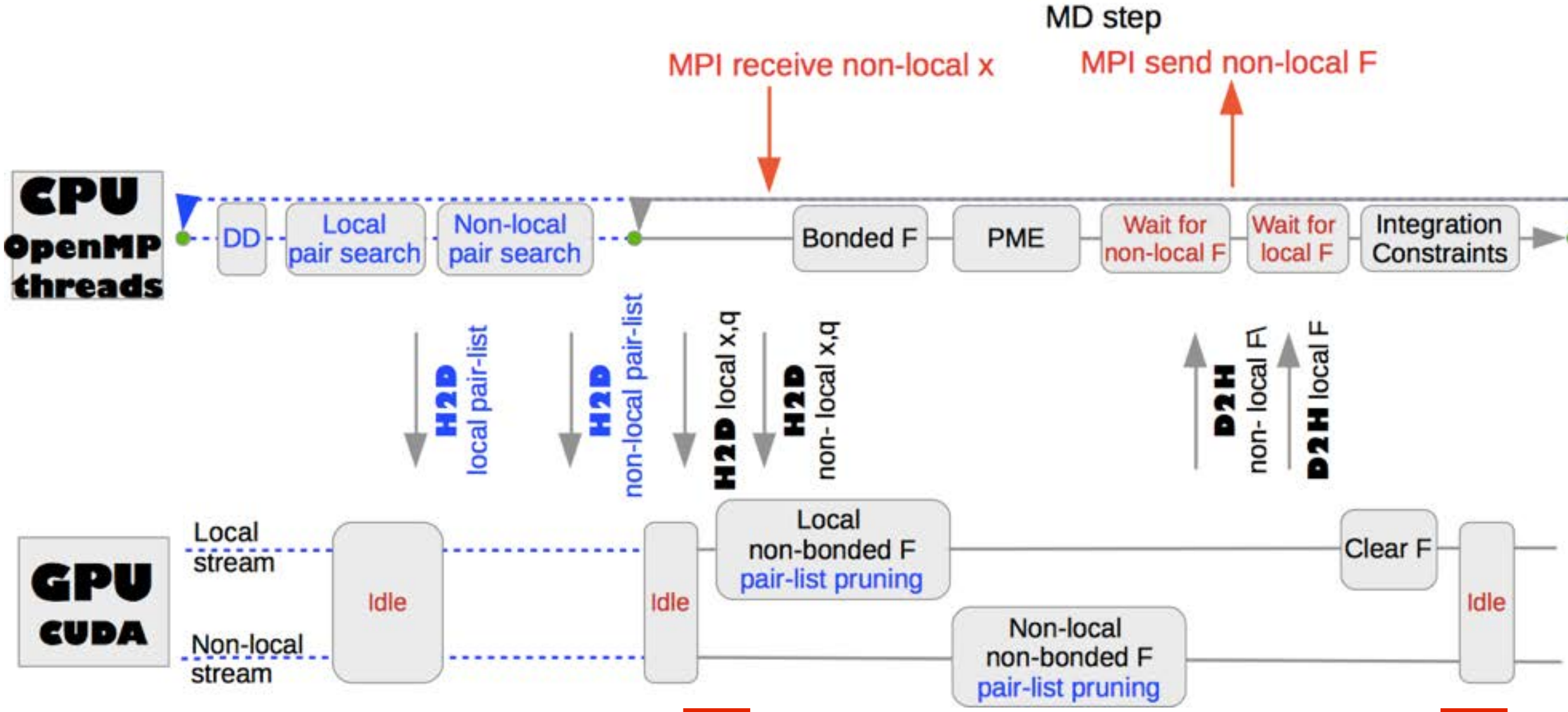




Gromacs-4.6 2nd-generation GPU acceleration:



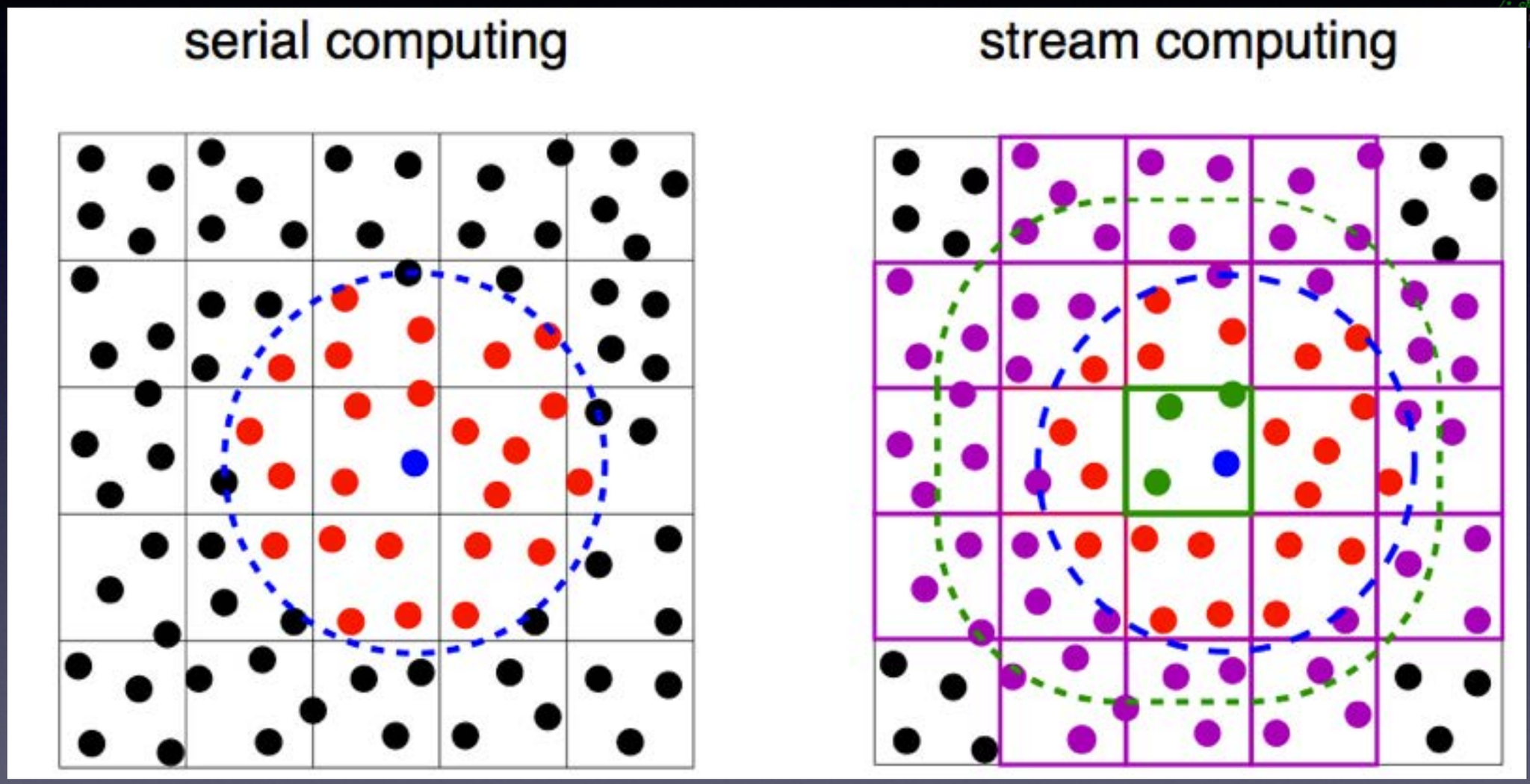
Heterogeneous CPU-GPU acceleration in GROMACS-4.6



Wallclock time for a step:
~0.5 ms if we want to simulate 1µs/day
We cannot afford to lose all previous acceleration tricks!

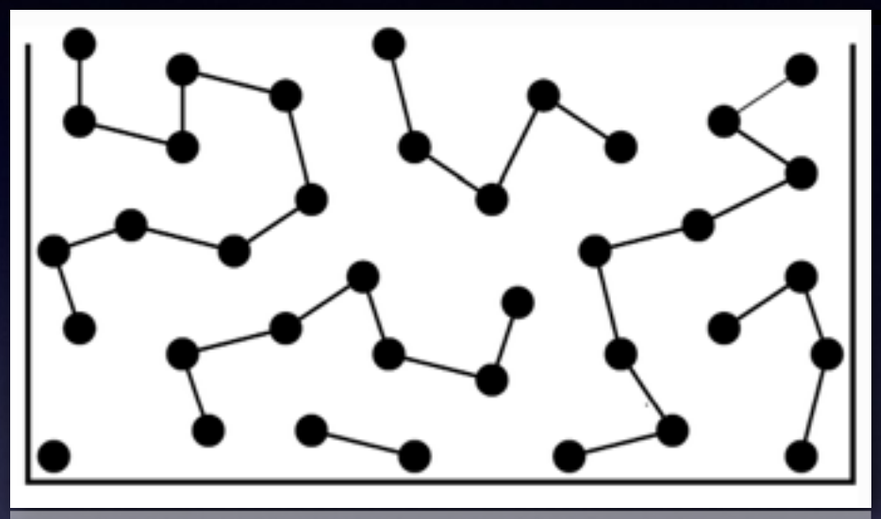
100-500 µs

Tiling circles is difficult!

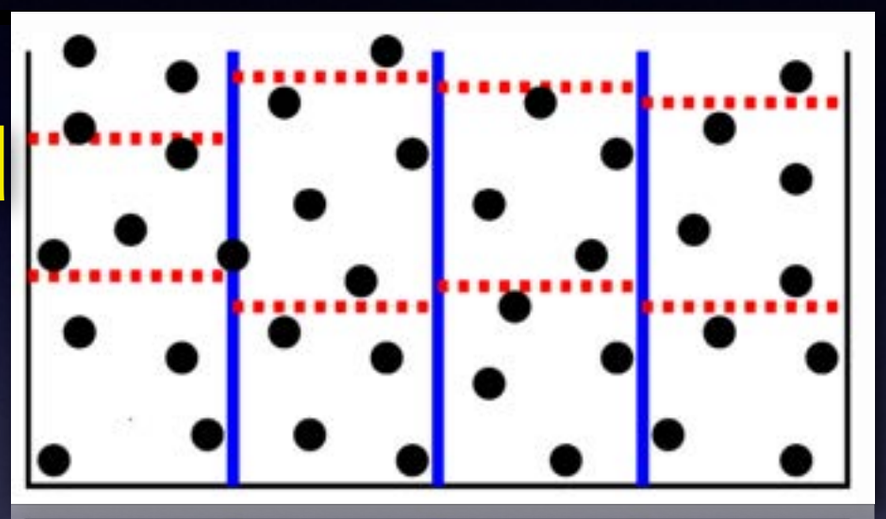


- You need a lot of cubes to cover a sphere
- Algorithms developed last 40 years suck

From neighborlists to cluster pair lists in GROMACS-4.6



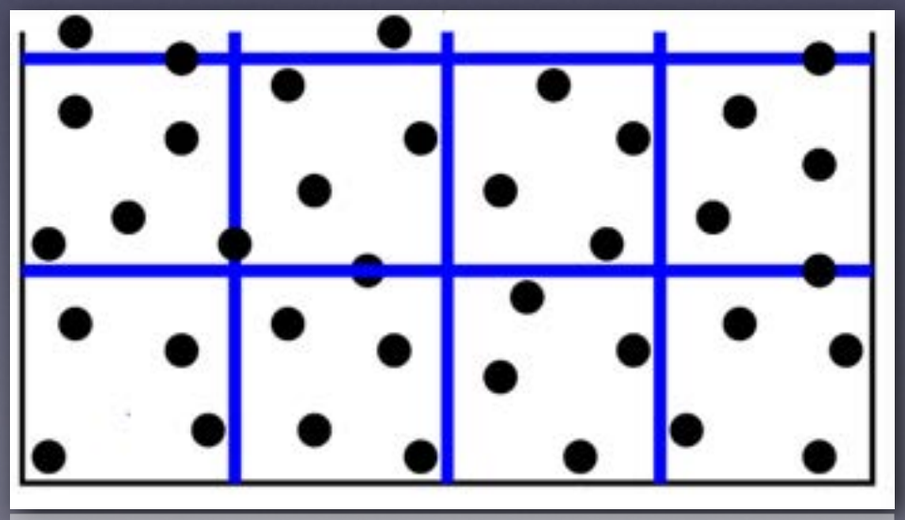
x,y grid
z sort
z bin



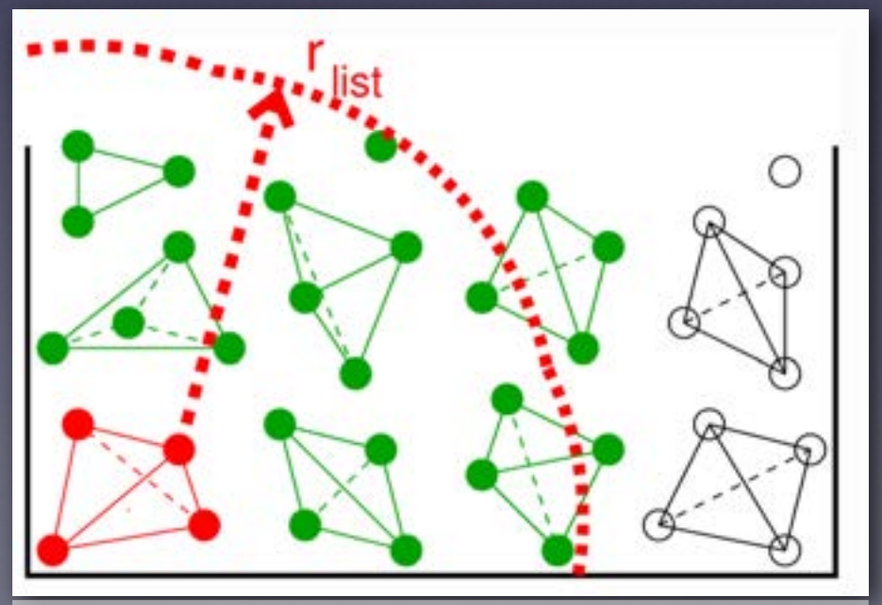
Organize as tiles with all-vs-all interactions:

X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X

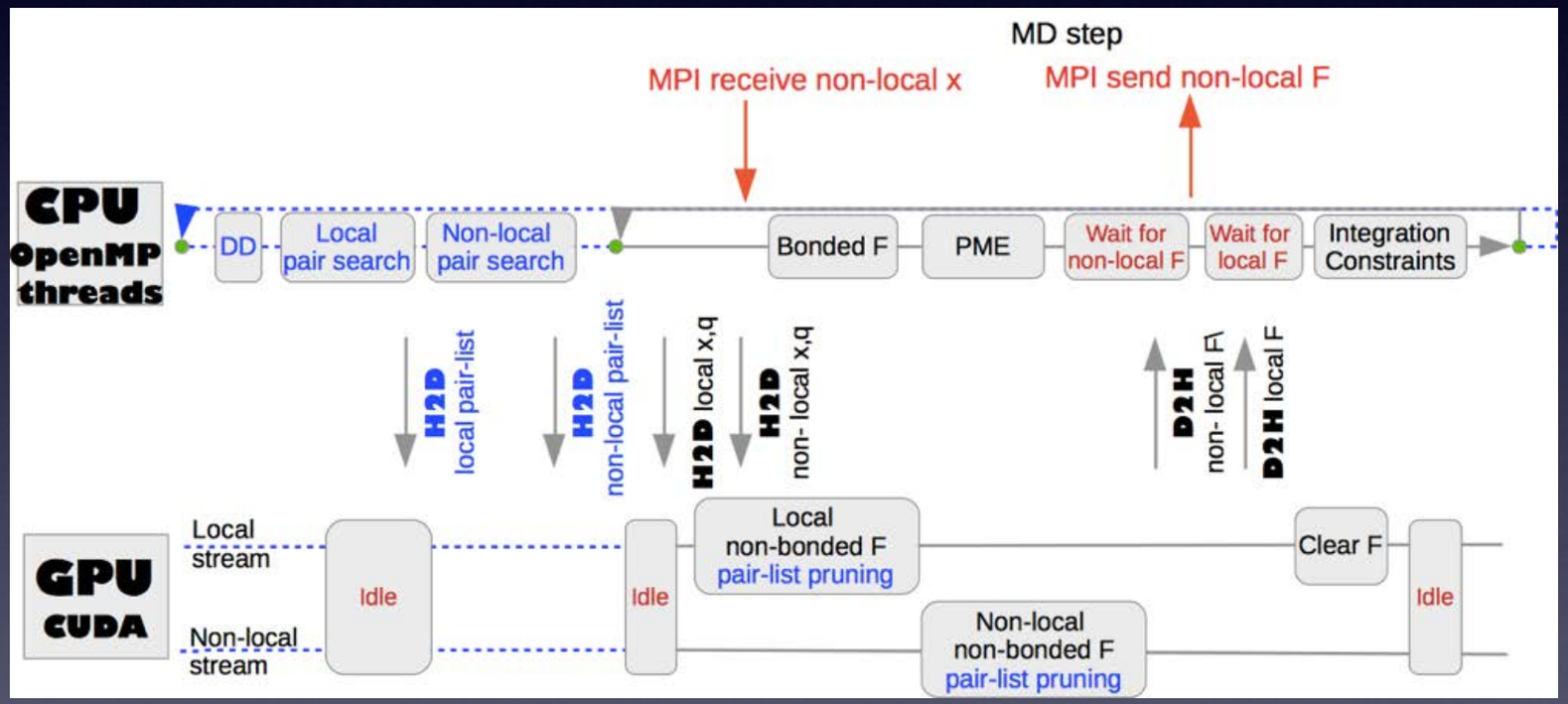
x,y,z gridding

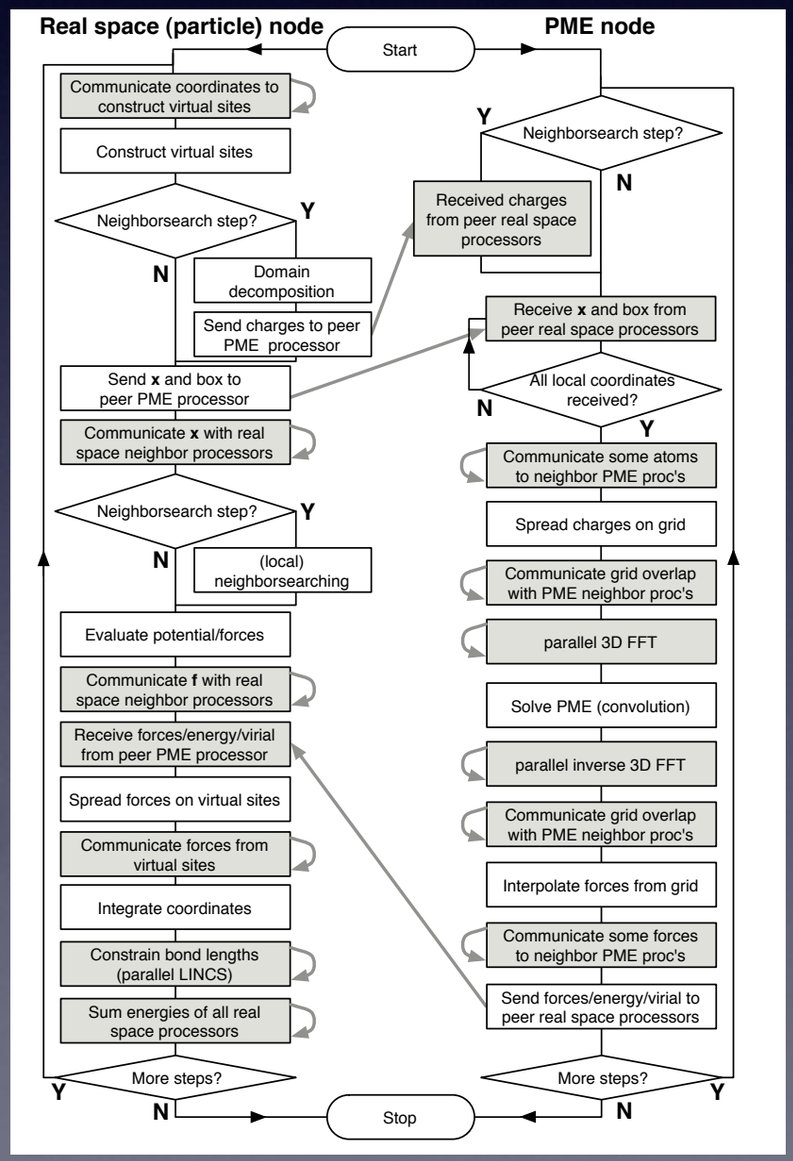


Cluster pairlist



but where does the CPU come in now?



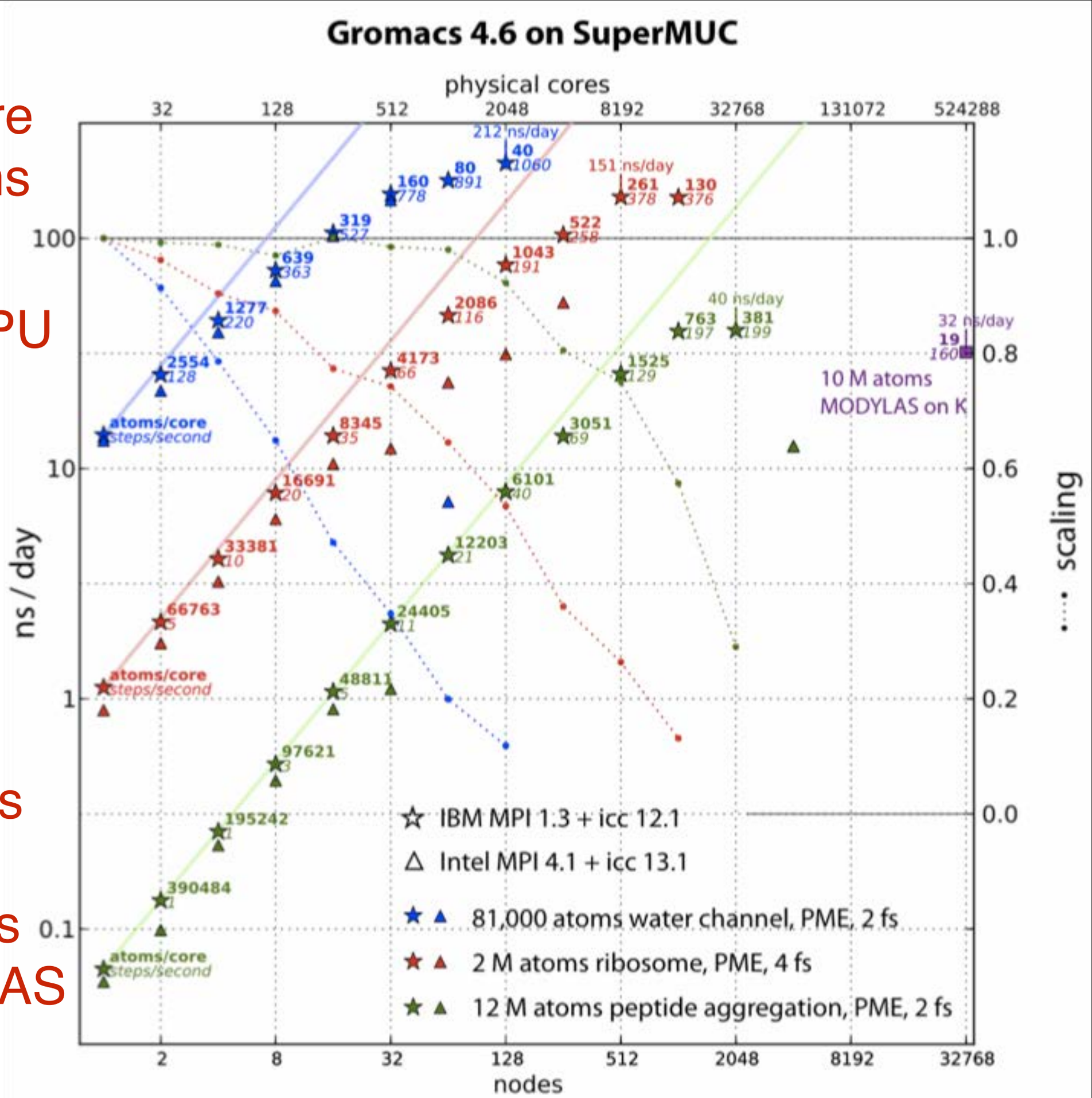


CPU SIMD units like streaming algorithms: Significant scaling improvements!

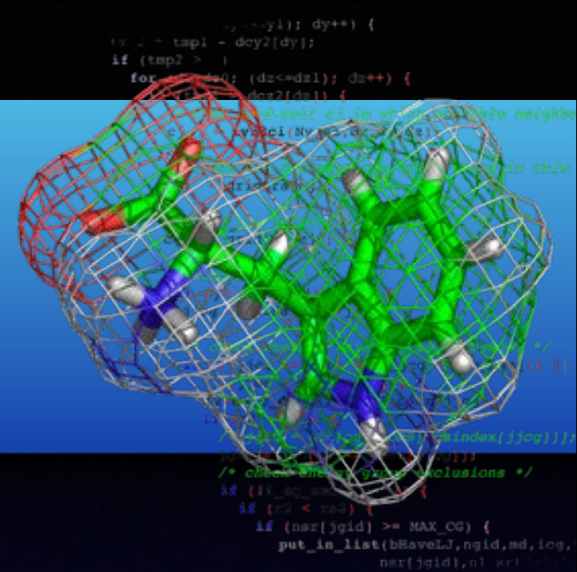
Strong scaling:
40-80 atoms/core
for small systems

~1300 atoms/GPU

Performance:
12k Xeon cores
running Gromacs
on SuperMUC
beats 880k cores
running MODYLAS
on K computer



Coding challenges



> 2 million lines of C/C++ code
Extremely tuned: SIMD, Kernel generators, >2 IPC
C++ modules, C kernels, MPI (MPMD), OpenMP, CUDA

Libraries have been disappointing;
not enough fine-grained control

Not just a matter of scaling: The fastest flops are the ones we avoid calculating

Lots of complex/smart algorithms

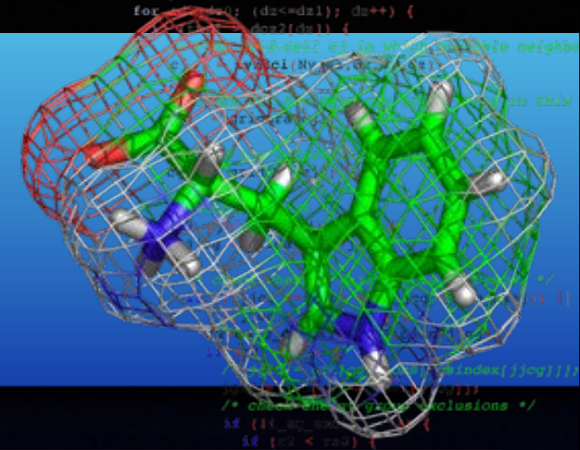
Mini-apps under development - devil is in the detail…

Applications care about performance - not scaling!

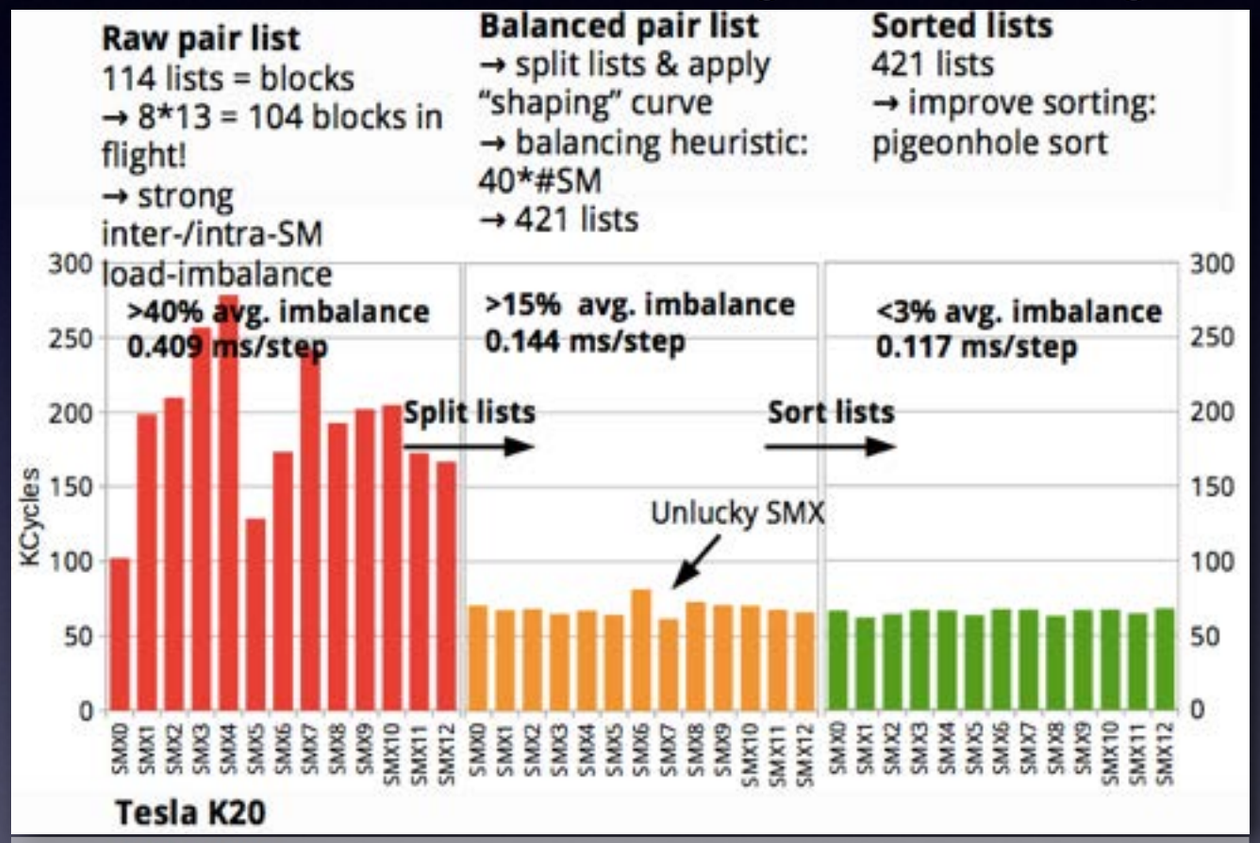
Much scientific software (including ours) originally written by amateurs

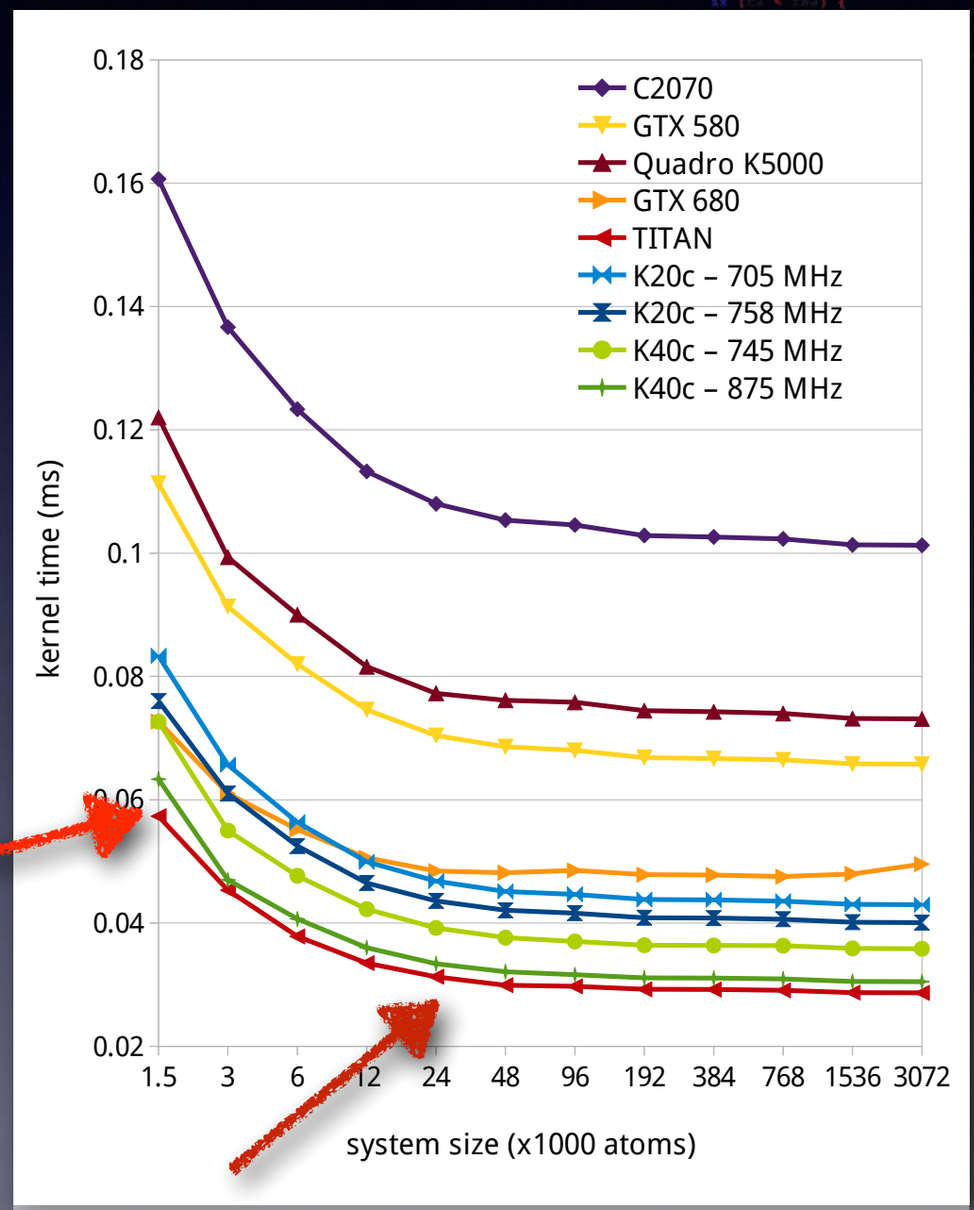
Finally becoming serious about QA

A lot of low-level tuning



GPU SMX scheduling/balancing

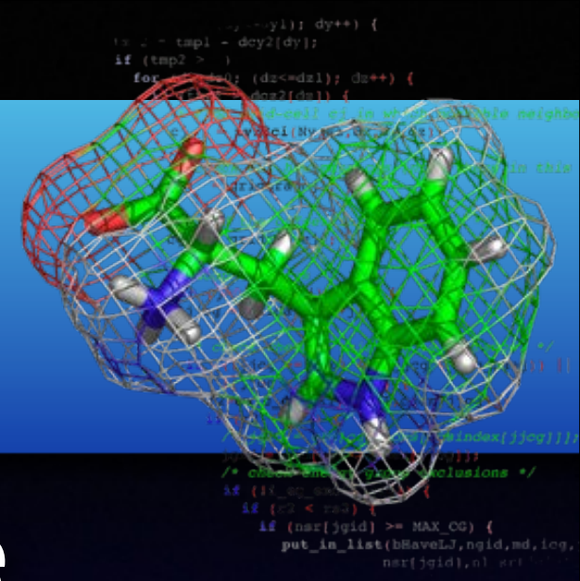




88µs actual time (1000 atoms)

If we solve all latency bottlenecks, we would approach 30µs

Hardware challenges



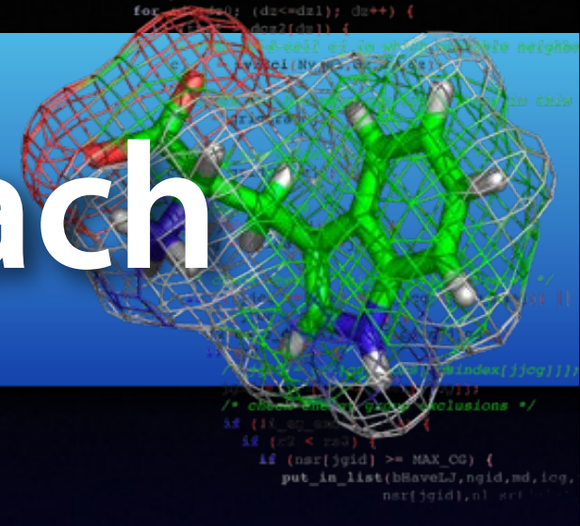
Extreme-scale HPC is expensive

A 60M core-hour project in PRACE might cost 2-3M EUR *today*

We will increasingly have to justify this in comparison with alternatives

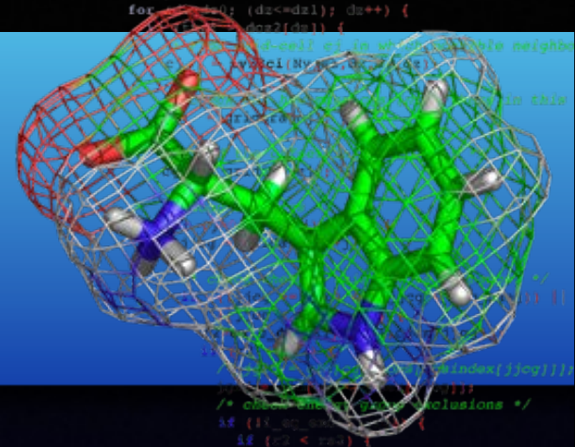
Exascale machines will likely have to be used for rapid time-to-solution, not to run single projects for weeks and months

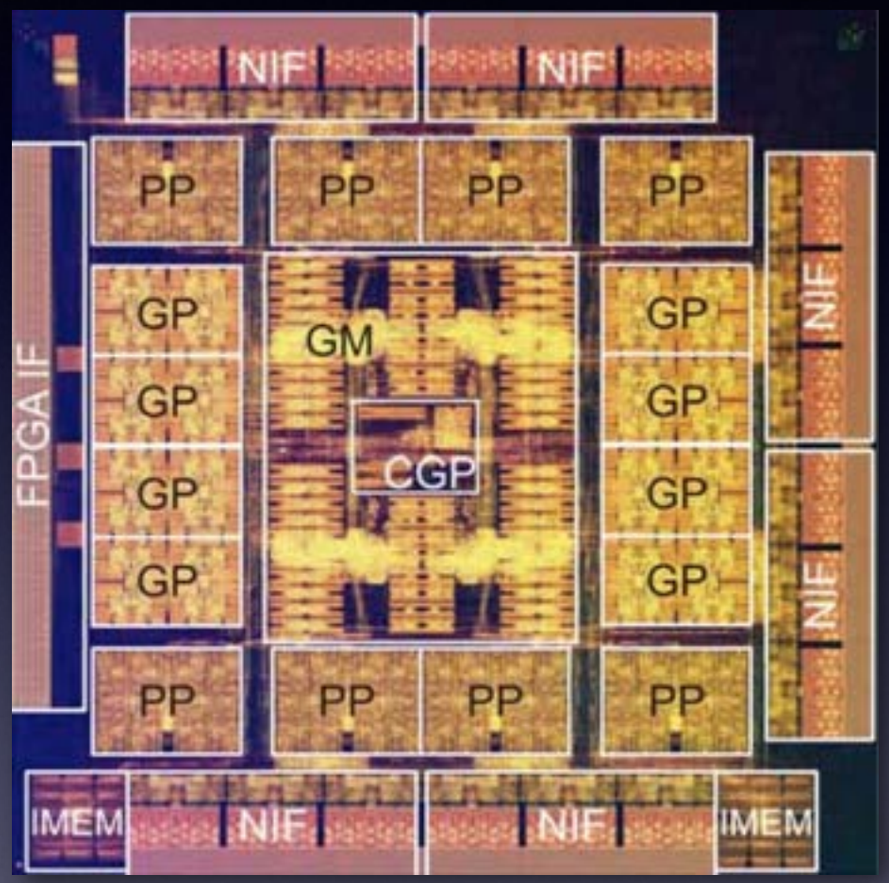
Our development approach



- Total rewrites difficult
  - QA challenges
  - Easily leads to focus on relative scaling (not perf.)
- Increasing modularization enables refactoring
  - 2005-2010: NT Domain Decomposition
  - 2008-2013: GPU/Streaming acceleration
  - 2010-2012: Heterogeneous acceleration
  - 2013-2014: Move stream version back to CPUs
  - 2014-: Fine-grained task-based scheduler (raw threads)
  - Nodes as a set of heterogeneous resources (LOC+TOC); Communication devices are also resources
- We live VERY close to the hardware
  - Most abstraction layers suck…

What can we learn from ASICs? ANTON, MDGRAPE4





2x2x2 chips in a node
64 cores & 8 pipelines / chip
Optical interconnects (~100ns)

Pipelines working on 8x8 atoms
Extremely fine-grained parallelism
~2.5 TFLOP / chip ~10µs/step

Drawback: ASICs highly inflexible, no general solution

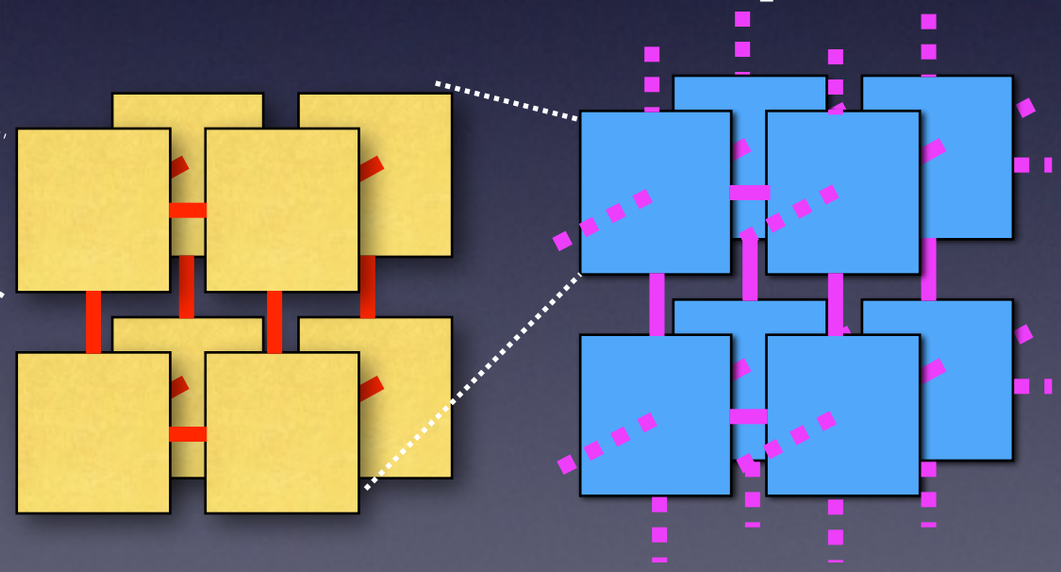
The importance of a balanced architecture:
We could reach similar raw FLOP levels with GPUs, but we are not be able to push them as efficiently today!

What machine does MD need?

Stop building HPC systems that consist of "N desktops"!
Go hierarchical, and fully expose hardware & connect

- **Even tighter connects further down**
- **Looser connects further up**

~64 LOC
~4 TOC
no PCIe!

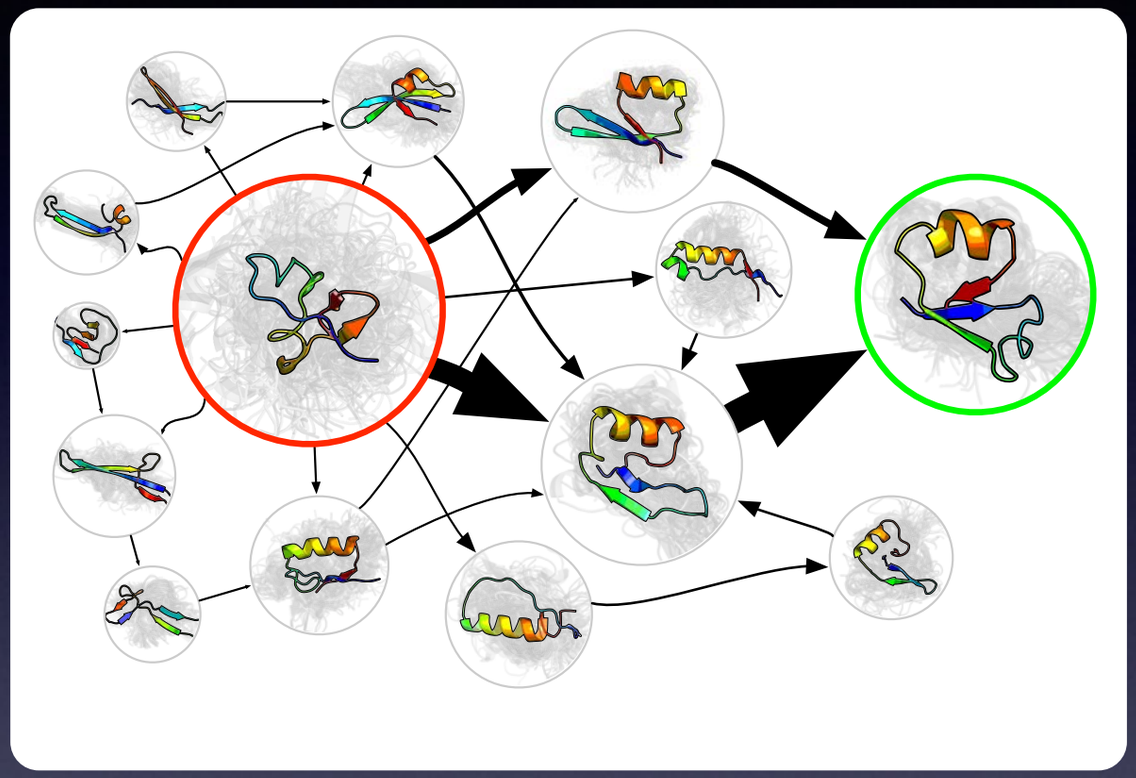


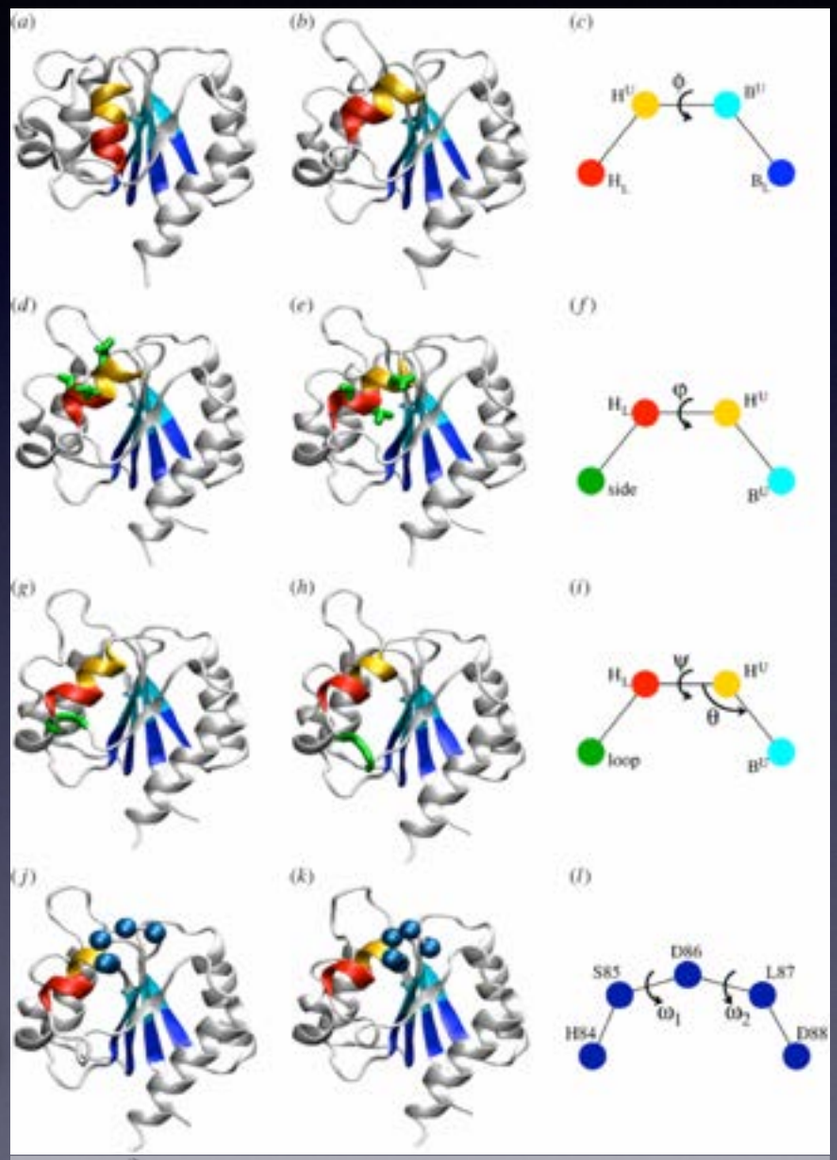
"Islands":
Say >4*4*4 units
>32768 cores
>2048 GPUs

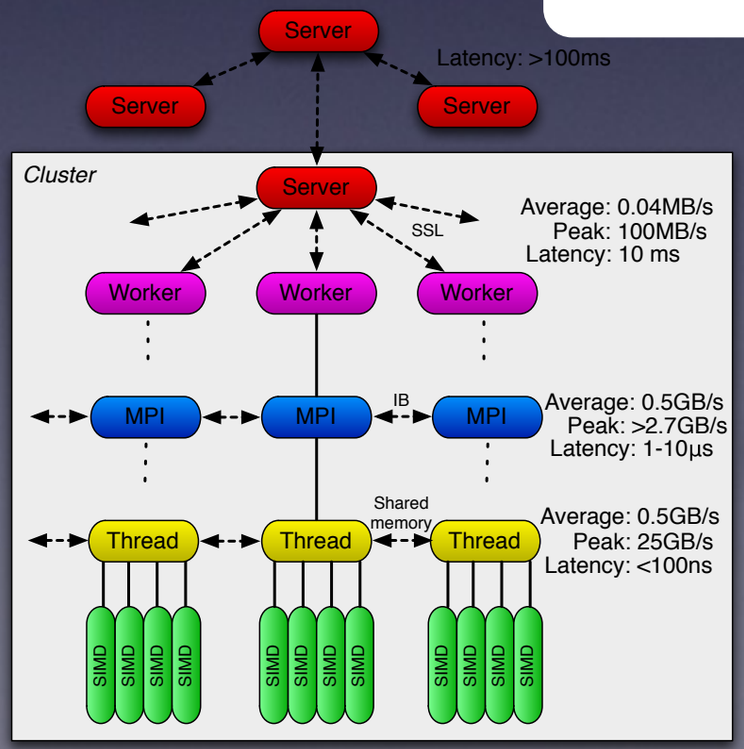
Optical, 100ns **<1µs, single-hop**

"If I had asked my customers what they wanted, they would have said a faster horse" [Henry Ford]

From ~100k cores to Exascale: Ensembles







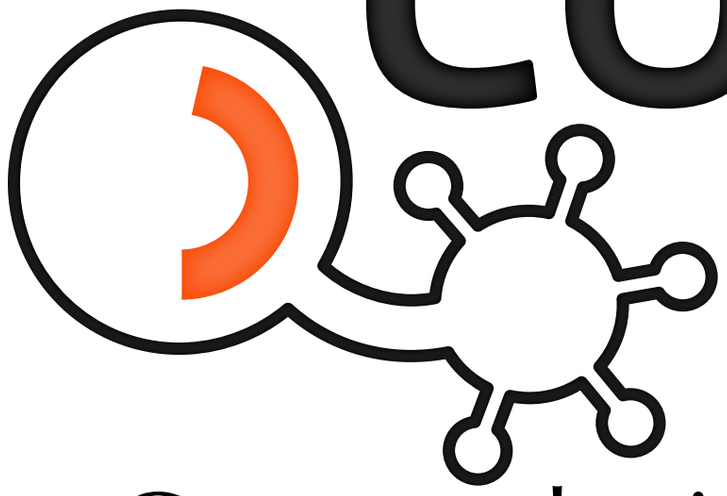
Milestoning

Markov State Models

Monte Carlo Sampling

Free Energies

Swarms / Transition pathways



copernicus

HPC ENSEMBLE COMPUTING

Current plugins:

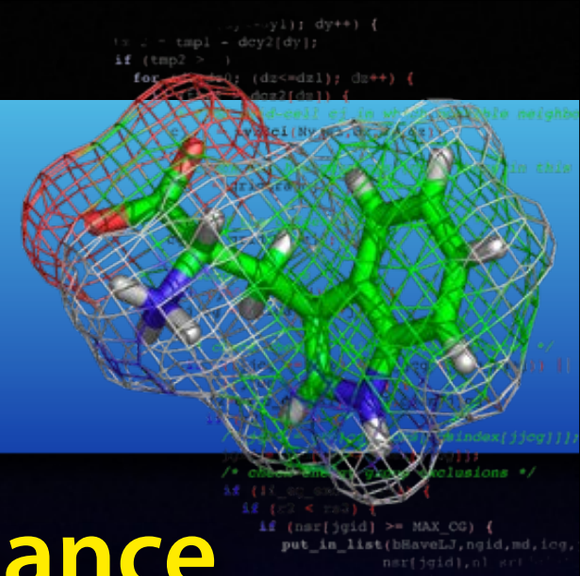
- Markov State Modeling
- Swarms
- Free Energies

Under development:

- Collective variables

Open source software available at
http://www.copernicus-computing.org/

Summary



STRONG scaling & absolute performance

Throw out all old algorithms

Heterogeneous acceleration

Hierarchical hardware needed

We live close to the hardware

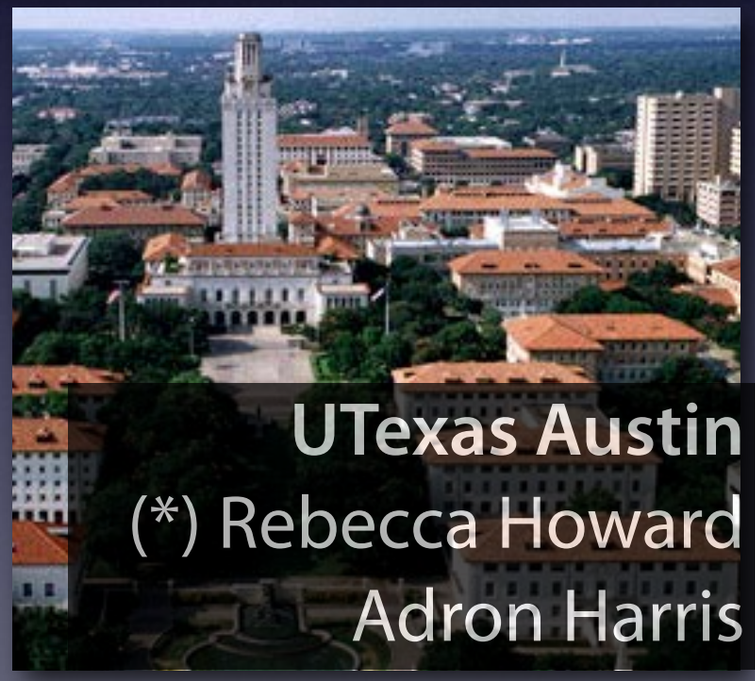
Task-based parallelism

Method Development:
Szilárd Páll (*)
Berk Hess (*)
Sander Pronk
Viveca Lindahl
Petter Johansson
Grant Rotskoff
Anders Gabrielsson
Christian Wennberg

Biophysics/ion channels:
Samuel Murail (*)
Torben Brö
Özge Yoluk (*)
Iman Pouya
Jens Carlsson
Sophie Schwaiger
Göran Klement
Magnus Andersson

Stanford University
James Trudell
Edward Bertaccini

Linköping University
Sara Börjesson
Fredrik Elinder



UTexas Austin
(*) Rebecca Howard
Adron Harris

Stiftelsen för Strategisk Forskning
erc European Research Council
7 SJUNDE RAMPROGRAMMET
National Institutes of Health
SeRC Swedish e-Science Research Centre
Vetenskapsrådet
ScalaLife