Advanced Resource Computing (ARC) Best Practices.

Wiki page: <https://rcs.ucalgary.ca/ARC_Cluster_Guide>

Notes: The wiki page has not been updated in awhile and is used as supplementary material for the construction of this document The following ARC best practices document is used as a resource to navigate the slurm HPC scheduler on ARC.

**Login node**

The login node can be used for short tasks such as copying, editing, or moving files, compiling programs and running short tests for developing programs. It is suggested that CPU intensive workloads are restricted to 15 minutes per ARC guidelines. For workloads greater than 15 minutes an interactive

The ARC login node may be used for such tasks as editing files, compiling programs and running short tests while developing programs. We suggest CPU intensive workloads on the login node be restricted to under 15 minutes as per [our cluster guidelines](https://rcs.ucalgary.ca/General_Cluster_Guidelines_and_Policies). For interactive workloads exceeding 15 minutes, use the [**salloc command**](https://rcs.ucalgary.ca/Running_jobs#Interactive_jobs) to allocate an interactive session on a compute node.

ARC Partitions and Resources

Commands to list some useful information of ARC resources.

arc.hardware

Description: This commands lists the partitions, number of nodes per partition, number of cpus per node, amount of memory per node in MB, and GPUs per node

output:

Node specifications per partition:

================================================================================

Partition | Node CPUs Memory GPUs Node list

| count /node (MB) /node

--------------------------------------------------------------------------------

bigmem | 2 80 3000000 fm[1-2]

| 1 40 4127000 a100:4 mmg1

| 1 40 8256000 a100:2 mmg2

cpu2013 | 14 16 120000 h[1-14]

cpu2017-bf05 | 16 28 245000 s[1-16]

| 20 28 188000 th[1-20]

cpu2019 | 40 40 185000 fc[22-61]

cpu2019-bf05 | 87 40 185000 fc[1-21,62-127]

cpu2021 | 17 48 185000 mc[1-11,14-19]

cpu2021-bf24 | 21 48 185000 mc[23-43]

| 7 48 381000 mc[49-55]

cpu2022 | 52 52 256000 mc[73-124]

cpu2022-bf24 | 16 52 256000 mc[57-72]

gpu-a100 | 6 40 515000 a100:2 mg[1-6]

gpu-v100 | 13 40 753000 v100:2 fg[1-13]

lattice | 196 8 11800 cn[169-364]

parallel | 572 12 23000 cn[0513-0544,0557-1096]

| 4 12 23000 m2070:2 cn[0553-0556]

single | 168 8 11800 cn[001-168]

synergy | 16 28 245000 s[1-16]

================================================================================

15 | 1253

arc.limits

Description: This command is useful for figuring out the Max Wall time (MaxWall) of each partition.

output:

PartitionName Flags MaxTRES MaxWall MaxTRESPU MaxSubmitPU MinTRES

0 normal 7-00:00:00 4000

1 cpu2019 cpu=240 7-00:00:00 cpu=240 4000

2 gpu-v100 DenyOnLimit cpu=80,gpu=4 1-00:00:00 cpu=160,gpu=8 4000 gpu=1

3 single cpu=200 7-00:00:00 cpu=200,node=30 4000

4 razi 7-00:00:00 4000

5 apophis 14-00:00:00 4000

6 razi-bf cpu=546 05:00:00 cpu=546 4000

7 apophis-bf cpu=280 05:00:00 cpu=280 4000

8 lattice cpu=408 7-00:00:00 cpu=408 4000

9 parallel cpu=624 7-00:00:00 cpu=624 4000

10 bigmem cpu=80 1-00:00:00 cpu=80,gpu=1 4000

11 cpu2013 7-00:00:00 4000

12 pawson 7-00:00:00 4000

13 pawson-bf cpu=480 05:00:00 cpu=480 4000

14 theia 7-00:00:00 4000

15 theia-bf cpu=280 05:00:00 4000

16 demo 7-00:00:00 4000

17 synergy 7-00:00:00 4000

18 synergy-bf cpu=448 05:00:00 cpu=448 4000

19 backfill05 cpu=1000 05:00:00 cpu=1000 4000

20 cpu2021 cpu=576 7-00:00:00 cpu=576 4000

21 backfill24 cpu=208 1-00:00:00 cpu=208 4000

22 sherlock 7-00:00:00 4000

23 wdf-zach 7-00:00:00 4000

24 wdf-think 7-00:00:00 4000

25 mtst 7-00:00:00

26 cpu2022 cpu=520 7-00:00:00 cpu=520 4000

27 gpu-a100 DenyOnLimit cpu=80 1-00:00:00 cpu=160

TRES=Trackable RESources

PU=Per User

arc.jobs

Description: This command displays the number of jobs running on each partition.

Output:

Jobs per partition:

==========================================================================================================

Partition | CPUs Jobs Users Wait MaxWait Effort Effort\_PD

| online Total Running Pending Completed Other (hrs) (hrs) (hrs) (hrs)

----------------------------------------------------------------------------------------------------------

bigmem | 240 5 2 3 0 0 4 0.0 0.0 13.7 13.1

cpu2013 | 224 0 0 0 0 0 0 0.0 0.0 0.0 0.0

cpu2017-bf05 | 1008 0 0 0 0 0 0 0.0 0.0 0.0 0.0

cpu2019 | 1600 335 98 235 2 0 17 0.5 6.1 183.7 90.8

cpu2019-bf05 | 3480 48 3 44 0 1 4 0.6 1.9 7.6 6.4

cpu2021 | 816 261 69 189 3 0 11 20.2 98.6 192.2 100.3

cpu2021-bf24 | 1344 218 12 186 20 0 4 0.0 0.3 60.4 56.8

cpu2022 | 2704 330 127 201 1 1 16 10.9 168.1 105.2 61.0

cpu2022-bf24 | 832 188 2 186 0 0 4 11.6 23.2 103.3 91.7

gpu-a100 | 240 1 1 0 0 0 1 0.0 0.0 0.1 0.0

gpu-v100 | 520 27 13 14 0 0 6 5.4 11.5 21.3 12.3

lattice | 1568 674 138 515 21 0 3 0.8 32.4 107.9 63.1

parallel | 6912 153 140 0 13 0 6 8.0 119.0 46.3 0.0

single | 1344 564 41 515 8 0 3 1.0 4.1 79.2 73.6

synergy | 448 188 17 161 10 0 3 56.6 169.9 279.7 181.2

----------------------------------------------------------------------------------------------------------

Total | 1544 668 796 78 2 52

Pending jobs:

==================================================================

Reason | Jobs Users Wait MaxWait

| (hrs) (hrs)

------------------------------------------------------------------

DependencyNeverSatisfied | 28 2 725.9 1665.5

JobArrayTaskLimit | 2 2 137.9 173.0

None | 675 3 0.9 2.0

PartitionTimeLimit | 3 2 262.4 628.0

Priority | 50 6 19.0 106.8

QOSMaxCpuPerUserLimit | 36 5 9.7 20.9

Resources | 2 2 0.3 0.5

------------------------------------------------------------------

Total / Avg | 796 17 29.3

arc.nodes

Description: This command lists the number of nodes allocated for wach partition.

Output:

Partitions: 15 (bigmem, cpu2013, cpu2017-bf05, cpu2019, cpu2019-bf05, cpu2021, cpu2021-bf24, cpu2022, cpu2022-bf24, gpu-a100, gpu-v100, lattice, parallel, single, synergy)

=======================================================================================

| Total Allocated Down Drained Draining Idle Mixed Reserved

---------------------------------------------------------------------------------------

bigmem | 4 0 2 0 0 0 2 0

cpu2013 | 14 0 0 0 0 14 0 0

cpu2017-bf05 | 36 7 0 0 0 19 10 0

cpu2019 | 40 28 4 0 0 0 8 0

cpu2019-bf05 | 87 40 3 27 5 11 1 0

cpu2021 | 17 1 0 0 0 0 16 0

cpu2021-bf24 | 28 0 0 0 1 17 10 0

cpu2022 | 52 3 0 8 0 0 41 0

cpu2022-bf24 | 16 4 0 0 0 8 4 0

gpu-a100 | 6 0 0 0 0 5 1 0

gpu-v100 | 13 0 1 1 0 0 11 0

lattice | 196 91 2 15 0 2 86 0

parallel | 576 196 71 121 0 97 87 4

single | 168 26 0 14 0 112 15 1

synergy | 16 7 0 0 0 0 9 0

---------------------------------------------------------------------------------------

logical total | 1269 403 83 186 6 285 301 5

|

physical total | 1253 396 83 186 6 285 292 5

**Interactive jobs**

Commands

salloc --time 5:00:00 --partition cpu2019 -n 8 --mem 38G

**salloc** --time 5:00:00 --partition cpu2017-bf05,cpu2019-bf05,cpu2021-bf24,cpu2019,cpu2021,cpu2022,cpu2013,bigmem,synergy -n 14 --mem 60G

Parameters:

--time – The amount of time to allocate for the salloc command. **Maximum is 5 hours** (i.e. 5:00:00)

--partition – The slurm partitions to allocate for the salloc command provided as a list delimited by commas “,”.

-n threads

--mem – The memory in RAM to allocate for the salloc command.

**Job submission**

Running a slurm job script file.

sbatch < slurm\_batch\_file.sh

**Batch jobs**

**Job arrays**

Job array submission header:

#SBATCH --ntasks=1

#SBATCH --cpus-per-task=14

#SBATCH --time=1-00:00:00

#SBATCH --mem=60G

#SBATCH --array=1-6%6

#SBATCH --output=ecami\_pangenomes\_job\_array.%A\_%a.out

#SBATCH --error=ecami\_pangenomes\_job\_array.%A\_%a.err

Slurm job script header options

#SBATCH --ntasks - are how many processes you want to use for parallel tasks.

#SBATCH --ntasks=1

Good for one node on a partition. (i.e. running a program using 14 cpu threads)

Use for interactive jobs, batch jobs, job arrays. using a single node.

#SBATCH --ntasks=N

N >=2

Good for parallel processing like mpi programming.

<https://stackoverflow.com/questions/39186698/what-does-the-ntasks-or-n-tasks-does-in-slurm>

#SBATCH --cpus-per-task=14

#SBATCH --time=1-00:00:00

#SBATCH --mem=60G

#SBATCH --array=1-6%6

#SBATCH --output=ecami\_pangenomes\_job\_array.%A\_%a.out

#SBATCH --error=ecami\_pangenomes\_job\_array.%A\_%a.err

Useful slurm commands;

1. sbatch – Used for submitting the slurm scripts.
2. squeue - .
3. salloc - .
4. sinfo –
5. seff - .

Slurm documentation manual website.

<https://slurm.schedmd.com/documentation.html>

Useful website resources.

1. <https://slurm.schedmd.com/tutorials.html>
2. <https://slurm.schedmd.com/pdfs/summary.pdf>
3. This is a good vim text editor hotkey shortcut blog. Will help with speeding up text file navigation when constructing slurm job scripts. <https://scaron.info/blog/vim-keyboard-shortcuts.html>

Useful examples:

singularity run -H $HOME -B /work:/work -B /bulk:/bulk /global/software/singularity/images/software/alphafold\_2.0.0.sif --fasta\_paths /work/sycuro\_lab/kevin/dominant\_lactobacillus\_genomes/L\_iners\_pullulanase\_blastn/lactobacillus\_iners\_GCF\_000179995.1\_LEAF\_3008A-a\_pula.fasta --output\_dir /work/sycuro\_lab/kevin/dominant\_lactobacillus\_genomes/alphafold\_results

(base) [kevin.muirhead@arc Cpn60]$ sbatch\_id=$(sbatch < bbmap\_cpn60\_db\_pct100.sh)

(base) [kevin.muirhead@arc Cpn60]$ echo $sbatch\_id

Submitted batch job 16568241