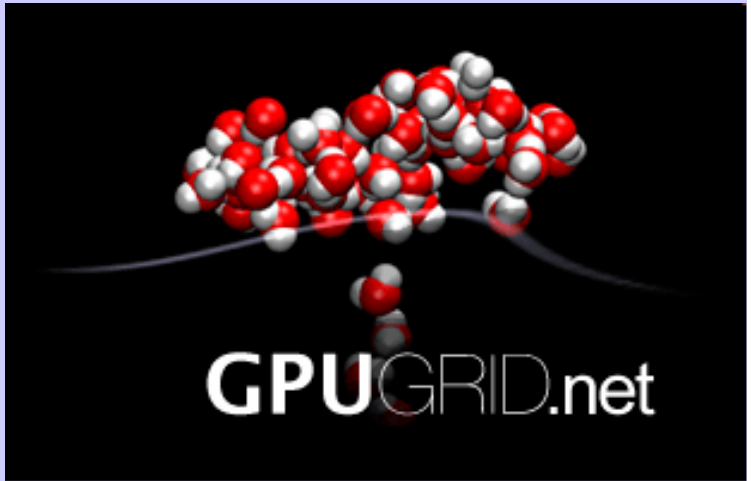
Submitting and managing distributed computations

The researcher's interface to a BOINC project

Toni Giorgino, I. Buch, K. Sadiq, M. J. Harvey, G. De Fabritiis

Computational Biochemistry and Biophysics Lab / GRIB-IMIM
Barcelona Biomedical Research Parc (PRBB)
www.gpugrid.net

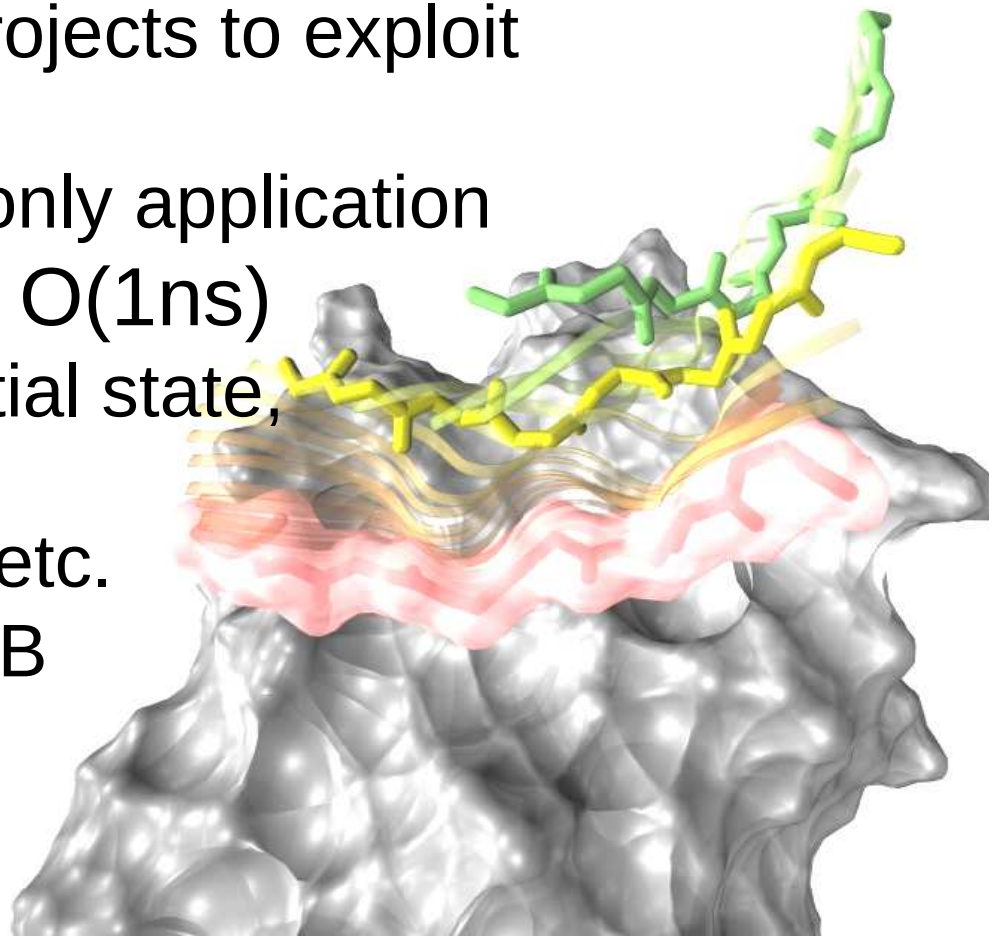
Thursday, 22 October 2009



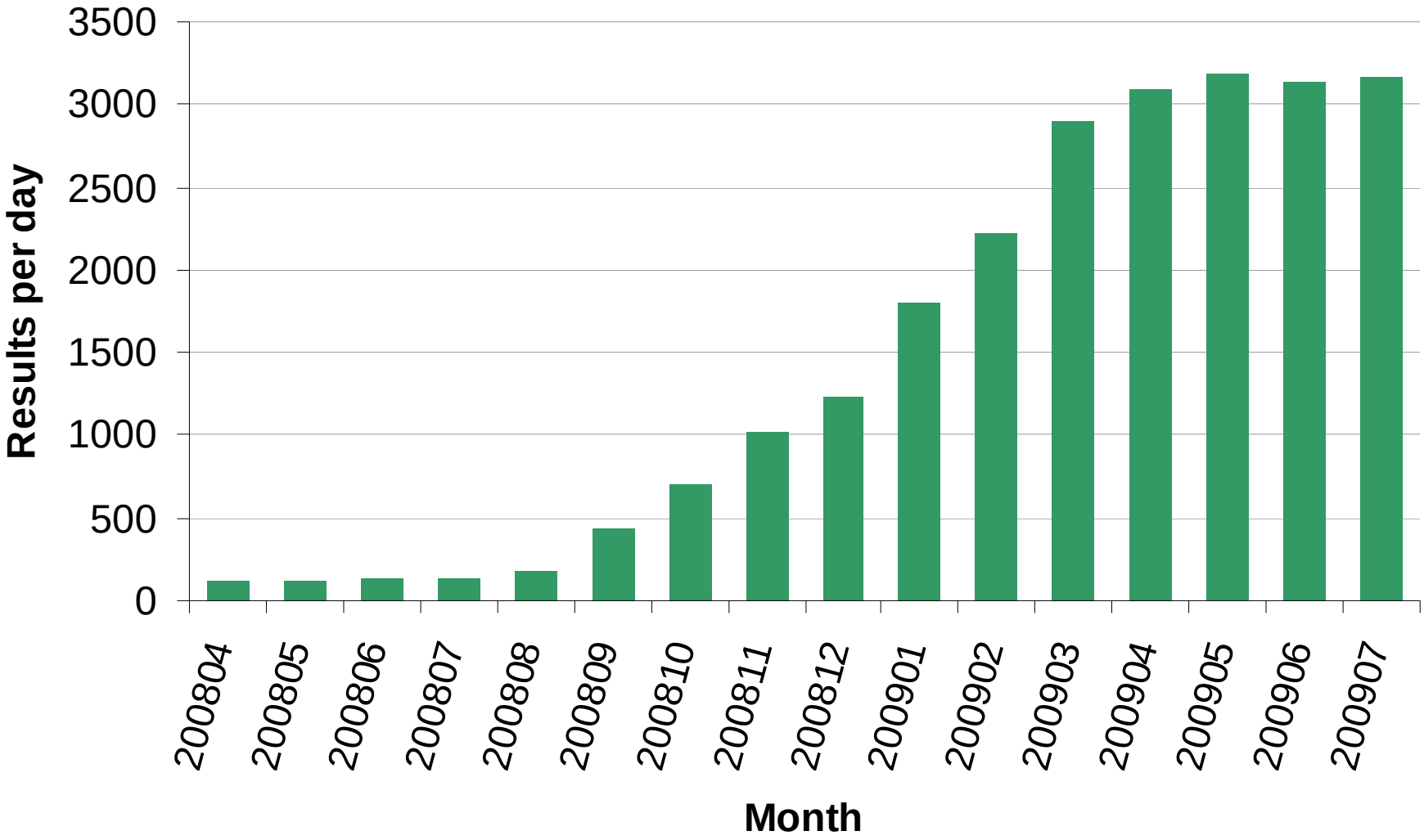
GPUGRID.net *-or-*
Remotely using BOINC as a
“virtual supercomputer”
for molecular simulations

GPUGRID.net

- Our aim: bridge the “scale gap” between biological models and the most detailed atomistic representations available
  - One of the first BOINC projects to exploit accelerated processors
  - Highly optimized CUDA-only application
- Each work-unit simulates O(1ns)
  - Requires parameters, initial state, simulation protocol, etc.
  - Returns final state, logs, etc.
  - ~5 PFLOP/WU, 8h, 30 MB



Workunit turnaround

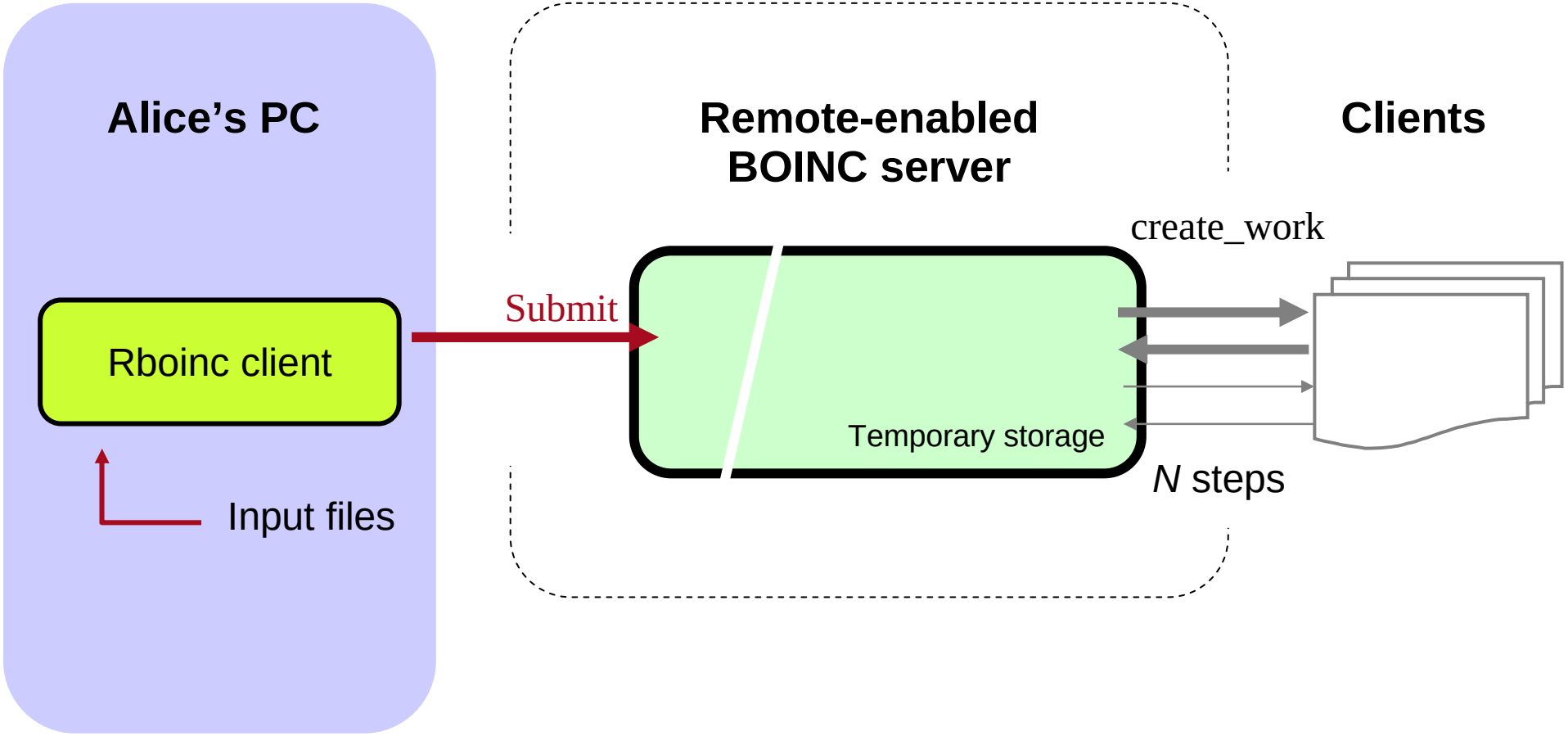


Submission, in practice - *or* -
Where do these WUs come from?

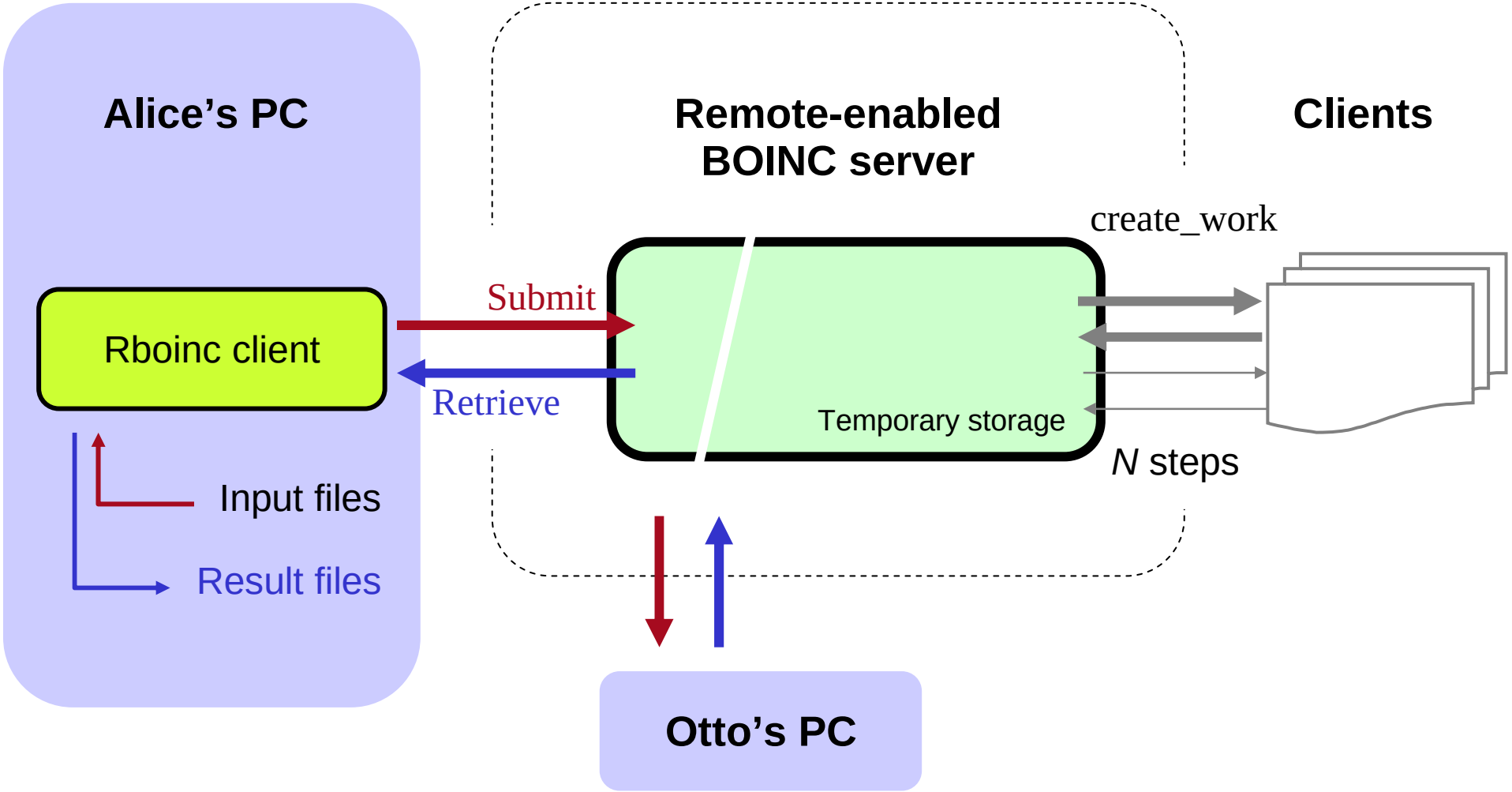
BOINC as a *virtual supercomputer*: overview

- We have *scientists* to create work-units
  - They do not access the server directly
  - Not necessarily BOINC- or even coding- experts
  - Could be at remote sites
- **Unrestricted access to server is ruled out**
- A *remote boinc client* allows them to
  - *submit* their computations
  - *retrieve* results
  - *manage* and *monitor* progress of WUs

Remote scientist interface: overview



Remote scientist interface: overview



Submission, in practice

- Alice can start an hypothetical simulation with this command line

Alice's terminal

```
rboinc_submit
      -url http://big.edu -auth …
      -app md
      -help_parameters
```

Submission, in practice

```
boinc_submit.pl \
        -url http://www.ps3grid.net:8383/rboinc_cgi \
        -app meta      -help_parameters

Remote application queue `meta'
Description: Standard ACEMD run with optional DCD and PLUMED
Application on server: `acemd'

Options defined for this application queue:
    -pdb_file           PDB structure
    -coor_file          Binary coordinates
    -hills_file         (optional) PLUMED metadynamics restart
    -idx_file           (optional) (undocumented)
    -metainp_file       (optional) PLUMED metadynamics configuration
    -conf_file          ACEMD input file
    -par_file           CHARMM parameters
    -psf_file           PSF topology
    -vel_file           Binary velocities
```

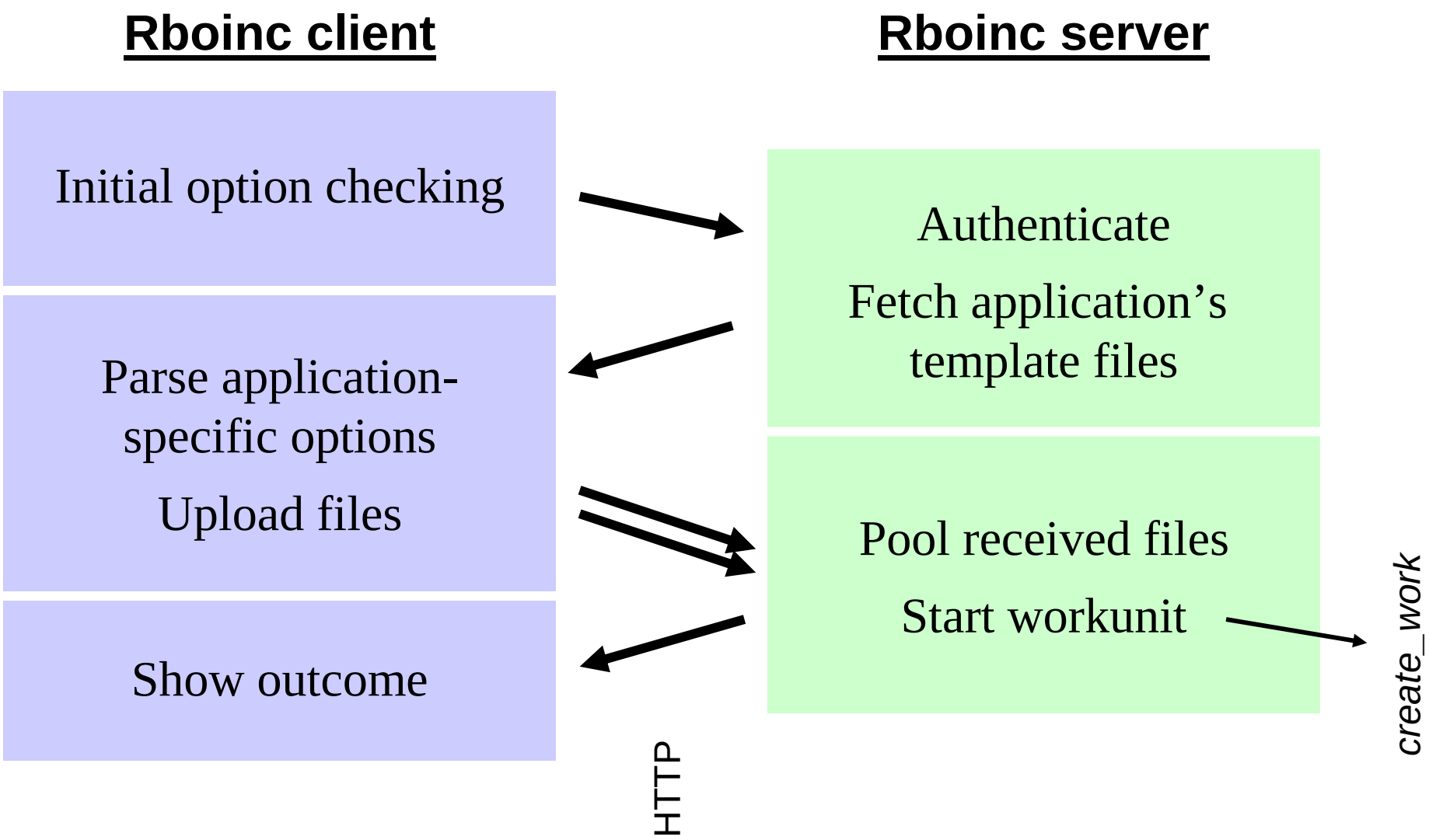
Submission, in practice

- Alice can start an hypothetical simulation with this command line

Alice's terminal

```
rboinc_submit
    -group AQUAPORIN -name SEQ1
    -url http://big.edu -auth …
    -app md –num_steps 20 -metadata …
    -pdb_file aquaporin.pdb
    -coor_file initial.coor […]
```

A look behind the scenes



Retrieving, in practice

- Now Alice wants to download the results computed so far
  - Files will be removed from the server

Alice's terminal

```
rboinc_retrieve
     -group AQUAPORIN [-name SEQ1]
     -url http://abigserver.edu -auth …
```

Monitoring

- Each WU is associated with a *scientist* and a *group*
- Useful for accounting and error-checking
  - How many credits did I “consume”?
  - Faulty WUs can be traced back
- Reports on demand
  - Which WUs am I running?
- Nightly reports
  - Who is running, how much, and at what error rate?

Nightly report: example

Per-user

scientist	sent	unsent	day_suc	day_unsuc	day_credits
ALICE	1527	1453	868	336	5134196
BOB	2305	447	1466	390	7704984

Per-user per-error

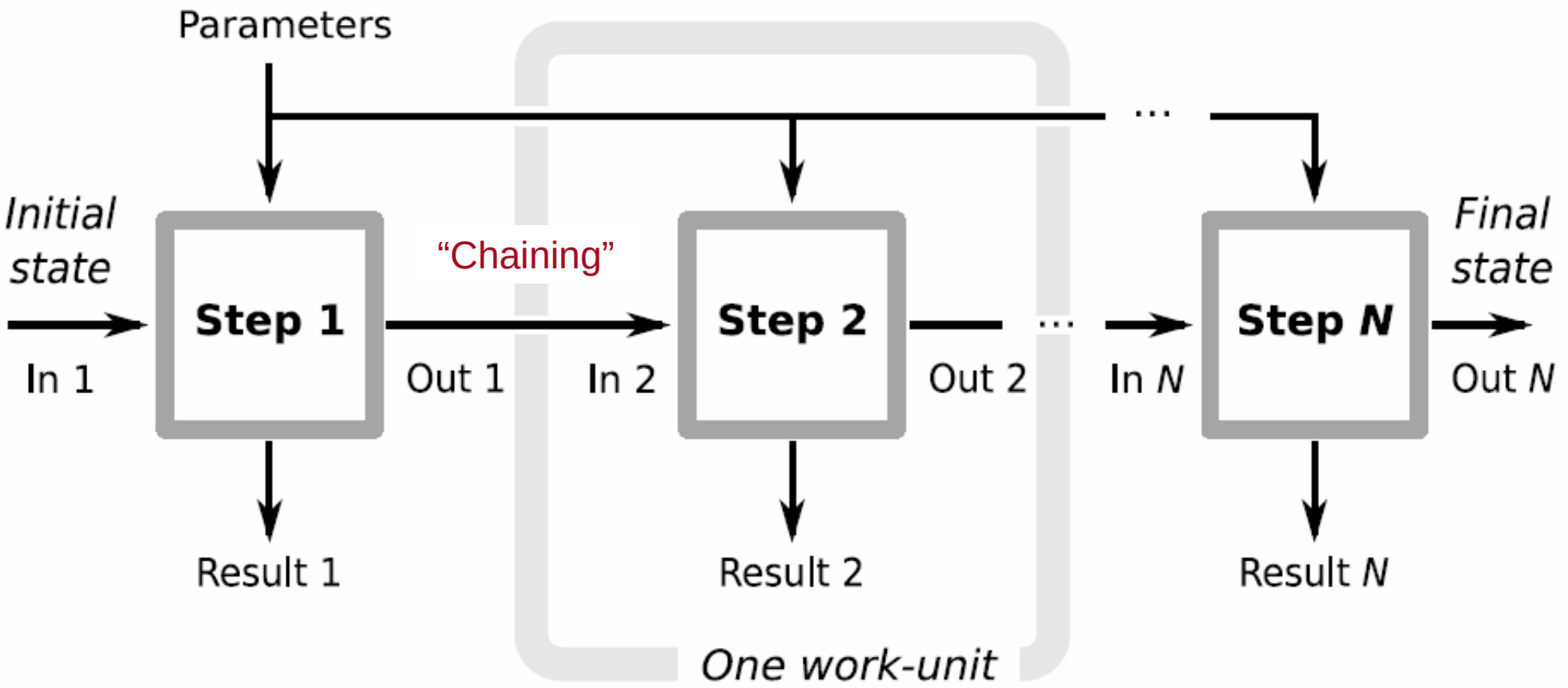
Code	Error	ALICE	BOB	[...]
-177	RSC_LIMIT_EXCEEDED	0	0	
-185	RESULT_START	0	1	
-186	RESULT_DOWNLOAD	12	9	
-187	RESULT_UPLOAD	0	0	
-197	ABORTED_VIA_GUI	17	21	
-198	INSUFFICIENT_RESOURCE	0	0	
-226	TOO_MANY_EXITS	1	13	
-233	UNSTARTED_LATE	7	8	
1	(Exit code by app)	151	180	
2	(Exit code by app)	0	0	
3	(Exit code by app)	37	28	
other	(Any other exit code)	111	130	

Per-group

group_name	sent	unsent	day_suc	day_unsuc
ALICE_BIND_11	81	20	51	21
ALICE_BIND_166_119	422	81	266	94
BOB_TEST_2	83	16	51	20
	[...]			

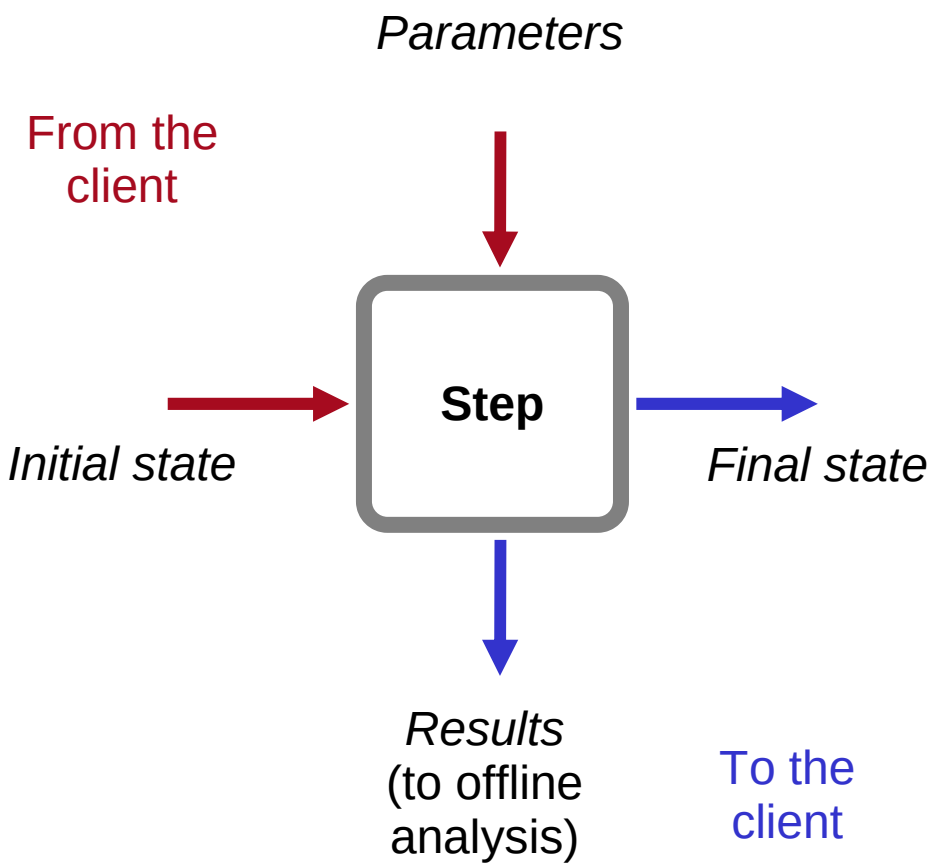
Remoting files *-or-*
the remote application description

Chains as dependent steps



Let's focus on one step: the application description

- Arrows are the *file interface* of a WU
- We have to *expose* input and output files over the network to Alice's PC
- Akin to *staging*
- Accomplished cleanly, extending BOINC *templates*



Extension of *wu_template*

- *Input files* are enabled
  by just adding the *rboinc* tag

rboinc_app_wu

```
[…]

<file_ref>
        <file_number>5</file_number>
        <open_name>structure.pdb</open_name>
        <rboinc parameter_name="pdb_file"
                parameter_description="PDB structure"/>
</file_ref>

[…]
```

Extension of *result_template*

- *Outputs* are enabled similarly, and optionally chained with the *chain* attribute

rboinc_app_result

```
[…]

<file_info>
    <name><OUTFILE_1/></name>
    <generated_locally/>
    <upload_when_present/>
    <max_nbytes>50000000</max_nbytes>
    <url><UPLOAD_URL/></url>
    <rboinc aliases=".coor" chain="2"/>
</file_info>

[…]
```

File pooling

- Usually, simulations share a lot of files
- We “pool” files within the same group
  - Equal files are reused
  - All files are automatically indexed by content: pooling is transparent for users
  - Saves O(1000) disk space
  - Files in upload and download directories are also linked to the pool, rather than copied

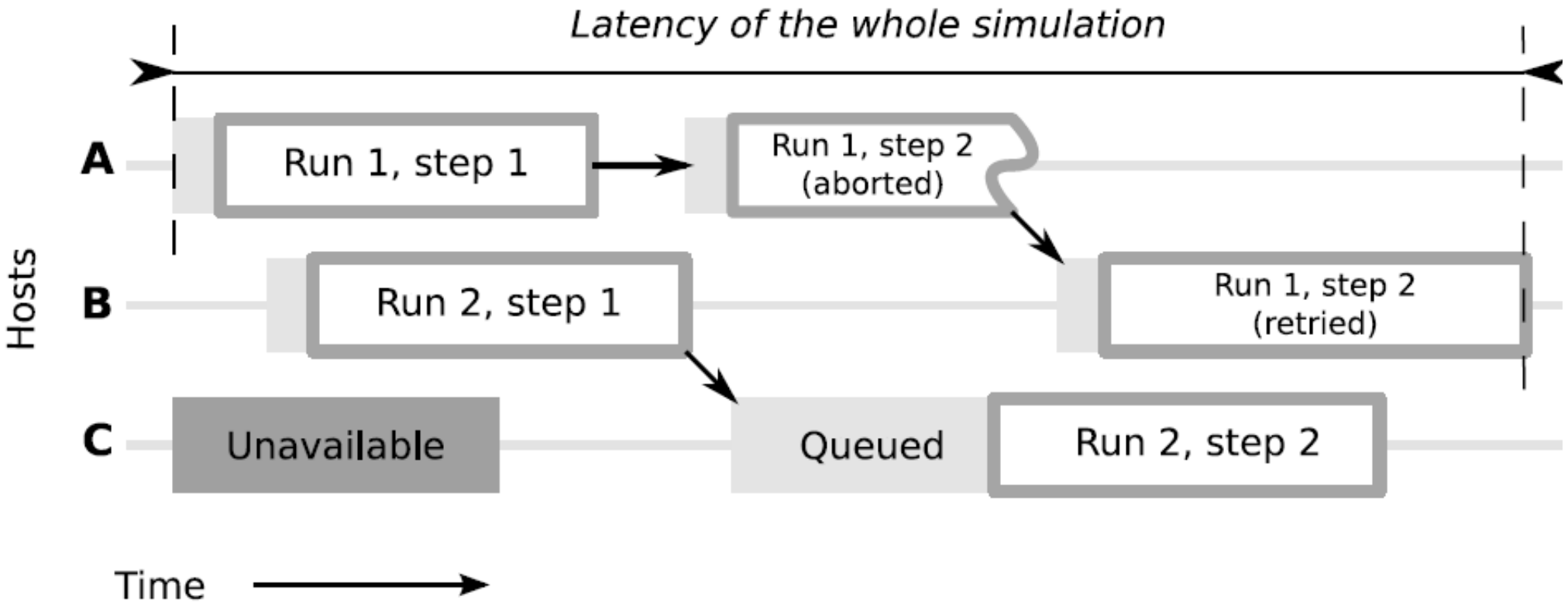
Clean storage layout

- On the server, groups stay organized in a natural hierarchy

```
Workflow/
   ALICE_GRAMICIDIN/
      SEQ1/
         (inputs for seq1)
      SEQ2/
         (inputs for seq2)
      pool/
         (pooled file storage)
      (results)
      process*
   ALICE_AQUAPORIN/…
```

Load balancing

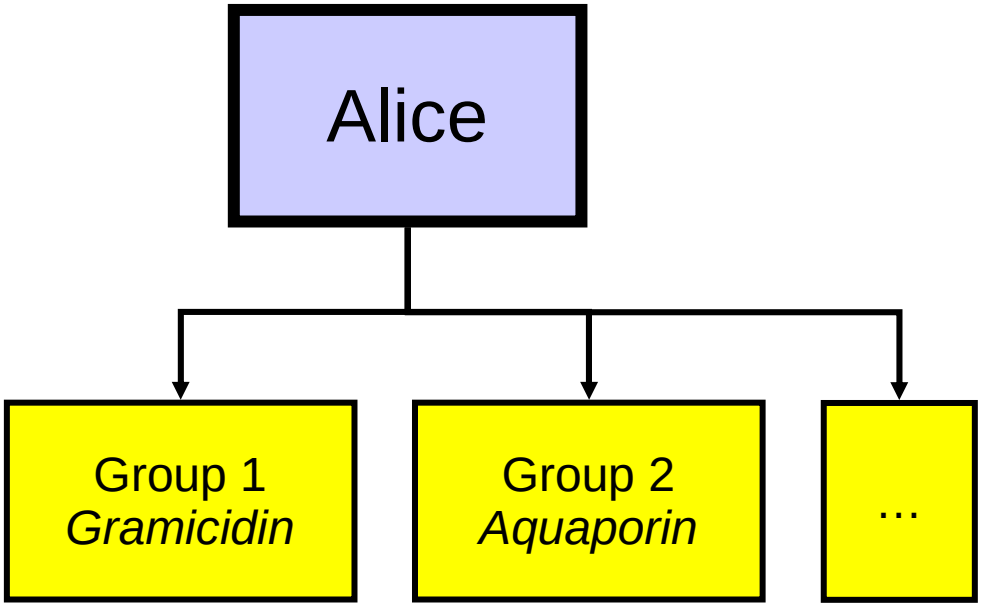
- Sometimes, one is willing sequences to proceed *on par*
- Load balancing raises the priority of “late” steps when they are generated



Job layout *-or-*
example of gpugrid.net at work

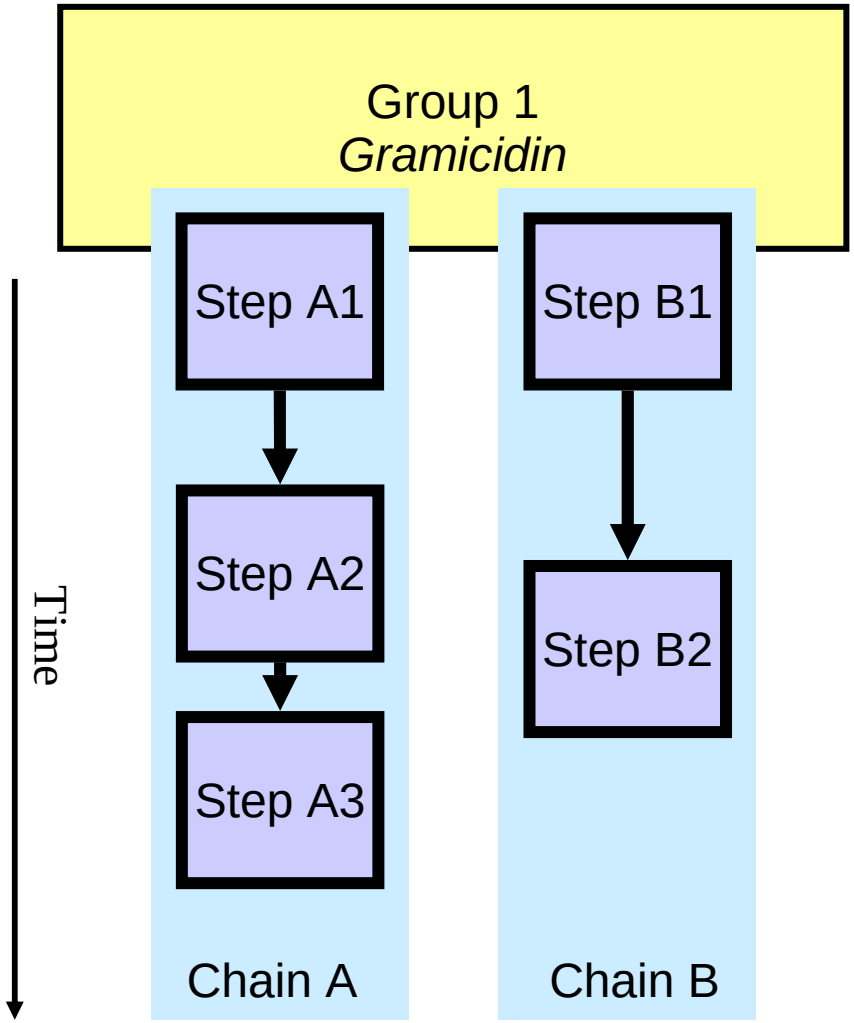
Group: an homogeneous set of simulations

- A scientist has administration rights on the *groups* he/she submitted
  - Create, then
  - Stop/delete them
  - Retrieve results
  - Be accounted for credits consumed…
  - …and errors

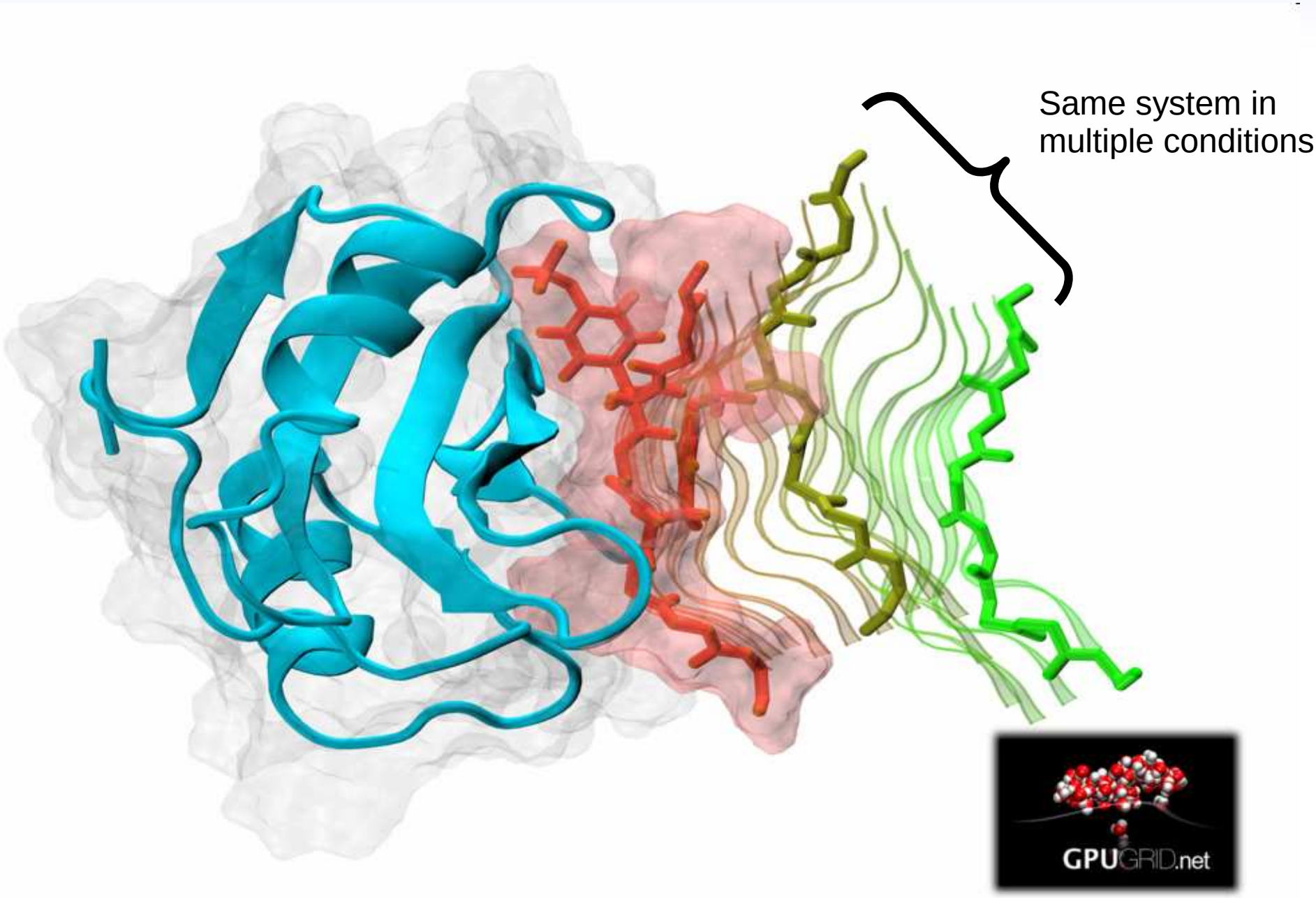


Computation chains

- Each group contains one or more *computation chains*
- Each chain consists of an ordered sequence of *steps*
- Steps are dependent: output of one is required to start the next

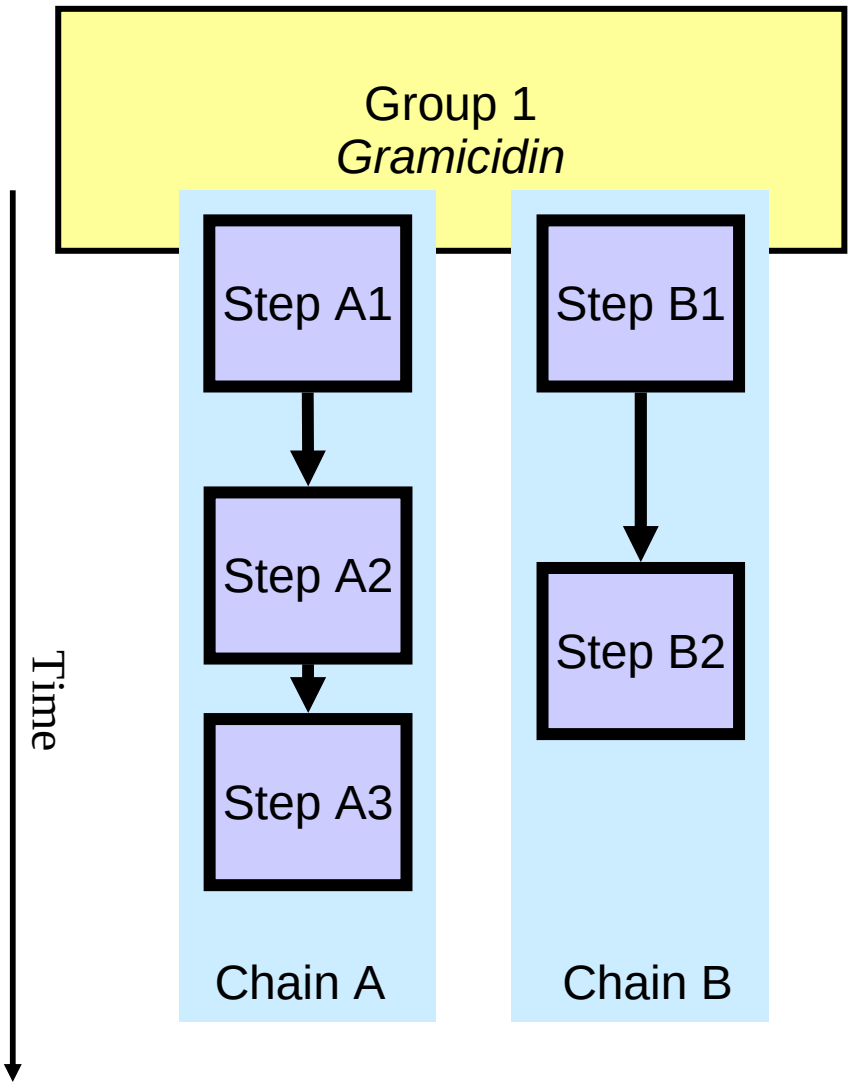


Example: one group, several chains



Computation chains

- Users can manage computations as groups or specific chains
- Group attributes:
  - Submitter, application description, pooled storage, load balancing, …
- Per-chain attributes:
  - Metadata, priority, resource limits, …



Status & outlook

Conclusions

- RBoinc provides a *generic*, *simple* and *efficient* remote interface for BOINC projects
  - Researchers become its *users*
  - Any BOINC project can be exposed as a “service” or even a “virtual supercomputer”
- Implementation stable and virtually complete
  - Currently used for **ALL** of gpugrid.net’s work
- To do
  - Test-run WUs before sending into the wild
  - Test it outside the group