Inserm

La science pour la santé
From science to health

Guide d'utilisation

Mise en œuvre du serveur Ollama pour tester des modèles LLM sur le cluster HPC GPU

Inserm - Cloud self-service - Offre HPC

Objectif du document

Ce tutoriel a pour objectif de vous guider pas-à-pas pour :

Déployer le serveur `Ollama` sur le cluster `HPC`.

Exécuter des modèles `LLM` sur des ressources `GPU`.

Utiliser un environnement conteneurisé via `Apptainer`.

Tester et évaluer des modèles dans un cadre sécurisé et maîtrisé.

Table des matières

1 Eléments contextuels

Mise en œuvre de Ollama sur le cluster HPC GPU

Présentation du cas d'usage

La plateforme HPC Inserm met à disposition des ressources de calcul mutualisées, incluant des accélérateurs GPU, accessibles via le portail OpenOnDemand.
Afin de permettre l'expérimentation et l'exécution de modèles d'intelligence artificielle générative (LLM), un environnement conteneurisé basé sur **Ollama** est proposé aux utilisateurs.

Cas d'usage visés

- Tests et validation de modèles LLM
- Expérimentation en environnement GPU
- Prototypage d'applications IA
- Évaluation des performances avant industrialisation

1.1 Apptainer/Singularity

Chaque nœud de calcul du cluster HPC intègre la solution de conteneurisation Apptainer. Grâce à des images spécifiques, il est ainsi possible d'enrichir dynamiquement l'environnement logiciel du nœud avec des outils non disponibles par défaut.

1.2 Accès GPU

La plateforme HPC est équipée de ressources GPU, réparties en partition selon leur spécificité.

Partition 2xH100-96

Cette partition est présentée à titre d'exemple : elle fait partie des partitions disponibles sur la plateforme, parmi d'autres configurations.

Composée de 2 nœuds de calculs fournissant chacun :

- 32 Cores CPU
- 2 GPUs H100 – 96Go.

Ressources dédiées

Après avoir réservé une partition, les ressources sont dédiées en intégralité à l'utilisateur qui a réservé le nœud.

- Durée maximum : 30 heures

⚠️**Point crucial : au-delà de cette durée, vos résultats de calcul seront perdus !** Assurez vous que vos calculs ne dépassent pas 30 heures. Au delà de cette durée, vous pouvez découper vos calculs et réserver des partitions à la suite.

1.3 Ressources de calcul : orchestrateur Slurm

L'accès et la réservation de nœuds de calcul d'une partition se fait au travers de l'orchestrateur Slurm et de ses commandes.

1.4 Modèles LLM Ollama

Modèles disponibles

Un certain nombre de modèles ont été mis à disposition sur la plateforme HPC via le répertoire /shared/ollama.

Restrictions réseau

La plateforme ne permet pas de télécharger de nouveaux modèles via internet.

Demande d'ajout de nouveaux modèles

L'ajout de nouveaux modèles devra faire l'objet d'une demande de mise à disposition.

2 Préparation à l'utilisation de Apptainer

Pour mettre en œuvre le serveur Ollama sur la plateforme HPC, les 4 étapes ci-après sont nécessaires. Ces étapes sont mises en œuvre par **Apptainer au travers des paramètres passés en ligne de commande**.

Toutefois, ces étapes conduisent à une commande relativement longue et complexe. Afin de simplifier son utilisation au quotidien, il est donc recommandé de **définir cette commande sous forme d'alias** dans votre fichier ~/.bashrc. Pour ce faire, vous pouvez passer ces 4 étapes et continuer le tutoriel.

1

Utiliser l'image conteneurisée Apptainer ./containers/ollama.sif

Le HPC permet de mettre en œuvre des applications en s'appuyant sur une technologie de conteneurisation, à savoir Apptainer. Cette approche permet d'exécuter des applications complexes dans un environnement maîtrisé, isolé et reproductible, tout en restant compatible avec les contraintes d'un cluster HPC (sécurité, performances, multi-utilisateurs).

L'image Apptainer ollama.sif contient l'application Ollama ainsi que l'ensemble de ses dépendances logicielles. Son utilisation permet :

- d'assurer la reproductibilité des calculs ;
- d'éviter toute installation directe sur les nœuds de calcul ;
- de garantir une portabilité entre les différents nœuds du cluster.

2

Monter le répertoire des modèles dans le conteneur --bind /shared/ollama/models:/models

Cette option permet de rendre accessible, depuis l'intérieur du conteneur, le répertoire contenant les modèles LLM stockés sur le système de fichiers partagé du HPC.

- `/shared/ollama/models` : répertoire partagé entre tous les nœuds de calcul
- `/models` : point de montage à l'intérieur du conteneur.

Cette approche évite la duplication des modèles sur chaque nœud et permet leur réutilisation par l'ensemble des utilisateurs.3

3

Définir les variables d’environnement NVIDIA

Ces variables sont indispensables pour garantir un fonctionnement correct des GPU NVIDIA au sein du conteneur.

Elles permettent notamment d’éviter certains dysfonctionnements connus liés aux drivers NVIDIA et à CUDA.

4

Autoriser l’utilisation des GPU dans Apptainer

Cette option active automatiquement le support GPU dans le conteneur en exposant les bibliothèques NVIDIA du système hôte. Sans cette option, l’application s’exécuterait sans accès GPU, même si des ressources GPU ont été réservées via Slurm.

Utilisation de Apptainer sous forme d'alias

1) Exécuter la commande ci-dessous pour définir les variables d'environnement Nvidia dans un fichier nommé «.nvidia-env » situé dans notre répertoire personnel ($HOME) :

```
$ cat ~/.nvidia-env
NCCL_DEBUG=NONE
NCCL_IB_DISABLE=1
NCCL_NET=socket
NCCL_P2P_DISABLE=1
```

2) Modifier le fichier .bashrc en ajoutant en fin de fihcier cette ligne déclarant l'alias :

```
alias ollm='apptainer exec --writable-tmpfs --nv --sharens \
--bind /shared/ollama/models/:$HOME/.ollama/models \
--env-file $HOME/.nvidia-env \
/containers/ollama.sif ollama'
```

3) Pour que la modification soit effective, exécuter la commande source `$HOME/.bashrc`

4) Une fois cet alias en place, lancer vos commandes Ollama. Par exemple :

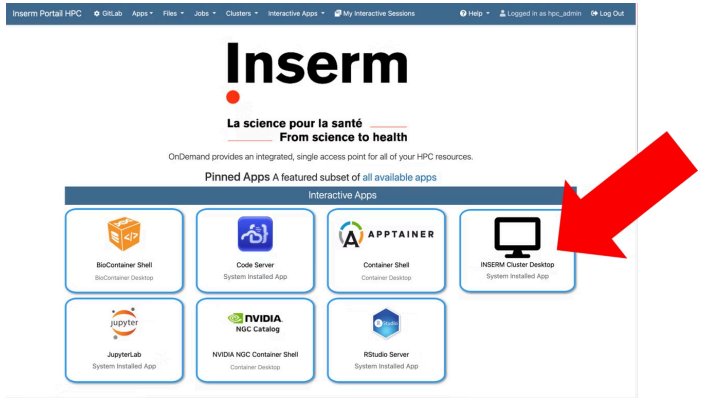
```
ollm serve
ollm list
ollm run llam
```

3 Mise en œuvre de Ollama

Pour bénéficier d'une interface graphique, nous pouvons réserver un nœud de calcul depuis l'interface OOD au travers du widget Inserm Cluster Desktop, puis y accéder via une session GUI.

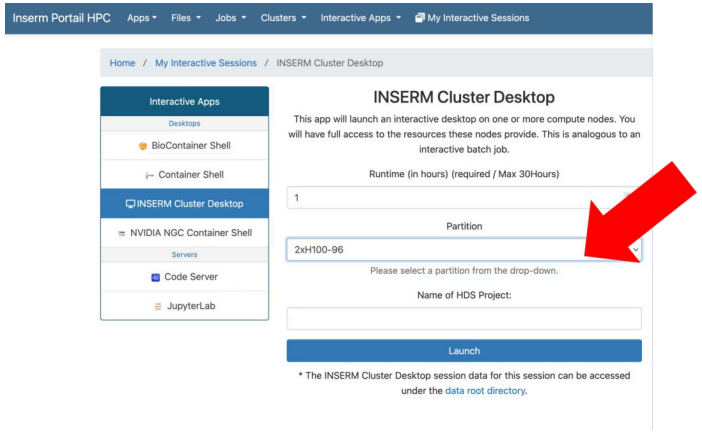
01

Choisir le widget Inserm Cluster Desktop



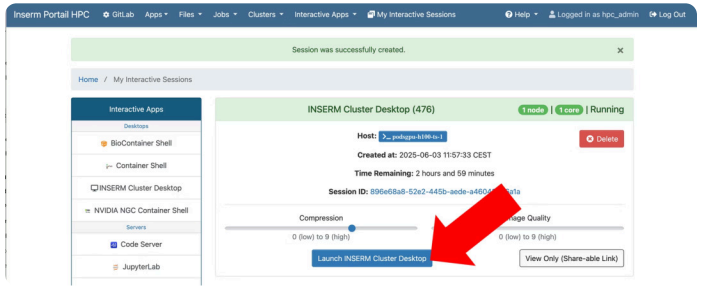
02

Sélectionner la partition GPU



03

Accéder à l'interface via le bouton : launch Inserm Cluster Desktop



Nous allons mettre en œuvre 3 terminaux Xterm :

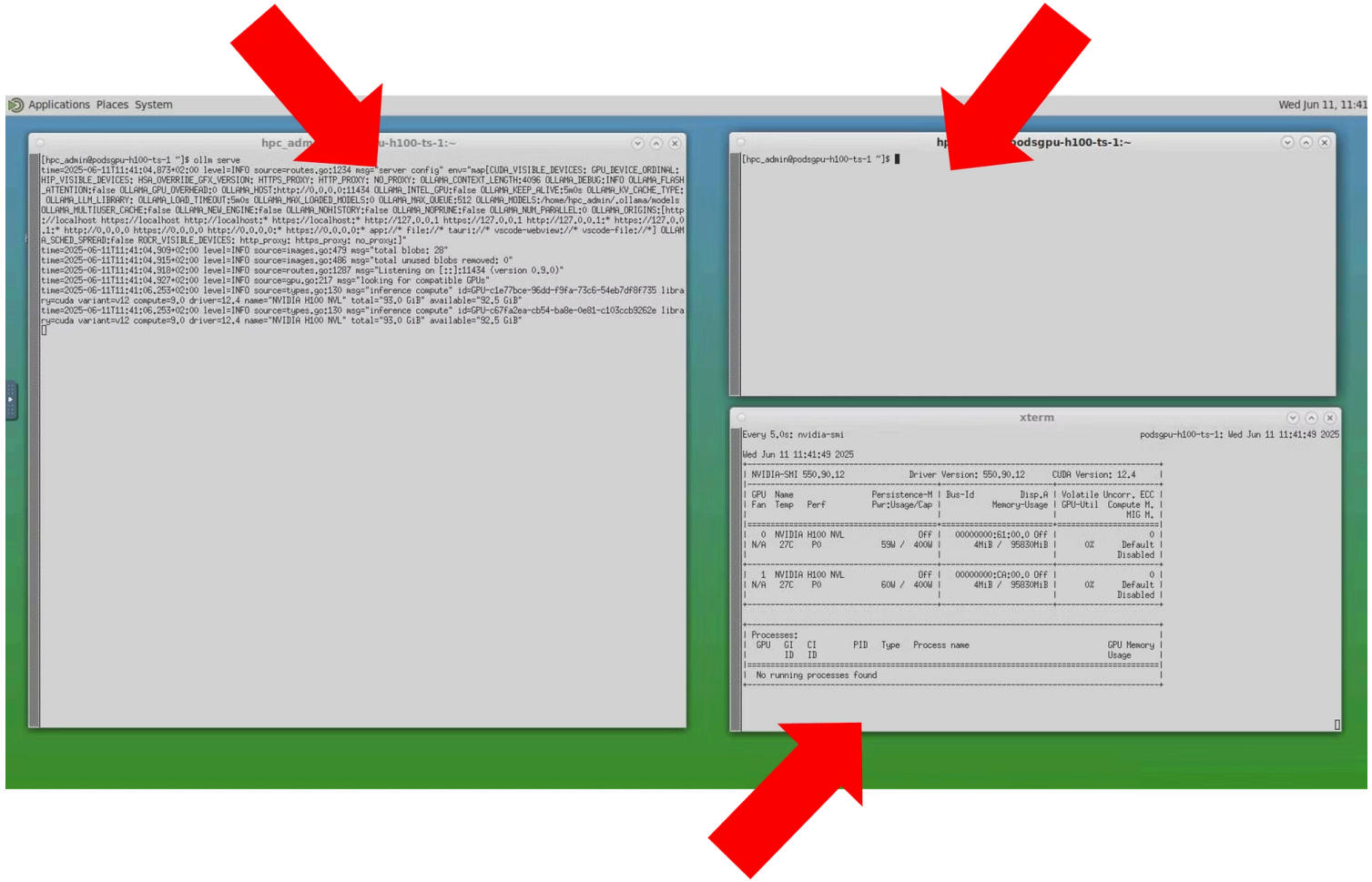
- Le premier terminal nous permettra de lancer le serveur Ollama : ollm serve
- Le deuxième terminal nous permettra d'interagir avec le serveur en tant que client : ollm list, ollm run,...
- Le troisième terminal nous donnera une vue sur l'utilisation CPU : watch -n 5 nvidia-smi

Note : si besoin, utiliser la commande ci-dessous pour charger votre environnement et accéder à vos alias :

$> source $HOME/.bashrc

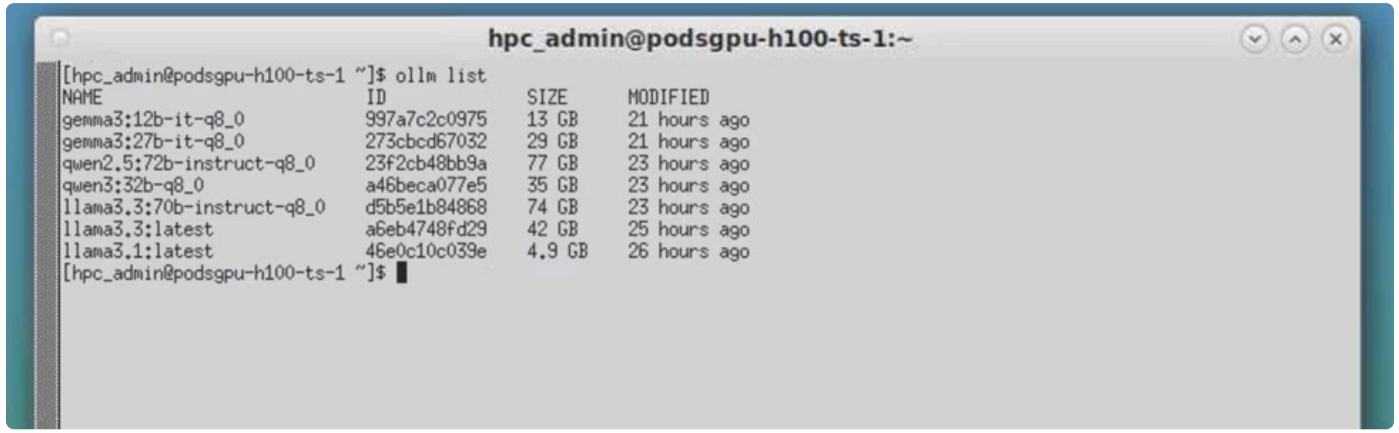
Serveur OLLAMA

Client OLLAMA



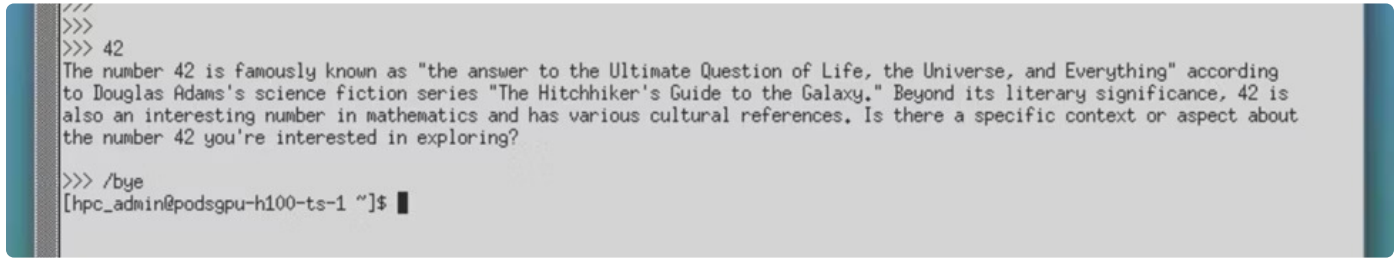
Vue GPUs via un "watch -n 5 nvidia-smi"

Exécuter la commande ollm list via le Client pour lister les modèles mis à disposition



Exécuter la commande ollm run pour mettre en œuvre un modèle :

Exécuter la commande pour interroger le modèle :



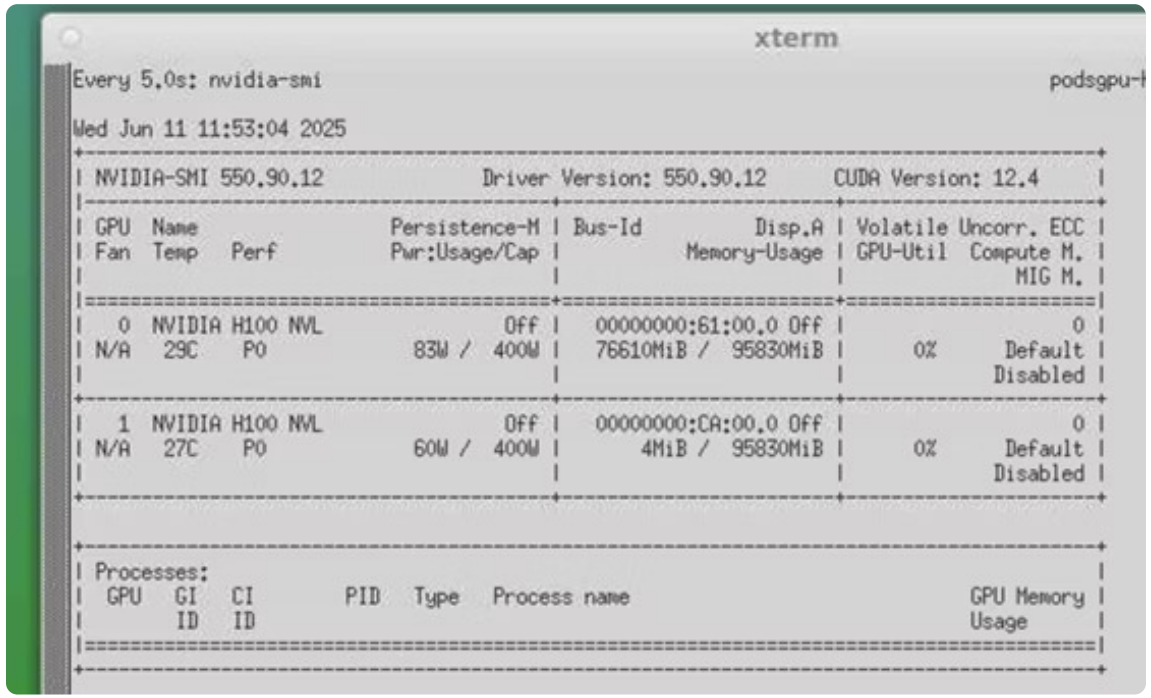
Exécuter la commande pour lister et arrêter les modèles exécutés :

```
[hpc_admin@podsgpu-h100-ts-1 ~]$
[hpc_admin@podsgpu-h100-ts-1 ~]$ ollm ps
NAME                        ID              SIZE     PROCESSOR    UNTIL
qwen2.5:72b-instruct-q8_0   23f2cb48bb9a    81 GB    100% GPU     4 minutes from now
[hpc_admin@podsgpu-h100-ts-1 ~]$
[hpc_admin@podsgpu-h100-ts-1 ~]$
[hpc_admin@podsgpu-h100-ts-1 ~]$ ollm stop qwen2.5:72b-instruct-q8_0
[hpc_admin@podsgpu-h100-ts-1 ~]$
[hpc_admin@podsgpu-h100-ts-1 ~]$
[hpc_admin@podsgpu-h100-ts-1 ~]$ ollm ps
NAME    ID    SIZE    PROCESSOR    UNTIL
[hpc_admin@podsgpu-h100-ts-1 ~]$
```

Le terminal Xterm du serveur délivre les logs :

```
load_tensors:        CUDA0 model buffer size = 72415.43 MiB
load_tensors:   CPU_Mapped model buffer size =  1262.25 MiB
time=2025-06-11T11:49:30.416+02:00 level=INFO source=server.go:625 msg="waiting for server to become available" status="llm server loading
model"
llama_context: constructing llama_context
llama_context: n_seq_max     = 2
llama_context: n_ctx         = 8192
llama_context: n_ctx_per_seq = 4096
llama_context: n_batch       = 1024
llama_context: n_ubatch      = 512
llama_context: causal_attn   = 1
llama_context: flash_attn    = 0
llama_context: freq_base     = 1000000.0
llama_context: freq_scale    = 1
llama_context: n_ctx_per_seq (4096) < n_ctx_train (32768) -- the full capacity of the model will not be utilized
llama_context:  CUDA_Host  output buffer size =     1.22 MiB
llama_kv_cache_unified: kv_size = 8192, type_k = 'f16', type_v = 'f16', n_layer = 80, can_shift = 1, padding = 32
llama_kv_cache_unified:      CUDA0 KV buffer size =  2560.00 MiB
llama_kv_cache_unified: KV self size  = 2560.00 MiB, K (f16): 1280.00 MiB, V (f16): 1280.00 MiB
llama_context:      CUDA0 compute buffer size =  1104.00 MiB
llama_context:  CUDA_Host compute buffer size =    32.01 MiB
llama_context: graph nodes  = 2966
llama_context: graph splits = 2
time=2025-06-11T11:49:38.467+02:00 level=INFO source=server.go:630 msg="llama runner started in 10.78 seconds"
[GIN] 2025/06/11 - 11:49:38 | 200 | 12.780765685s |       127.0.0.1 | POST     "/api/generate"
[GIN] 2025/06/11 - 11:49:49 | 200 |      88.911µs |       127.0.0.1 | HEAD     "/"
[GIN] 2025/06/11 - 11:49:49 | 200 |     146.343µs |       127.0.0.1 | GET      "/api/ps"
[GIN] 2025/06/11 - 11:49:54 | 200 |      71.549µs |       127.0.0.1 | HEAD     "/"
[GIN] 2025/06/11 - 11:49:54 | 200 |   12.144548ms |       127.0.0.1 | POST     "/api/generate"
[GIN] 2025/06/11 - 11:49:58 | 200 |      55.617µs |       127.0.0.1 | HEAD     "/"
[GIN] 2025/06/11 - 11:49:58 | 200 |      32.824µs |       127.0.0.1 | GET      "/api/ps"
```

La vue GPU nous indique l'utilisation/ la consommation :



Exécuter <CTRL-C> dans le xterm exécutant ollm serve pour arrêter le serveur Ollama.

4 Conseils d'utilisation de Apptainer

1

Sur la plateforme HPC de l'Inserm, **l'utilisation de « apptainer instance » n'est pas possible :** il faut utiliser **Apptainer (shell ou run).**

2

Si dans un exec ou un shell apptainer vous exécutez un **processus en background** (utilisation de &), assurez-vous de **supprimer le processus** par : pkill -P $$ avant de quitter le container.

Dans le cas contraire, le processus continuera d'être exécuté sur le noeud.

Si vous avez des questions, contacter vos référents habituels sur le HPC.