Gromacs at CSC

Gromacs 5, CSC 23.11.2015

Atte Sillanpää

First: check the info page

https://research.csc.fi/-/gromacs

Choosing the right platform

- Sisu
  - Batch jobs
  - Massively parallel: 72-1008 cores/job (but upto 9000 cores/job after confirming that your jobs parallelize that far, see instructions)
  - Haswell architecture (24 cores/node)
- Taito
  - Batch jobs
  - Serial and parallel: 1- 672 cores/job (but rather run big jobs in Sisu)
  - Haswell and Sandy Bridge architecture (24, or 16 cores/node)
- Taito-shell
  - Interactive work (analysis) max 4 cores/job
  - Shared resource
- Taito-GPU
  - Serial and parallel jobs using K40 GPGPU-cards and slowish CPUs
  - Efficiency depends on job type

Batch vs. interactive use

- Login node: submit batch job

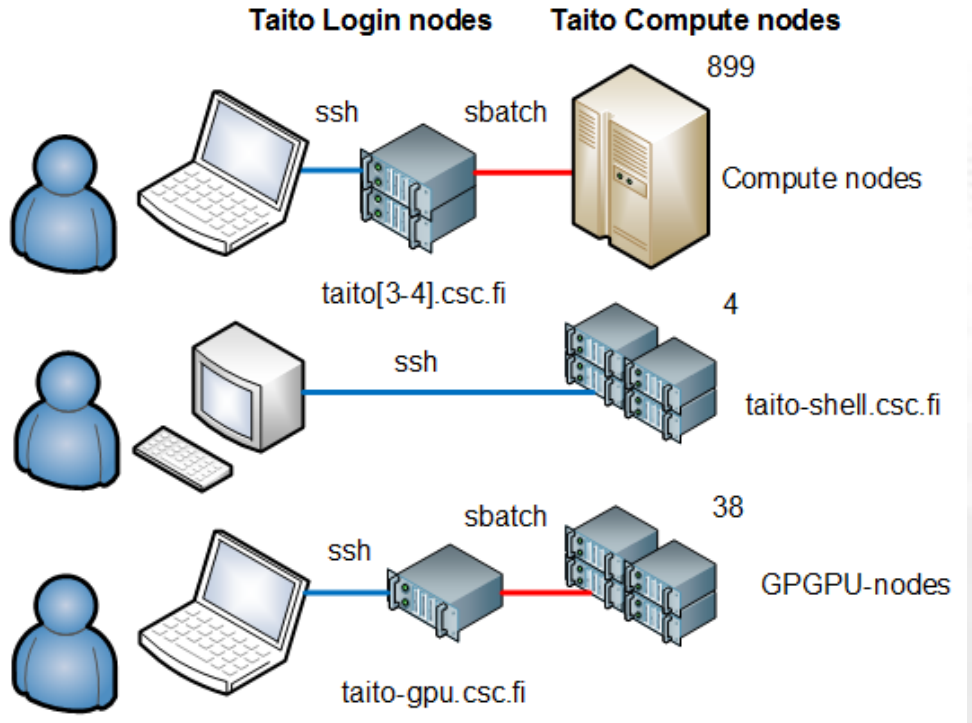
```
sbatch job_script.sh
```

- taito-shell: run directly

```
./my_prog &
```

- taito-gpu: submit to GPGPU nodes

```
sbatch job_script.sh
```



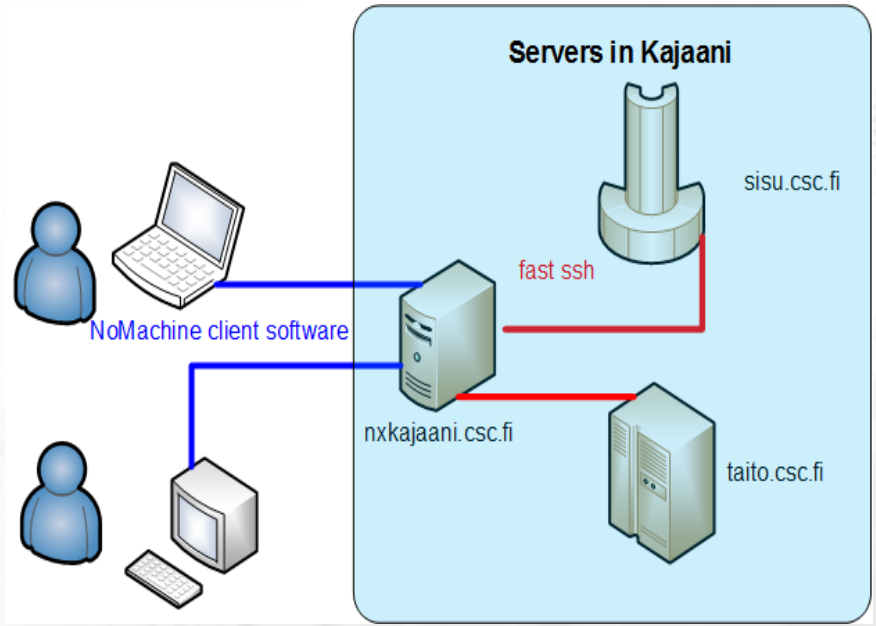
Youtube video on submitting batch jobs at CSC

Accessing platforms

- **ssh** username@taito.csc.fi
  - **ssh -Y** username@taito.csc.fi for X11 access
  - Needs an X11 emulator on Windows – not recommended
- NoMachine -> nxkajaani.csc.fi → then ssh
  - Recommended for interactive work
  - Fluent X11 experience
  - Needs installing the NoMachine client (Admin rights)
  - You can also suspend the session and continue later

NoMachine remote desktop

- Client connection between user and gateway
- Good performance even with slow network
- Ssh from gateway to server (fast if local)
- Persistent connection
- Suspendable
  - Continue later at another location
- Read the instructions…
  - ssh-key, keyboard layout, mac specific workarounds, …
- Choose an application or server to use (right click)



Using Gromacs at CSC

- Initialise Gromacs
  - **`module avail gromacs`**
  - **`module load gromacs/version`**
  - **`module load gromacs-env/version`**
- Keep files in working directory
  - **`cd $WRKDIR`**
  - Save important (final) results in IDA or HPC_ARCHIVE

Specifying resources

- # of cores
  - Either serial, or full nodes
  - In taito multiple of 16 or 24. Haswell and Sandy Bridge can't be mixed
  - For parallel jobs, in Taito add **`#BATCH –exclusive`** or reserve nodes (instead of cores)
  - Test performance to see how many cores can be used efficiently
- Memory
  - Gromacs needs usually very little memory
  - The default is 512MB/core, this is ok
  - Actual usage can be checked afterwards with **`sacct`**, see FAQ
- Runtime
  - Shorter queues give resources faster
  - Don't ask for too much "just in case" (but this is not so critical)
  - Use **`-maxh [hours]`** for mdrun to end job gracefully if unsure of duration, then continue
  - Instructions on restarting/extending simulations
- Architecture
  - On Sisu all nodes are Haswell
  - On Taito, some Sandy Bridge (**`#SBATCH -C snb`**) some Haswell (**`#SBATCH -C hsw`**)
  - Choose "snb" version of Gromacs for Sandybridge and "hsw" for Haswell
  - "hsw" won't run on Sandybridge, and "hsw" on Haswell will give better performance

Hardware for many needs

Node type	Number of nodes	cores / node	Total cores	Memory / node
Sandy Bridge login node	4	16	64	64 / 192 GB
Haswell compute node	397	24	9528	128 GB
Haswell big memory node	10	24	240	256 GB
Sandy Bridge compute node	496	16	7936	64 GB
Sandy Bridge big memory node	16	16	256	256 GB
Sandy Bridge huge memory node	2	32	64	1,5 TB
Intel Xeon E5-2620V2 node + 2 Kepler K40 GPGPUs	38	12	456	32 GB
Haswell compute node (Sisu)	1688	24	40512	64 GB

Tip: Check server load from https://sui.csc.fi/web/guest/host-monitor

Optimizations vs. architecture, Taito

48 cores	Sandy nodes	Haswell nodes
Sandy gromacs	25.8 * ns/day	41.7 ns/day
Haswell Gromacs	Won't run	44.8 ns/day

144 cores	Sandy nodes	Haswell nodes
Sandy gromacs	96.1 † ns/day	106.7 ns/day
Haswell Gromacs	Won't run	111.4 ns/day

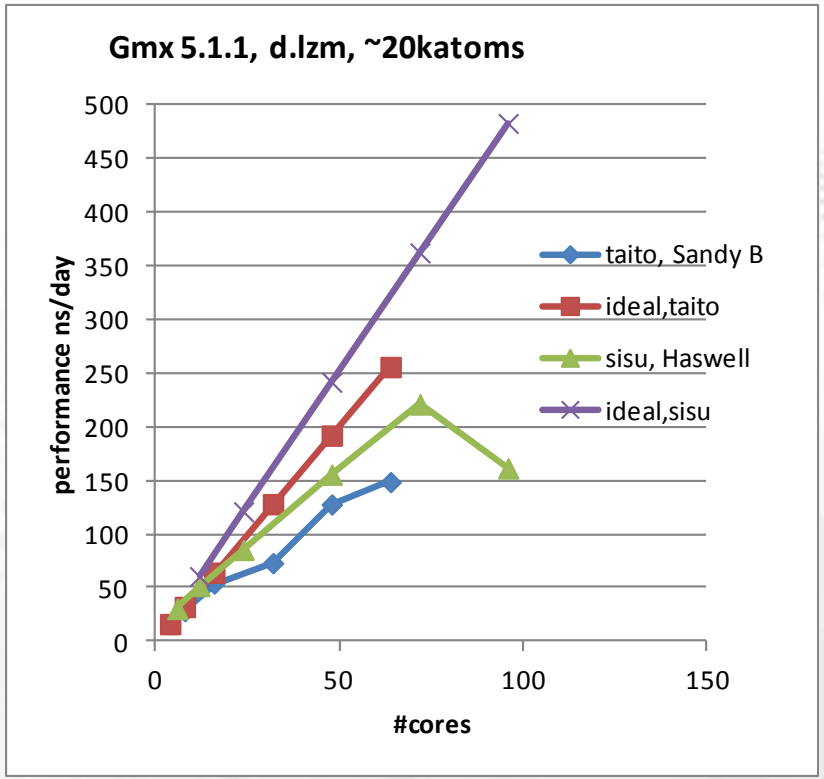
* 3 nodes on Sandy Bridge architecture, 2 on Haswell
† 9 nodes on Sandy Bridge architecture, 6 on Haswell
System: "dppc", 121856 particles, membrane in water, PME, verlet cutoff, NPT, v-rescale, 3 minute run (too short for sustained performance)

Performance tests

- Production system (or as similar as possible)
- Run for short time with increasing # of cores (`-maxh 0.1`)
- Speed should increase at least 1.5 fold, when #cores is doubled
- Better to run many jobs simultaneously than one using more cores
- Think about optimal usage of the whole resource

taito, Sandy B	ns/day	ideal,taito	speedup	1=perf. scaling
4	16.0	16.0		
8	28.1	32.0	1.76	0.88
16	53.9	63.9	1.92	0.96
32	72.8	127.9	1.35	0.68
48	127.6	191.8	1.75	1.17
64	148.4	255.8	1.16	0.87

sisu, Haswell	ns/day	ideal,sisu	speedup	1=perf. scaling
6	30.2			
12	51.5	60.4	1.71	0.85
24	85.2	120.8	1.65	0.83
48	155.4	241.5	1.82	0.91
72	220.9	362.3	1.42	0.95
96	161.2	483.0	0.73	0.55



Queue priorities

- CSC uses Fair share prioritization
  - Recent usage decreases priority
  - Waiting in queue increases priority
- Backfiller runs the job if a slot is available
  - A number of lower priority jobs will be tried
  - Short jobs with modest memory requirements may run with a lower priority
- Test queue is only for testing (no tens of 30 minute jobs)
- Big memory requirements may leave nodes partially empty
- Limited resources in bigmem and hugemem nodes/partitions
- Perhaps better to run 3 day (taito) or 1 day (sisu) jobs and restart/extend
  - This can be scripted, but be careful!

Sisu batch example

```
#!/bin/bash -l
#SBATCH -t 00:30:00
#SBATCH -p test
#SBATCH -J GMX
#SBATCH -o ogmx.%j
#SBATCH -e egmx.%j
#SBATCH -N 8
#SBATCH --no-requeue
#SBATCH --mail-type=END
#SBATCH --mail-user=your.email@there.fi

(( ncores = SLURM_NNODES * 24 ))

export OMP_NUM_THREADS=1

module load gromacs/5.1.1

aprun -n $ncores gmx_mpi mdrun -s topol -dlb yes –maxh 0.5
```

Note: be careful with copy-paste from pdf: some characters may look ok, but are not understood by the shell!

Taito batch script example (Sandy)

```
#!/bin/bash -l
#SBATCH -t 00:06:00
#SBATCH -p test
#SBATCH -C snb
#SBATCH -J GMX
#SBATCH -o ogmx.%j
#SBATCH -e egmx.%j
#SBATCH -N 2
#SBATCH --ntasks-per-node=16

module load gromacs-env/5.1.1-snb

((ncores=16*SLURM_NNODES))
export OMP_NUM_THREADS=1

srun gmx_mpi mdrun -s topol -dlb yes -deffnm v_511_snb_$ncores -maxh 0.05
```

Taito batch script example (Haswell)

```
#!/bin/bash -l
#SBATCH -t 00:06:00
#SBATCH -p test
#SBATCH -C hsw
#SBATCH -J GMX
#SBATCH -o ogmx.%j
#SBATCH -e egmx.%j
#SBATCH -N 2
#SBATCH --ntasks-per-node=24

module load gromacs-env/5.1.1-hsw

((ncores=24*SLURM_NNODES))
export OMP_NUM_THREADS=1

srun gmx_mpi mdrun -s topol -dlb yes -deffnm v_511_snb_$ncores -maxh 0.05
```

Array job example

```
#!/bin/bash
#SBATCH -J array_job
#SBATCH -o array_job_out_%A_%a.txt
#SBATCH -e array_job_err_%A_%a.txt
#SBATCH -t 02:00:00
#SBATCH --mem-per-cpu=512
#SBATCH --array=1-50
#SBATCH -n 1
#SBATCH -p serial

# move to the directory where the data files locate
cd data_dir
export OMP_NUM_THREADS=1
# run the command

srun gmx mdrun -nt 1 topol_"$SLURM_ARRAY_TASK_ID".tpr \
     -deffnm out_"$SLURM_ARRAY_TASK_ID" -maxh 2
```

Note: this script expects to find topol_1.tpr, topol_2.tpr … topol_50.tpr

GPGPU example

```
#!/bin/bash -l
#SBATCH -t 00:30:00
#SBATCH -p gpu
#SBATCH -J GMX
#SBATCH -o ogmx.%j
#SBATCH -e egmx.%j
#SBATCH -N 4
#SBATCH --ntasks-per-node=2
#SBATCH --exclusive
#SBATCH --gres=gpu:2

module load gromacs-env
export OMP_NUM_THREADS=6
((tasks=2*SLURM_NNODES))

srun --gres=gpu:2 -n $tasks mdrun_mpi -ntomp $OMP_NUM_THREADS -pin on \
 -s topol.tpr -dlb auto
```

Note: submit job from taito-gpu.csc.fi !

Chaining long jobs (restarts) in Sisu

```
#!/bin/bash -l
#SBATCH -t 24:00:00
#SBATCH -p small
#SBATCH -J GMX1
#SBATCH -N 20
#SBATCH --mail-type=END
#SBATCH --mail-user=your.email@there.fi

module load gromacs/5.1.1
((ncores=24*SLURM_NNODES))
export OMP_NUM_THREADS=1
job=your-jobname # edit this
aprun -n $ncores gmx_mpi mdrun -dlb yes \
     -deffnm $job -maxh 24 -noappend
```

```
> sbatch script1.sh
Submitted batch job 400105
```

```
#!/bin/bash -l
#SBATCH -t 24:00:00
#SBATCH -p small
#SBATCH -J GMX2
#SBATCH -N 20
#SBATCH --mail-type=END
#SBATCH --mail-user=your.email@there.fi
#SBATCH --dependency=afterok:400105

module load gromacs/5.1.1
((ncores=24*SLURM_NNODES))
export OMP_NUM_THREADS=1
job=your-jobname
prevjobname=`ls -t ${job}.part*.log| head -1`
ok=`grep "Finished mdrun " ${prevjobname} | wc -l `

if [ -s "${job}.cpt" ] && [ $ok == 1 ]
then
  echo "previous job likely ended ok"
  aprun -n $ncores gmx_mpi mdrun -s topol -dlb yes -deffnm \
      ${job} -cpi ${job} -maxh 24 -noappend
else
  exit # something wrong with previous job
fi

sleep 20 # wait a bit before submitting the next job
```

- *Note: be careful with copy-paste from pdf: some characters may look ok, but are not understood by the shell*
- *--dependency=afterok:<JOBID> knows only about SLURM-problems, you need to check also if mdrun had problems.*
- *First try with 2 minute jobs in the test queue – don't waste your time…*

```
> sbatch script2.sh
Submitted batch job 400110
> squeue -u pelle
pelle@sisu-login2:/wrk/pelle/subdir> squeue -u pelle
JOBID NAME ... ST         REASON           START_TIME        PRIORITY
400110 GMX1    PD         Dependency       N/A               4368
400105 GMX2    R          None             2015-11-20T1      4368
```

VMD at CSC → taito-shell

- Use NoMachine to log in
- Choose taito-shell from mouse menu, log in, then in command line:

```
> module load vmd
> vmd structure.gro
```