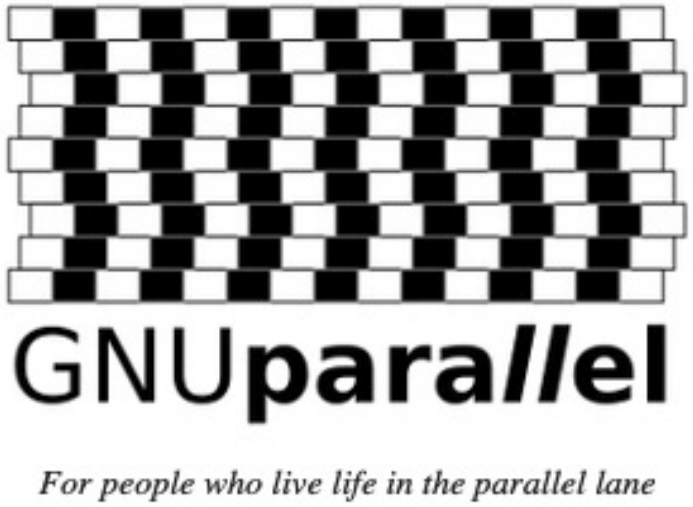
Presentation on Gnu parallel

By Kevin Fotso

What is gnu parallel?



- It is a shell tool for executing tasks/jobs in parallel using one or more nodes
- Very efficient for jobs that do the same scripts with different inputs

Strengths of gnu parallel

- Easily scales to very large number of tasks
- Easily scales to multiple nodes
- Efficient use of SLURM resources

Cons of gnu parallel

- Does not easily balance work across multiple nodes
- Careful optimization of input/output files is required
- Scaling up requires careful consideration of I/O performance
- Bash scripting familiarity recommended

Example case to experiment gnu parallel (1)

- Requesting nodes and loading gnu parallel

```
[kefo9343@login10 ~]$ acompile --ntasks=4 --time=01:00:00
acompile: submitting job... salloc --nodes=1 --partition=aco
e=acompile --bell --oversubscribe srun --pty /bin/bash
salloc: Granted job allocation 1219511
salloc: Waiting for resource configuration
salloc: Nodes c3cpu-a2-u7-4 are ready for job
```

- Requesting nodes and loading gnu parallel

```
[kefo9343@c3cpu-a2-u7-4 ~]$ ml gnu_parallel
```

Parallelizing echo (1)

- **“:::”** is where we are giving the argument parameters to echo

```
[kefo9343@c3cpu-a2-u7-4 ~]$ parallel echo ::: A B C
A
B
C
```

Parallelizing echo (2)

- We can give two lists of parameters

```
[kefo9343@c3cpu-a2-u7-4 ~]$ parallel echo ::: A B C C D E F ::: 1 2 3 4 5 6
A 3
A 1
A 2
A 4
A 5
```

Parallelizing echo (3)

- Or even 3!!!

```
[kefo9343@c3cpu-a2-u7-4 ~]$ parallel echo ::: A B C C D E F ::: 1 2 3 4 5 6 ::: alpha beta
A 1 alpha
A 1 beta
A 2 beta
```

Parallelizing echo (4)

- We can find text files in a directory and echo them

```
[kefo9343@c3cpu-a2-u7-4 gnu_parallel]$ find . -name "*.txt" | parallel echo find files {}
find files ./nodelist.txt
find files ./test1.txt
find files ./output_gnu_.txt
find files ./test3.txt
find files ./test4.txt
find files ./test2.txt
```

Parallelizing echo (5)

- **Another way to give arguments is by using brackets**

```
[kefo9343@c3cpu-a2-u7-4 gnu_parallel]$ parallel python3 hello_dummy.py {} ::: {1..1000}
Hello world from task: 1
Hello world from task: 2
Hello world from task: 3
Hello world from task: 4
Hello world from task: 6
Hello world from task: 8
Hello world from task: 7
```

Parallelizing echo (6)

- **Files can be used as parameter inputs. In that case, ‘::::’ is used.**

```
[kefo9343@c3cpu-a2-u7-4 gnu_parallel]$ parallel python3 hello_dummy.py {} :::: arg_file.txt
Hello world from task: 2
Hello world from task: 34
Hello world from task: 1
Hello world from task: 7667
```

Parallelizing multiple nodes

- **Gnu parallel needs to know the address of all the nodes involved.**

```
[kefo9343@c3cpu-a2-u7-4 gnu_parallel]$ scontrol show hostname > $SLURM_SUBMIT_DIR/nodelist.txt
```

- **'—sshloginfile' will help gnu parallel read the list of nodes involved and '—wd' will let gnu parallel know the working directory**

```
[kefo9343@c3cpu-a2-u7-4 gnu_parallel]$ parallel --sshloginfile nodelist.txt --wd $SLURM_SUBMIT_DIR python3 hell
o_dummy.py {} :::: arg_file.txt
```

Advanced commands

- '—joblog' helps you keep track of your tasks.
- '-j' represents the number of jobs/tasks per node
- '--env PATH' exports the PATH environment. It can be very useful when working with anaconda for instance

```
^C[kefo9343@c3cpu-a2-u7-4 gnu_parallel]parallel --joblog job.log -j 3 --env PATH --sshloginfile nodelist.txt --wd $SLU
RM_SUBMIT_DIR python3 hello_dummy.py {} :::: arg_file.txt
Hello world from task: 1
Hello world from task: 2
Hello world from task: 34
Hello world from task: 76
Hello world from task: 68
Hello world from task: 7667
```

Advanced gnu parallel on Alpine

- ‘—delay’ helps delay the execution for each task by 0.2 sec to mitigate I/O related issues
- Here for each task we cd to a directory and we run the Rscript using the parameter number as input (from 1 to 1000)

```
# GNU parallel for the main psa

my_parallel="parallel --env PATH --delay .2 -j $SLURM_NTASKS_PER_NODE --wd $TMPDIR --sshloginfile nodelist.txt --joblog job.log"
my_srun="srun --export=all --exclusive -n1 --cpus-per-task=$SLURM_CPUS_PER_TASK --cpu-bind=cores"

$my_parallel '$my_srun cd /scratch/alpine/kfotso@xsede.org/reduce_psa/{}; pwd; Rscript PSA.R {}' ::: {1..1000}
```

Sources:

- https://www.gnu.org/software/parallel/
- https://docs.nersc.gov/jobs/workflow/gnuparallel/
- https://www.youtube.com/watch?v=Rl06WD60afA
- https://curc.readthedocs.io/en/latest/software/GNUParallel.html?highlight=gnu%20parallel#using-gnu-parallel