Swarm on the Biowulf2 Cluster

Dr. David Hoover, SCB, CIT, NIH
staff@hpc.nih.gov
September 24, 2015

What is swarm?

- Wrapper script that simplifies running individual commands on the Biowulf cluster

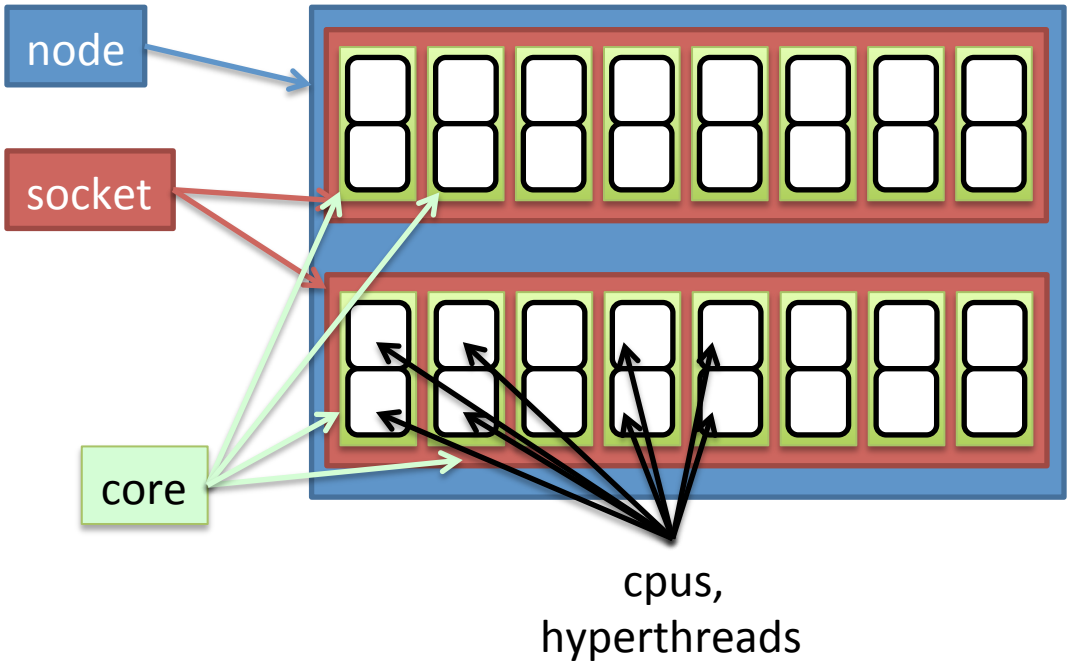
Documentation

- 'man swarm'
- 'swarm --help'
- https://hpc.nih.gov/apps/swarm.html

Differences in Biowulf 1 -> 2

- Jobs are allocated cores, rather than nodes
- All resources must be allocated
- All swarms are job arrays

Architectural Digest



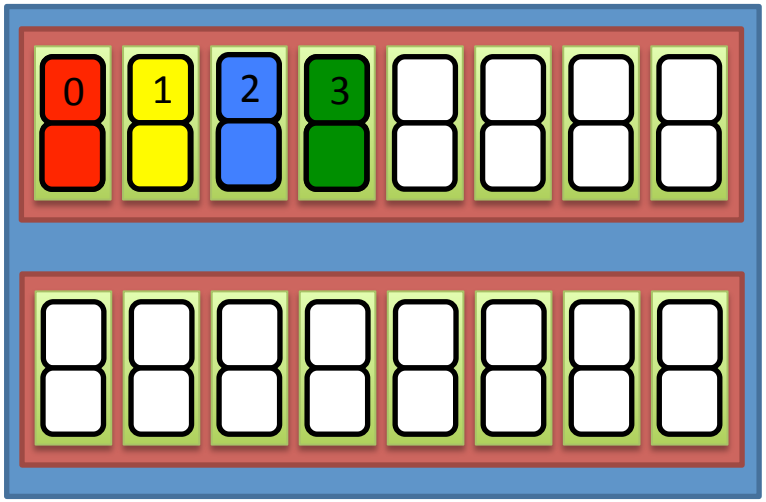
Simple swarm

subjobs

0	hostname ; uptime
1	hostname ; uptime
2	hostname ; uptime
3	hostname ; uptime

swarm

swarm -t 2



single-threaded swarm

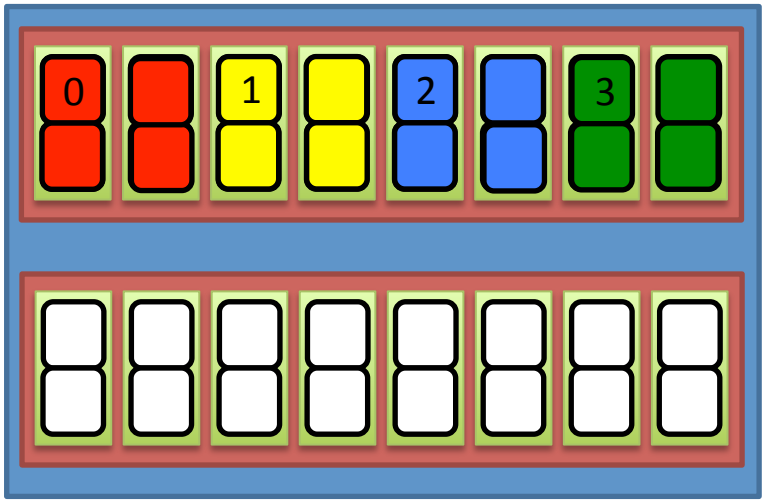
Simple swarm

subjobs

0	hostname ; uptime
1	hostname ; uptime
2	hostname ; uptime
3	hostname ; uptime

swarm -t 3

swarm -t 4



multi-threaded swarm

Basic swarm usage

- Create swarmfile (file.swarm)

```
cd /home/user/dir1 ; ls -l
cd /home/user/dir2 ; ls -l
cd /home/user/dir3 ; ls -l
cd /home/user/dir4 ; ls -l
```

- Submit and wait for output

```
$ swarm -f file.swarm
1234567
$ ls
swarm_1234567_0.o    swarm_1234567_1.o    swarm_1234567_2.o
swarm_1234567_0.e    swarm_1234567_1.e    swarm_1234567_2.e
```

Standard swarm options

- -f : swarmfile, list of commands to run
- -g : GB per process (NOT PER NODE OR JOB!)
- -t : threads/cpus per process (DITTO!)

```
command –p 4 –i /path/to/input1 –o /path/to/output1
command –p 4 –i /path/to/input2 –o /path/to/output2
command –p 4 –i /path/to/input3 –o /path/to/output3
command –p 4 –i /path/to/input4 –o /path/to/output4
```

```
$ swarm –f file.swarm –g 4 –t 4
```

-t auto

- Autothreading still enabled

```
java –Xmx8g –jar /path/to/jarfile –opt1 –opt2 –opt3
```

```
$ swarm –f file.swarm –g 4 –t auto
```

- This allocates an entire node to each process

-b, --bundle

- Bundling is slightly different
- swarms of > 1000 commands are autobundled
- --autobundle is deprecated

--singleout

- Concatenate all .o and .e into single files
- Not entirely reliable!
- CANCELLED and TIMEOUT will lose all output
- Better to use --logdir (described later)

Miscellaneous

- --usecsh
- --no-comment, --comment-char
- --no-scripts, --keep-scripts
- --debug, --devel, --verbose, --silent

--devel

```
$ swarm --devel -f file.swarm -b 4 -g 8 -v 4
-------------------------------------------------------
SWARM
├── subjob 0:  4 commands (1 cpu, 8.00 gb)
├── subjob 1:  4 commands (1 cpu, 8.00 gb)
├── subjob 2:  4 commands (1 cpu, 8.00 gb)
├── subjob 3:  4 commands (1 cpu, 8.00 gb)
-------------------------------------------------------
4 subjobs, 16 commands, 0 output file
16 commands run in 4 subjobs, each command requiring 8 gb and
1 thread, running 4 processes serially per subjob
sbatch --array=0-3 --job-name=swarm --output=/home/user/test/
swarm_%A_%a.o --error=/home/user/test/swarm_%A_%a.e --cpus-
per-task=1 --mem=8192 --partition=norm --time=16:00:00 /
spin1/swarm/user/I8DQDX4O.batch
```

No more .swarm directories

- swarm scripts are written to a central, shared area
- Each user has their own subdirectory

```
$ tree /spin1/swarm/user
/spin1/swarm/user
├── 2341529
│   ├── cmd.0
│   ├── cmd.1
│   ├── cmd.2
│   └── cmd.3
└── 2341529.batch
```

--license

- --license replaces -R or --resource

```
$ swarm –f file.swarm --license=matlab
```

--module

- --module now requires comma-delimited list, rather than space delimited list:

```
$ swarm –f file.swarm --module python,samtools,bwa,tophat
```

--gres

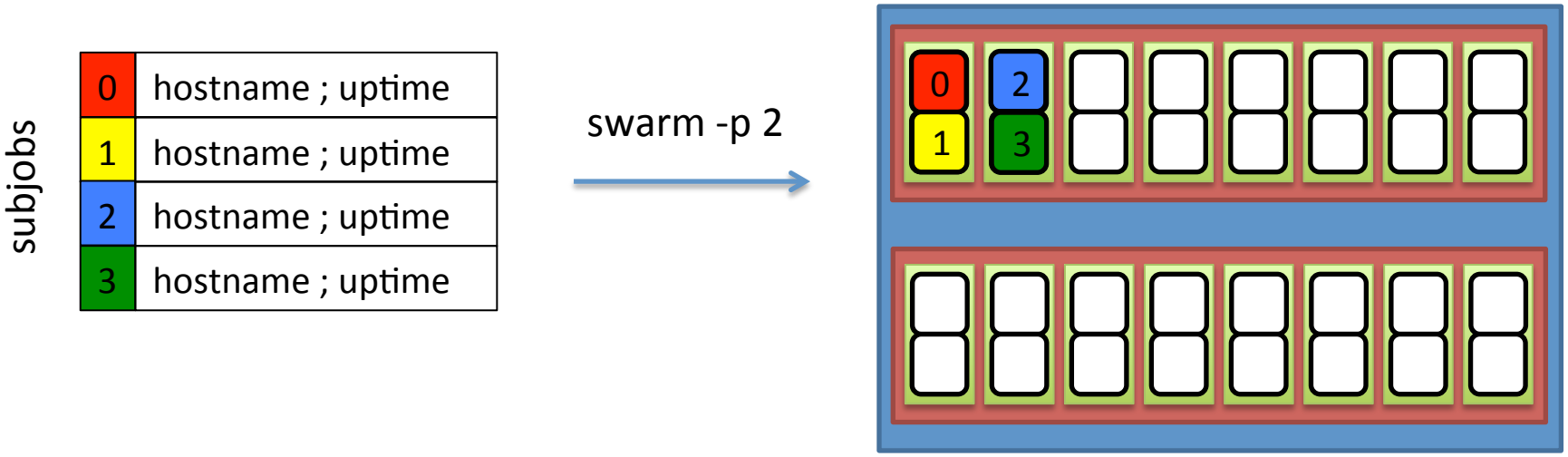
- --gres stands for "generic resources"
- Is used for allocating local disk space
- Replaces --disk

```
$ swarm -f file.swarm --gres=lscratch:50
```

- /lscratch/$SLURM_JOBID
- Above example gives 50GB of scratch space
- See https://hpc.nih.gov/docs/userguide.html#local

-p, --processes-per-subjob

- For single-threaded commands
- -p can only be 1 or 2



--logdir

- Redirects .o and .e files to a directory
- The directory must first exist

```
$ mkdir /data/user/trashbin
$ swarm –f files.swarm --logdir /data/user/trashbin
1234567
$ ls
file.swarm
$ ls /data/user/trashbin
swarm_1234567_0.o    swarm_1234567_1.o    swarm_1234567_2.o
swarm_1234567_0.e    swarm_1234567_1.e    swarm_1234567_2.e
```

--time

- All jobs must have a walltime now
- --time for swarm is per command, not per swarm or per subjob
- --time multiplied by bundle factor

```
$ swarm -f file.swarm --devel --time=01:00:00
32 commands run in 32 subjobs, each command requiring 1.5 gb and 1 thread, allocating 32
cores and 64 cpus
sbatch --array=0-31 ... --time=01:00:00 /spin1/swarm/hooverdm/iMdaW6dO.batch
$ swarm -f file.swarm --devel --time=01:00:00 -b 4
32 commands run in 8 subjobs, each command requiring 1.5 gb and 1 thread, running 4
processes serially per subjob
sbatch --array=0-7 ... --time=04:00:00 /spin1/swarm/hooverdm/zYrUbkiO.batch
```

Primary sbatch options

- --job-name
- --dependency
- --time, --gres
- --partition
- --qos

ALL sbatch options

- --sbatch

```
$ swarm -f file.swarm --sbatch "--mail-type=FAIL --
export=var=100,nctype=12 --workdir=/data/user/test"
```

- Type 'man sbatch' for more information

--prologue and --epilogue

- way too difficult to implement
- conflicts with --prolog and --epilog options to srun

--W block=true

- sbatch does not allow blocking
- must use srun instead

-R, --resource

- gpfs is available on all nodes
- Replaced by a combination of --license, --gres, and --constraint

Examples

- single-threaded commands

```
$ swarm –f file.swarm
```

- multi-threaded commands

```
$ swarm –f file.swarm –t 4
```

- large memory, single-threaded

```
$ swarm –f file.swarm –g 32
```

Examples

- >10K single-threaded commands

```
$ mkdir /data/user/bigswarm
$ swarm –f file.swarm --job-name bigswarm --logdir /data/
user/bigswarm
```

- wait for it and deal with the output …

```
$ cd /data/user/bigswarm
$ cat *.e > bigswarm.err
$ cat *.o > bigswarm.out
$ rm *.{e,o}
```

Examples

- Large temp files

```
export TMPDIR=/lscratch/$SLURM_JOBID ; command -opt 1
export TMPDIR=/lscratch/$SLURM_JOBID ; command -opt 2
export TMPDIR=/lscratch/$SLURM_JOBID ; command -opt 3
export TMPDIR=/lscratch/$SLURM_JOBID ; command -opt 4
```

```
$ swarm -f file.swarm --gres=lscratch:200
```

Examples

- Dependencies

```
$ sbatch script_1.sh
10000
$ swarm -f file.swarm --dependency=afterany:10000
10001
$ swarm -f file2.swarm --dependency=afterany:10001
10002
$ sbatch sbatch_2.sh --dependency=afterany:10002
10003
```

Examples

- Long-running processes

```
$ swarm -f file.swarm --time=4-00:00:00
```

Defaults and Limits

- 1,000 subjobs per swarm
- 4,000 jobs per user max
- 30,000 jobs in slurm max
- 1.5 GB/process default, 1 TB max
- 0 GB/disk, 800 GB max
- 4 hours walltime, 10 days max
- batchlim and freen

Monitoring Swarms

- current and running jobs:
  - squeue
  - sjobs
  - jobload
- historical
  - sacct
  - jobhist

Stopping Swarms

- scancel

Complex Examples

- Rerunnable swarm

```
[[ -e file1.flag ]] || ( command1 && touch file1.flag )
[[ -e file2.flag ]] || ( command2 && touch file2.flag )
[[ -e file3.flag ]] || ( command3 && touch file3.flag )
[[ -e file4.flag ]] || ( command4 && touch file4.flag )
```

- -e tests if the file exists

Complex Examples

- Very long command lines

```
cd /data/user/project; KMER="CCCTAACCCTAACCCTAA"; \
jellyfish count -C -m ${#KMER} \
-t 32 \
-c 7 \
-s 1000000000 \
-o /lscratch/$SLURM_JOBID/39sHMC_Tumor_genomic \
<(samtools bam2fq /data/user/bam/0A4HMC/DNA/genomic/39sHMC_genomic.md.bam ); \
echo ${KMER} | jellyfish query /lscratch/$SLURM_JOBID/39sHMC_Tumor_genomic_0 \
> 39sHMC_Tumor_genomic.telrpt.count
```

Complex Examples

- Comments

```
# This is for the first file

command -i infile1 -o outfile1 -p 2 -r /path/to/file

# This is for the next file

command -i infile2 -o outfile2 -p 2 -r /path/to/another/file
```

- --comment-char, --no-comment

Complex Examples

- Environment variables
- Defined BEFORE the job; passed to the swarm:

```
$ swarm –f file.swarm --sbatch "--export=FILE=/path/to/
file,DIR=/path/to/dir,VAL=12345"
```

- Defined WITHIN the job; part of the swarmfile

```
export TMPDIR=/lscratch/$SLURM_JOBID ; command –opt 1
export TMPDIR=/lscratch/$SLURM_JOBID ; command –opt 2
export TMPDIR=/lscratch/$SLURM_JOBID ; command –opt 3
export TMPDIR=/lscratch/$SLURM_JOBID ; command –opt 4
```

Blank Space for More Examples

Questions? Comments?

staff@helix.nih.gov