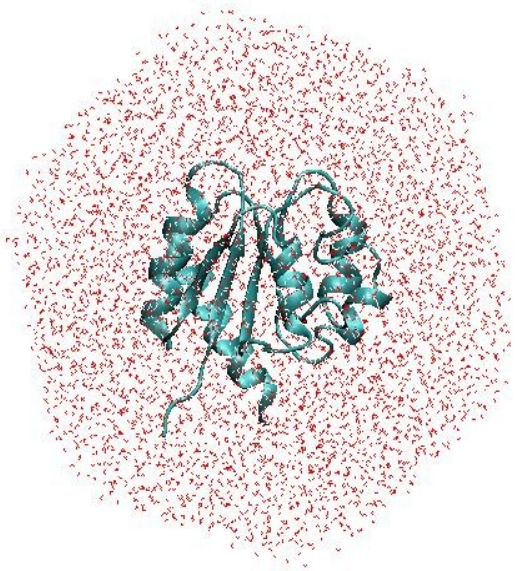
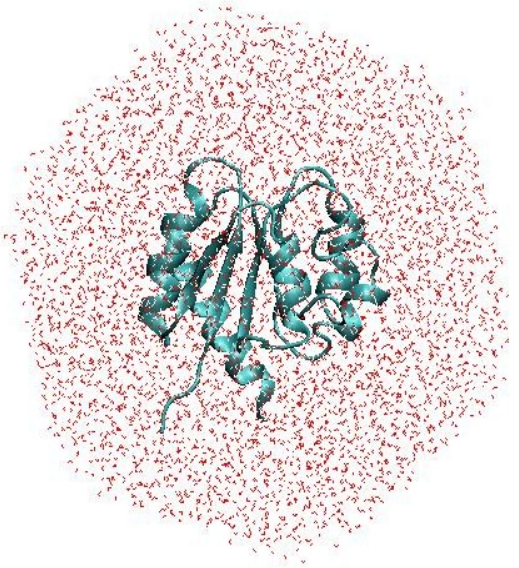
Protein molecular dynamics on OSG using CHARMM



Structure -> Dynamics -> Function



Timescales of protein motion:

femto-pico: bond vibrations, angle bending

pico-nano: loop motions, surface sidechains, water penetration

nano-micro: folding in small peptides, helix-coil transitions

micro-seconds: conformational rearrangements, protein folding, catalysis

Physical complexity:
various shapes, sizes,
bound non-protein molecules

Environment: water,
membrane, pH, ions,
gases, small molecules,
macromolecules

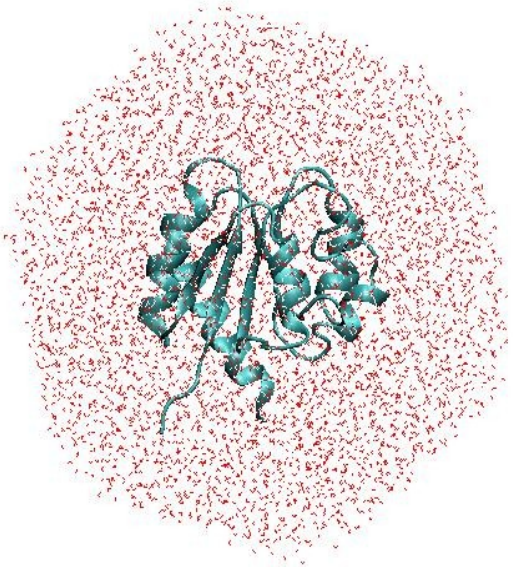
Molecular dynamics simulations

All atoms described explicitly (including water molecules, ions).

Interaction between atoms through empirical potentials:
bonded terms: bond vibrations, angle bending, dihedrals ...
non-bonded terms: electrostatic, van der Waals.

Time evolution of the system obtained through integration of Newton's equation of motion.

Integration timestep is 1-2 fs. Motions at the order of ns, or 10-100 ns are accessible through MD simulations.



Why we need the grid?

* Achieve statistically meaningful results
  (most experimental techniques deal with ensembles).
  This will become possible for processes that occur on
  timescales of 10-100 ns (water penetration).

* Increase probability of observation of processes that occur on timescales
  longer than microseconds: protein folding,
  protein conformational transitions.

* Simulate related proteins (comparative study)

* Simulate proteins under slightly different conditions (e.g., with bound
  protons or small molecules)

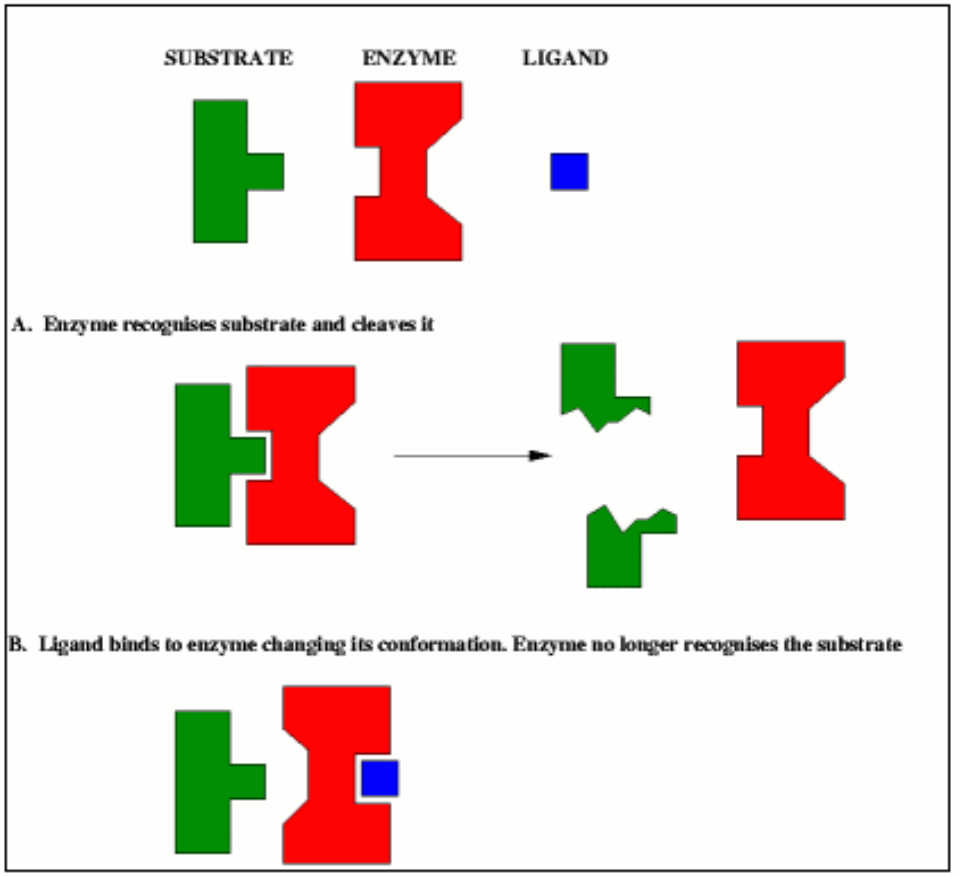
Understanding protein conformations

Understand effects of long time dynamics on structure and function.

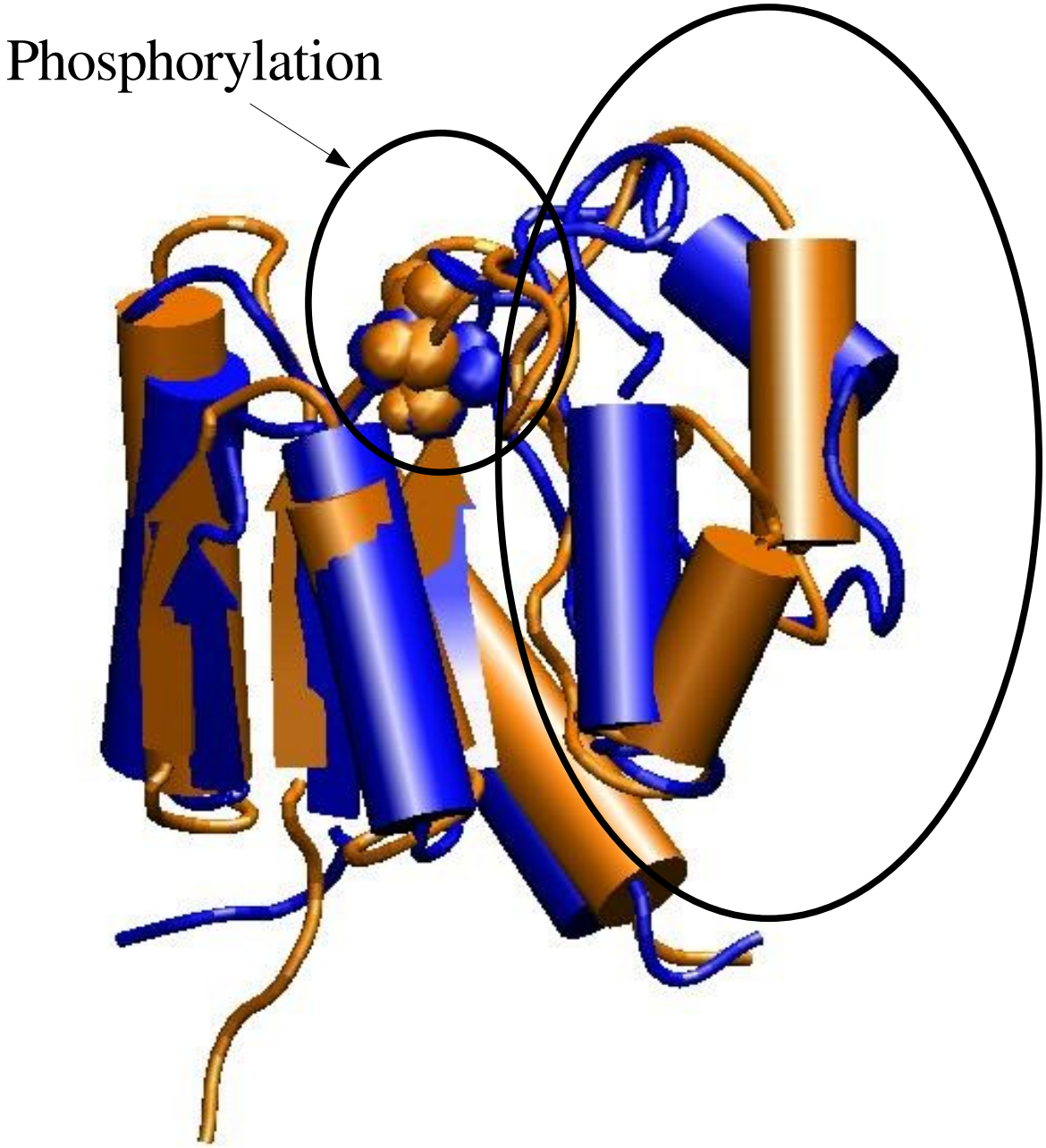
and protein conformational transitions

Protein conformation can be changed through changes in environment (such as pH) or binding of small molecules.

This can be used as a mechanism of **CONTROL** of protein activity.



Conformational changes induced by phosphporylation



Phosphorylation
favors **active** vs inactive
conformation.

There are two NMR structures
of the active form.

Run simulations for:
1) active1, phosphorylated
2) active1, unphosphorylated
3) active2, phosphorylated
4) active2, unphosphorylated
5) inactive, phosphorylated
6) inactive, unphosphorylated

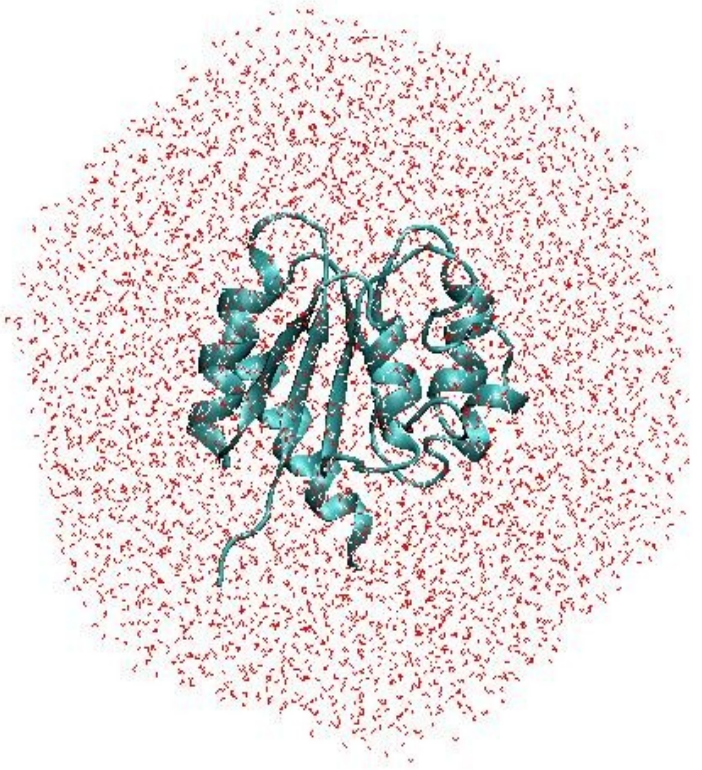
What is CHARMM?

CHARMM is a **general** and **flexible** software application for modeling the structure and behavior of molecular systems. More information is available at http://www.charmm.org.

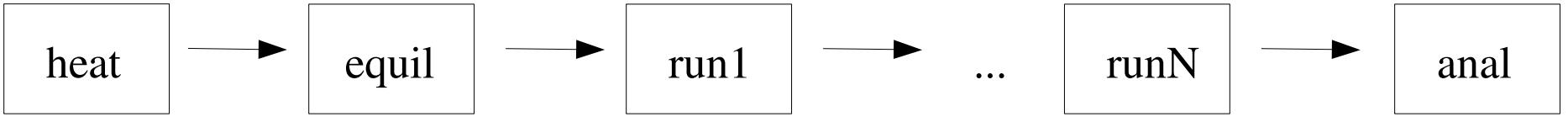
* variety of systems: small molecules - large oligomeric proteins in its solvent environment
* QM/MM potentials
* energy minimizations, molecular dynamics, vibrational analysis
* variety of analysis tools

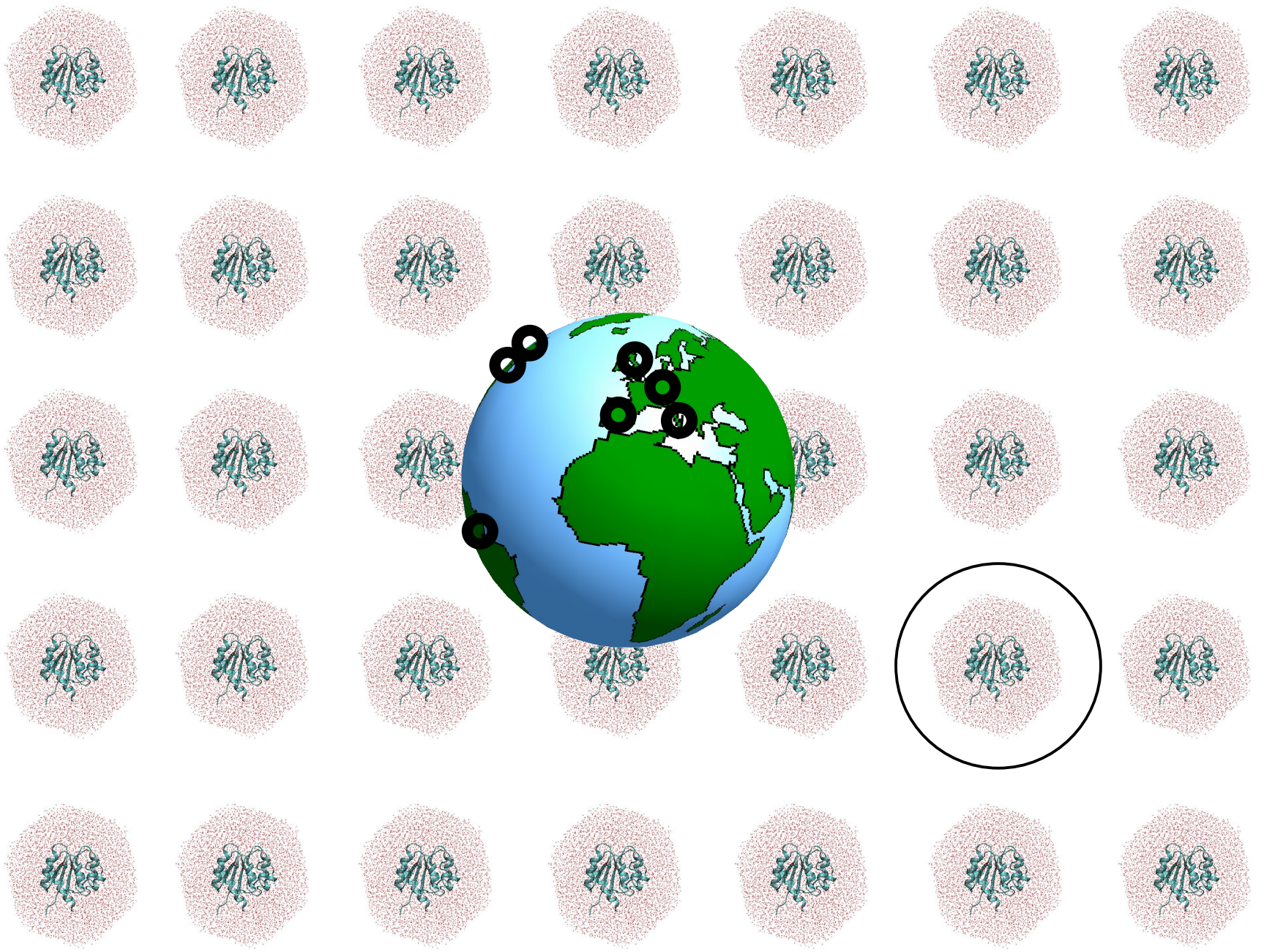
System setup

2,000 protein atoms
+16,000 water atoms
=18,000 atoms



Typical sequence of a CHARMM molecular dynamics job





With a software that can babysit the jobs while we sleep ...

software managing your jobs

Input

Output

Implementation of CHARMM on the OSG

What do we need to have (requirements)?

- A way to set up various run parameters.
- Ability to submit and track many jobs.
- Easy access to input and output files from the grid.

What application specific challenges must we deal with?

- The framework must allow for maximum flexibility since CHARMM can do many things.
- Efficient handling of many input and output files.
- Figuring out queue lengths and resource limitations and tailoring jobs to them.
- Restarting failed jobs.

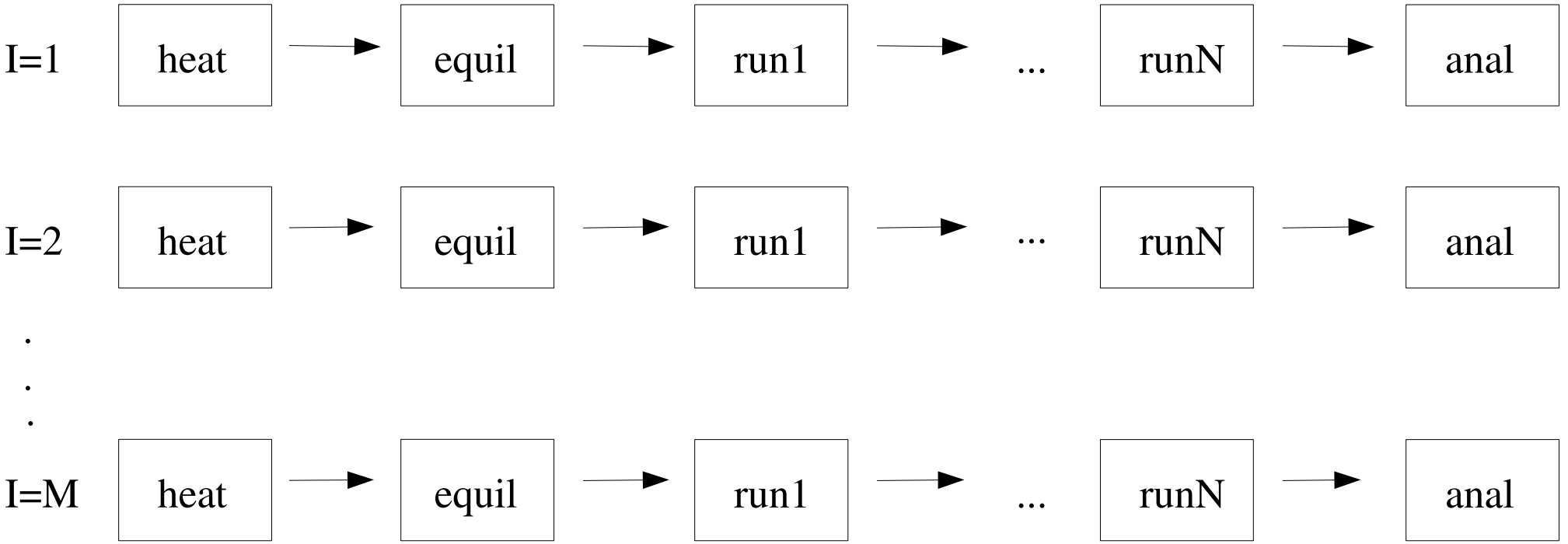
Solution: Use PanDA and a custom set of management scripts

The Scheduler Interface

- We use the PanDA front end.
- We also use TestPilot and run our own pilot scheduler for maximum control.
- Users can track jobs via a Web interface.

CHARMM Job Management

- Thread and wave model. Each independent case is a thread and each step in the analysis is a wave. Each job can have many threads with the same waves.
- The individual jobs keep track of their state information and pass it to the next wave in the thread.
- Each job automatically submits the next wave in the thread upon its own completion.



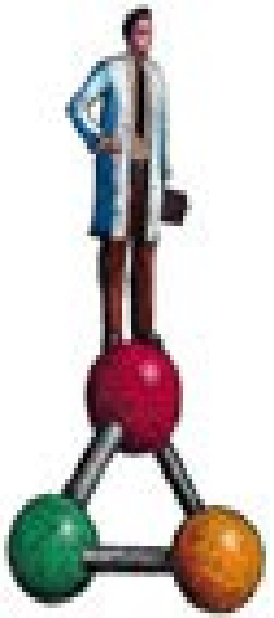
Job definition from the researcher's point of view

The following steps are required to set up and submit a job:

1. Obtain CHARMM and the PandaForCharmm software.
2. Create the various input scripts needed for the jobs.
3. Pack these and other necessary files into a tar.gz to be extracted on the execution host.
4. Modify parameters in charmmJob.sh, example:

Example thread and wave parameters:

```
export tarball=ana2.tar.gz
export exe=c33a2-lrg.one
export jobname=ana3
export threadparams="I=[jobid]"
export inpscripts="heat=heat.inp,equ=eq.inp,md=run.inp"
export threaddef="heat,equ*2,md*2"
```



5. Run charmmJob.sh
6. Watch your jobs run!

The Web Interface (constructed by Torre Wenaus)

Configuration

Update

Panda monitor

Quick guide, twiki

Jobs - search
Recent running, activated, waiting, assigned, defined, finished, failed jobs
Select analysis, production, test jobs

Quick search
Job
Dataset
Task
File

Summaries
Blocks: days
Errors: days
Nodes: days
Daily usage

Tasks - search
Generic Task Req
EvGen Task Req
CTBsim Task Req
Task list
Task browser

Datasets - search
Dataset browser
New datasets
Panda subscriptions
All subscriptions

Sites - see all
BNL BU IU OU SLAC UC UMICH UTA LCG NG

Applications
CHARMM

Dashboards: Production DDM Sites & Grids Analysis Physics data Task definition Usage & Quotas TestPilot Plots ArdaDash

CHARMM job overview

Recent CHARMM job submitters: Ana Damjanovic Benjamin Timothy Allen Miller

Recent CHARMM pilots: all submitted (8) scheduled (0) running (4) finished failed aborted

Queues used by CHARMM

Recent CHARMM jobs (last 3 days): ana1 ana2 ana3

All jobs:

	jobs	active	run	finish	fail
Totals:	*50*	*0*	*42*	*8*	*0*

Jobname ana1:

Wave	jobs	active	run	finish	fail	PandaIDs
Totals:	*23*	*0*	*22*	*1*	*0*	
4	1	0	0	1	0	1049437
3	4	0	4	0	0	1008684 1009260 1009327 1010430
2	3	0	3	0	0	998138 999868 1002365
1	12	0	12	0	0	989027 992341 992342 993107 993108 995119 995421 996927 997229 998134 998137 998139
0	3	0	3	0	0	1039253 1039254 1039255

Jobname ana2:

Wave	jobs	active	run	finish	fail	PandaIDs
Totals:	*10*	*0*	*10*	*0*	*0*	
0	10	0	10	0	0	1055685 1055686 1055687 1055688 1055689 1055690 1055691 1055692 1055693 1055694

Jobname ana3:

Wave	jobs	active	run	finish	fail	PandaIDs
Totals:	*17*	*0*	*10*	*7*	*0*	
2	3	0	3	0	0	1067943 1068544 1069746
1	4	0	1	3	0	1069745 1061206 1061207 1061208
0	10	0	6	4	0	1056295 1056296 1056297 1056298 1056299 1056303 1056300 1056301 1056302 1056304

Where we are and where we want to go

Currently:

- Basic set up of the thread and wave model is completed and we've tested our own scripts extensively.
- We have started production runs with fifty threads of twelve waves.
- 100K step jobs are taking about 1 day to finish. This means we can simulate 1 ns per thread in 180 to 300 hours of wall time!

Future Directions

- Ability to introduce “branches” in the script sequence, to allow, for example, extra analysis of “interesting” structures.
- Better tracking of in-progress jobs along with failure detection and possible correction.
- A graphical front-end for job definition and submission.
- Gaining a better understanding of various sites and queues so we can better match jobs to resources.

Thank You!

NIH
Bernard Brooks

Fermilab/Johns Hopkins University
Petar Maksimovic

Open Science Grid
Wensheng Deng (BNL)
Torre Wenaus (BNL)
Frank Wuerthwein (UCSD)