Job Scheduling on IBEX

Mohsin Ahmed Shaikh

Computational Scientists @ KSL

What is a Job Scheduler?

- HPC resources at KSL must be busy at maximum capacity including nights and weekends
  - Fair distribution of resource
- How we achieve it:
  - Queues and schedulers help increase utilization
- You may need to change your workflow:
  - You may need to wait for your turn because resources in use
  - Automate where ever possible for scheduler to run on your behalf
  - Have your work queued to maximize utilization

Ibex Cluster

- Heterogeneous CPU architecture
  - ✓ Intel Cascade Lake – 106 nodes (40 cores, 2 sockets per node)
  - ✓ Intel Skylake – 106 nodes (40 cores, 2 sockets per node)
  - ✓ AMD EPYC ROME – 108 nodes (128 cores, 2 sockets per node)
- **He**terogeneous GPU architecture
  - ✓ Volta (v100) - 38 nodes (4 or 8 GPUs cards per node)
  - ✓ Turing (rtx2080ti) - 4 nodes (8 GPUs cards per node)
  - ✓ Pascal (gtx1080ti, P100, P6000) - 19 nodes (2-8 GPUs per node)
- Large memory nodes
  - ✓ 3 TB nodes – 18 nodes (32- 48 cores per node, Intel Skylake and Cascadelake)

SLURM

- A resource manager
  - Manages more work than the resource by scheduling queues of work
- Supports complex scheduling of algorithms

- Provides:
  - Way to describing and submitting a resource request
  - Way to prescribing how to run workload on allocated resource
  - Way to monitoring the state of the submitted ”job”
  - Way to account for the resources used (charging system)

Querying system resources

sinfo

- A concise view of the system resources and their state/availability

```
> sinfo

PARTITION   AVAIL  TIMELIMIT  NODES  STATE NODELIST

……..

batch*         up 14-00:00:0     44  alloc cn509-03-l,cn509-03-r,cn509-04-l,cn509-04-r,cn509-09-l,cn509-09-
r,cn509-19-l,cn509-20-r,cn509-22-r,cn509-23-r,cn509-29-r,cn512-05-r,cn512-08-l,cn512-16-l,cn603-04-r,cn603-05-
l,cn603-05-r,cn603-06-l,cn603-08-r,cn603-09-l,cn603-20-l,cn603-20-r,cn603-26-l,cn605-10-l,cn605-15-r,cn605-16-
l,cn605-17-l,cn605-19-r,cn605-25-l,dgpu501-26,gpu502-11,lm508-[02,04,06,08,10,12],lm602-[10,12,14,16,18,20,24]

batch*         up 14-00:00:0     38   idle cn509-11-l,cn509-11-r,cn509-12-l,cn509-16-l,cn512-03-r,cn512-04-
l,cn512-07-r,cn512-12-r,cn512-13-r,cn512-14-l,cn512-14-r,cn512-26-r,cn603-08-l,cn603-11-r,cn603-14-r,cn603-15-
r,cn603-18-l,cn603-26-r,cn603-29-r,cn605-04-r,cn605-05-l,cn605-05-r,cn605-06-l,cn605-06-r,cn605-11-l,cn605-12-
r,cn605-16-r,cn605-21-r,cn605-22-l,cn605-22-r,cn605-23-l,cn605-23-r,cn605-24-l,cn605-26-r,cn605-27-l,dgpu501-
14,gpu510-12,lm602-08

debug          up    2:00:00      3   idle cn603-14-l,dgpu609-14,gpu510-32

……..

gpu4           up    4:00:00      7  alloc dgpu501-26,gpu210-10,gpu211-10,gpu213-[02,18],gpu214-14,gpu502-11

gpu4           up    4:00:00      2   idle dgpu501-14,gpu510-12
```

ginfo

- An in-house tool developed to query the status of GPU resources on Ibex cluster

```
> ginfo
GPUS currently in use:
GPU Models:          Total  Used  Free
gtx1080ti               64    60     4
p100                    12     9     3
p6000                    4     2     2
rtx2080ti               32    15    17
v100                   274   226    48
             Total:    386   312    74
```

```
> ginfo --help
```

squeue

- Shows the list of jobs in the queue along with information about the request and its current state

```
> squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           9204779     batch NO_MoSe2 nadhrega  R   14:46:40      1 dbn404-16-r
           9118758  gpu_wide     bash     zhup  R 2-22:03:07      1 gpu214-14
           9118404     batch     bash  zhanb0b  R 2-22:26:04      1 gpu208-02
           9195595     batch dtchFlow   parawr  R   19:13:39      1 gpu510-12
           9194968     batch      aug   parawr  R   19:29:43      1 gpu510-07
         9234033_7     batch   s3knn2    qiang  R    1:42:16      1 gpu211-02
         9234033_8     batch   s3knn2    qiang  R    1:42:16      1 gpu211-02
         9234033_9     batch   s3knn2    qiang  R    1:42:16      1 gpu211-02
        9234033_10     batch   s3knn2    qiang  R    1:42:16      1 gpu609-02
           9087308     batch Haplotyp alsaedsb PD       0:00      1 (DependencyNeverSatisfied)
           9087317     batch Haplotyp alsaedsb PD       0:00      1 (DependencyNeverSatisfied)
```

- You can use filters on the list:
  - -u $USERNAME – show jobs by a user
  - -p $PARTITION – show jobs on a specific partition
  - -j $JOBID – show status of a specific job

Specifying resources

Resource pools

- Partitions
  - Queues with names adhering to a scheduling policy e.g.
    - batch, debug
- Features/Constraints
  - Some partitions have types of one resources, e.g.
    - CPU architectures
    - GPU architecture
    - collections of node with various amount of memory

```
> sinfo -Ne -o %N,%t,%f
NODELIST,STATE,AVAIL_FEATURES
cn509-03-1,mix,dragon,cascadelake,cpu_intel_gold_6248,intel,ibex2019,nogpu,nolmem,local_200G,local_400G,local_500G,local_950G
……
gpu208-02,mix,dragon,ibex2018,nolmem,cpu_intel_platinum_8260,intel,gpu,intel_gpu,local_200G,local_400G,local_500G,gpu_v100,v100,nossh,ref_32T,gpu_ai
```

Requestable resources

- CPUs
- GPUs
- Memory
- Wall time

Requesting resource

- You can either run your jobs in batch mode or interactive mode:
- Implications:
- Batch mode
  - You will need a script with resource request and the commands to run on that resource once allocated
  - Scheduler will run the script on your behalf once the requested resources are available
  - Resources can be requested for longer durations (several hours)
- Interactive
  - Resource requests are usually small and short
  - You run each command by typing it interactively
  - Useful for prototyping and debugging

JobScript (Ibex)

```
--------- jobscirpt.slurm ------
#!/bin/bash -l

#SBATCH --job-name=myfirstjob
#SBATCH --time=01:00:00
#SBATCH --partition=batch
#SBATCH --ntasks=1
#SBATCH --gres=gpu:1
#SBATCH --constraint=v100
module load cuda
srun ./helloworld
-------------------------------
```

> **sbatch** jobscript.slurm

sbatch

- Command to submit your **jobscript** to SLURM:
- Upon successful submission a unique job ID is assigned
- Job is queued and awaits allocation of the requested resources
- On Ibex jobs accounting is not enabled. Although a priority is assigned to each job, it is not used. This may change in future.

- In general, short and small jobs are to schedule

scancel

- SLURM command to cancel a queued job:

```
> squeue -u $USER
      JOBID       USER ACCOUNT           NAME  ST REASON    START_TIME  TIME  TIME_LEFT NODES
   13723548   shaima0d     k01 jobscript.slur  PD Priority  N/A         0:00      10:00     2

> scancel 13723548

> squeue -u $USER
      JOBID       USER ACCOUNT           NAME  ST REASON    START_TIME  TIME  TIME_LEFT NODES
```

salloc (Ibex)

- Command to request allocation of resource for interactive use:
  - Primary be used for prototyping and/or debugging your workflow

```
> salloc -p debug --ntasks=1 --gres=gpu:1 --constraint=v100 -t 00:10:00
salloc: Pending job allocation 7329981
salloc: job 7329981 queued and waiting for resources
salloc: job 7329981 has been allocated resources
salloc: Granted job allocation 7329981
salloc: Waiting for resource configuration
salloc: Nodes gpu104-12 are ready for job
> module load cuda
> srun -n 1 deviceQuery/deviceQuery
deviceQuery/deviceQuery Starting...
 CUDA Device Query (Runtime API) version (CUDART static linking)
Detected 1 CUDA Capable device(s)
Device 0: "Tesla V100-PCIE-32GB"
  CUDA Driver Version / Runtime Version          10.1 / 9.2
  CUDA Capability Major/Minor version number:    7.0
  ……
> exit
```

Using allocated resources

srun

- **Once allocated**, `srun` command can be used to launch your application on to the compute resources

```
> salloc --partition=debug --ntasks=4 –cpus-per-task=10 --time=00:10:00
  salloc: Granted job allocation 12140840
> srun -n 1 –c 1 ./pre_processing_step
> export OMP_NUM_THREADS=10
> srun –n 4 –c 10 ./solver
> exit
  salloc: Relinquishing job allocation 11762274
  salloc: Job allocation 11762274 has been revoked.
```

- Each `srun` command is considered as "job step" for the corresponding allocation by SLURM
- When a job step completes, the allocation does not automatically terminate
- This means you can run multiple job steps with different configurations.

Monitoring and account your jobs

squeue

- You can query the state of your job using squeue
- Common filters include
  - by user
  - by partition

```
> squeue -u alsaedsb
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           9087308     batch Haplotyp alsaedsb PD       0:00      1 (DependencyNeverSatisfied)
           9087317     batch Haplotyp alsaedsb PD       0:00      1 (DependencyNeverSatisfied)
           9197195     batch Joint-GV alsaedsb  R   18:38:25      1 dbn302-31-r
```

```
> squeue -j 9197195
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           9197195     batch Joint-GV alsaedsb  R   18:39:17      1 dbn302-31-r
```

scontrol

- `scontrol` command, among other things, allows user to show parameters of request and allocated resource for a job in queue (in any state i.e running, pending, etc)

```
> scontrol show job 9197195
   JobId=9197195 JobName=Joint-GVCFs
   UserId=alsaedsb(157323) GroupId=g-alsaedsb(1157323) MCS_label=N/A
   Priority=81 Nice=0 Account=default QOS=normal
   JobState=RUNNING Reason=None Dependency=(null)
   Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
   RunTime=18:40:26 TimeLimit=1-00:00:00 TimeMin=N/A
   SubmitTime=2020-02-08T17:07:42 EligibleTime=2020-02-08T17:07:42
   AccrueTime=2020-02-08T17:07:42
   StartTime=2020-02-08T17:07:43 EndTime=2020-02-09T17:07:43 Deadline=N/A
   ……
   NumNodes=1 NumCPUs=16 NumTasks=1 CPUs/Task=16 ReqB:S:C:T=0:0:*:*
   TRES=cpu=16,mem=115G,node=1,billing=16
   Socks/Node=* NtasksPerN:B:S:C=0:0:*:* CoreSpec=*
   MinCPUsNode=16 MinMemoryNode=115G MinTmpDiskNode=0
   Features=intel&nolmem&nogpu DelayBoot=00:00:00
   OverSubscribe=OK Contiguous=0 Licenses=(null) Network=(null)
   Command=(null)
   WorkDir=/encrypted/genomics/Sakhaa/Saudi_Population/Spring/Scripts
```

sacct

- Displays accounting command which tells about the resources used by the job and its job steps.

```
> sacct -u shaima0d
       JobID    JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
9230749      jupyter_s+      debug    default         36     FAILED      1:0
9230752      jupyter_s+      debug    default         16    TIMEOUT      0:0
9230752.bat+      batch               default         16  CANCELLED     0:15
9230752.ext+     extern               default         16  COMPLETED      0:0
```

- The accounting information persists after the life of the job
- Common filters include `-u` for "by user" and `-j` for "by jobID"

Example Jobscripts

Example – OpenMP jobs on Ibex

- A jobscript running an OpenMP code on a Ibex with 4 OpenMP threads
  - NOTE: nodes are shared on Ibex (must define the request properly)

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=4
#SBATCH --constraint=intel

module load gcc/6.4.0

export OMP_NUM_THREADS=4
export OMP_PLACES=cores
export OMP_PROC_BIND=close
srun -c 4 ./helloworld_omp
```

Example – MPI jobs on Ibex

- A jobscript running MPI code on a Ibex with 32 MPI tasks on same node
  - NOTE: nodes are shared on Ibex (must define the request properly)

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --ntasks=32
#SBATCH --ntasks-per-node=32
#SBATCH --constraint=intel

module load gcc/6.4.0
module load openmpi
srun -n 32 ./helloworld_mpi
```

Example – large memory jobs on Ibex

- Normal compute nodes have memory up to ~ **360GB** per node.
- “large memory job” is a label that’s assigned to your job by SLURM if you ask for memory => **370G**

- Use `--mem=####G` to request nodes with large memory.
- Upon submission via `sbatch` or `salloc`, SLURM will notify the following message:

```
sbatch: job tagged as large memory
```

- When you don't specify `--mem`, the **default memory** allocation will be **2GB**

Example – 1 GPU jobs on Ibex

- Running a CUDA code on a single GPU
  - NOTE: nodes are shared on Ibex (must define the request properly)
  - All GPU jobs should be submitted from `glogin.ibex.kaust.edu.sa` login node

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --ntasks=1
#SBATCH --gres=gpu:1
#SBATCH --constraint=v100
#SBATCH --mem=64G

module load cuda/10.2.89

srun ./helloworld_cuda
```

Example – 4 GPUs jobs on Ibex

- Running a CUDA code on 4 GPUs on single Ibex node
  - NOTE: nodes are shared on Ibex (must define the request properly)
  - All GPU jobs should be submitted from `glogin.ibex.kaust.edu.sa` login node

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --ntasks=4
#SBATCH --gres=gpu:4
#SBATCH --constraint=v100
#SBATCH --mem=64G

module load cuda/10.2.89

srun -n 4 ./helloworld_multigpu
```

Common Constraints on Ibex

- For CPU only jobs, use jobscript generator: https://www.hpc.kaust.edu.sa/ibex/job
- You may use **`sinfo -o %N,%f`** to list constraints for each node in Ibex
- For advice on templates for GPU jobs (DL training) please attend the afternoon session on **Deep Learning Best Practices**

- **Any Intel architecture:**
  `#SBATCH --constraint=intel`
- **Intel Skylake:**
  `#SBATCH --constraint=skylake`
- **Intel Cascade Lake:**
  `#SBATCH --constraint=cascadelake`
- **Any AMD architecture:**
  `#SBATCH --constraint=amd`
- **AMD ROME:**
  `#SBATCH --constraint=rome`

- **Any GPU Architecture:**
  - `#SBATCH --gres=gpu:1`
  - `#SBATCH --constraint=gpu`
- **Volta V100, 8 GPUs per node**
  - `#SBATCH --gres=gpu:8`
  - `#SBATCH --constraint=v100`
- **Large Memory**
  - `#SBATCH --mem=2T`
- **Local storage**
  - `#SBATCH --constraint=local_500G`

Why my job is not running?

- When the estimated start time of your pending job is not available, you can get more details and reasons for your job not running:

- By typing *squeue --job <jobid >–l* , you will get the following output along with the reason for your job not running.

```
> squeue -j 9235760 -l
Sun Feb  9 11:57:39 2020
             JOBID PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
           9235760     batch     vasp  agapevk  PENDING       0:00 3-00:00:00      4 (Resources)
```

- To understand the reason more:

```
> man squeue
```

And search for **JOB REASON CODES**

Resources The job is waiting for resources to become available.