HPC usage in the University of Luxembourg Soft Matter Theory Group

Joshua T. Berryman, Muhammad Anwar, Mohit Dixit, Sven Dorosz, Anja Kuhnhold, Marko Mravlak, Amirhossein Taghavi, Tanja Schilling

Overview

Computational Challenges in Soft Matter
- Free Energy Estimation
- Reaction Pathways

Methods In Use

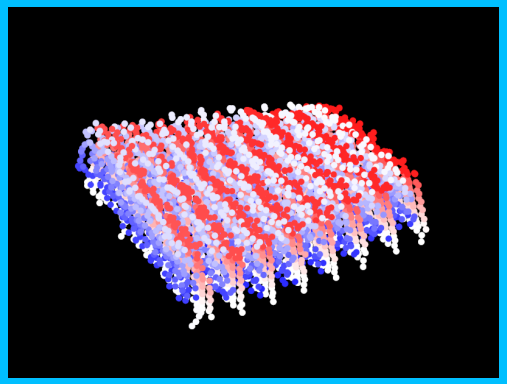
Methods And Cluster Usage Patterns
- Codes Used
- Compilation
- Launch Scripts

Free Energy Estimation

$$Z \propto \int d\vec{x} d\vec{p}\, e^{-\mathcal{H}(\vec{x},\vec{p})}$$

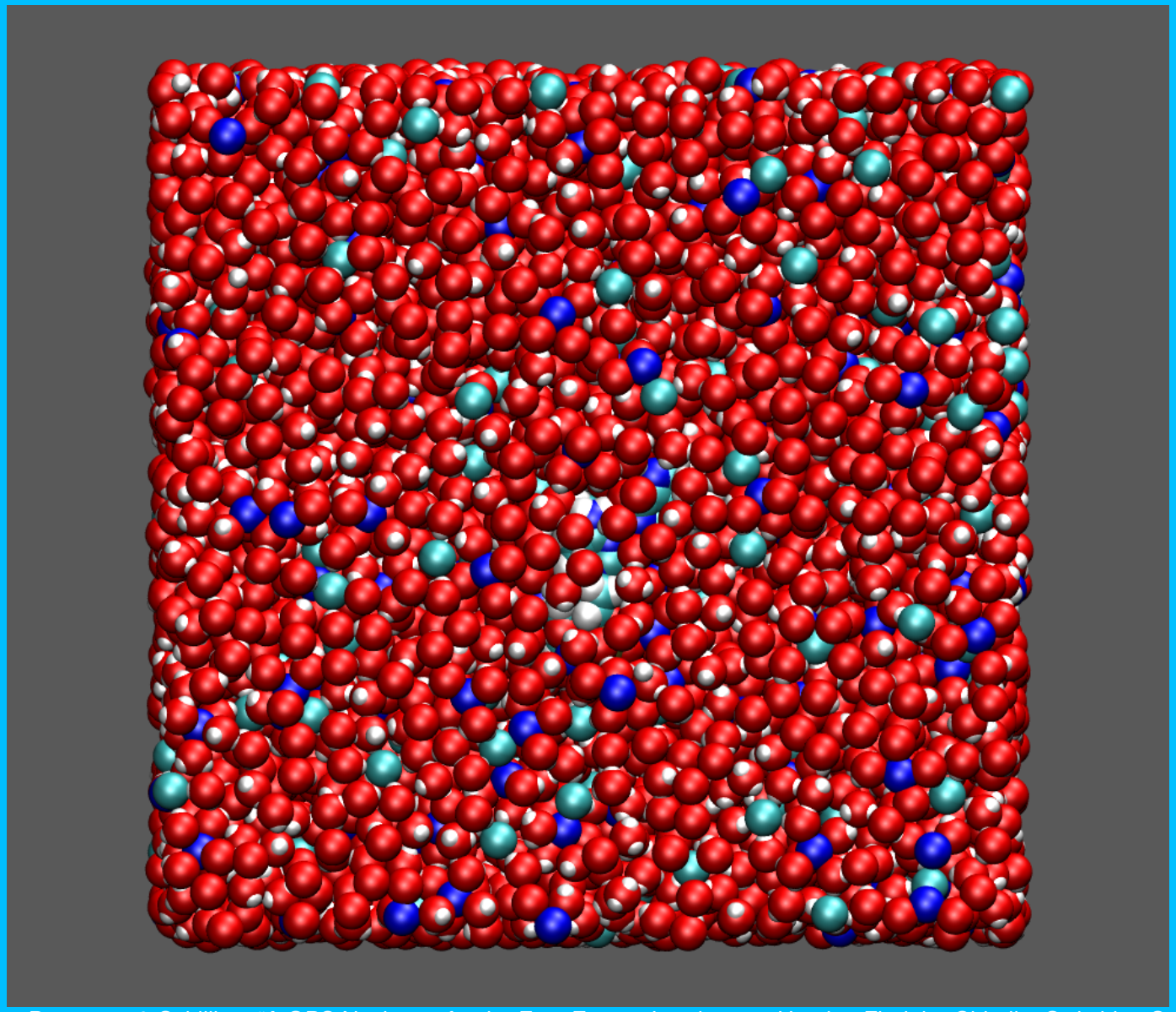
$$A = -k_B T \ln(Z)$$

- PCA to get the normal modes of the dynamics: equivalent to fitting a multivariate Gaussian to Z.
- Many, many other methods . . .



Lara, Reynolds, Berryman, Zhang, Xu, Mezzenga, "ILQINS Hexapeptide, Identified in Lysozyme Left-Handed Helical Ribbons and Nanotubes, Forms Right-Handed Helical Ribbons and Crystals." JACS 2014.

Atomistic DNA in High Salt



9441 waters

30113 atoms

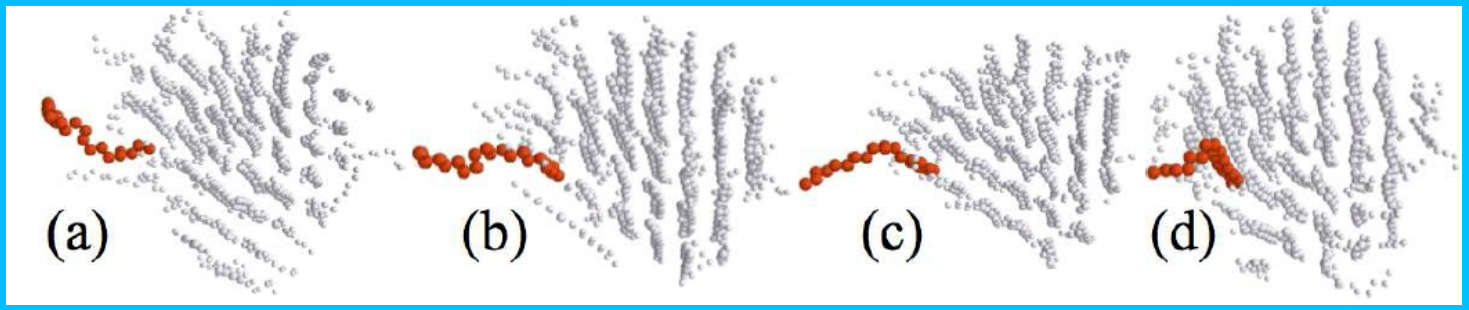
10^{-19} mol...

10^{-8} sec...

Berryman & Schilling, “A GPS Navigator for the Free Energy Landscape, Used to Find the Chirality-Switching Salt Concentration of DNA” J. Chem. Theory Comput. 2013.

Reaction Pathways

Free energy is only properly defined at thermodynamic equilibrium: to study transitions in collective behaviour, need to take a view of ‘pathways’ instead of ‘landscapes’:



- Brute force MD (e.g. Alkane nucleation pathway above).
- Also rare event methods.

Muhammad Anwar, Francesco Turci and Tanja Schilling, “Crystallization mechanism in melts of short n-alkane chains”

J. Chem. Phys. 2013

Methods In Use on UL HPC

Topic	Method	Parallelism	Username	Papers 2014-2015	
Phase Diagrams	MC	Total	sdorosz	Dorosz et al.	Soft Matter 2014
				Case et al.	AMBER 2015
				Berryman	Phys. Proc. 2014
		GPU		Lara et al.	JACS 2014
Reaction Paths	MD	12 cores/run	fturci	Turci & Schilling	J. Chem. Phys. 2014
			fturci	Turci et al.	J. Chem. Phys. 2014
		Total	manwar	Anwar et al.	J. Chem. Phys. 2014
			sdorosz	Dorosz & Schilling	J. Crystall. Proc. and Tech. 2014
			mradu	Radu et al.	Europhys. Lett. 2014
		Asynchronous	jberryman	Kratzer et al.	Comput. Phys. Commun. 2014

Username	CPU time 2013
sdorosz	195 years 307 days
manwar	128 years 105 days
jberryman	103 years 262 days

This year? Considerably less: (2nd: manwar, 3rd: sdorosz, 5th: jberrryman).

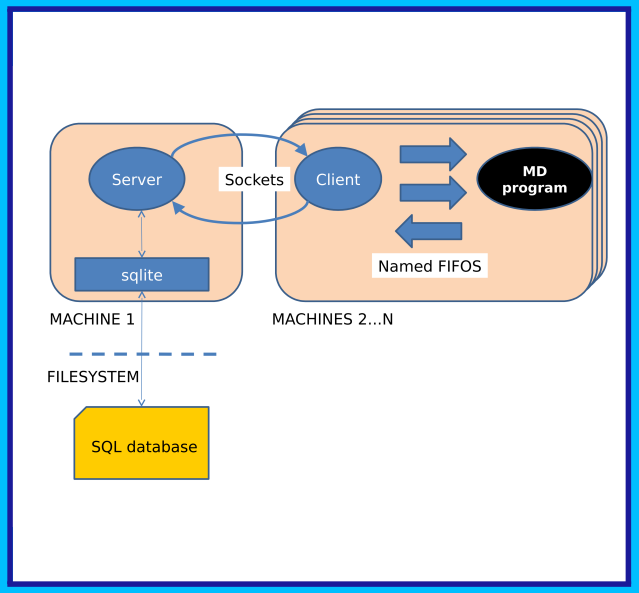
Codes Used

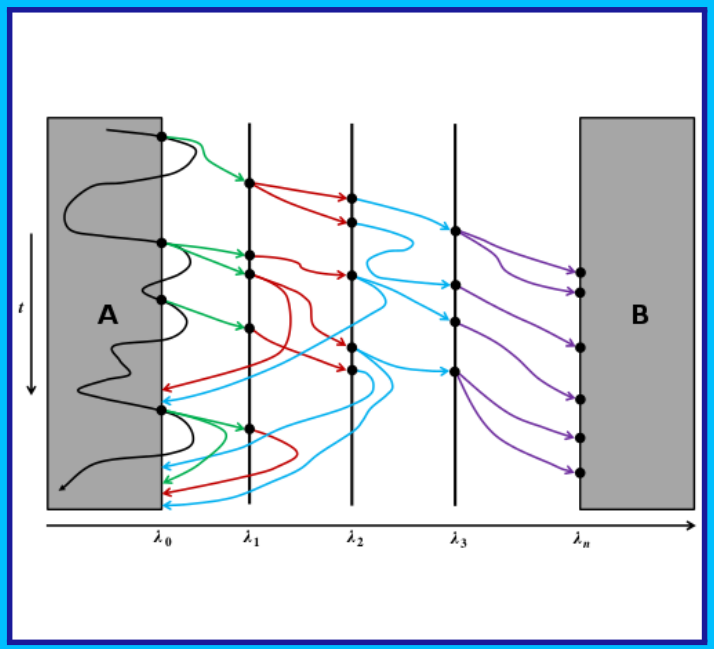
Codes:

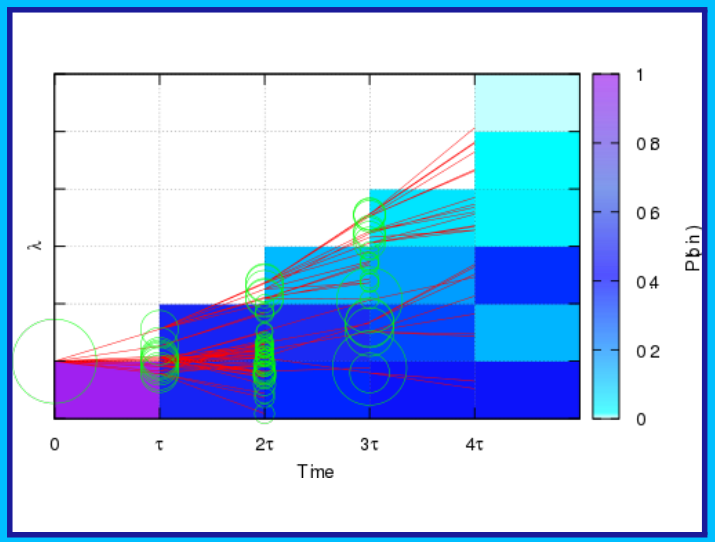
- Anwar uses ESPResSoMD, own build. **icc+impi**. Standard 12-core one node job script.
- Sven uses his own codes. **icc**. Farms groups of serial jobs.
- I use AMBER, own build. **icc+impi+(CUDA sometimes)**. 4-36 cores. Job scripts to follow.
- . . . group employs asynchronous parallelism using FRESHS to act as a wrapper for all of the above.

FRESHS

- GPL python application for rare event sampling.
- Intended as a very open collaboration, currently Kratzer, Berryman, Taudt, Zeman & Arnold.
- http://www.freshs.org







FRESHS Launch Script

Current best-practice FRESHS job script:

```
#bin/bash

##clip the first and last host ids from the list:
NODES=$(cat $OAR_NODEFILE)
SERVER_HOST=$(echo $NODES | awk 'print $1')
LAST_NODE=$(echo $NODES | awk 'print $NF')
NODES=$(echo $NODES | awk 'for(i=2;i<NF;i++)printf "%s ",$i')

##launch the server
oarsh $SERVER_HOST \
   "python $FRESHS/server/main_server.py \
   –db-folder $DB_STAGE –config-folder $CONF –config $inFile \
   >/dev/null 2>server.log" &
```

. . . continued

. . . continued from previous slide

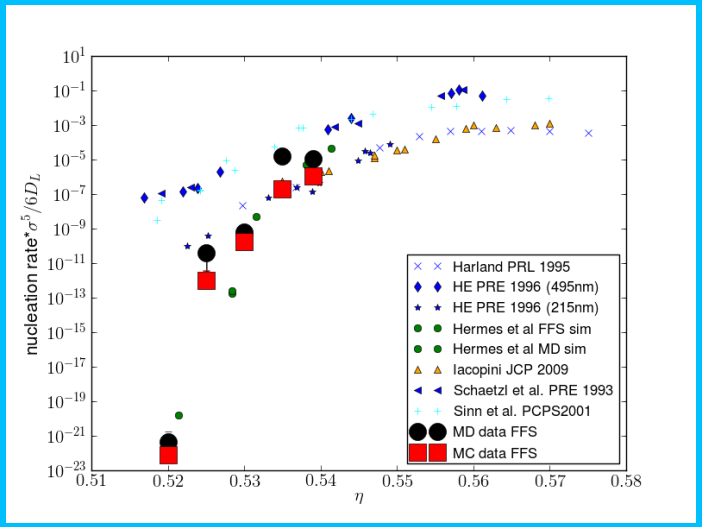
```
##launch the clients

sleep 2
count=0
for node_host in $NODES
do
   oarsh “$node_host” \
      “nice python $FRESHS/client/main_client.py –server $SERVER_HOST \
         >client$count.log 2>&1” &
   count=$[count + 1]
done

oarsh “$LAST_NODE” \
   “nice python $FRESHS/client/main_client.py –server $SERVER_HOST \
      >client$count.log 2>&1”
```

FRESHS Load Types

FRESHS Hard-Sphere Nucleation calculation by Sven:



- Usage on ganglia: ≈10% on 1 node. Code spends most of its time in comms, starting/stopping executables or blocking waits: average fullness of pipelines is small.
- Time to run for data above: ≈1 day.
- Time to get by brute force: never.

FRESHS Load Types

FRESHS calculations can be compute-bound, comms-bound or i/o-bound (SPRES). Comms can use tcp sockets (typically between nodes), named FIFOS (typically within nodes) or even just shared files (best when state info is large but infrequently visited).

HPC systems are **not** optimised for any of these types of comms.

- compute-bound: haven't yet observed this.
- comms-bound: FFS, typically.
- i/o bound: SPRES, typically.

The best strategy for i/o bound calculations so far has been to save to node-local SSD drives, then compress-and-move to project directories as a background process.

The whole thing has been made complicated by NFS and Lustre limitations.

CUDA Performance and Constraints

- AMBER
  - Basic features only are available so far.
  - $\approx$10$\times$ speedup for (1 core + 1 GPU) vs. (12 cores).
  - Memory limitations: $\approx$30k atoms. Cards have approx 6GB (vs. 24GB for nodes) so this is odd.
- ESPResSoMD:
  - advanced features only are available so far. . .

Compilation (intel):

As we understand it, best practice *currently* for any code is to use intel compilers and MPI:

```
$ oarsub -I
$ module load mpi/impi
$ source compilervars.sh intel64
$ # module load CUDA
$ export MPICC=mpicc
$ export MPIEXEC=mpirun
$ export CC=icc
$ export F90=ifort
$ make
```

MPI Launch Scripts

Current best-practice MPI job script in our group isn't very pretty:

```
#bin/bash -l
oarsub -l "nodes=2/core=12,walltime=12" \
   ". /etc/profile.d/lmod.sh; \
   ". /etc/profile.d/resif.sh; \
   module load mpi/impi; \
   mpirun -hostfile $OAR_NODEFILE $my_exe_name"
```