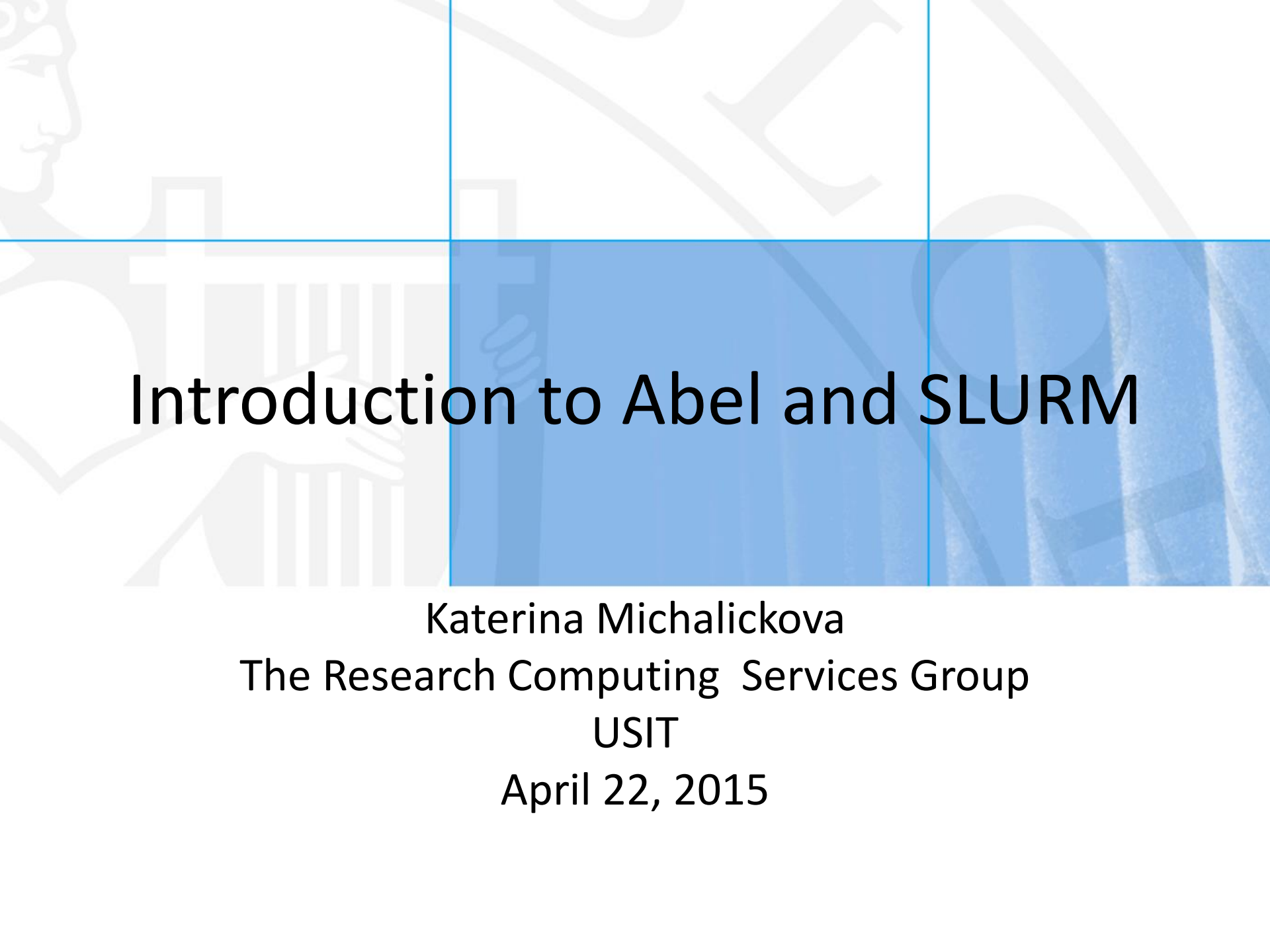




Introduction to Abel and SLURM

Katerina Michalickova

The Research Computing Services Group

USIT

April 22, 2015

Topics

- The Research Computing Services group
- Abel details
- Logging in
- Copying files
- Running a simple job
- Queuing system
- Job administration
- User administration
- Parallel jobs



The Research Computing Services

Seksjon for IT i Forskning

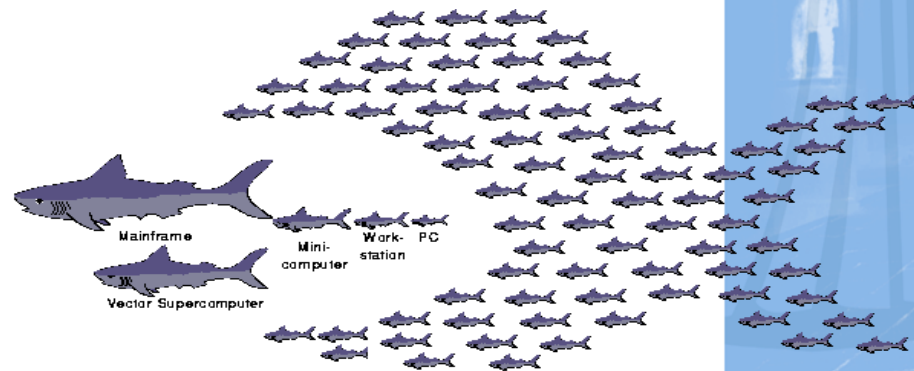
- The RCS group provides access to IT resources and high performance computing to researchers at UiO and to NOTUR users
- <http://uio.no/hpc>
- Part of USIT
- Write to us: hpc-drift@usit.uio.no

The Research Computing Services

- operation of Abel - a computer cluster
- Abel user support
- data storage
- secure data storage
- Lifeportal – portal to life sciences applications on Abel
- statistical support
- qualitative methods
- advanced user support (one on one work with scientists)
- visualization

Abel

- Computer cluster
- Enables parallel computing
- Science presents multiple problems of parallel nature
 - Sequence database searches
 - Genome assembly and annotation
 - Molecular simulations



What makes a bunch of computers into a useful computer cluster

- Hardware
 - nodes connected by high-speed network
 - all nodes have an access to a common file system
- Software
 - Operating system (Rocks flavor of Linux) enables identical mass installations
 - Queuing system enables timely execution of many concurrent processes
- Read about Abel

<http://www.uio.no/hpc/abel>

Abel in numbers



- Nodes - 650+
- Cores - 10000+
- Total memory - ~40 TB
- Total storage - ~400TB
- # 96 at top500.org (2012)

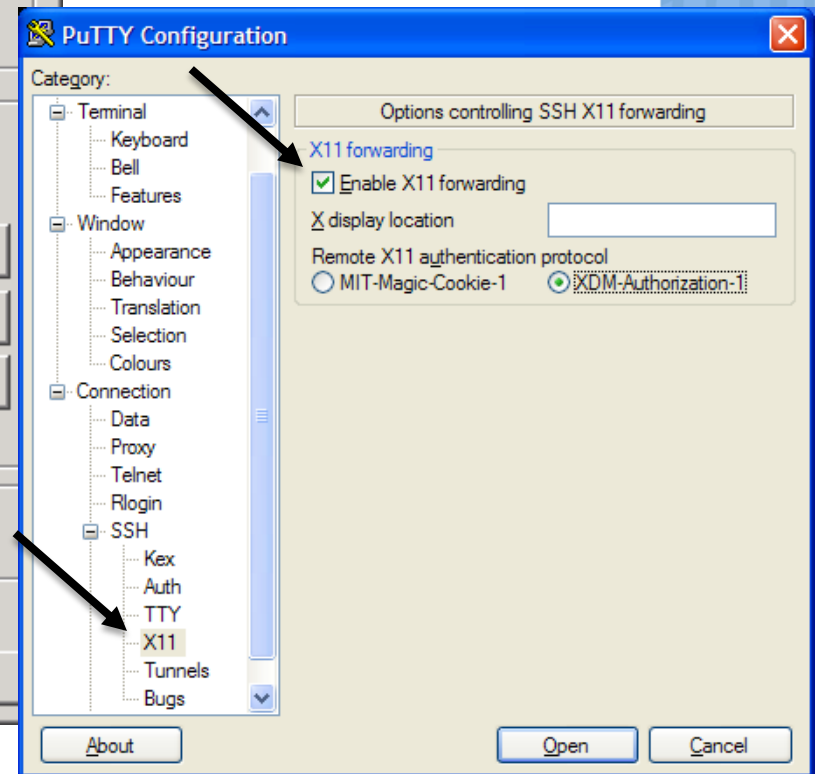
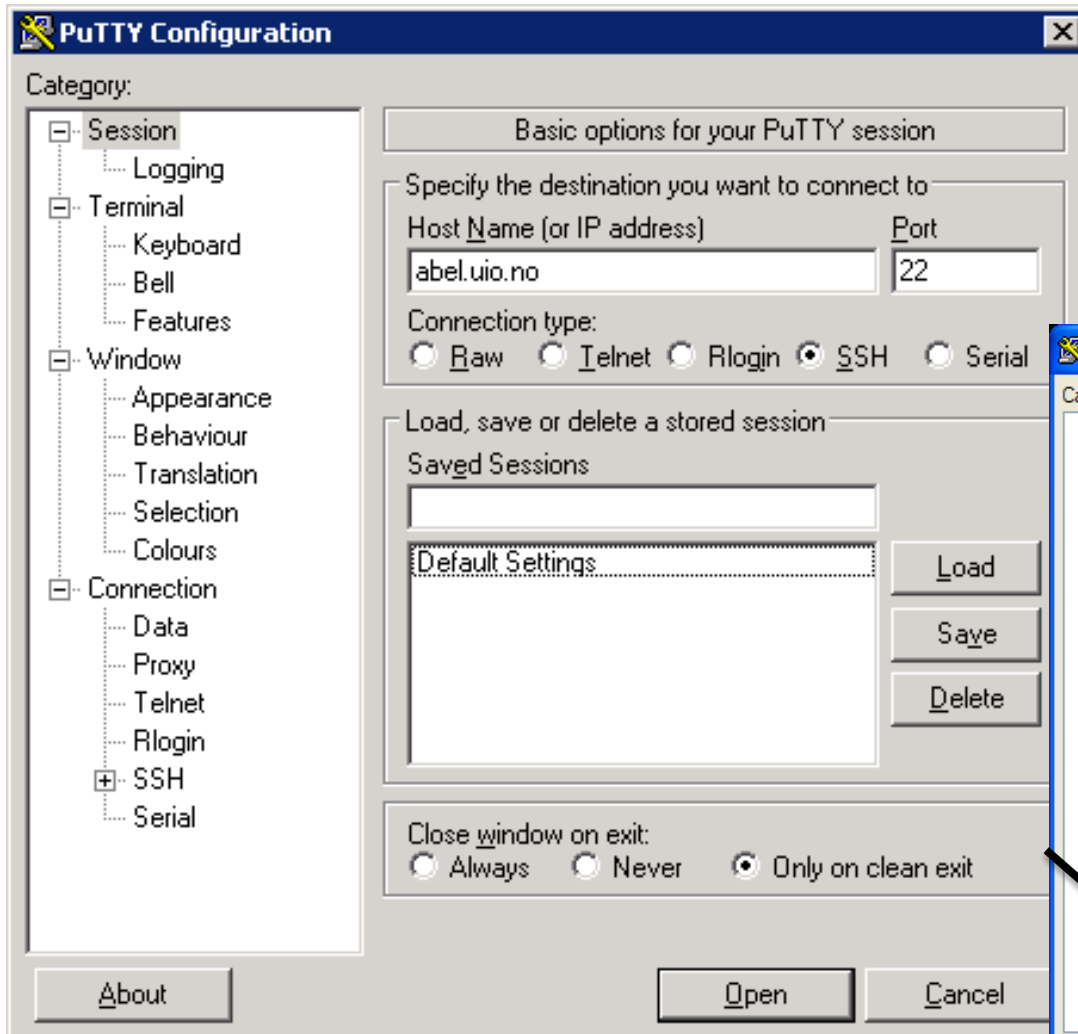
Accessing Abel

- If you are working or studying at UiO, you can have an Abel account directly from us.
- If you are Norwegian scientist (or need large resources), you can apply through NOTUR – <http://www.notur.no>
- Write to us for information: hpc-drift@usit.uio.no
- Read about getting access: <http://www.uio.no/hpc/abel/help/access>

Log into Abel

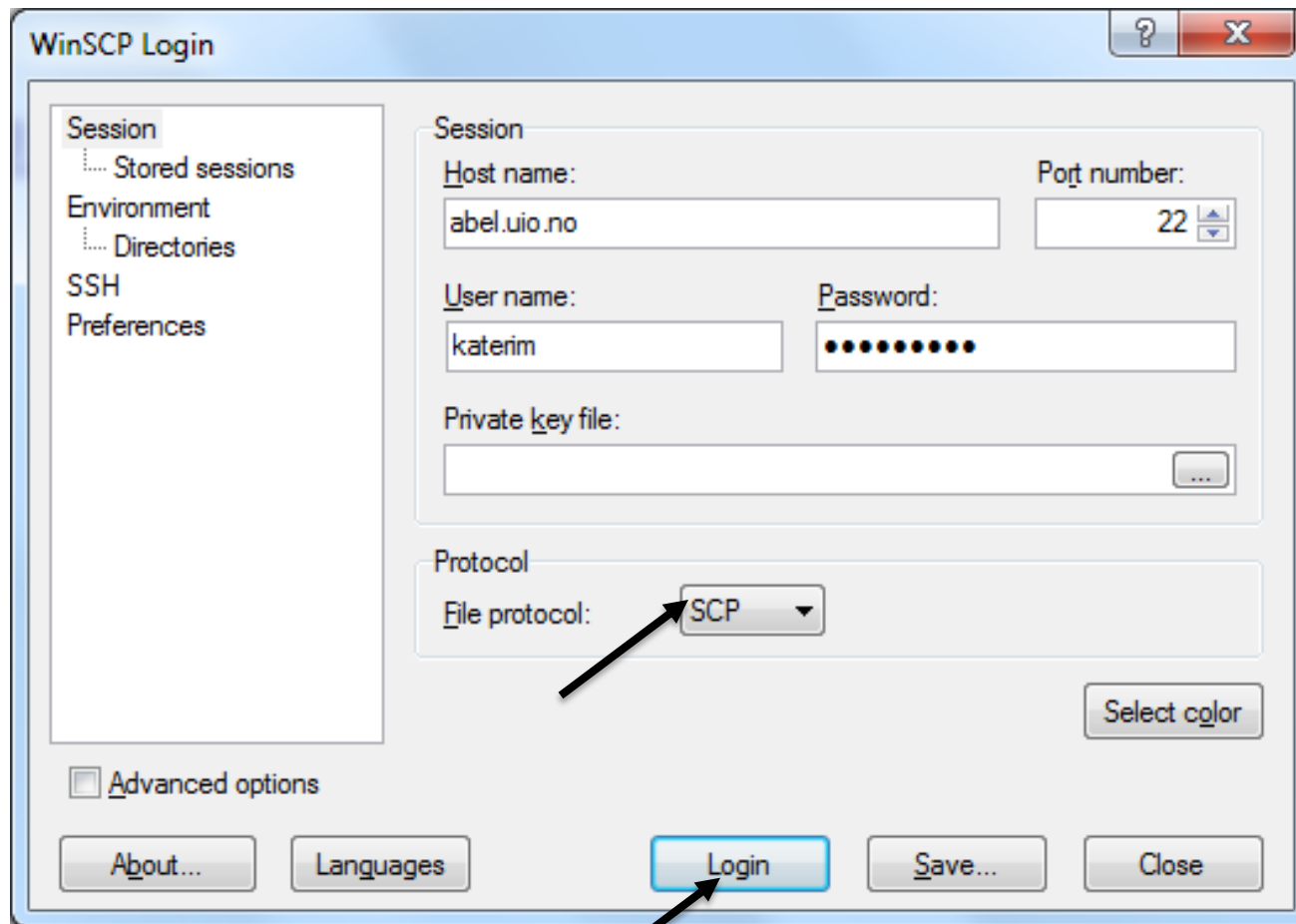
- If on Windows, download
 - Putty for connecting
 - WinSCP for copying files
- On Unix systems, open a terminal and type:
`ssh -Y username@abel.uio.no`
- Use your **UiO username and password**

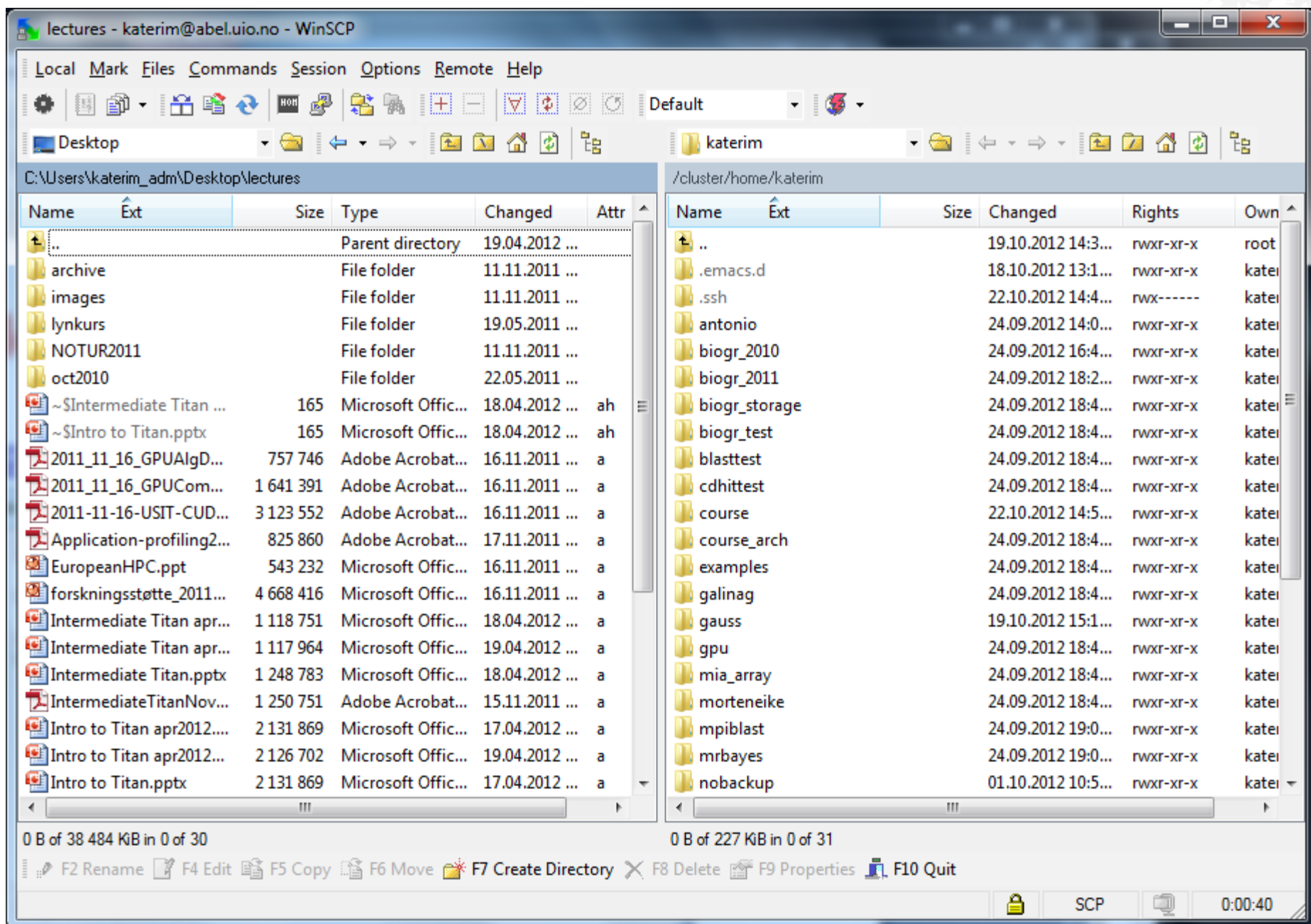
Putty <http://www.putty.org/>



File upload/download - WinSCP

- <http://winscp.net/eng/download.php>





File upload/download on command line

- Unix users can use secure copy or rsync commands
 - Copy myfile.txt from the current directory on your machine to your home area on Abel:

```
scp myfile.txt user_name@abel.uio.no:~
```

- For large files, use rsync command:

```
rsync -z myfile.tar user_name@abel.uio.no:~
```

- Available on Abel:
 - <http://www.uio.no/hpc/abel/help/software>
- Software on Abel is organized in modules.
 - List all software (and version) organized in modules:


```
module avail
```
 - Load software from a module:


```
module load module_name
```
- If you cannot find what you looking for: ask us

<http://www.uio.no/hpc/abel/help/software>

- Software on Abel is organized in modules.

- List all software (and version) organized in modules:

module avail

- Load software from a module:

```
module load module name
```

- If you cannot find what you looking for: **ask us**

ask us

[illegible]

Your own software

You can install own software in your home area

- We provide
 - Perl and Python interpreters
 - C, C++, Fortran compilers (GNU and Intel)
 - numerous libraries

Using Abel

- When you log into Abel, you are either on login0 or login1 compute node.
- It is not allowed to execute programs (jobs) directly on the login nodes.
- Jobs are submitted to Abel via the queuing system.
- The login nodes are just for logging in, copying files, editing, compiling, running short tests (no more than a couple of minutes), submitting jobs, checking job status, etc.
- If command line is needed, use **qlogin**.

Computing on Abel

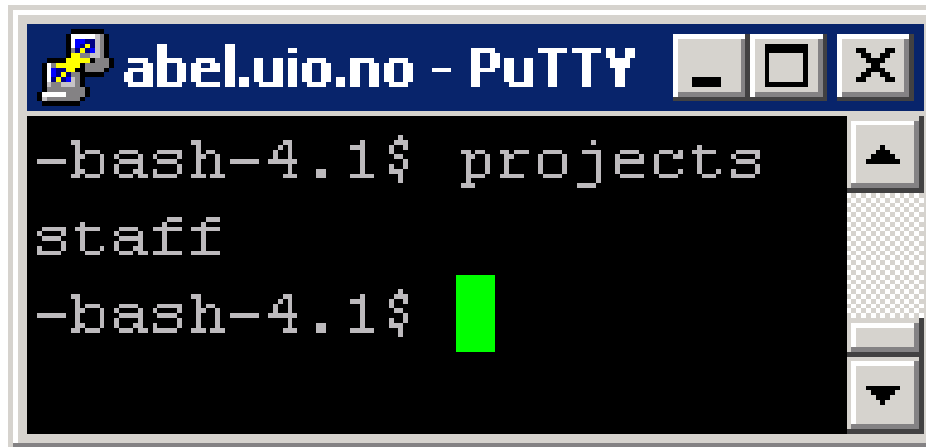
- Submit a job to the queuing system
 - Software that executes jobs on available resources on the cluster (and much more)
 - SLURM - Simple Linux Utility for Resource Management
- Communicate with the queuing system using a shell (or job) script
- Read tutorial:
<http://www.uio.no/hpc/abel/help/user-guide>

Job script

- Job script - shell script including the command that one needs to execute
- EXTRA comments read by the queuing system – “#SBATCH --xxxx”
- Compulsory values:
 - #SBATCH --account
 - #SBATCH --time
 - #SBATCH --mem-per-cpu
- Setting up a job environment
 - source /cluster/bin/jobsetup

Project/Account

- Each user belongs to a project on Abel
- Each project has set resources
- Learn about your project(s):
 - Use: **projects**



```
abel.uio.no - PuTTY
-bash-4.1$ projects
staff
-bash-4.1$
```

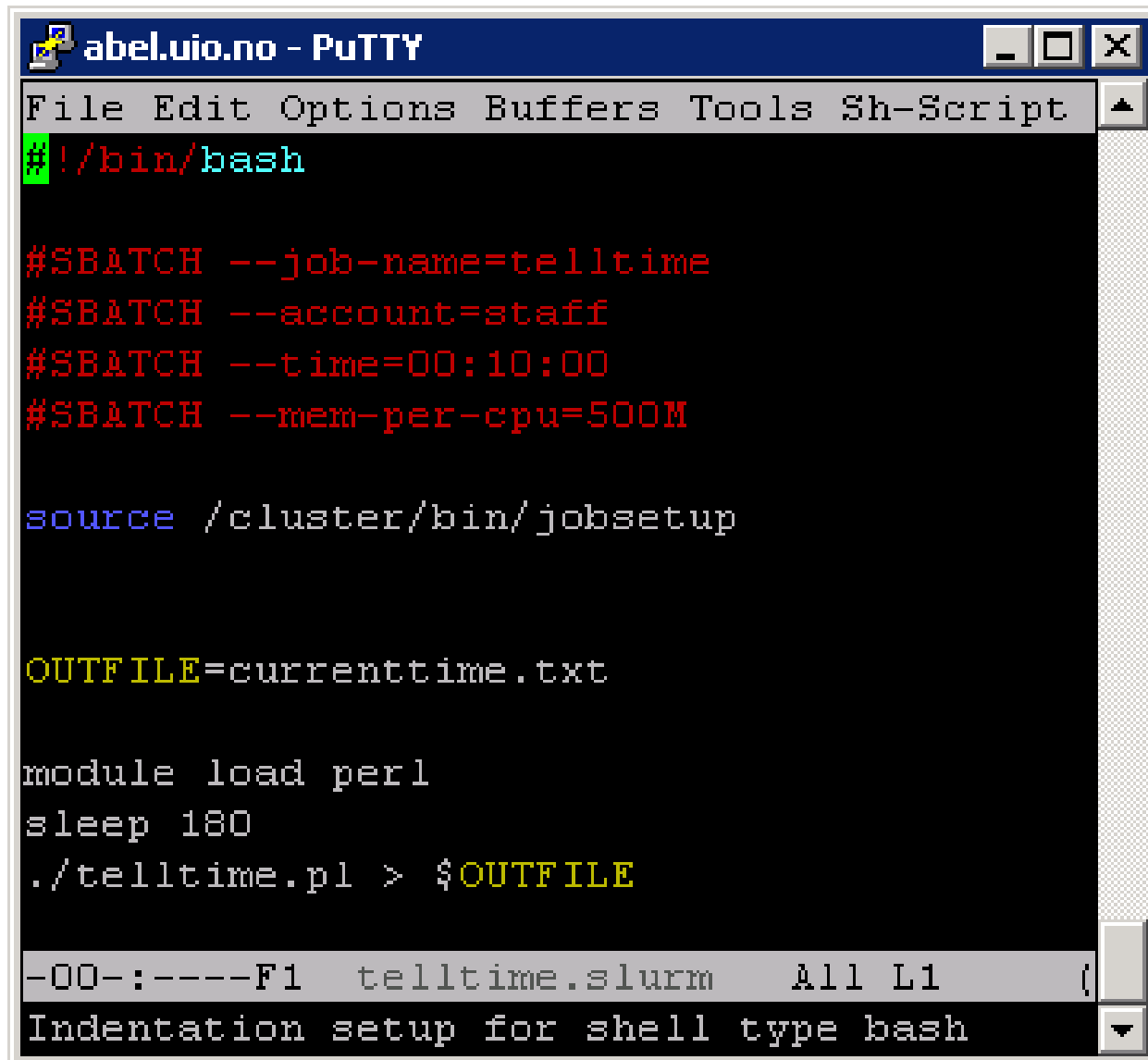
Minimal job script

```
#!/bin/bash
## Job name:
#SBATCH --job-name=jobname
## Project:
#SBATCH --account=ui0
## Wall time:
#SBATCH --time=hh:mm:ss
## Max memory
#SBATCH --mem-per-cpu=max_size_in_memory

## Set up environment
source /cluster/bin/jobsetup

## Run command
./executable > outfile
```

Example job script



The image shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The window contains a Slurm job script. The script starts with a shebang line, followed by four SBATCH directives for job name, account, time, and memory. It then sources a job setup script, sets an output file, loads the perl module, sleeps for 180 seconds, and runs a perl script. The bottom of the window shows a status bar with job information and a message about indentation setup.

```
File Edit Options Buffers Tools Sh-Script
#!/bin/bash

#SBATCH --job-name=telltime
#SBATCH --account=staff
#SBATCH --time=00:10:00
#SBATCH --mem-per-cpu=500M

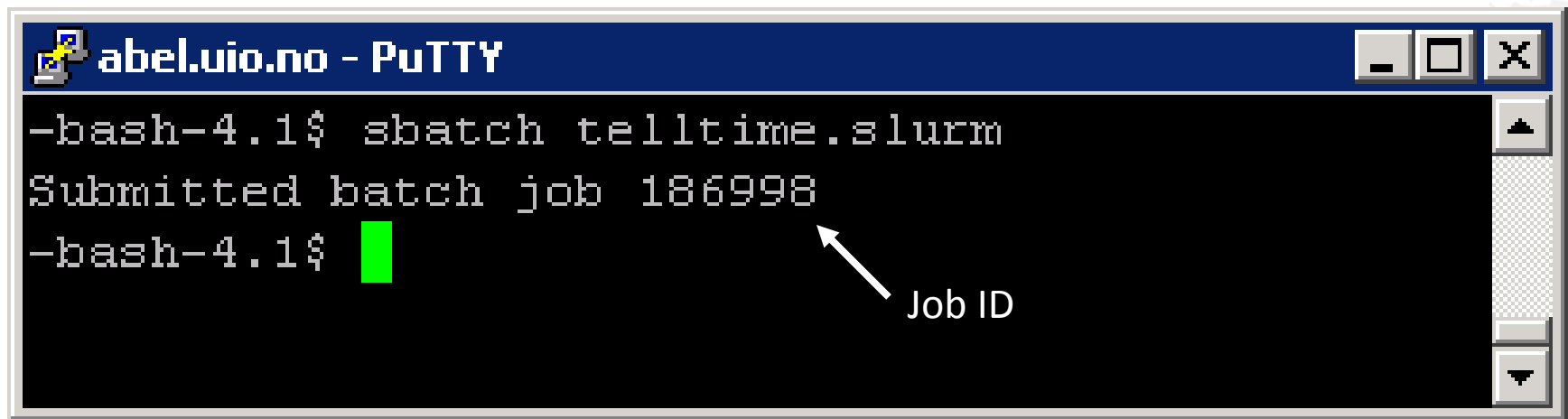
source /cluster/bin/jobsetup

OUTFILE=currenttime.txt

module load perl
sleep 180
./telltime.pl > $OUTFILE

-00-:----F1  telltime.slurm  All L1  (
Indentation setup for shell type bash
```

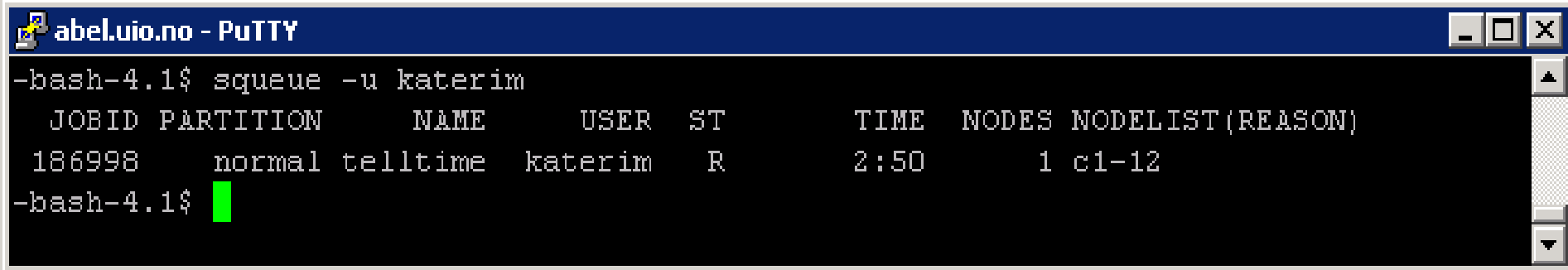
Submitting a job - sbatch

A terminal window titled 'abel.uio.no - PuTTY' with standard window controls. It shows a command prompt where the user has entered 'sbatch telltime.slurm'. The system has responded with 'Submitted batch job 186998'. A green cursor is visible on the line following the prompt. An arrow points from the text 'Job ID' to the number '186998' in the output.

```
abel.uio.no - PuTTY
-bash-4.1$ sbatch telltime.slurm
Submitted batch job 186998
-bash-4.1$ █
```

Job ID

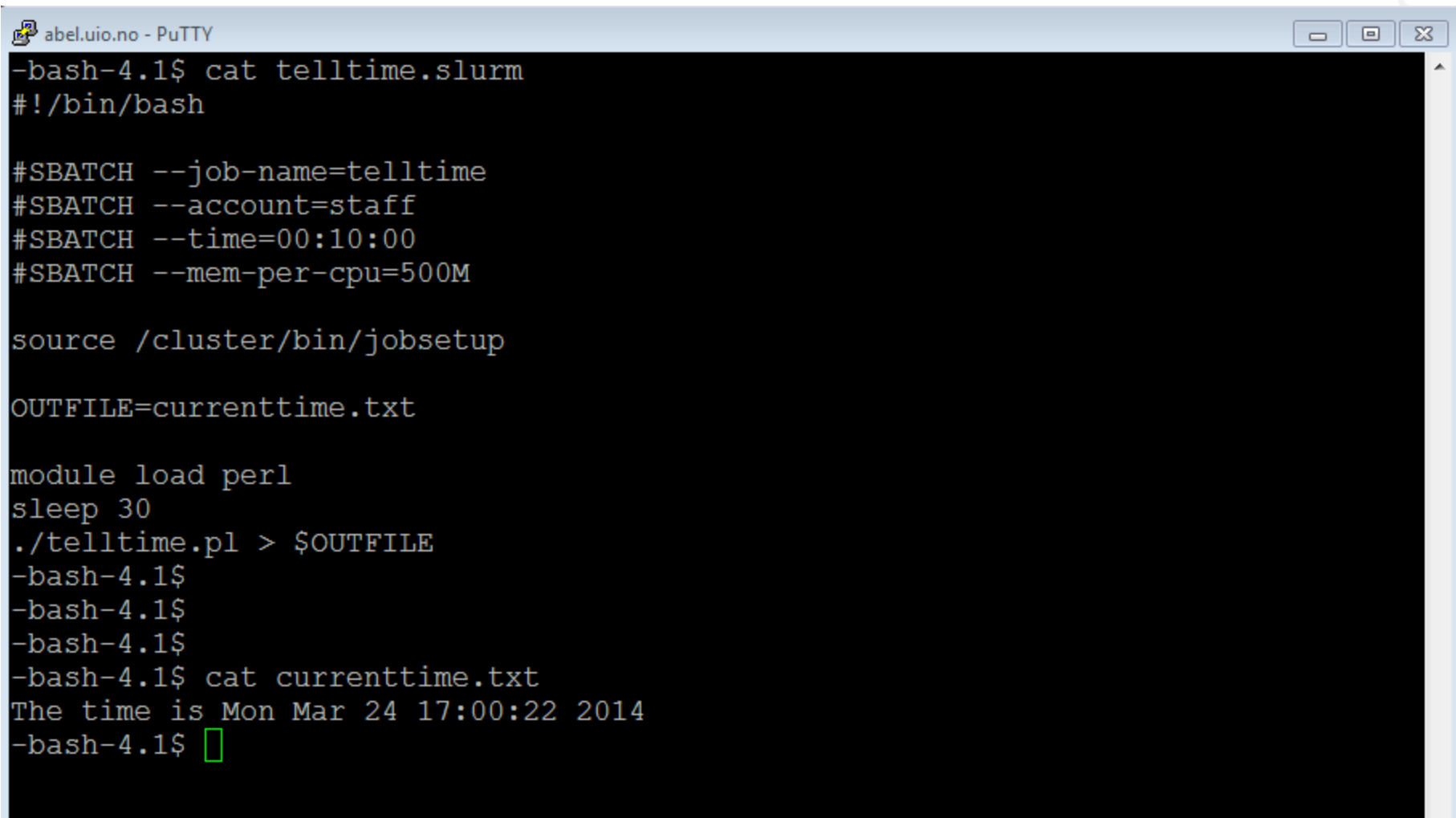
Checking a job - squeue



The screenshot shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The terminal displays the command `squeue -u katerim` and its output, which is a table of job information for user 'katerim'. The table has columns for JOBID, PARTITION, NAME, USER, ST, TIME, NODES, and NODELIST(Reason). One job is listed: JOBID 186998, PARTITION normal, NAME teltime, USER katerim, ST R, TIME 2:50, NODES 1, and NODELIST c1-12. The prompt `-bash-4.1$` is followed by a green cursor.

```
abel.uio.no - PuTTY
-bash-4.1$ squeue -u katerim
  JOBID PARTITION    NAME     USER  ST       TIME  NODES NODELIST(REASON)
 186998   normal  teltime  katerim  R       2:50      1  c1-12
-bash-4.1$
```

Checking the results of a job



The image shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The terminal displays the contents of a file named 'telltime.slurm' and the output of a script named 'telltime.pl'. The script sets up a Slurm job, sleeps for 30 seconds, and then writes the current time to a file named 'currenttime.txt'. The final output shows the time as 'Mon Mar 24 17:00:22 2014'.

```
-bash-4.1$ cat telltime.slurm
#!/bin/bash

#SBATCH --job-name=telltime
#SBATCH --account=staff
#SBATCH --time=00:10:00
#SBATCH --mem-per-cpu=500M

source /cluster/bin/jobsetup

OUTFILE=currenttime.txt

module load perl
sleep 30
./telltime.pl > $OUTFILE
-bash-4.1$
-bash-4.1$
-bash-4.1$
-bash-4.1$ cat currenttime.txt
The time is Mon Mar 24 17:00:22 2014
-bash-4.1$
```

Troubleshooting

- Every job produces a log file: **slurm-jobID.out**
- Check this file for error messages
- If you need help, list jobID or paste the slurm file into your e-mail

Use of the SCRATCH area

```
#!/bin/sh
#SBATCH --job-name=YourJobname
#SBATCH --account=YourProject
#SBATCH --time=hh:mm:ss
#SBATCH --mem-per-cpu=max_size_in_memory
source /cluster/bin/jobsetup

## Copy files to work directory:
cp $SUBMITDIR/YourData $SCRATCH

## Mark outfiles for automatic copying to $SUBMITDIR:
chkfile YourOutput

## Run command
cd $SCRATCH
executable YourData > YourOutput
```

Interactive use of Abel - qlogin

- Send request for a resource (e.g. 4 cores)
- Work on command line when a node becomes available
- Example - book one node (or 32 cores) on Abel for your interactive use for 1 hour:

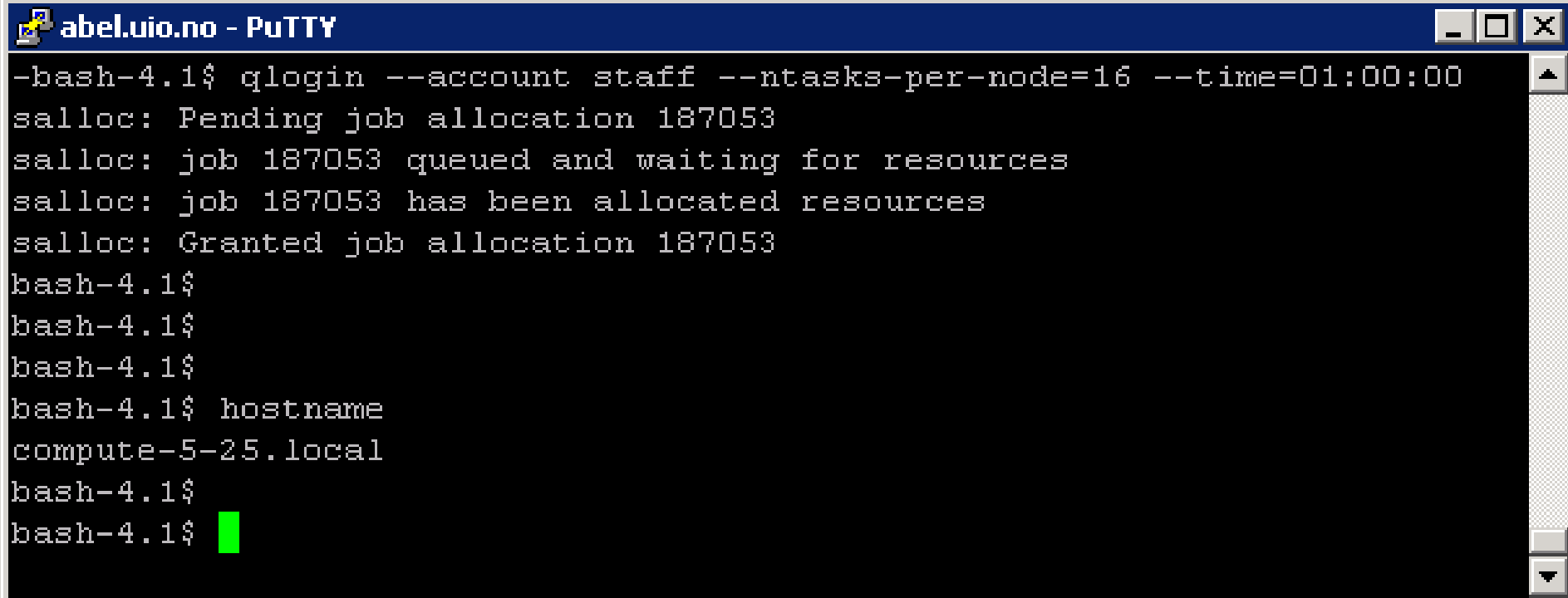
qlogin --account=your_project --ntasks-per-node=32 --time=01:00:00

- Run “**source /cluster/bin/jobsetup**” after receiving allocation

- For more info, see:

<http://www.uio.no/hpc/abel/help/user-guide/interactive-logins.html>

Interactive use of Abel - qlogin



```
abel.uio.no - PuTTY
-bash-4.1$ qlogin --account staff --ntasks-per-node=16 --time=01:00:00
salloc: Pending job allocation 187053
salloc: job 187053 queued and waiting for resources
salloc: job 187053 has been allocated resources
salloc: Granted job allocation 187053
bash-4.1$
bash-4.1$
bash-4.1$
bash-4.1$ hostname
compute-5-25.local
bash-4.1$
bash-4.1$ █
```



Queuing system

- Lets you specify resources that your program needs.
- Keeps track of which resources are available on which nodes, and starts your job when the requested resources are available.
- Abel uses the Simple Linux Utility for Resource Management - SLURM <https://computing.llnl.gov/linux/slurm/>
- A job is started by sending a job script to SLURM with the command **sbatch**. Resources are requested within the script
- For full list of options see:

[http://www.uio.no/hpc/abel/help/user-guide/job-scripts.html#Useful sbatch parametres](http://www.uio.no/hpc/abel/help/user-guide/job-scripts.html#Useful_sbatch_params)

Ask SLURM for the right resources

- Project
- Memory
- Time
- Queue
- CPUs
- Nodes
- Combination thereof
- Standard stream redirect



sbatch - project

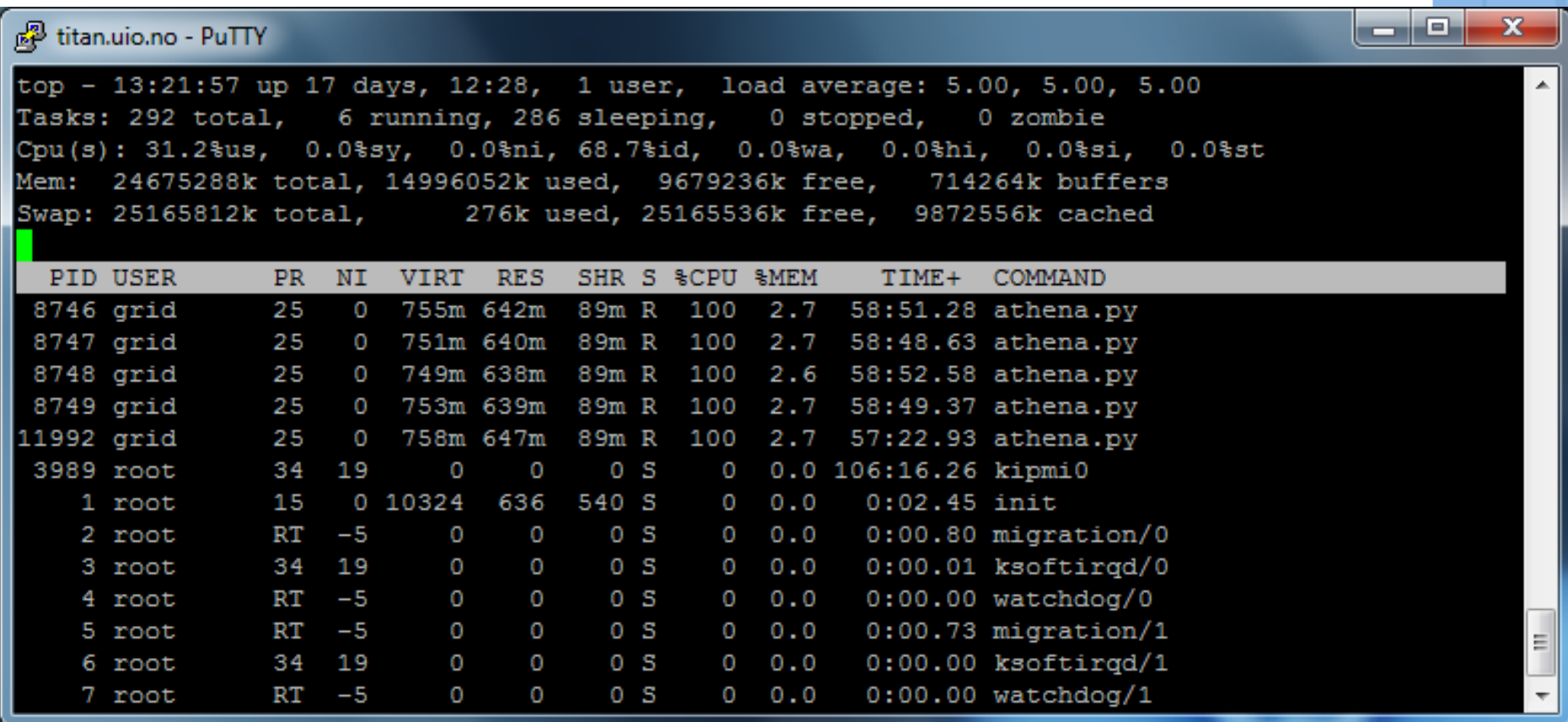
- **#SBATCH --account=Project** Specify the project to run under.
- Every Abel user is assigned a project. Use command **projects** to find out which project you belong to.
 - UiO scientists/students can use the uio project
 - It is recommended to seek additional resources if planning intensive work. Application for compute hours and data storage can be placed with the Norwegian metacenter for computational science (NOTUR)
<http://www.notur.no/>.
- **#SBATCH --job-name=jobname** Job name

sbatch - memory

- **#SBATCH --mem-per-cpu=Size** Memory required per allocated core (format: 2G or 2000M)
 - How much memory should one specify? The maximum usage of RAM by your program (plus some). Exaggerated values might, delay the job start.
- **#SBATCH --partition=hugemem**
 - If you need more than 61GB of RAM on a single node.

Top and mem-per-cpu

- maximum usage of virtual RAM by your program



```
titan.uio.no - PuTTY
top - 13:21:57 up 17 days, 12:28,  1 user,  load average: 5.00, 5.00, 5.00
Tasks: 292 total,   6 running, 286 sleeping,   0 stopped,   0 zombie
Cpu(s): 31.2%us,  0.0%sy,  0.0%ni, 68.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  24675288k total, 14996052k used,  9679236k free,   714264k buffers
Swap: 25165812k total,    276k used, 25165536k free,  9872556k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 8746 grid       25   0 755m 642m  89m  R   100   2.7   58:51.28 athena.py
 8747 grid       25   0 751m 640m  89m  R   100   2.7   58:48.63 athena.py
 8748 grid       25   0 749m 638m  89m  R   100   2.6   58:52.58 athena.py
 8749 grid       25   0 753m 639m  89m  R   100   2.7   58:49.37 athena.py
11992 grid       25   0 758m 647m  89m  R   100   2.7   57:22.93 athena.py
 3989 root        34  19    0    0    0  S    0   0.0  106:16.26 kipmi0
     1 root        15   0 10324  636  540  S    0   0.0    0:02.45 init
     2 root         RT  -5     0    0    0  S    0   0.0    0:00.80 migration/0
     3 root        34  19     0    0    0  S    0   0.0    0:00.01 ksoftirqd/0
     4 root         RT  -5     0    0    0  S    0   0.0    0:00.00 watchdog/0
     5 root         RT  -5     0    0    0  S    0   0.0    0:00.73 migration/1
     6 root        34  19     0    0    0  S    0   0.0    0:00.00 ksoftirqd/1
     7 root         RT  -5     0    0    0  S    0   0.0    0:00.00 watchdog/1
```

sbatch - time

- **#SBATCH --time=hh:mm:ss** Wall clock time limit on the job
- Some prior testing is necessary. One might, for example, test on smaller data sets and extrapolate. As with the memory, unnecessarily large values might delay the job start.
- **#SBATCH --begin=hh:mm:ss** Start the job at a given time (or later)
- Maximum time for a job is 1 week (168 hours). If more needed, use **--partition=long**

sbatch – CPUs and nodes

- Does your program support more than one CPU?
 - If so, do they have to be on a single node?
 - How many CPUs will the program run efficiently on?
-
- **#SBATCH --nodes=Nodes** Number of nodes to allocate
 - **#SBATCH --ntasks-per-node=Cores** Number of cores to allocate within each allocated node
 - **#SBATCH --ntasks=Cores** Number of cores to allocate

sbatch – CPUs and nodes

- If you just need some cpus, no matter where:

#SBATCH --ntasks=17

- If you need a specific number of cpus on each node

#SBATCH --nodes=8 --ntasks-per-node=4

- If you need the cpu's on a single node

#SBATCH --nodes=1 --ntasks-per-node=8

SBATCH - interconnect

- **#SBATCH --constraint=ib** Run job on nodes with infiniband
 - Gigabit Ethernet on all nodes
 - All nodes on Abel are equipped with InfiniBand (56 Gbits/s)
 - Select if you run MPI jobs

sbatch - constraints

- `#SBATCH --constraint=feature` Run job on nodes with a certain feature - *ib*, *rackN*.
- `#SBATCH --constraint=ib&rack21` If you need more than one constraint

in case of multiple specifications, the later overrides the earlier

sbatch - files

- **#SBATCH --output=file** Send 'stdout' (and stderr) to the specified file (instead of slurm-xxx.out)
- **#SBATCH --error=file** Send 'stderr' to the specified file
- **#SBATCH --input=file** Read 'stdin' from the specified file

sbatch – low priority

- **#SBATCH --qos=lowpri** Run a job in the lowpri queue
 - Even if all of your project's cpus are busy, you may utilize other cpus
 - Such a job may be terminated and put back into the queue at any time.
 - If possible, your job should ensure its state is saved regularly, and should be prepared to pick up on where it left off.

Inside the job script

- All jobs must start with the bash-command:
`source /cluster/bin/jobsetup`
- A job-specific scratch-directory is created for you on /work partition. The path is in the environment variable \$SCRATCH.
- We recommend using this directory especially if your job is IO intensive. You can copy results back to your home-directory when the job exits using `chkfile` in your script.

Environment variables

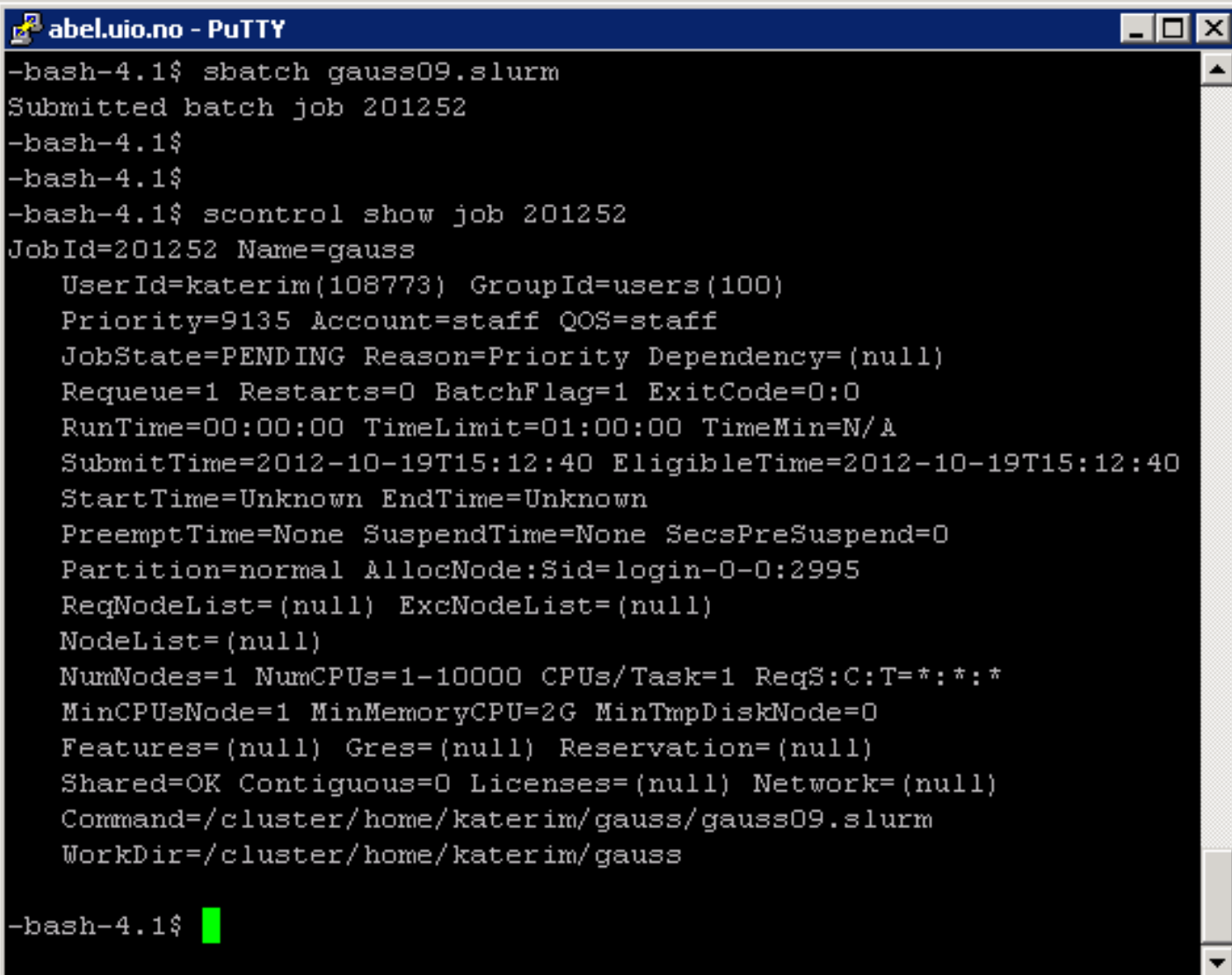
- SLURM_JOBID – job-id of the job
- SCRATCH – name of job-specific scratch-area
- SLURM_NPROCS – total number of cpus requested
- SLURM_CPUS_ON_NODE – number of cpus allocated on node
- SUBMITDIR – directory where sbatch were issued
- TASK_ID – task number (for arrayrun-jobs)



Job and user administration

- `scancel jobid` Cancel a job
- `scancel --user=me` Cancel all your jobs

Job details - scontrol show job



```
abel.uio.no - PuTTY
-bash-4.1$ sbatch gauss09.slurm
Submitted batch job 201252
-bash-4.1$
-bash-4.1$
-bash-4.1$ scontrol show job 201252
JobId=201252 Name=gauss
  UserId=katerim(108773) GroupId=users(100)
  Priority=9135 Account=staff QOS=staff
  JobState=PENDING Reason=Priority Dependency=(null)
  Requeue=1 Restarts=0 BatchFlag=1 ExitCode=0:0
  RunTime=00:00:00 TimeLimit=01:00:00 TimeMin=N/A
  SubmitTime=2012-10-19T15:12:40 EligibleTime=2012-10-19T15:12:40
  StartTime=Unknown EndTime=Unknown
  PreemptTime=None SuspendTime=None SecsPreSuspend=0
  Partition=normal AllocNode:Sid=login-0-0:2995
  ReqNodeList=(null) ExcNodeList=(null)
  NodeList=(null)
  NumNodes=1 NumCPUs=1-10000 CPUs/Task=1 ReqS:C:T=*:*:~
  MinCPUsNode=1 MinMemoryCPU=2G MinTmpDiskNode=0
  Features=(null) Gres=(null) Reservation=(null)
  Shared=OK Contiguous=0 Licenses=(null) Network=(null)
  Command=/cluster/home/katerim/gauss/gauss09.slurm
  WorkDir=/cluster/home/katerim/gauss

-bash-4.1$
```

```
abel.uio.no - PuTTY
-bash-4.1$ squeue -t R| more
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	ODELIST (REASON)
170901	long	pt_kosto	globus	R	3-03:15:24	1	c15-2
170892	long	pt_kosto	globus	R	3-03:19:24	1	c15-32
170879	long	pt_kosto	globus	R	3-03:23:23	1	c16-33
169917	long	pt_cheun	globus	R	3-06:23:19	1	c16-33
167724	long	pt_rbelm	globus	R	3-19:49:24	1	c15-4
164099	long	pt_moham	globus	R	4-01:35:24	1	c16-29
163791	long	pt_marta	globus	R	4-02:21:24	1	c15-8
163756	long	pt_marta	globus	R	4-02:25:09	1	c15-1
163742	long	pt_marta	globus	R	4-02:27:13	1	c16-29
159984	long	159982.2	globus	R	4-07:49:20	1	c16-30
159983	long	159982.1	globus	R	4-07:49:22	1	c16-30
159982	long	pt_jesus	globus	R	4-07:49:24	1	c15-33
159444	long	pt_zrudi	globus	R	4-15:27:24	1	c15-5
157822	long	pt_moham	globus	R	4-19:51:24	1	c16-31
157800	long	pt_moham	globus	R	4-19:53:24	1	c15-29
157137	long	pt_fabel	globus	R	5-02:03:24	1	c16-29
139520	long	pt_ipvs2	globus	R	6-11:48:54	1	c16-33
130367	long	130364.1	globus	R	7-05:52:59	1	c16-36
130368	long	130364.2	globus	R	7-05:52:59	1	c16-36
130364	long	pt_jesus	globus	R	7-05:53:23	1	c15-33
120674	long	120669.2	globus	R	7-21:49:19	1	c15-8
120673	long	120669.1	globus	R	7-21:49:21	1	c16-33
93223	long	pt_astri	globus	R	6-06:35:40	1	c16-33
95159	long	pt_tschw	globus	R	6-06:35:40	1	c15-6
92546	long	pt_rjsor	globus	R	6-07:46:40	1	c15-8
92547	long	pt_rjsor	globus	R	6-07:46:40	1	c15-8
97783	long	pt_zrudi	globus	R	9-15:59:24	1	c15-8
97751	long	pt_lutja	globus	R	9-16:13:23	1	c15-8
96180	long	pt_mheik	globus	R	9-19:09:18	1	c15-5
96004	long	pt_mheik	globus	R	9-19:20:50	1	c15-5
95232	long	pt_tschw	globus	R	9-21:55:23	1	c15-5
95230	long	pt_tschw	globus	R	9-21:55:24	1	c15-5
93422	long	pt_mkoll	globus	R	10-02:25:24	1	c15-5
102039	long	pt_gunnh	globus	R	9-00:15:23	1	c15-32
24320	long	pt_corne	globus	R	15-05:21:18	1	c15-32
24321	long	pt_corne	globus	R	15-05:21:18	1	c15-33

See the queue - squeue

- [-j *jobids*] show only the specified jobs
- [-w *nodes*] show only jobs on the specified nodes
- [-A *projects*] show only jobs belonging to the specified projects
- [-t *states*] show only jobs in the specified states (pending, running, suspended, etc.)
- [-u *users*] show only jobs belonging to the specified users All specifications can be comma separated lists

Examples:

- `squeue -j 4132,4133` shows jobs 4132 and 4133
- `squeue -w compute-23-11` shows jobs running on compute-23-11
- `squeue -u foo -t PD` shows pending jobs belonging to user 'foo'
- `squeue -A bar` shows all jobs in the project 'bar'

-bash-4.1\$ qsumm --nonzero

Account	Limit	nRun	nPend	lRun	lPend
akvaforsk	48	50	16	.	.
atlas-low	2500	.	.	744	.
cicero	256	208	48	.	.
geofag	206	64	.	.	.
ife	96	80	.	144	.
iha	48	7	16	21	.
metagen	96	10	.	.	.
Sum notur	9075	6265	3601	.	.
Sum grid	1196	57	.	.	.
Sum atlas	1195	57	.	.	.
atlas-ana	1195	12	.	.	.
nn2952k	1195	45	.	.	.
nn2849k	9075	128	3072	.	.
nn2875k	9075	64	.	.	.
nn2936k	9075	12	.	.	.
nn4604k	9075	192	.	.	.
nn4605k	9075	4	.	.	.
nn4653k	9075	12	.	.	.
nn4654k	9075	296	1	.	.
nn4683k	9075	144	256	.	.
nn4708k	9075	42	.	.	.
nn9110k	9075	768	256	.	.
nn9139k	9075	1	.	.	.
nn9180k	9075	640	.	.	.
nn9184k	9075	393	.	.	.
nn9188k	9075	128	.	.	.
nn9201k	9075	20	.	.	.
nn9203k	9075	100	.	.	.
nn9208k	9075	2	.	.	.
nn9244k	9075	349	16	.	.
nn9249k	9075	1	.	.	.
nn9259k	9075	32	.	.	.
nn9264k	9075	352	.	.	.
nn9267k	9075	96	.	.	.
nn9283k	9075	376	.	.	.
nn9302k	9075	8	.	.	.
nn9303k	9075	2048	.	.	.
nrr	128	104	8	.	.
quiet	260	256	.	.	.
staff	10000	40	.	.	.
uio	1000	106	19	.	.

Total sum	11325	7190	3708	909	0

-bash-4.1\$

See the projects - qsumm

- `--nonzero` only show accounts with at least one running or pending job
- `--pe` show processor equivalents (PEs) instead of CPUs
- `--memory` show memory usage instead of CPUs
- `--group` do not show the individual Notur and Grid accounts
- `--user=username` only count jobs belonging to username
- `--help` show all options

project and cost

```
abel.uio.no - PuTTY
-bash-4.1$ projects
staff
-bash-4.1$
-bash-4.1$ cost -p staff
Report for account staff on login-0-0.local
Allocation period 2012-2 (starts 2012-10-01)
Last updated on Fri Oct 19 01:24:40 2012
=====
Account                                CPU hours
=====
No usage is registered for project staff.
=====
-bash-4.1$
-bash-4.1$
-bash-4.1$ cost -p nn9259k
Report for account nn9259k on login-0-0.local
Allocation period 2012-2 (starts 2012-10-01)
Last updated on Fri Oct 19 01:24:40 2012
=====
Account                                CPU hours
=====
nn9259k      usage                      0
nn9259k      reserved                   NA
nn9259k      quota (pri)                50000
nn9259k      quota (unpri)              NA
=====
-bash-4.1$ █
```



Strength of cluster computing

- Large problems (or parts of) can be divided into smaller tasks and executed in parallel
- Types of parallel applications:
 - Divide input data and execute your program on all subsets (arrayrun)
 - Execute your program (or parts of) in parallel (MPI or OpenMP programming)

Arrayrun and TASK_ID variable

- TASK_ID is an environment variable, it can be accessed by all scripts during the execution of arrayrun
 - 1st run – TASK_ID = 1
 - 2nd run – TASK_ID = 2
 - Nth run – TASK_ID = N
- TASK_ID can be used to name input and output files
- Accessing the value of the TASK_ID variable:
 - In shell script : \$TASK_ID
 - In perl script: ENV{TASK_ID}

Arrayrun – worker script

```
#!/bin/sh
#SBATCH --account=YourProject
#SBATCH --time=hh:mm:ss
#SBATCH --mem-per-cpu=max_size_in_memory
#SBATCH --partition=lowpri
source /cluster/bin/jobsetup

DATASET=dataset.$TASK_ID
OUTFILE=result.$TASK_ID

cp $SUBMITDIR/$DATASET $SCRATCH
chkfile $OUTFILE

cd $SCRATCH
executable $DATASET > $OUTFILE
```

Arrayrun – submit script

```
#!/bin/sh
#SBATCH --account=YourProject
#SBATCH --time=hh:mm:ss (longer than worker script)
#SBATCH --mem-per-cpu=max_size_in_memory (low)
source /cluster/bin/jobsetup
```

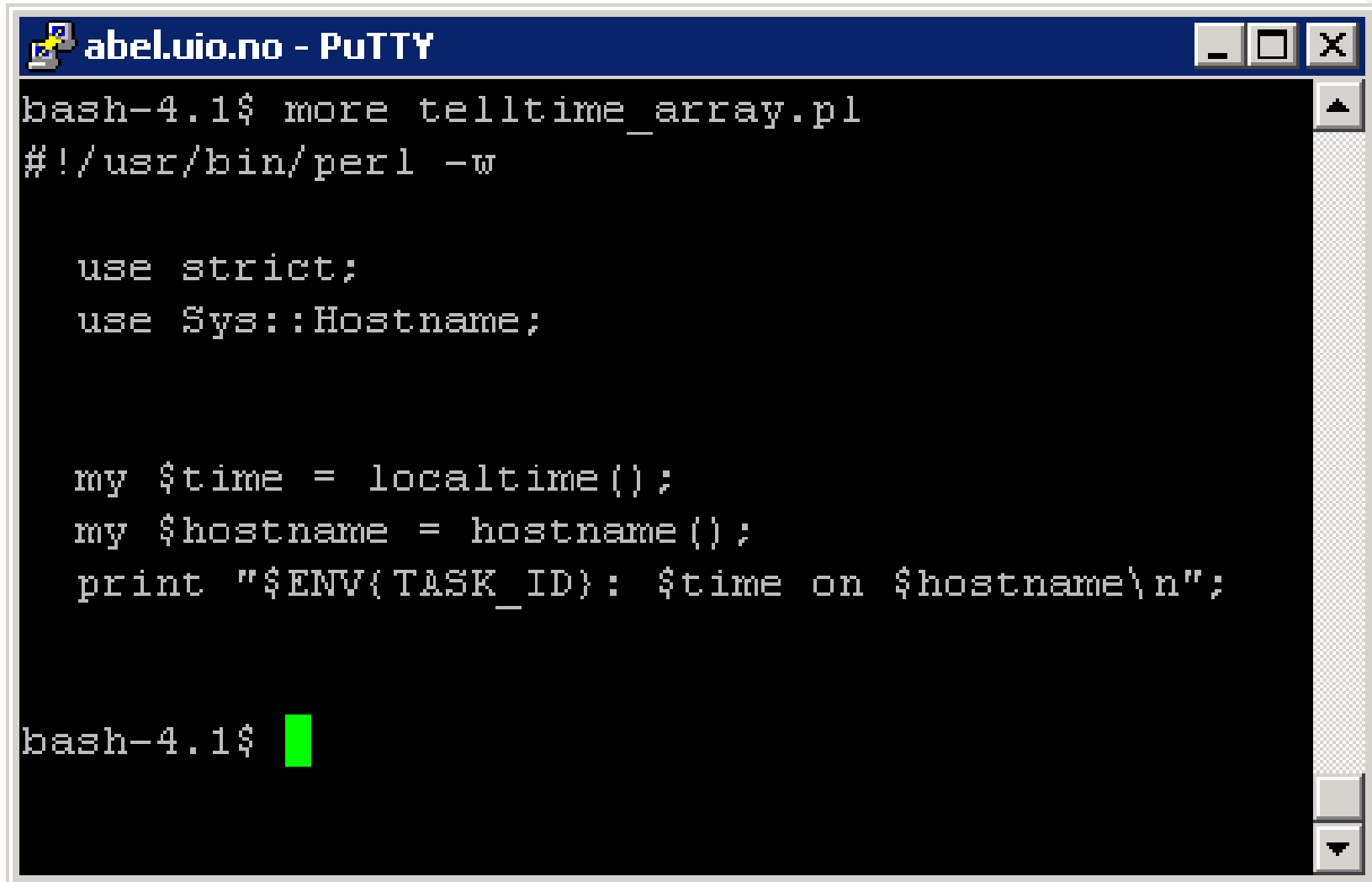
arrayrun *1-200* workerScript

1,4,42	1, 4, 42
1-5	1, 2, 3, 4, 5
0-10:2	0, 2, 4, 6, 8, 10
32,56,100-200	32, 56, 100, 101, 102, ..., 200

!no spaces, decimals, negative numbers

Example 1 - executable

Print out TASK_ID variable



The image shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The terminal displays the execution of a Perl script named 'teltime_array.pl'. The script uses the 'strict' module and 'Sys::Hostname' to retrieve the system's hostname. It then captures the local time and prints the 'TASK_ID' environment variable along with the current time and hostname. The terminal prompt 'bash-4.1\$' is visible at the bottom, followed by a green cursor.

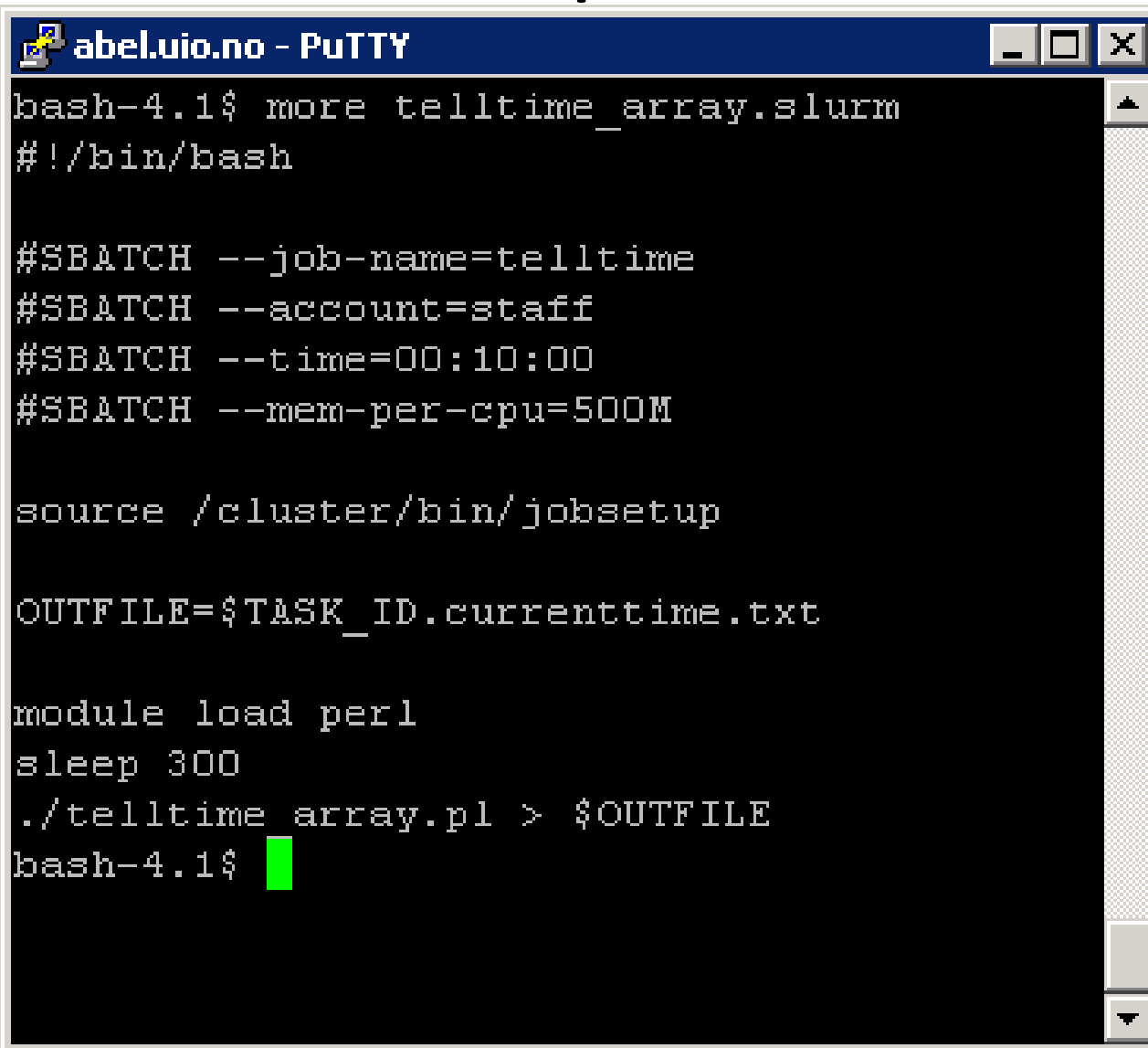
```
bash-4.1$ more teltime_array.pl
#!/usr/bin/perl -w

use strict;
use Sys::Hostname;

my $time = localtime();
my $hostname = hostname();
print "$ENV{TASK_ID}: $time on $hostname\n";

bash-4.1$
```

Example1 -worker script



```
bash-4.1$ more telltime_array.slurm
#!/bin/bash

#SBATCH --job-name=telltime
#SBATCH --account=staff
#SBATCH --time=00:10:00
#SBATCH --mem-per-cpu=500M

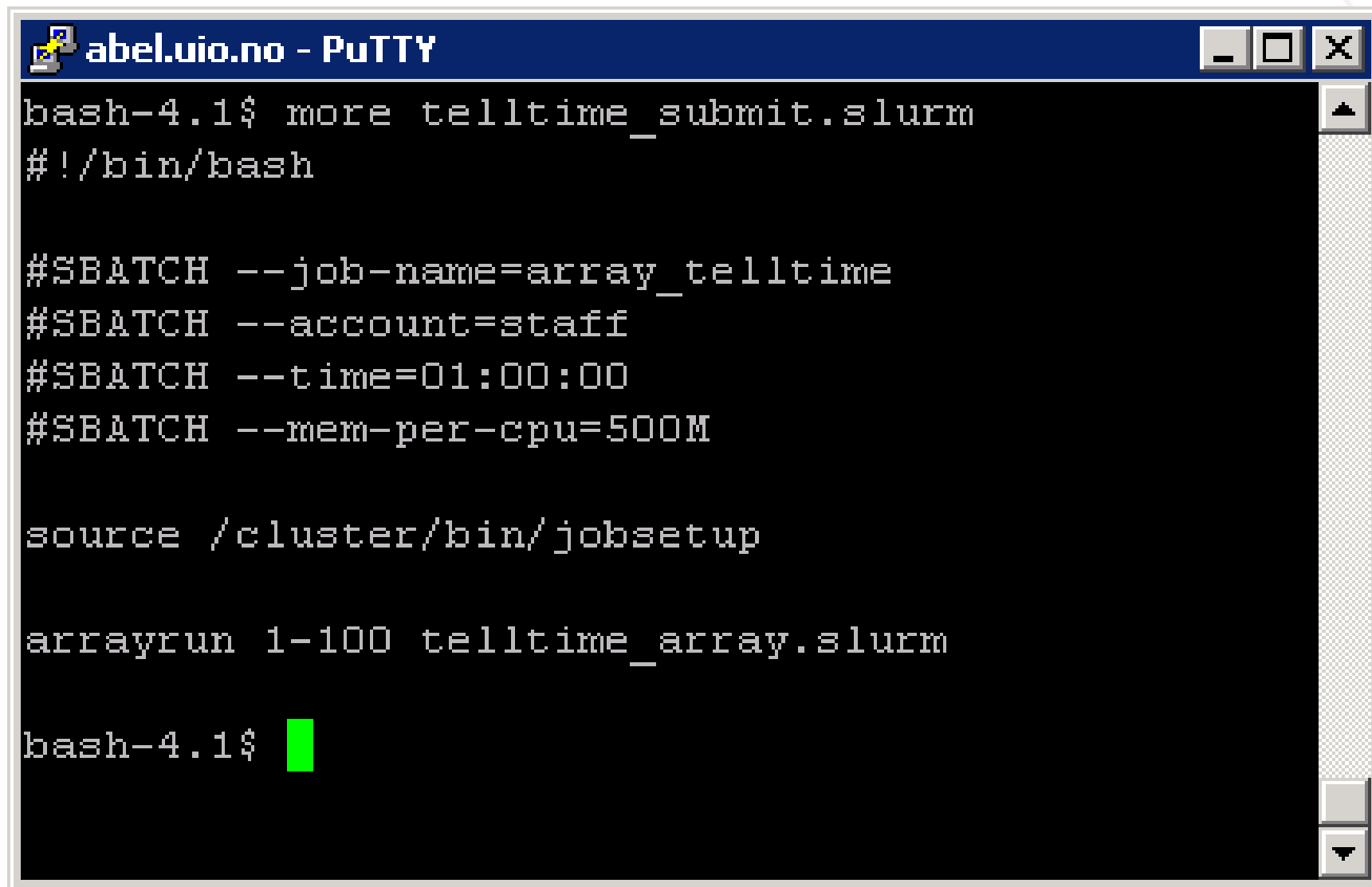
source /cluster/bin/jobsetup

OUTFILE=$TASK_ID.currenttime.txt

module load perl
sleep 300
./telltime_array.pl > $OUTFILE
bash-4.1$
```

Added TASK_ID
to the name
of the output

Example1 - submit script



The image shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The terminal displays the command 'more telltime_submit.slurm' and the contents of the script. The script includes Slurm directives for job name, account, time, and memory, followed by a source command and an arrayrun command. The prompt 'bash-4.1\$' is followed by a red cursor.

```
abel.uio.no - PuTTY
bash-4.1$ more telltime_submit.slurm
#!/bin/bash

#SBATCH --job-name=array_telltime
#SBATCH --account=staff
#SBATCH --time=01:00:00
#SBATCH --mem-per-cpu=500M

source /cluster/bin/jobsetup

arrayrun 1-100 telltime_array.slurm

bash-4.1$
```

Submitting and checking an arrayrun job

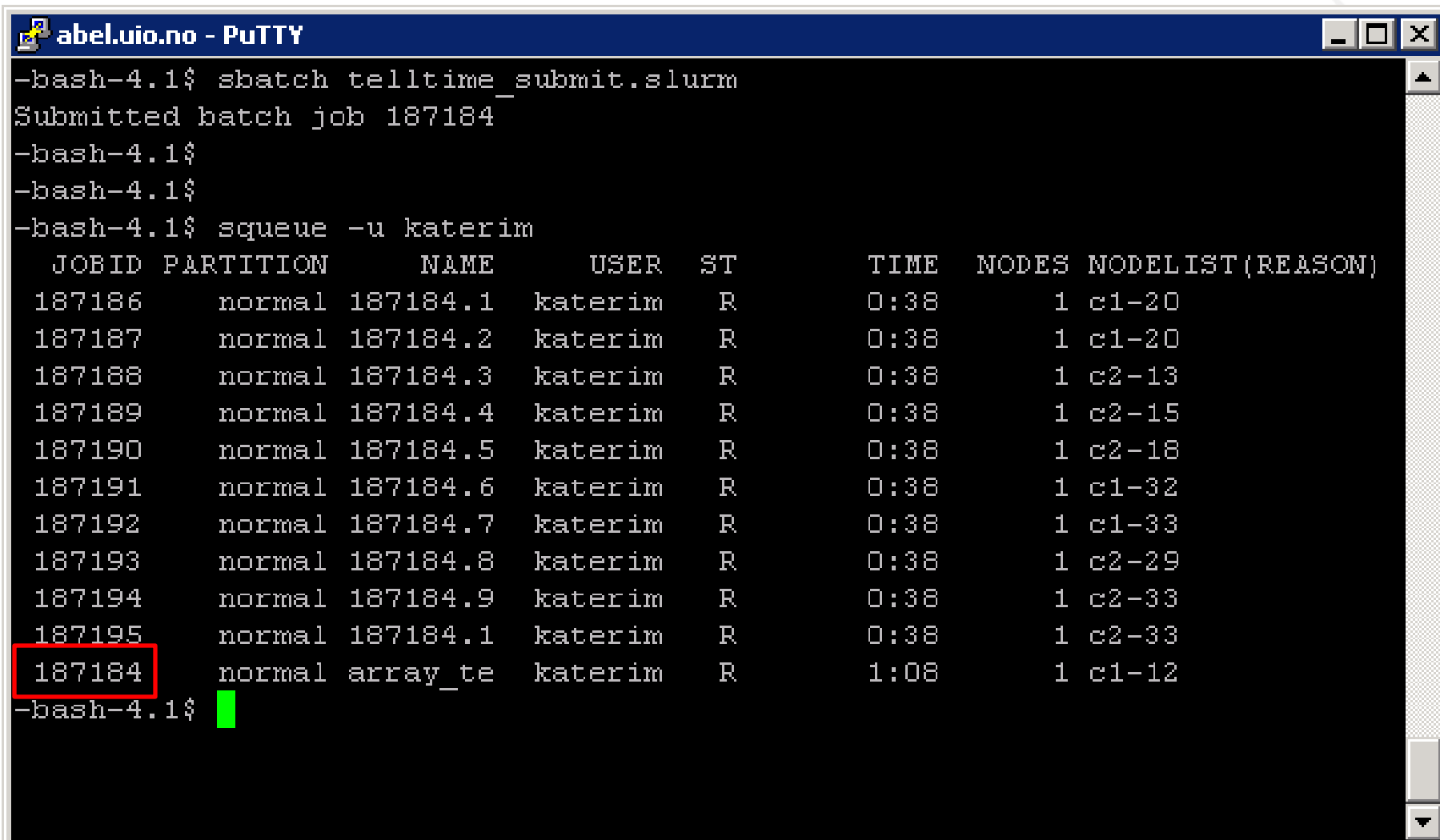
```
abel.uio.no - PuTTY
-bash-4.1$ sbatch telltime_submit.slurm
Submitted batch job 187184
-bash-4.1$
-bash-4.1$
-bash-4.1$ squeue -u katerim
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(Reason)
187186	normal	187184.1	katerim	R	0:38	1	c1-20
187187	normal	187184.2	katerim	R	0:38	1	c1-20
187188	normal	187184.3	katerim	R	0:38	1	c2-13
187189	normal	187184.4	katerim	R	0:38	1	c2-15
187190	normal	187184.5	katerim	R	0:38	1	c2-18
187191	normal	187184.6	katerim	R	0:38	1	c1-32
187192	normal	187184.7	katerim	R	0:38	1	c1-33
187193	normal	187184.8	katerim	R	0:38	1	c2-29
187194	normal	187184.9	katerim	R	0:38	1	c2-33
187195	normal	187184.1	katerim	R	0:38	1	c2-33
187184	normal	array_te	katerim	R	1:08	1	c1-12

```
-bash-4.1$
```

Cancelling an arrayrun

scancel 187184



The screenshot shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The user has submitted a batch job and is viewing its status. The job is an array job with 12 elements. The status of the array job is shown in the last row of the table, which is highlighted with a red box.

```
-bash-4.1$ sbatch telltime_submit.slurm
Submitted batch job 187184
-bash-4.1$
-bash-4.1$
-bash-4.1$ squeue -u katerim
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
187186	normal	187184.1	katerim	R	0:38	1	c1-20
187187	normal	187184.2	katerim	R	0:38	1	c1-20
187188	normal	187184.3	katerim	R	0:38	1	c2-13
187189	normal	187184.4	katerim	R	0:38	1	c2-15
187190	normal	187184.5	katerim	R	0:38	1	c2-18
187191	normal	187184.6	katerim	R	0:38	1	c1-32
187192	normal	187184.7	katerim	R	0:38	1	c1-33
187193	normal	187184.8	katerim	R	0:38	1	c2-29
187194	normal	187184.9	katerim	R	0:38	1	c2-33
187195	normal	187184.1	katerim	R	0:38	1	c2-33
187184	normal	array_te	katerim	R	1:08	1	c1-12

```
-bash-4.1$
```

Looking at the results...

```
abel.uio.no - PuTTY
-rw-r--r-- 1 katerim users 51 Oct 18 15:46 99.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 90.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 85.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 86.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 84.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 83.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 81.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 88.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 89.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 82.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:43 87.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 71.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 72.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 73.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 74.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 75.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 76.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 77.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 78.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 79.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:39 80.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 66.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 67.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 61.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 65.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 64.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 70.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 63.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 68.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 62.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:36 69.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:32 53.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:32 51.currenttime.txt
-rw-r--r-- 1 katerim users 51 Oct 18 15:32 54.currenttime.txt
More...
```

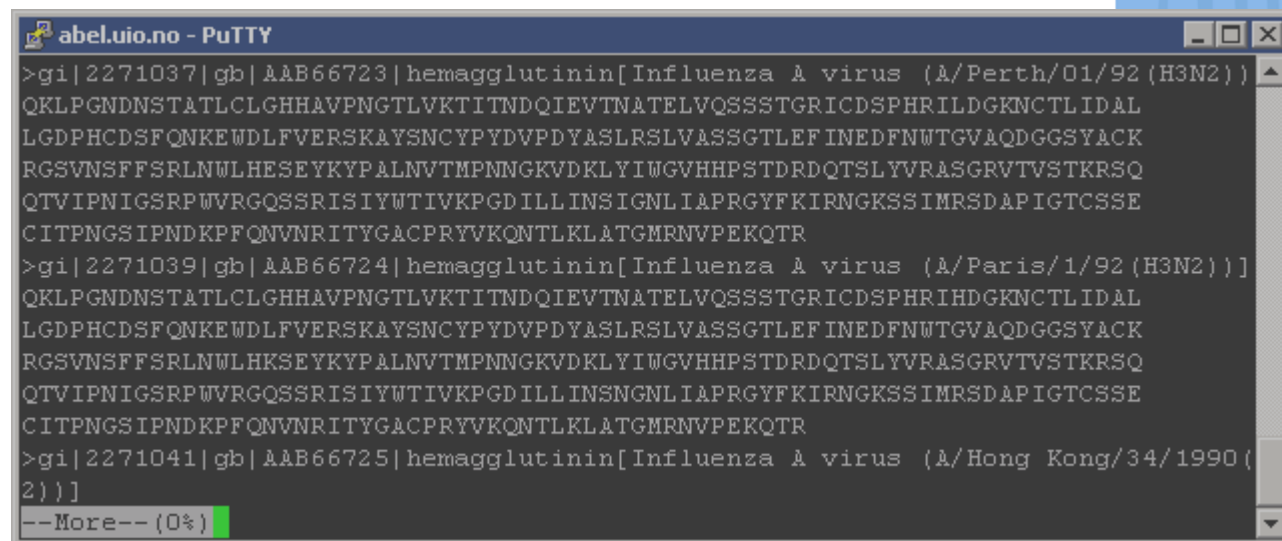
abel.uio.no - PuTTY

```
1: Thu Oct 18 15:11:23 2012 on compute-1-20.local
2: Thu Oct 18 15:11:24 2012 on compute-1-20.local
3: Thu Oct 18 15:11:23 2012 on compute-2-13.local
4: Thu Oct 18 15:11:24 2012 on compute-2-15.local
5: Thu Oct 18 15:11:23 2012 on compute-2-18.local
6: Thu Oct 18 15:11:23 2012 on compute-1-32.local
7: Thu Oct 18 15:11:23 2012 on compute-1-33.local
8: Thu Oct 18 15:11:24 2012 on compute-2-29.local
9: Thu Oct 18 15:11:24 2012 on compute-2-33.local
10: Thu Oct 18 15:11:27 2012 on compute-2-33.local
11: Thu Oct 18 15:14:23 2012 on compute-2-33.local
12: Thu Oct 18 15:14:53 2012 on compute-4-29.local
13: Thu Oct 18 15:14:55 2012 on compute-4-31.local
14: Thu Oct 18 15:14:53 2012 on compute-4-31.local
15: Thu Oct 18 15:14:58 2012 on compute-3-28.local
16: Thu Oct 18 15:14:54 2012 on compute-3-20.local
17: Thu Oct 18 15:14:53 2012 on compute-3-20.local
18: Thu Oct 18 15:14:54 2012 on compute-3-20.local
19: Thu Oct 18 15:14:56 2012 on compute-3-20.local
20: Thu Oct 18 15:14:53 2012 on compute-3-17.local
21: Thu Oct 18 15:17:53 2012 on compute-1-20.local
22: Thu Oct 18 15:17:53 2012 on compute-1-32.local
23: Thu Oct 18 15:18:26 2012 on compute-2-33.local
24: Thu Oct 18 15:18:23 2012 on compute-2-33.local
25: Thu Oct 18 15:18:23 2012 on compute-2-13.local
26: Thu Oct 18 15:18:23 2012 on compute-2-15.local
27: Thu Oct 18 15:18:23 2012 on compute-2-18.local
28: Thu Oct 18 15:18:25 2012 on compute-3-17.local
29: Thu Oct 18 15:18:27 2012 on compute-3-17.local
30: Thu Oct 18 15:18:23 2012 on compute-3-17.local
31: Thu Oct 18 15:27:12 2012 on compute-1-33.local
32: Thu Oct 18 15:21:53 2012 on compute-2-33.local
33: Thu Oct 18 15:21:53 2012 on compute-3-35.local
34: Thu Oct 18 15:21:53 2012 on compute-4-29.local
35: Thu Oct 18 15:27:12 2012 on compute-4-31.local
36: Thu Oct 18 15:21:53 2012 on compute-4-31.local
37: Thu Oct 18 15:21:59 2012 on compute-3-28.local
38: Thu Oct 18 15:21:53 2012 on compute-3-17.local
39: Thu Oct 18 15:27:12 2012 on compute-3-17.local
40: Thu Oct 18 15:21:56 2012 on compute-3-17.local
```

--More-- (39%)

Array run example 2

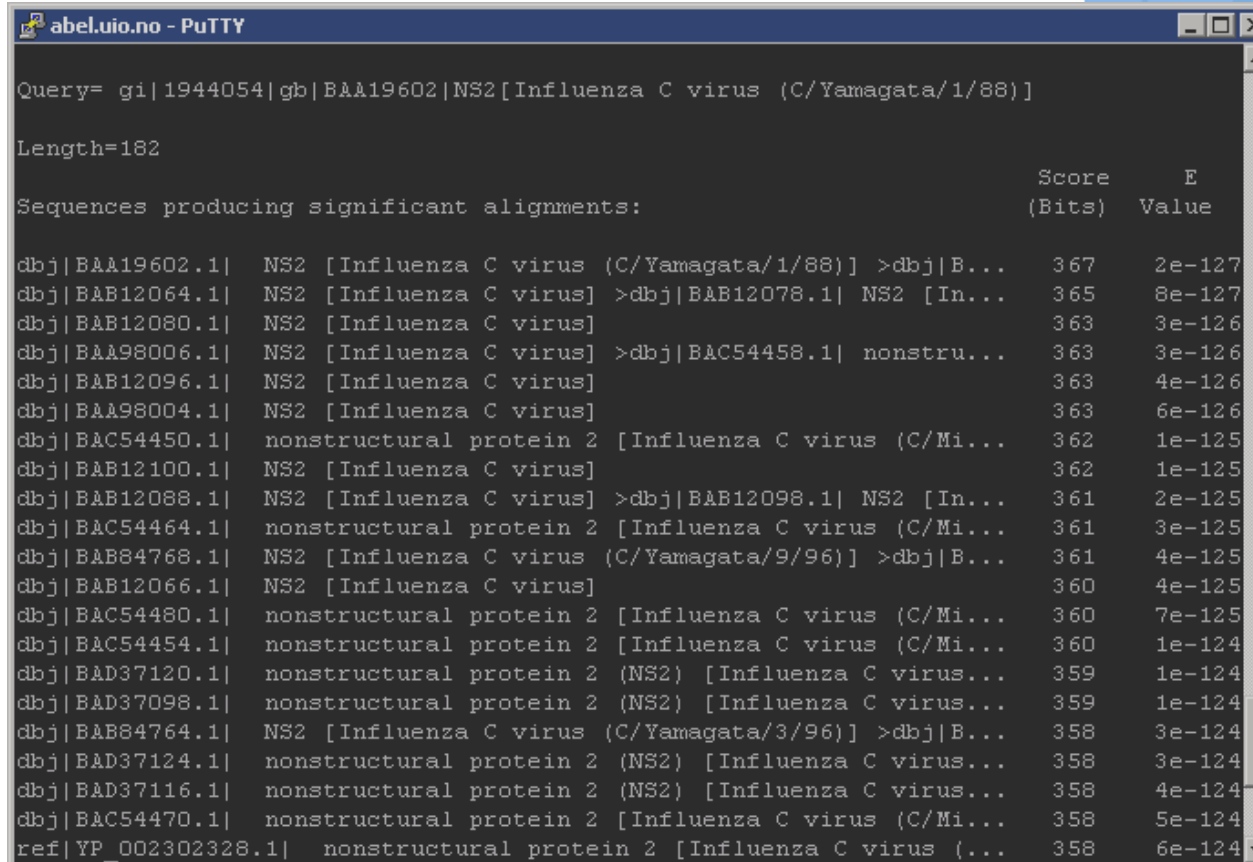
- BLAST - sequence similarity search program
<http://blast.ncbi.nlm.nih.gov/>
- Input
 - biological sequences
<ftp://ftp.ncbi.nih.gov/genomes/INFLUENZA/influenza.faa>
 - Database of sequences <ftp://ftp.ncbi.nih.gov/blast/db/>



```
abel.uio.no - PuTTY
>gi|2271037|gb|AAB66723|hemagglutinin[Influenza A virus (A/Perth/01/92 (H3N2))]
QKLP GNDNSTATLCLGHHAVPNGTLVKTITNDQIEVTNATELVQSSSTGRICDSPHRILDGKNCTLIDAL
LGDPHCDSFQNKEDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEFINEDFNWTGVAQDGGSYACK
RGSVNSFFSRLNWLHESEYKYPALNVTMPNNGKVDKLYIWGVVHPSTDRDQTSLYVRASGRVTVSTKRSQ
QTVIPNIGSRPWVRGQSSRSISYWTIVKPGDILLINSIGNLIAPRGYFKIRNGKSSIMRSDAPIGTCSE
CITPNGSIPNDKPFQVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>gi|2271039|gb|AAB66724|hemagglutinin[Influenza A virus (A/Paris/1/92 (H3N2))]
QKLP GNDNSTATLCLGHHAVPNGTLVKTITNDQIEVTNATELVQSSSTGRICDSPHRIHDGKNCTLIDAL
LGDPHCDSFQNKEDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEFINEDFNWTGVAQDGGSYACK
RGSVNSFFSRLNWLHKSEYKYPALNVTMPNNGKVDKLYIWGVVHPSTDRDQTSLYVRASGRVTVSTKRSQ
QTVIPNIGSRPWVRGQSSRSISYWTIVKPGDILLINSNGNLIAPRGYFKIRNGKSSIMRSDAPIGTCSE
CITPNGSIPNDKPFQVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>gi|2271041|gb|AAB66725|hemagglutinin[Influenza A virus (A/Hong Kong/34/1990 (
2))]
--More-- (0%)
```

Array run example 2

- Output
 - sequence matches
 - probabilistic scores
 - sequence alignments



The screenshot shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The query is 'gi|1944054|gb|BAA19602|NS2[Influenza C virus (C/Yamagata/1/88)]' with a length of 182. The results are a table of sequences producing significant alignments, sorted by score. The table has three columns: 'Sequences producing significant alignments:', 'Score (Bits)', and 'E Value'. The results list various Influenza C virus NS2 proteins and nonstructural proteins 2 from different strains, with scores ranging from 358 to 367 bits and E-values from 2e-127 to 6e-124.

```
Query= gi|1944054|gb|BAA19602|NS2[Influenza C virus (C/Yamagata/1/88)]
Length=182

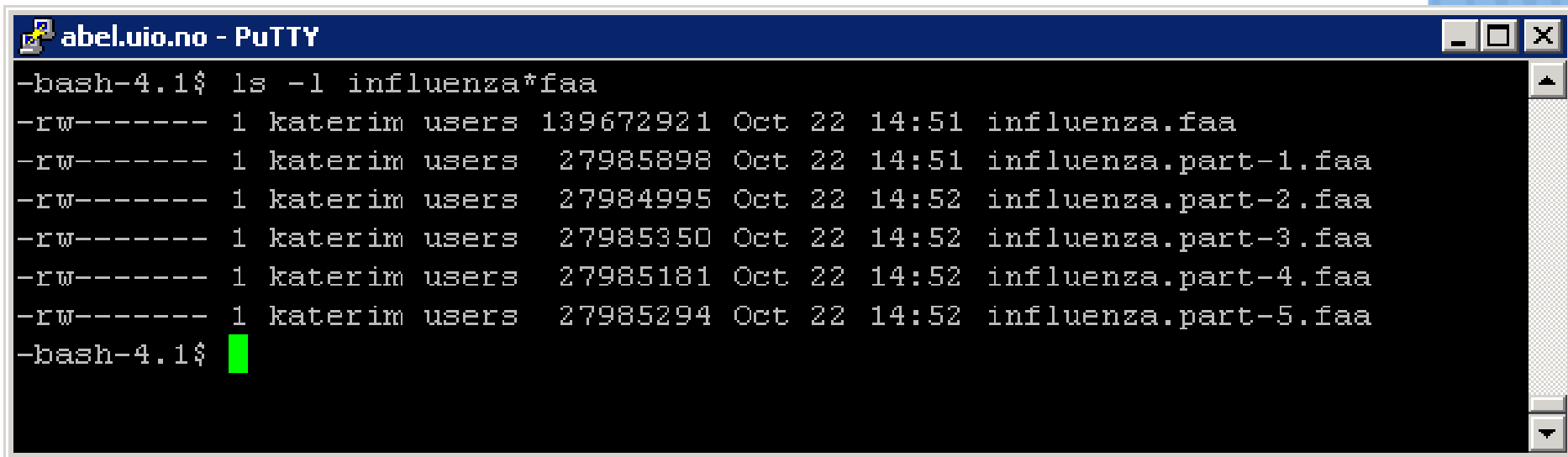
Sequences producing significant alignments:
```

Sequences producing significant alignments:	Score (Bits)	E Value
dbj BAA19602.1 NS2 [Influenza C virus (C/Yamagata/1/88)] >dbj B...	367	2e-127
dbj BAB12064.1 NS2 [Influenza C virus] >dbj BAB12078.1 NS2 [In...	365	8e-127
dbj BAB12080.1 NS2 [Influenza C virus]	363	3e-126
dbj BAA98006.1 NS2 [Influenza C virus] >dbj BAC54458.1 nonstru...	363	3e-126
dbj BAB12096.1 NS2 [Influenza C virus]	363	4e-126
dbj BAA98004.1 NS2 [Influenza C virus]	363	6e-126
dbj BAC54450.1 nonstructural protein 2 [Influenza C virus (C/Mi...	362	1e-125
dbj BAB12100.1 NS2 [Influenza C virus]	362	1e-125
dbj BAB12088.1 NS2 [Influenza C virus] >dbj BAB12098.1 NS2 [In...	361	2e-125
dbj BAC54464.1 nonstructural protein 2 [Influenza C virus (C/Mi...	361	3e-125
dbj BAB84768.1 NS2 [Influenza C virus (C/Yamagata/9/96)] >dbj B...	361	4e-125
dbj BAB12066.1 NS2 [Influenza C virus]	360	4e-125
dbj BAC54480.1 nonstructural protein 2 [Influenza C virus (C/Mi...	360	7e-125
dbj BAC54454.1 nonstructural protein 2 [Influenza C virus (C/Mi...	360	1e-124
dbj BAD37120.1 nonstructural protein 2 (NS2) [Influenza C virus...	359	1e-124
dbj BAD37098.1 nonstructural protein 2 (NS2) [Influenza C virus...	359	1e-124
dbj BAB84764.1 NS2 [Influenza C virus (C/Yamagata/3/96)] >dbj B...	358	3e-124
dbj BAD37124.1 nonstructural protein 2 (NS2) [Influenza C virus...	358	3e-124
dbj BAD37116.1 nonstructural protein 2 (NS2) [Influenza C virus...	358	4e-124
dbj BAC54470.1 nonstructural protein 2 [Influenza C virus (C/Mi...	358	5e-124
ref YP_002302328.1 nonstructural protein 2 [Influenza C virus (...]	358	6e-124

Parallelizing BLAST

- Split the query database
 - Perl fasta splitter from

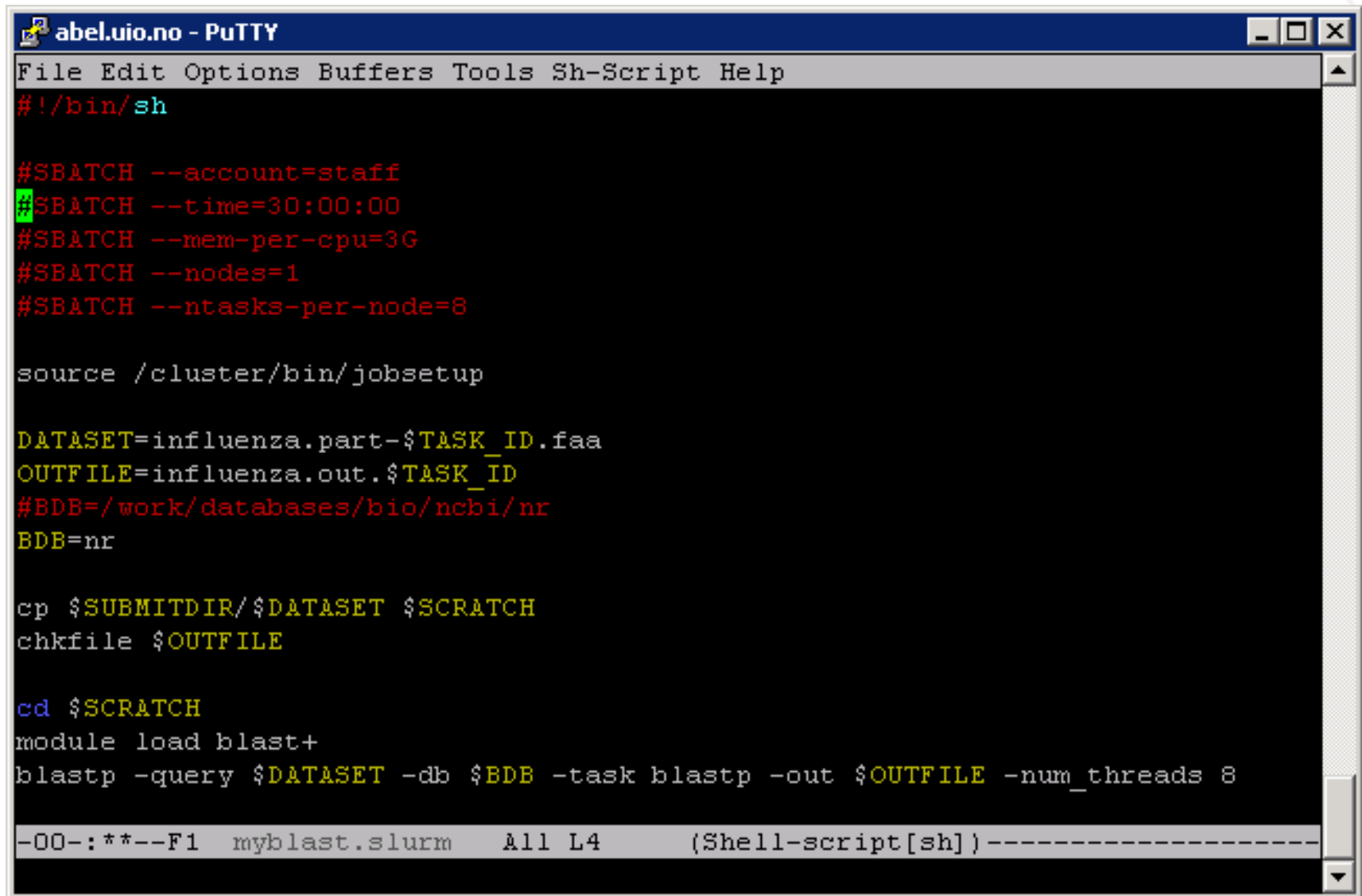
<http://kirill-kryukov.com/study/tools/fasta-splitter/>



The screenshot shows a PuTTY terminal window titled 'abel.uio.no - PuTTY'. The terminal displays the output of the command 'ls -l influenza*faa'. The output lists six files: 'influenza.faa' and five split files ('influenza.part-1.faa' through 'influenza.part-5.faa'). Each file entry includes permissions, size, owner, group, date, time, and filename. The terminal prompt is '-bash-4.1\$' followed by a green cursor.

```
abel.uio.no - PuTTY
-bash-4.1$ ls -l influenza*faa
-rw----- 1 katerim users 139672921 Oct 22 14:51 influenza.faa
-rw----- 1 katerim users  27985898 Oct 22 14:51 influenza.part-1.faa
-rw----- 1 katerim users  27984995 Oct 22 14:52 influenza.part-2.faa
-rw----- 1 katerim users  27985350 Oct 22 14:52 influenza.part-3.faa
-rw----- 1 katerim users  27985181 Oct 22 14:52 influenza.part-4.faa
-rw----- 1 katerim users  27985294 Oct 22 14:52 influenza.part-5.faa
-bash-4.1$
```

Abel worker script



```
abel.uio.no - PuTTY
File Edit Options Buffers Tools Sh-Script Help
#!/bin/sh

#SBATCH --account=staff
#SBATCH --time=30:00:00
#SBATCH --mem-per-cpu=3G
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=8

source /cluster/bin/jobsetup

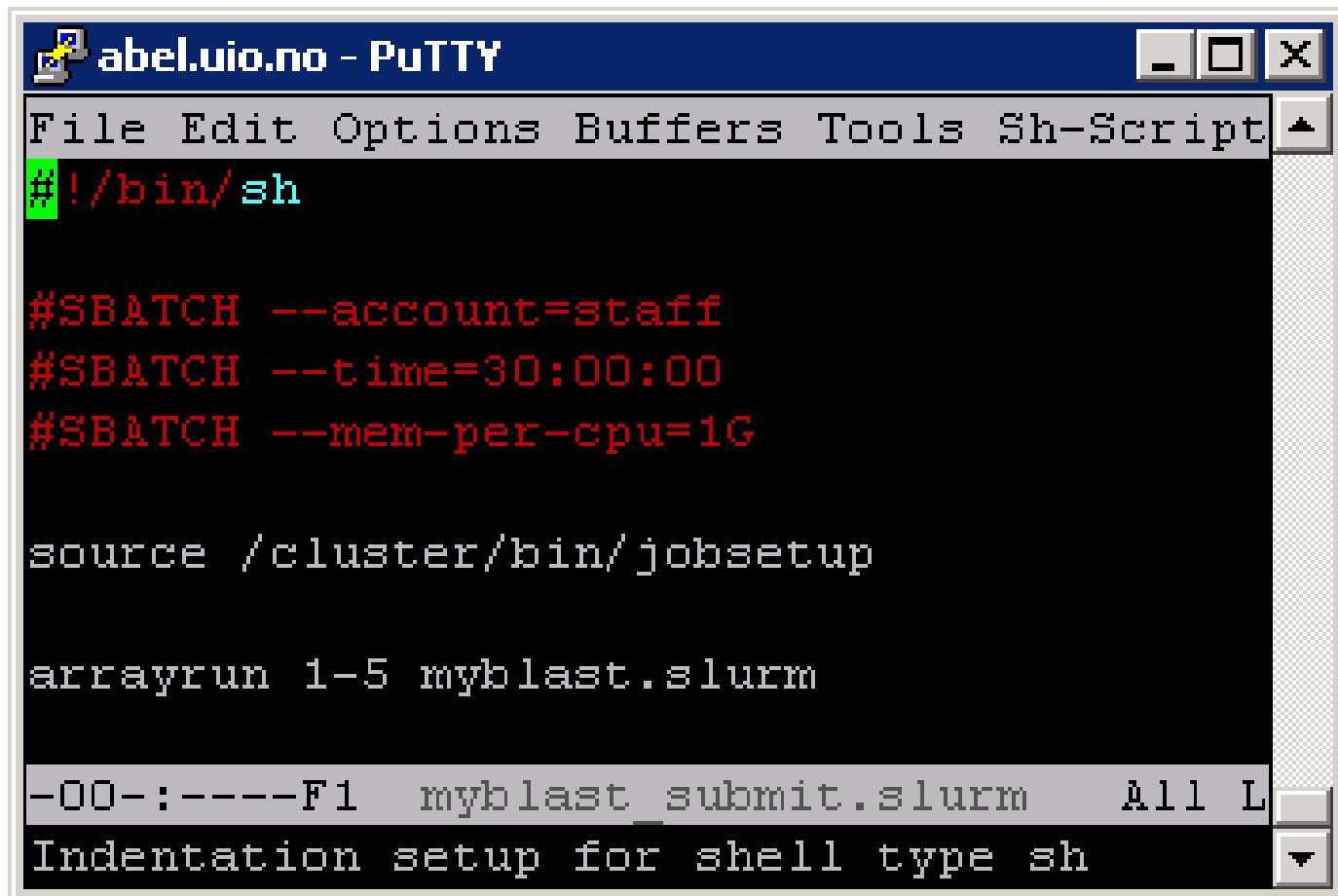
DATASET=influenza.part- $\$TASK\_ID$ .faa
OUTFILE=influenza.out. $\$TASK\_ID$ 
#BDB=/work/databases/bio/ncbi/nr
BDB=nr

cp  $\$SUBMITDIR/\$DATASET$   $\$SCRATCH$ 
chkfile  $\$OUTFILE$ 

cd  $\$SCRATCH$ 
module load blast+
blastp -query  $\$DATASET$  -db  $\$BDB$  -task blastp -out  $\$OUTFILE$  -num_threads 8

-OO-: **--F1 myblast.slurm All L4 (Shell-script[sh])-----
```

Abel submit script



```
abel.uio.no - PuTTY
File Edit Options Buffers Tools Sh-Script
#!/bin/sh

#SBATCH --account=staff
#SBATCH --time=30:00:00
#SBATCH --mem-per-cpu=1G

source /cluster/bin/jobsetup

arrayrun 1-5 myblast.slurm

-00-:----F1  myblast submit.slurm  All L
Indentation setup for shell type sh
```

Abel in action

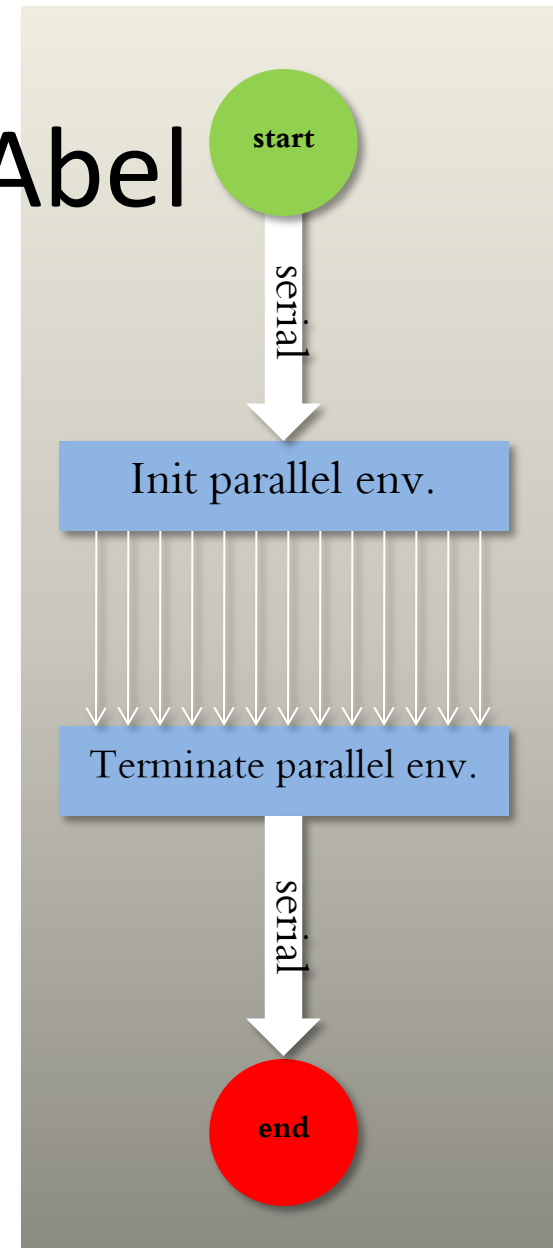
```
abel.uio.no - PuTTY
-bash-4.1$ ls -l
total 273118
drwxr-xr-x 2 katerim users      12 Oct 23 09:48 archive
-rw----- 1 katerim users      743 Oct 22 14:51 course.txt
drwxr-xr-x 2 katerim users     105 Oct 22 15:07 db
-rwx----- 1 katerim users    7976 Oct 22 14:51 fasta_splitter.pl
-rw----- 1 katerim users 139672921 Oct 22 14:51 influenza.faa
-rw----- 1 katerim users 27985898 Oct 22 14:51 influenza.part-1.faa
-rw-r--r-- 1 katerim users  59805 Oct 22 15:09 influenza.part-1.faa~
-rw----- 1 katerim users 27984995 Oct 22 14:52 influenza.part-2.faa
-rw----- 1 katerim users 27985350 Oct 22 14:52 influenza.part-3.faa
-rw----- 1 katerim users 27985181 Oct 22 14:52 influenza.part-4.faa
-rw----- 1 katerim users 27985294 Oct 22 14:52 influenza.part-5.faa
-rw-r--r-- 1 katerim users    416 Oct 23 09:50 myblast.slurm
-rw-r--r-- 1 katerim users    416 Oct 22 15:39 myblast.slurm~
-rw-r--r-- 1 katerim users    143 Oct 23 09:50 myblast_submit.slurm
-rw-r--r-- 1 katerim users    143 Oct 22 19:55 myblast_submit.slurm~
-bash-4.1$ grep -c ">" influenza.part-1.faa
57252
-bash-4.1$ sbatch myblast_submit.slurm
Submitted batch job 243856
-bash-4.1$ squeue -u katerim
  JOBID PARTITION    NAME     USER  ST       TIME  NODES NODELIST(REASON)
  243863    normal 243856.5  katerim  R         0:39      1 c4-20
  243862    normal 243856.4  katerim  R         0:41      1 c4-20
  243860    normal 243856.3  katerim  R         0:43      1 c4-19
  243858    normal 243856.2  katerim  R         0:45      1 c4-19
  243857    normal 243856.1  katerim  R         0:47      1 c4-18
  243856    normal myblast_  katerim  R         0:49      1 c1-35
-bash-4.1$ █
```

```
total 519080
drwxr-xr-x 3 katerim users      22 Apr 16 2013 .
drwxr-xr-x 4 katerim users      27 Mar 24 2014 ..
drwxr-xr-x 2 katerim users      12 Oct 23 2012 archive
-rw----- 1 katerim users      743 Oct 22 2012 course.txt
-rwx----- 1 katerim users     7976 Oct 22 2012 fasta_splitter.pl
-rw----- 1 katerim users 139672921 Oct 22 2012 influenza.faa
-rw-r--r-- 1 katerim users 123846232 Oct 24 2012 influenza.out.1
-rw-r--r-- 1 katerim users  32347422 Oct 24 2012 influenza.out.2
-rw-r--r-- 1 katerim users  31381629 Oct 24 2012 influenza.out.3
-rw-r--r-- 1 katerim users  21865726 Oct 24 2012 influenza.out.4
-rw-r--r-- 1 katerim users  42332986 Oct 24 2012 influenza.out.5
-rw----- 1 katerim users  27985898 Oct 22 2012 influenza.part-1.faa
-rw----- 1 katerim users  27984995 Oct 22 2012 influenza.part-2.faa
-rw----- 1 katerim users  27985350 Oct 22 2012 influenza.part-3.faa
-rw----- 1 katerim users  27985181 Oct 22 2012 influenza.part-4.faa
-rw----- 1 katerim users  27985294 Oct 22 2012 influenza.part-5.faa
-rw-r--r-- 1 katerim users      425 Oct 23 2012 myblast.slurm
-rw-r--r-- 1 katerim users      143 Oct 23 2012 myblast_submit.slurm
-rw-r--r-- 1 katerim users  157180 Oct 24 2012 slurm-243856.out
-rw-r--r-- 1 katerim users    633 Oct 24 2012 slurm-243857.out
-rw-r--r-- 1 katerim users    633 Oct 24 2012 slurm-243858.out
-rw-r--r-- 1 katerim users    633 Oct 24 2012 slurm-243860.out
-rw-r--r-- 1 katerim users    633 Oct 24 2012 slurm-243862.out
-rw-r--r-- 1 katerim users    633 Oct 24 2012 slurm-243863.out
-bash-4.1$
```

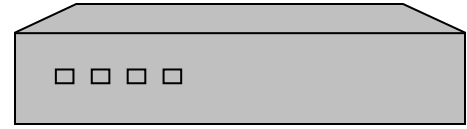


Parallel jobs on Abel

- Two kinds of parallel jobs
 - Single node
 - OpenMP
 - Multiple nodes
 - MPI



Single node



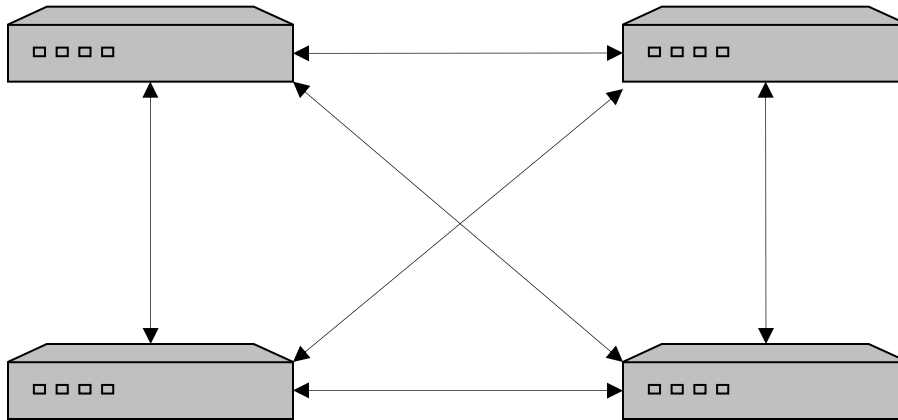
- Shared memory is possible
 - Threads
 - OpenMP
- Message passing
 - MPI

OpenMP job script

```
[olews@login-0-1 OpenMP]$ cat hello.run
#!/bin/bash
#SBATCH --account=staff
#SBATCH --time=00:01:00
#SBATCH --mem-per-cpu=100M
#SBATCH --ntasks-per-node=4 --nodes=1
source /cluster/bin/jobsetup
export OMP_NUM_THREADS=$SLURM_CPUS_ON_NODE
./hello.x
```

Multiple nodes

- Distributed memory
 - Message passing, MPI



MPI on Abel

- we support Open MPI
 - module load openmpi
- use mpicc and mpif90 as compilers
- use the same MPI module for compilation and execution
- read <http://hpc.uio.no/index.php/OpenMPI>
- special concern – distributing your files to all nodes

sbcast myfiles.* \$SCRATCH

- jobs specifying more than one node automatically get

#SBATCH --constraint=ib

MPI job script

```
#!/bin/bash
#SBATCH --account=staff
#SBATCH --time=0:01:0
#SBATCH --mem-per-cpu=100M
#SBATCH --ntasks-per-node=1
#SBATCH --nodes=4
source /cluster/bin/jobsetup

module load openmpi
mpirun ./hello.x
```

Notur – apply for more resources on Abel

- The Norwegian metacenter for computational science
- <http://www.notur.no/>
- Large part of our funding is provided through Notur
- You can apply for:
 - Abel compute hours
 - Data storage
 - Advanced user support

Thank you.



