# BEGINNER'S GUIDE TO HPC FOR NLP RESEARCH

Using the Flemish Supercomputer (VSC) at UGent

*LT3 · Language and Translation Technology Team · UGent*

GHENT UNIVERSITY

# WHAT'S IN THIS GUIDE

## Understanding HPC Tiers

Tier1 vs Tier2 and what LT3 uses

## Getting Access

Account setup and joining the LT3 VO

## Infrastructure

Nodes, clusters, modules, and GPU clusters

## Data Directories

$VSC_HOME, $VSC_DATA, $VSC_SCRATCH and quotas

## Connecting

Terminal, web portal, and VS Code tunnel

## Running Jobs

Job system, PBS scripts, GPU jobs, and more

GHENT UNIVERSITY

# UNDERSTANDING HPC TIERS

## TIER 2 (DEFAULT)

- Available to all UGent researchers
- Great for learning HPC workflows
- GPU clusters: joltik, accelgor, litleo
- Start here even if you have Tier1 access

## TIER 1 (ADVANCED)

- Specialized hardware, more power
- Requires approved grant proposal
- Evaluated on scientific merit
- Same concepts, some differences

**This guide focuses on Tier 2**

GHENT UNIVERSITY

# GETTING ACCESS

### Create VSC Account

Go to:
account.vscentrum.be
Select UGent, log in via CAS

→

### Wait for Approval

Usually 1–2 business days.
You'll receive an email with your vscXXXXX ID.

→

### Join LT3's VO

On account.vscentrum.be:
Click 'New/Join VO'
Group: gvo00042

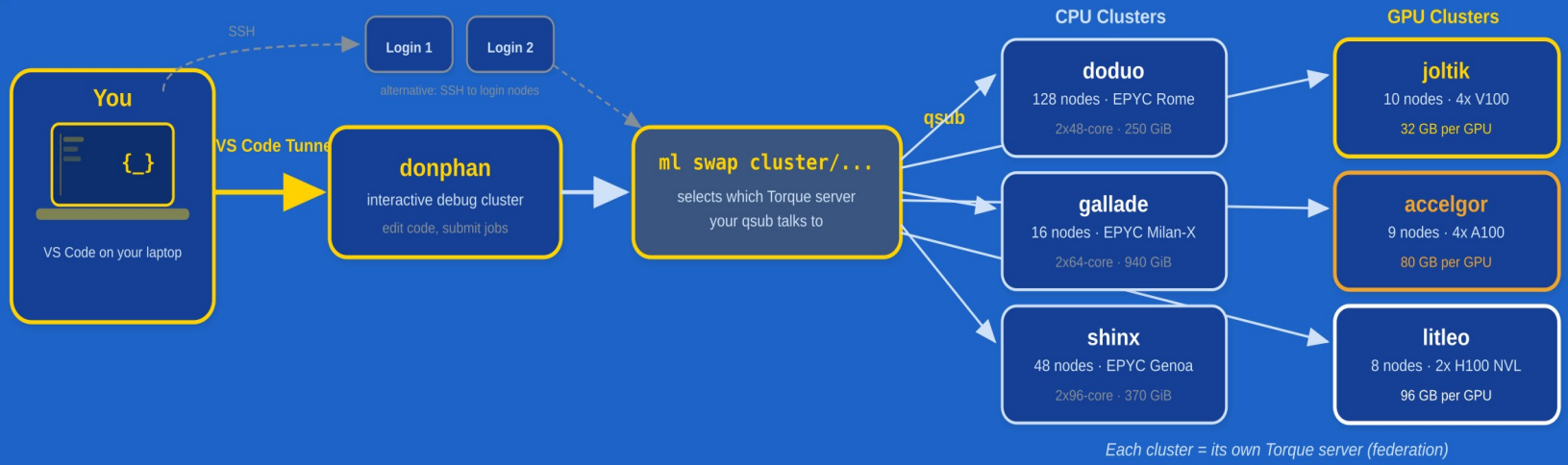
→

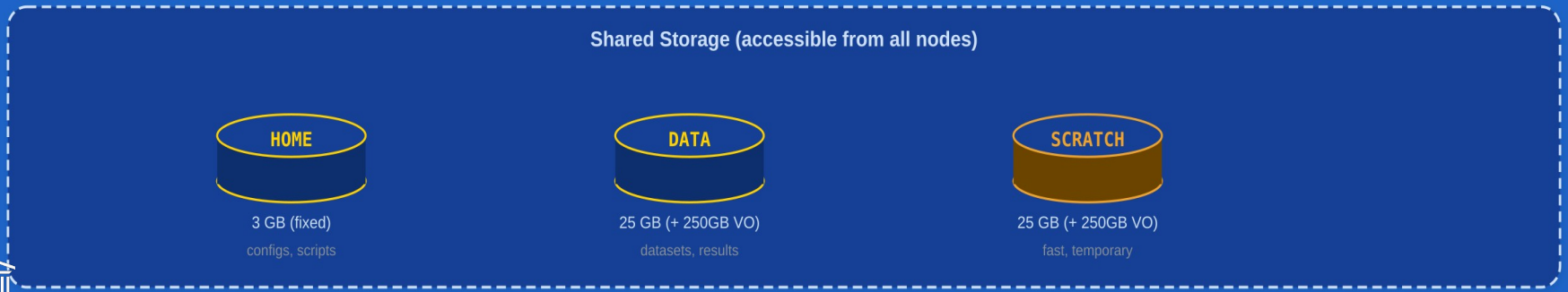
### Ready to Go!

Log in at:
login.hpc.ugent.be
(web portal) or via SSH

GHENT UNIVERSITY

# HPC INFRASTRUCTURE: THE BIG PICTURE




GHENT UNIVERSITY

# KEY TERMINOLOGY

When you connect to the HPC, you access a terminal with multiple clusters, each containing nodes that run modules.

## NODES

Individual computers.
Login nodes = entry points.
Compute nodes = where jobs run.
Debug nodes = interactive work (donphan).

## CLUSTERS

Groups of nodes with specific hardware.
Each has its own Torque server.
Switch with: ml swap cluster/...
E.g. doduo, accelgor, litleo.

## MODULES

Software you load on demand:
module load Python/3.10.4-...
module avail  (search available)
module list  (show loaded)

# GPU CLUSTERS ON TIER 2

## joltik

**10 nodes**
2× 16-core Intel Xeon Gold 6242
**4× NVIDIA Volta V100**
32 GB GPU memory per card
256 GiB memory per node

```
module swap cluster/joltik
```

## accelgor

**9 nodes**
2× 24-core AMD EPYC 7413
**4× NVIDIA Ampere A100**
80 GB GPU memory per card
500 GiB memory per node

```
module swap cluster/accelgor
```

## litleo

**8 nodes**
1× 48-core AMD EPYC 9454P
**2× NVIDIA H100 NVL**
96 GB GPU memory per card
315 GiB memory per node

```
module swap cluster/litleo
```

GHENT UNIVERSITY

*Fun fact: all cluster names at UGent HPC are based on Pokemon characters!*

# DATA DIRECTORIES ON THE HPC

### $VSC_HOME

Config files, small scripts

3 GB (fixed) · All sites

user/Gent/xxx/vscXXXXX

### $VSC_DATA

Datasets, results, logs

25 GB (+ 250GB VO) · All sites

data/Gent/xxx/vscXXXXX

### $VSC_SCRATCH

Fast temporary storage

25 GB (+ 250GB VO) · Active jobs

scratch/Gent/xxx/vscXXXXX

**With LT3 VO (gvo00042) you also get:**

**$VSC_DATA_VO_USER**

Shared data within LT3

**$VSC_SCRATCH_VO_USER**

Fast shared scratch within LT3

Monitor quotas at account.vscentrum.be · Soft quota can be temporarily exceeded; hard quota cannot

*Check your quotas with: my_dodrio_quota | Keep models in $VSC_DATA or $VSC_SCRATCH, not $VSC_HOME (only 3 GB).*

GHENT UNIVERSITY

# CONNECTING TO THE HPC

## Terminal (SSH)

ssh vscXXXXX@
login.hpc.ugent.be
Requires SSH key pair.
Direct terminal access.

## Web Portal

login.hpc.ugent.be
No SSH key needed.
Browser-based shell,
Jupyter, file manager.

RECOMMENDED

## VS Code Tunnel

Recommended for us.
Full IDE on debug node.
File explorer + terminal.
See next slide for setup.

*Note: VS Code tunnel connects to donphan (debug node), so you can run simple computations directly.*

GHENT UNIVERSITY

# VS CODE TUNNEL: SETUP

login.hpc.ugent.be

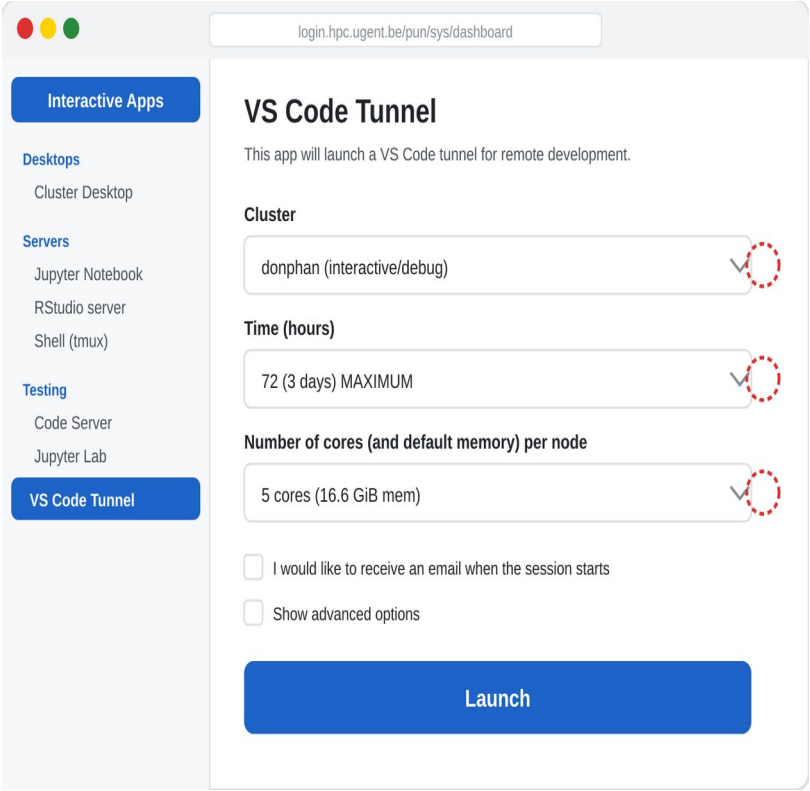
Go to login.hpc.ugent.be → Interactive Apps → VS Code Tunnel

Set cluster: donphan (debug), time (72h), cores (4–5)

Click Launch, wait for session to become active

Click Connect, choose Microsoft Account (UGent)

Open VS Code locally → Remote-Tunnels: Connect




*Tip: Set up a VS Code workspace file to easily navigate $VSC_HOME, $VSC_DATA, and $VSC_SCRATCH.*

GHENT UNIVERSITY

# THE JOB SYSTEM

You don't run programs directly. You pick a cluster, then submit jobs to its queue.


`qstat` · `qdel <job_id>`

monitor or cancel at any time

## FEDERATION

Each cluster has its own Torque server (not like Slurm)

## MODULE SWAP

ml swap cluster/accelgor redirects qsub to that server

## SUBMIT + MONITOR

qsub job.pbs to submit
qstat to check status

## FAIR-SHARE

Less past usage = higher priority for you

# ESSENTIAL HPC COMMANDS

| COMMAND | WHAT IT DOES |
| --- | --- |
| `qsub job.pbs` | Submit a job to the queue |
| `qstat` | Check job status (qstat -f for details) |
| `qdel <job_id>` | Cancel a running or queued job |
| `module load Python/3.10.4-...` | Load a software module |
| `module swap cluster/accelgor` | Target a cluster's Torque server (required before qsub) |
| `qsub -I -l nodes=1:ppn=4:gpus=1` | Start an interactive session (with GPU) |
| `nvidia-smi` | Check GPU status and usage |
| `cd $PBS_O_WORKDIR` | Go to the job submission directory |

# ANATOMY OF A PBS JOB SCRIPT

```
#!/bin/bash -l
#PBS -l nodes=1:ppn=4
#PBS -l mem=16gb
#PBS -l walltime=00:05:00
#PBS -N my_job_name
#PBS -m abe

module load Python/3.10.4-GCCcore-11.3.0
cd $PBS_O_WORKDIR

python my_script.py
```

- Login shell interpreter
- Request 1 node with 4 CPU cores
- Request 16 GB of memory
- Max runtime: 5 minutes (max is 72h)
- Job name (shows up in qstat)
- Email on abort / begin / end

- Load software, go to working dir, run script

Do NOT put cluster/partition in the script.
Use ml swap cluster/... before running qsub.

GHENT UNIVERSITY

# RUNNING A GPU JOB

Key changes: gpus=1 in PBS header, CUDA module. Cluster choice is NOT in the script -- use ml swap before qsub.

```
#!/bin/bash -l
#PBS -l nodes=1:ppn=8:gpus=1
#PBS -l mem=16gb
#PBS -l walltime=00:25:00
#PBS -N GPU_PyTorch_Job
#PBS -m abe

module load PyTorch/2.1.2-foss-\
 2023a-CUDA-12.1.1
cd $PBS_O_WORKDIR
nvidia-smi
python gpu_script.py
```

**In your terminal (not the script!):**

```
ml swap cluster/accelgor
qsub gpu_pytorch_job.pbs
# or cluster/litleo for H100s
```

**Always test first!**

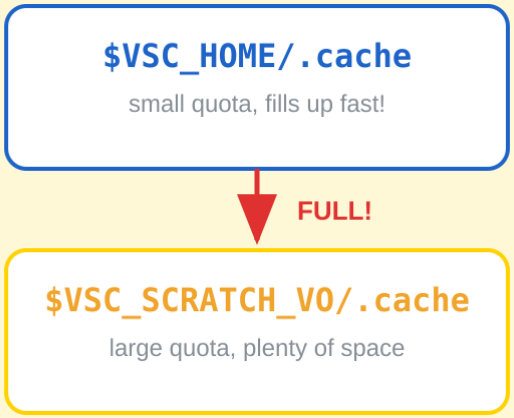
Use an interactive session to
catch errors before waiting in queue:

```
qsub -I -l nodes=1:ppn=4:gpus=1
```

GHENT
UNIVERSITY

# MANAGING STORAGE: SYMLINKS

Common problem: $VSC_HOME fills up fast with .cache and .local files from models and packages.




**Solution: Symbolic Link**

```
mv $VSC_HOME/.cache $VSC_SCRATCH_VO_USER
ln -s $VSC_SCRATCH_VO_USER/.cache $VSC_HOME/.cache
```

Now .cache writes go to SCRATCH transparently. Same trick works for .local!
Don't forget to source ~/.bashrc after adding environment variables.

*Don't forget: source ~/.bashrc after adding environment variables.*

GHENT UNIVERSITY

# SETTING UP VIRTUAL ENVIRONMENTS

## CONDA

```
module swap cluster/accelgor
qsub -I -l nodes=1:ppn=4:gpus=1

module load Anaconda3/2023.03-1
conda init
conda create --name myenv
conda activate myenv
pip install -r requirements.txt
```

## VENV

```
module swap cluster/accelgor
qsub -I -l nodes=1:ppn=4:gpus=1

module load Python/3.10.4-GCCcore-11.3.0
python -m venv myenv
source myenv/bin/activate
pip install -r requirements.txt
```

*Set up envs in an interactive GPU session so CUDA packages install correctly. Conda envs live in $VSC_HOME -- symlink .conda to $VSC_DATA if quota fills up.*

# RESOURCES & QUICK REFERENCE

### LT3 HPC Wiki Guide

vtcserv.ugent.be/lt3wiki/...

Full step-by-step HPC guide for LT3

### UGent HPC Docs

docs.hpc.ugent.be

Official docs, tutorials, and FAQ

### VSC Docs

docs.vscentrum.be

Flemish Supercomputer Center docs

### LT3 GitHub Repo

github.com/lt3

Code, scripts, and examples

### Account & Quotas

account.vscentrum.be

Manage your VSC account and VO

### Cluster Status

ugent.be/hpc/en/infrastructure

Current hardware specs

Quick start: `module swap cluster/accelgor | qsub job.pbs | qstat | my_dodrio_quota | login.hpc.ugent.be`

Acknowledgement: The computational resources (Stevin Supercomputer Infrastructure) and services were provided by the VSC, funded by Ghent University, FWO, and the Flemish Government (dept. EWI).

GHENT UNIVERSITY