جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Core Labs and Research Infrastructure

Job Scheduling on IBEX

Mohsin Ahmed Shaikh

Computational Scientists @ KSL

KSL Systems -- Ibex Cluster

- **Heterogeneous CPU architecture**
  - ✓ Intel Skylake – 106 nodes (32-40 cores, 2 sockets per node)
  - ✓ Intel Cascade Lake – 106 nodes (48 cores, 2 sockets per node)
  - ✓ AMD Rome – 108 nodes (128 cores, 2 sockets per node)
- **He**terogeneous GPU architecture
  - ✓ Ampere (A100) - 44 nodes (4 or 8 GPUs cards per node)
  - ✓ Volta (v100) - 38 nodes (4 or 8 GPUs cards per node)
  - ✓ Turing (rtx2080ti) - 3 nodes (8 GPUs cards per node)
  - ✓ Pascal (gtx1080ti, p100, p6000) - 20 nodes (2-8 GPUs per node)
- Large memory nodes
  - ✓ 3 TB nodes – 18 nodes (32- 48 cores per node, Intel Skylake and Cascadelake)
  - ✓ 2 TB nodes – 3 nodes (32 – 80 cores per nodes, Intel Skylake and Westmear)
  - ✓ 750GB – 1.5TB nodes – 13 nodes (64 cores per node, AMD Abu Dhabi)

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

On Ibex:

```
> sinfo -Nel
Mon Dec  9 13:11:27 2019
NODELIST      NODES   PARTITION       STATE CPUS    S:C:T MEMORY TMP_DISK WEIGHT AVAIL_FE REASON
besest514-03      1      batch*       mixed   80   8:10:1 154000        0   1540 ibex2017 none
cn603-02-l        1      batch*       mixed   40   2:20:1 375618        0    100 dragon,c none
cn603-02-r        1      batch*       mixed   40   2:20:1 375618        0    100 dragon,c none
cn603-03-l        1      batch*   allocated   40   2:20:1 375618        0    100 dragon,c none
cn603-03-r        1       c2034   allocated   40   2:20:1 375618        0    100 dragon,c none
…..
sdlm111-20        1      batch*        idle   64   4:16:1 101423        0   1014 ibex2017 none
sdlm111-22        1      batch*        idle   64   4:16:1 101423        0   1014 ibex2017 none
sdlm112-18        1      batch*     drained   64   4:16:1 510000        0    511 ibex2017 ZHARDWARE:[2019-11-2
```

ginfo

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

```
> ginfo -d
GPU Model          Used    Idle   Drain    Down   Maint   Total
gtx1080ti            22      35       7       0       0      64
p100                  8      12       0       0       0      20
p6000                 2       2       0       0       0       4
rtx2080ti             9      15       0       0       0      24
v100                208      55      11       0       0     274
       Totals:      249     119      18       0       0     386
```

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)
          22067281     batch      HRL      bob  R    3:59:57      1 gpu214-10
          22067282     batch      HRL      bob  R    3:59:57      1 gpu214-06
          22067280     batch      HRL      bob  R    4:00:00      1 gpu214-18
          22067276     batch      HRL      joe  R    4:00:03      1 gpu609-08
      22068721_833     batch   b2test     jane  R       1:42      1 cn513-21-l
      22068721_834     batch   b2test     jane  R       1:42      1 cn513-21-l
```

- 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 pariticular job

Specifying resources

Requestable resources

- CPUs / GPUs
- Memory
- Local disk space
- 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)
  - Remote servers (e.g. Jupyter/R/VS code servers) should also be run as batch job
- Interactive
  - Resource requests are usually small and short
  - You run each command by typing it interactively
  - Useful for prototyping and debugging

Example jobscript

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

#SBATCH --job-name=myfirstjob
#SBATCH --time=04:00:00
#SBATCH --partition=batch
#SBATCH --ntasks=1
#SBATCH --gpus=1
#SBATCH --constraint=v100
module load cuda
./my_application [blah]
-------------------------------

> sbatch jobscript.slurm
```

Example jobscript

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

#SBATCH --job-name=myfirstjob
#SBATCH --time=04:00:00
#SBATCH --partition=batch
#SBATCH --ntasks=1
#SBATCH --gpus=1
#SBATCH --constraint=v100
module load cuda
./my_application [blah]
--------------------------------
```

```
> sbatch jobscript.slurm
```

sbatch

- Command to submit your **jobscript** to SLURM:
- Returns a unique job ID is assigned
- Job status Pending (PD) until resources are available
- On Ibex jobs accounting is not enabled.
  - Although a priority is assigned to each job, it is not used.
  - This may change in near future.
- In general, short and small jobs are to schedule

salloc

- Command to request allocation of resource for interactive use
- Primarily used for prototyping or debugging a workflow

```
> salloc --gpus=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
```

salloc

> 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
  Total amount of global memory:                 32480 MBytes (34058272768 bytes)
  (80) Multiprocessors, ( 64) CUDA Cores/MP:     5120 CUDA Cores
….
  deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 10.1, CUDA Runtime
Version = 9.2, NumDevs = 1
Result = PASS
```

scancel

- SLURM command to cancel a queued job:

```
> squeue –u $USER
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          22073487     batch test.slu shaima0d  R       0:13      1 cn513-10-l

> scancel 22073487

> squeue –j 22073487
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
```

Using allocated resources

srun

- In a jobscript, srun can be used to configure how allocated resources are used

For example:

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --ntasks=2
#SBATCH --cpus-per-task=4

module load gcc/6.4.0
srun -n 1 -c 1 ./my_single_thread_app
export OMP_NUM_THREADS=4
srun -n 2 -c 4 ./my_multithreaded_app
```

Monitoring and account your jobs

squeue

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

```
> 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, amongst 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
   SuspendTime=None SecsPreSuspend=0 LastSchedEval=2020-02-08T17:07:43
   Partition=batch AllocNode:Sid=dbn503-33-r.ibex.kaust.edu.sa:6787
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=dbn302-31-r
   BatchHost=dbn302-31-r
   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
   StdErr=/encrypted/genomics/Sakhaa/Saudi_Population/Spring/Saudi_KSU_pop/LOGS/Joint-GVCFs.%J.err
   StdIn=/dev/null
   StdOut=/encrypted/genomics/Sakhaa/Saudi_Population/Spring/Saudi_KSU_pop/LOGS/Joint-GVCFs.%J.out
```

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

module load gcc/6.4.0

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

Example – MPI jobs on Ibex

- A jobscript running MPI code on 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

module load openmpi/4.0.3
module load gcc/11.2.2
mpirun -n 32 ./my_mpi_application
```

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.
  - ✓ When you don't specify `--mem`, the **default memory** allocation will be **2GB**
  - ✓ Upon submission via `sbatch` or `salloc`, SLURM will notify the following message:
  - ✓ `sbatch`: job tagged as large memory

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)
  - Submit from `glogin.ibex.kaust.edu.sa` login node

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --gpus=1

module load cuda/11.2.2
./my_cuda_app
```

Example – GPU jobs with Conda environments

- Pythonic CUDA codes installed via conda environments can also use GPUs

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --gpus=1

source ~/miniconda3/bin/activate my_cuda_env
python train.py
```

Example – multi-GPU on one node

- Running a CUDA code on 4 GPUs on single Ibex node

```
#!/bin/bash
#SBATCH --time=00:10:00
#SBATCH --cpus-per-task=4
#SBATCH --mem=64G
#SBATCH --gpus=4
#SBATCH --gpus-per-node=4
#SBATCH --constraint=v100

source ~/miniconda3/bin/activate my_cuda_env
srun -c 4 python train.py
```

Common Constraints on Ibex

For CPU only jobs use **jobscript generator:** https://www.hpc.kaust.edu.sa/ibex/job

For advice on templates for GPU jobs (DL training) please attend the Data-science On-boarding

- **Any Intel Architecture:**
  - `#SBATCH --constraint=[intel]`
- **Intel Skylake:**
  - `#SBATCH --constraint=[cpu_intel_gold_6148]`
- **Intel IveyBridge:**
  - `#SBATCH --constraint=[cpu_intel_e5_2680_v2|cpu_intel_e5_2670_v2]`
- **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`

Know your workload

- Test how long you job takes before submitting multiple jobs
  - `time –p python train.py`
- Test
  - How much memory does your workload requires
  - How many CPUs does your worload requires
    - Do a scaling test

CPUs	Time to solution (seconds)
2	1800
4	900
8	500
16	375
32	350

Contact us !

- For any issue, contact the team:
  - Please share the job id, the system, the error message....
  - For Ibex ibex@hpc.kaust.edu.sa

- Check our training website : hpc.kaust.edu.sa/ibex/training

Thanks !

Q&A