GPU sharing on orca

orca has two NVIDIA H100 NVL GPUs (~95 GB VRAM each, 190 GB total), shared between the lab's LLM inference service and everyone's research jobs (training runs, GPU-accelerated experiments). There is no job scheduler — sharing works by the conventions on this page.

What's already running

The LLM service (LiteLLM → llama-swap) uses the GPUs *elastically*:

- **Every LLM model unloads after 1 idle hour.** If nobody has used the service for an hour, GPU usage drops to near zero and essentially all 190 GB is available for your jobs. The next LLM request just pays a ~1-minute reload.
- **While in use:** the main chat model — `qwen3.8:27b-q8_0`, Qwen3.8-27B at Q8_0 quantization (unsloth's Dynamic GGUF build, whose multi-token-prediction heads give it speculative self-decoding at ~2× speed) — takes ~71 GB (~28 GB weights + a large shared KV cache, split across both cards), and loading one of the largest models can temporarily bring total usage to ~180 GB. A large-model load simply *fails* if your job holds the space — during a reserved run that's expected, not an emergency.
- Speech-to-text (WhisperX) takes ~4 GB while loaded, and it too unloads after 1 idle hour (~30 s reload on the next request).

Practical meaning: `nvidia-smi` will show usage that varies over time and isn't anyone's job. Free VRAM is genuinely free to use.

Rules of thumb (always apply)

1. **Check before you allocate.** Run `nvidia-smi` and look at both memory *and* utilization on each card. Don't start a large job on a card that's busy.
2. **Pin your job to one card** with `CUDA_VISIBLE_DEVICES=0` (or `=1`) unless you actually need both. This keeps the other card clean for the LLM service and other users. (PyTorch needs no other configuration — it allocates memory incrementally as it goes.)
3. **Checkpoint long runs.** There is no preemption, but the machine can be rebooted for maintenance; anything running longer than ~12 h should be resumable.

Reservations

Small jobs (up to ~40 GB VRAM, a few hours): just run them — no reservation needed. That fits comfortably alongside the LLM service even at its busiest.

Anything bigger or longer: post in the Zulip `#compute-gpu` channel, stating **which GPU(s), roughly how much VRAM, and until when**. That's the whole process — first post wins, coordinate in-thread if two runs collide.

During a reservation, the reservation wins. The largest LLM models may fail to load for other users while a big reserved run holds the space; the main chat model keeps working either way (it reloads into whatever room remains after an idle unload). If you need the big LLM models and a reservation is active, wait it out or ask in the thread.

Revision #3
Created 26 August 2026 03:51:38 by Adarsh Pyarelal
Updated 26 August 2026 06:55:02 by Adarsh Pyarelal