

Mila compute cheat sheet

by IDT team

getting help

- search in `docs.mila.quebec`
- search for specific strings (or error messages) on the Mila slack
- visit `#mila-cluster` and `#compute-canada` (for DRAC)
- visit specific tool channels such as `#pytorch` and `#jax`
- open an IT support ticket by emailing `it-support@mila.quebec`
- go to the IDT office hours (Tuesday 3PM-5PM, Wednesday 2PM-4PM) by just walking into the IDT lab (room A.17) or on Google Meet
- contact DRAC support at `support@tech.alliancecan.ca`

milatools

Quick way to setup SSH to Mila cluster

```
uv tool install milatools; mila init
```

Open VSCode connected to an interactive session on compute node

```
mila code /path/work --alloc [salloc arguments]
```

Milatools can also connect directly to a DRAC cluster (e.g. rorqual)

```
mila code /path/work --cluster=rorqual --alloc  
--account=<account> [other salloc arguments]
```

Inside VSCode you can also open a remote SSH to `mila-cpu` to automatically create an interactive session to a CPU node (configured by `mila init`). You can also `ssh mila-cpu` from a terminal.

Do not run a program that takes more than a few seconds on a login node. Do not edit files remotely with VSCode directly on login nodes.

modules

(NB: Many of those modules are outdated.)

```
module avail           Displays all the available modules  
module load <module>  Loads <module>  
module spider <module> Shows details about <module>  
module load python/3.10 Load python 3.10 to use it  
module load httpproxy  Allows Wandb and Comet on DRAC
```

DRAC clusters (including PAICE) tend to have the same modules, which differ a bit from those on the Mila cluster.

Slurm commands

```
salloc --gres=gpu:1 -c 2 --mem=12000  
Get an interactive job with one GPU, 2 CPUs and 12000 MB RAM  
sbatch job.sh  
Submit a batch job (same options as salloc)  
srun <command>  
Run a command (once per task) inside a job  
sattach --pty <jobid>.0  
Attach to a running job  
sinfo  
Status of all nodes  
savail  
List available gpu (Mila only)  
partition-stats [-v]  
Similar functionality to savail (DRAC only)  
scancel <jobid>  
Cancel a job  
squeue -u $USER or squeue --me  
Summary status of all YOUR active jobs  
squeue -j <jobid>  
Summary status of a specific job  
squeue -O jobid,name,username,partition,state,timeused,nodelist,gres,tres  
Status of all jobs including requested resources  
(see the SLURM squeue doc for all output options)  
scontrol show job <jobid>  
Detailed status of a running job  
sacct -j <job_id> -o NodeList  
Get the node where a finished job ran  
sacct -u $USER -S <start_time> -E <stop_time>  
Find info about old jobs  
sacct -oJobID,JobName,User,Partition,Node,State  
List of current and recent jobs
```

Remember that every map is a simplification of reality. This is a cheat sheet for Mila students using Slurm, not a full tutorial, and also not a Linux/Git/PyTorch guide. See docs.mila.quebec/home/cheatsheet for pdf, along with errata. Anticipate one update per year.

The complete up-to-date documentation at docs.mila.quebec.

You can also get [resource files](#) there to feed your favorite agent if you prefer that.

PREPARED ON

2026-05-05

BEST BEFORE

2027-04-01

sbatch / salloc commands

```
-n, --ntasks=<number>      Number of task in your script, usually =1  
-C, --cpus-per-task=<ncpus> Number of cores for each task  
-t, --time=<time>          Time requested for your job  
--mem=<size[units]>        Memory requested for all your tasks  
--gres=<list>              Select generic resources such as GPUs:  
--gres=gpu:GPU_MODEL  
-P, --partition=<name>     Partition for resource sharing (Mila cluster only)  
--account=<name>          DRAC allocation for resources (DRAC only)  
-x, --exclude=<nodes>     Exclude certain nodes from job submission
```

sbatch script example

```
#!/bin/bash  
#SBATCH --ntasks=1           # Default 1 task, optional  
#SBATCH --cpus-per-task=2    # Ask for 2 CPUs  
#SBATCH --gres=gpu:1         # Ask for 1 GPU (any type)  
#SBATCH --mem=10G            # Ask for 10 GB of RAM  
#SBATCH --time=3:00:00       # The job will run for 3 hours  
#SBATCH -o /network/scratch/<u>/<username>/slurm-%j.out  
  
# Copy your dataset on the compute node  
cp /network/datasets/<dataset> $SLURM_TMPDIR  
# Launch your job, tell it to save the model in $SLURM_TMPDIR  
# and look for the dataset into $SLURM_TMPDIR  
srun uv run python main.py --path $SLURM_TMPDIR  
# Copy whatever you want to save on $SCRATCH  
cp $SLURM_TMPDIR/<to_save> /network/scratch/<u>/<username>/
```

multi-GPU, multi-node 1 node with 1 GPU

See docs.mila.quebec/examples/distributed/index.html for minimalist standalone code.

```
#SBATCH --gpus-per-task=rtx8000:1  
#SBATCH --cpus-per-task=4  
#SBATCH --ntasks-per-node=1  
#SBATCH --mem=16G  
#SBATCH --time=00:15:00
```

1 node with 4 GPUs 2 nodes with 4 GPUs each

```
#SBATCH --gpus-per-task=rtx8000:1 #SBATCH --gpus-per-task=rtx8000:1  
#SBATCH --cpus-per-task=4          #SBATCH --cpus-per-task=4  
#SBATCH --ntasks-per-node=4        #SBATCH --ntasks-per-node=4  
#SBATCH --mem=16G                  #SBATCH --nodes=2  
#SBATCH --time=00:15:00            #SBATCH --mem=16G  
#SBATCH --time=00:15:00            #SBATCH --time=00:15:00
```

If you have N parallel jobs that each require 1 GPU, don't try to schedule them in a multi-GPU way. Submit many separate jobs, maybe use job arrays, or consider packing many experiments in a single job with 1GPU.

checkpointing, profiling, scaling

To run large-scale experiments, you need to

- profile your code to make sure you properly use the GPUs allocated (i.e. "GPU Utilization"),
- use checkpoints properly to resume your experiments when they crash or get preempted,
- package your experiments correctly in Slurm jobs to make full use of powerful GPUs.

Easy ways to measure "GPU Utilization" include Wandb, `nvidia-smi` and the DRAC "portal". See also docs.mila.quebec/examples/good_practices/checkpointing/index.html for an example of proper checkpointing. Avoid accumulating too many checkpoints files. They take a lot of storage space.

A common issue at Mila is that junior researchers don't write proper checkpointing for their training experiments. This leads to wasted resources when jobs are preempted after many hours and then cannot resume properly. To avoid preemption, some researchers stick to "unkillable" partitions on the Mila cluster, which **severely limits their ability to run experiments**.

Research involves exploring and testing ideas that don't necessarily work out in the end. This is a good use of the cluster when done properly. Mila is a research institute.

The Mila cluster is available for **all students (co-)supervised by a Mila core prof** and for Mila employees. Not MsPro students, nor students of non-core Mila profs.

Node Name	Qty	GPU N	GPU Model	Mem (GB)	CPU Cores	Mem (GB)	Tmp (TB)	SLURM features	optimal bundle GPU:CPU:RAM
GPU compute nodes									
cn-a[001-011]	11	8x	RTX8000	48	40	384	3.6	turing,48gb	1 : 5 : 47G
cn-b[001-005]	5	8x	V100	32	40	384	3.6	volta,nvlink,32gb	1 : 5 : 47G
cn-c[001-040]	40	8x	RTX8000	48	64	384	3	turing,48gb	1 : 8 : 47G
cn-g[001-029]	29	4x	A100	80	64	1024	7	ampere,nvlink,80gb	1 : 16 : 248G
cn-i001	1	4x	A100	80	64	1024	3.6	ampere,80gb	1 : 16 : 250G
cn-j001	1	8x	A6000	48	64	1024	3.6	ampere,48gb	1 : 8 : 125G
cn-k[001-004]	4	4x	A100	40	48	512	3.6	ampere,nvlink,40gb	1 : 12 : 125G
cn-l[001-091]	92	4x	L40S	48	48	1024	7	lovelace,48gb	1 : 12 : 250G
cn-n[001-002]	2	8x	H100	80	192	2048	35	hopper,nvlink,80gb	1 : 24 : 268G

DGX Systems									
cn-d[001-002]	2	8x	A100	40	128	1024	14	ampere,nvlink,dgx,40gb	1 : 16 : 124G
cn-d[003-004]	2	8x	A100	80	128	2048	28	ampere,nvlink,dgx,80gb	1 : 16 : 250G
cn-e[002-003]	2	8x	V100	32	40	512	7	volta,nvlink,dgx,32gb	1 : 5 : 62G

CPU compute nodes									
cn-f[001-004]	4	-	-	-	32	256	10	rome	0 : 1 : 7G
cn-h[001-004]	4	-	-	-	64	768	7	milan	0 : 1 : 11G
cn-m[001-004]	4	-	-	-	96	1024	7	sapphire	0 : 1 : 10GB

As a reference for orders of magnitude, the Mila cluster has 1020 GPUs in total, accessible by ~700 researchers, with yields 6141 RGUs if we use DRAC's definition of Reference GPU Units.

partition name	max resource usage	max time	note
unkillable	6 CPUs, mem=32G, 1 GPU	2 days	
unkillable-cpu	2 CPUs, mem=16G	2 days	CPU-only jobs
short-unkillable	exactly 4 GPU (see note below)	3 hours (!)	multi-GPU only
main	8 CPUs, mem=48G, 2 GPUs	5 days	
main-cpu	8 CPUs, mem=64G	5 days	CPU-only jobs
long	no limit of resources	7 days	
long-cpu	no limit of resources	7 days	CPU-only jobs

Jobs on the Mila cluster are all *preemptible*, except for those in the *unkillable* partitions. This means that they can be terminated and requeued automatically to allow higher-priority jobs to run (based on partition preemption order *unkillable > main > long*). There is a very limited number of jobs that can run in *unkillable* partitions.

The *short-unkillable* partition is a weird creature. It is designed to run jobs with exactly 4 GPUs (only A100L, L40S and H100) uninterrupted. One possible use for that is debug jobs on the Mila cluster with 4x H100 to later run on the *tamia* cluster.

path	storage/nodes	speed	backup?	mounted
\$HOME	100GB / 1M	low	yes	all nodes
\$SCRATCH	5TB / infity	high	no	all nodes
\$SLURM_TMPDIR	-	highest	no	cn-*
/network/projects	varies	medium	no	all nodes
/network/datasets	read-only	high	no	all nodes
/network/weights	read-only	high	no	all nodes
\$ARCHIVE	5TB	low	no	login-*

Use `disk-quota` to see your current usage of storage (\$HOME and \$SCRATCH).

STACC

coming September 2026

- 21 nodes with 8 x B200 (168 GPUs total)
- 12 nodes with 8 x RTX6000 (96 GPUs total)

STACC will have a similar configuration to the Mila/Tamia clusters with the same login credentials, but it will be shared 75% for Mila researchers and 25% for startups. Check in September 2026 for more info.

DRAC

docs.alliancecan.ca/wiki

		(shared mega-allocation)		(your supervisor's default allocation)		unrestricted			
		rrg-bengioy-ad_gpu	rrg-bengioy-ad_cpu	def-yourprof-gpu	def-yourprof-cpu	GPU types	internet?	Wandb?	Comet?
rorqual		1172 RGUs	263	?	?	4x H100-80GB	no	httpproxy (limited)	httpproxy
fir		2090 RGUs	0	?	?	4x H100-80GB	yes	yes	yes
nibi		363 RGUs	0	?	?	8x H100-80GB	yes	yes	yes
trillium, trillium-gpu		375 RGUs	788	?	?	8x H100-80GB	no	?	?

		GPU devices	RGU equivalent	CPU-only cores	GPU types	Wandb?	Comet?
PAICE (tier1+tier2)	tamia	143	1738	435	4x H100-80GB	no	httpproxy (limited)
(tier 3)	killarney	75	794	0	L40S, H100	yes (for now)	?
(tier 3)	vulcan	82	850	0	L40S	no	httpproxy (limited)

Values listed in this table represent 85% of Tamia (i.e. tier1+tier2), 10% of Killarney and 10% of Vulcan (both tier3 for Mila profs).

DRAC (Digital Research Alliance of Canada, formerly known as Compute Canada) offers access to compute clusters to all researchers in Canada. Any prof with an account on DRAC can add ("sponsor") whoever they want on their "default" allocation, but they will usually add their own students. Additionally, **all students supervised by a Mila core prof** and all Mila employees can be added to a "mega allocation" under Yoshua Bengio's name. See https://docs.mila.quebec/Extra_compute.html#account-creation.

DRAC uses a concept of "Reference GPU Unit" (RGU) to measure allocated GPUs in a way that attributes a different costs to GPUs based on their performance and memory. For example, P100-12GB is 1 RGU. A100-40GB is 4.0 RGUs. H100-SXM5-80GB is 12.17 RGUs.

PAICE clusters are part of DRAC, but they are built for CIFAR AI Chairs. Tamia for Mila, Killarney for Vector and Vulcan for AMIL.

scheduler issues

There have been issues with scheduling on our rrg-bengioy allocation, leading to long queues and our jobs not getting scheduled properly. We are going try a new system with DRAC this year to mitigate that issue using a dedicated Slurm partition. Refer to online documentation for more details.

Independant to that, when you submit jobs that are easy to schedule, the scheduler will prioritize you. Request GPU:CPU:RAM ratios that are conducive to sharing a node with other jobs, and shorter durations (chunking jobs can work well).

time bins <=3h <=12h <=24h <=72h <=168h

GPU:CPU:RAM bundles

Rorqual 1 : 12 : 124G

Nibi 1 : 14 : 250G

Fir 1 : 12 : 250G

Tamia 1 : 12 : 124G (full node only, <=24h)

DRAC compute nodes have similar roles as the Mila cluster for \$HOME, \$SCRATCH and \$SLURM_TMPDIR. See also \$HOME/projects/<account>/<your_username>. Use `diskusage_report` to see usage.