Cluster de cómputo y SLURM

Seminario 7 de mayo de 2024

¿Qué es slurm?

Sistema de gestión y administración de trabajos en clusters de pequeño y gran tamaño.

Características más relevantes:

1. Nos permite asignar el acceso a recursos de cómputo durante un tiempo establecido. Este acceso puede ser tanto restringido como abierto.
2. Framework para adaptar el trabajo a nuestras necesidades.
3. Implementa una cola de trabajo configurable con prioridades por usuario.
4. Control e información sobre los trabajos.

Nuevo hardware

slurm
workload manager

Procesador AMD ROME 7232P UP
8C/16T 3.1GHz 32MB
32GB RAM (ampliable)
1x NVMe 480GB M.2 para S.O.
12x HD 16TB SAS 12Gbs
Dual 10GBase-T LAN ports
1x RJ45 Dedicated IPMI LAN port

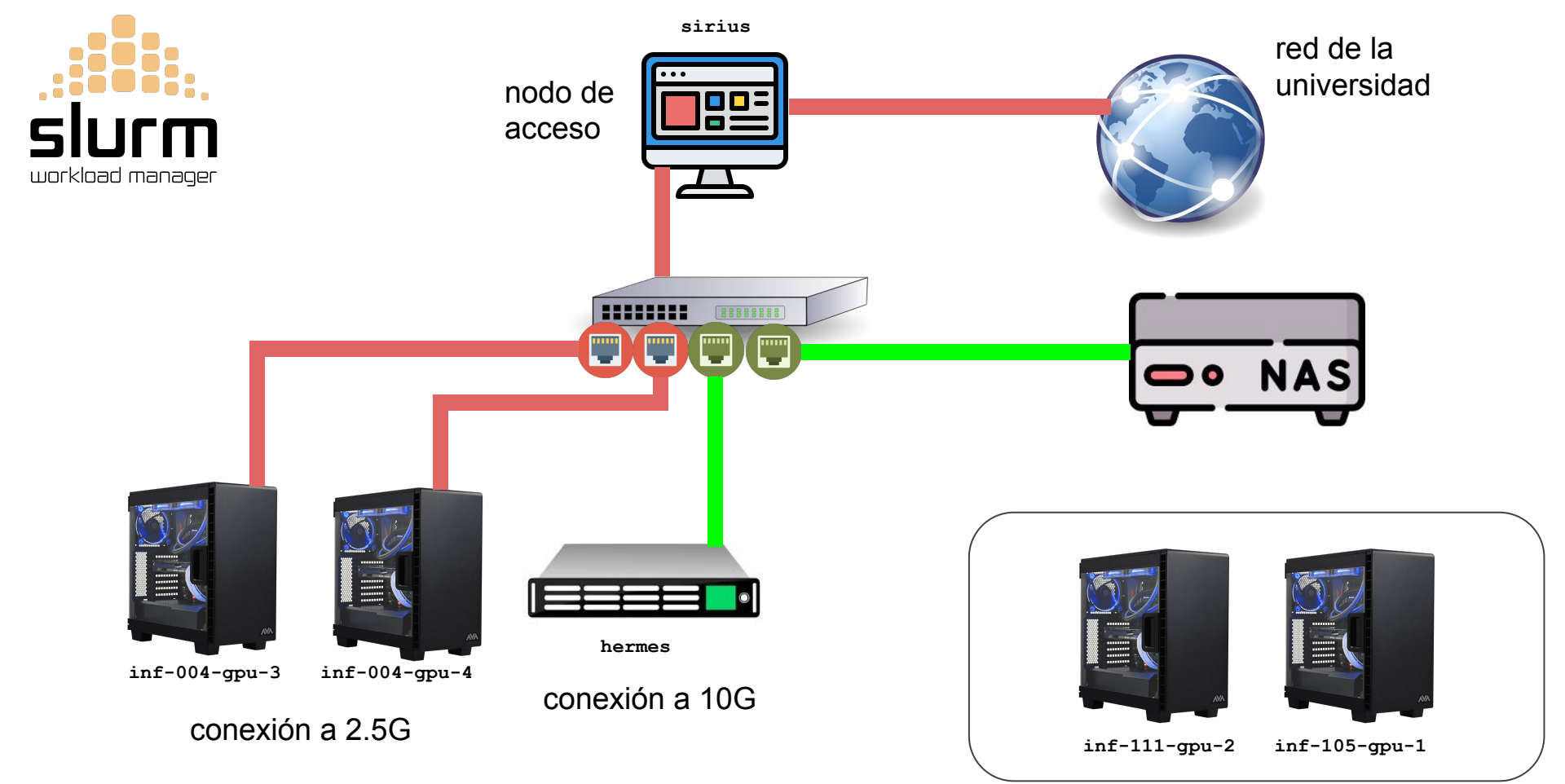
126,71 TB efectivos

slurm
workload manager

10x Adaptable Fiber/RJ45
2x RJ45 10G

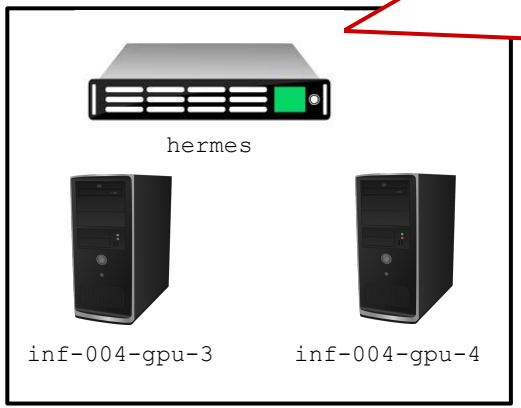
Conexiones a:
1 G
2,5G
5G
10G

Reorganización de las máquinas

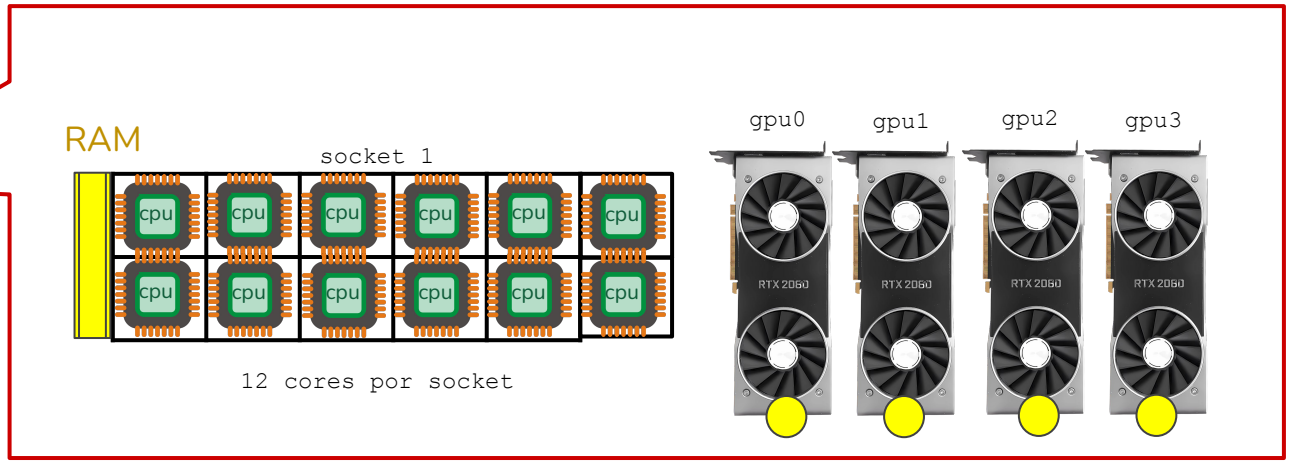

slurm
workload manager
sirius
nodo de acceso
red de la universidad
NAS
inf-004-gpu-3
inf-004-gpu-4
hermes
conexión a 2.5G
conexión a 10G
inf-111-gpu-2
inf-105-gpu-1

Recordatorio - Repaso

Nivel de nodos



Nivel Intra-node



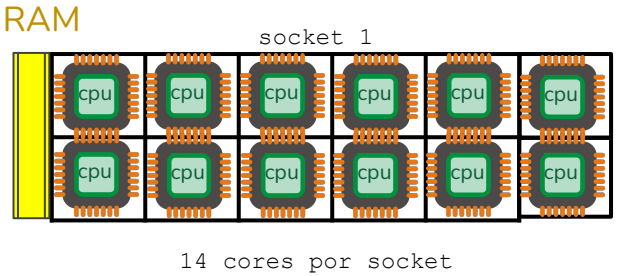
¿qué podemos controlar?

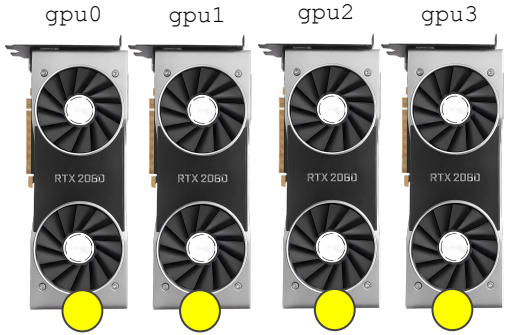
- Nodos
- Sockets
- Cores
- Threads
- Memoria
- GPUs
- Exclusividad

¿Cómo funciona un trabajo?

¿Cómo funciona un trabajo?

hermes

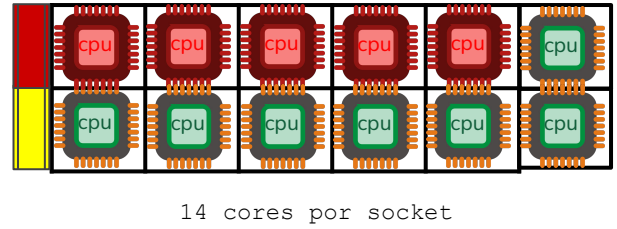


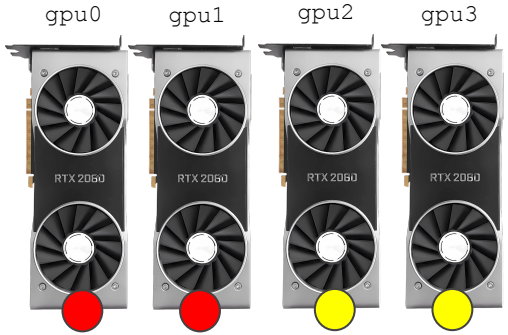


¿Cómo funciona un trabajo?

hermes

–nodes=1
–gres=gpu:A30:2
–p=hermes
–time=16:00:00
–cores-per-socket=5

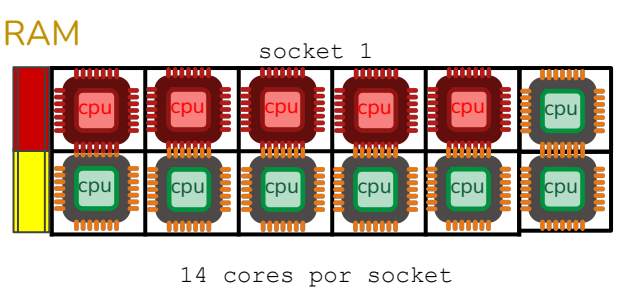


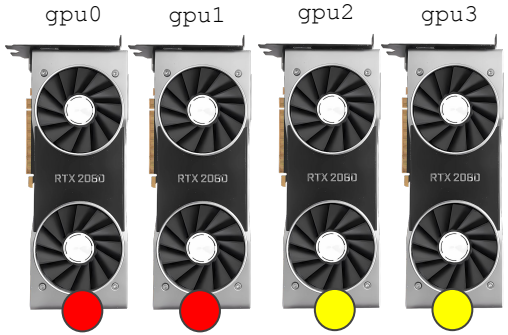


¿Cómo funciona un trabajo?

hermes

–nodes=1
–gres=gpu:A30:2
–p=hermes
–time=16:00:00
–cores-per-socket=5
–ntasks-per-node=2



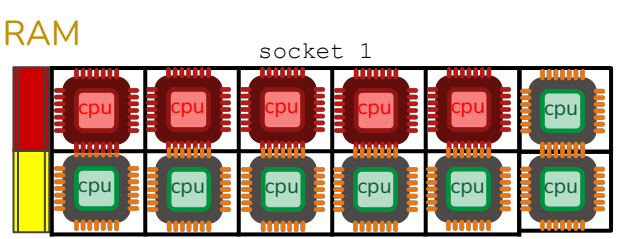


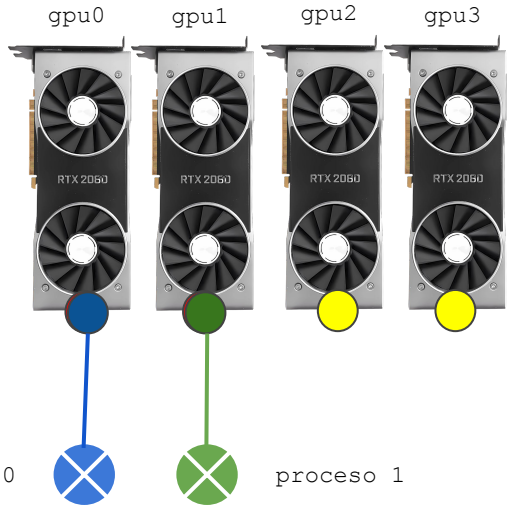
proceso 0 proceso 1

¿Cómo funciona un trabajo?

hermes

–nodes=1
–gres=gpu:A30:2
–p=hermes
–time=16:00:00
–cores-per-socket=5
–ntasks-per-node=2
–gpus-per-task=1

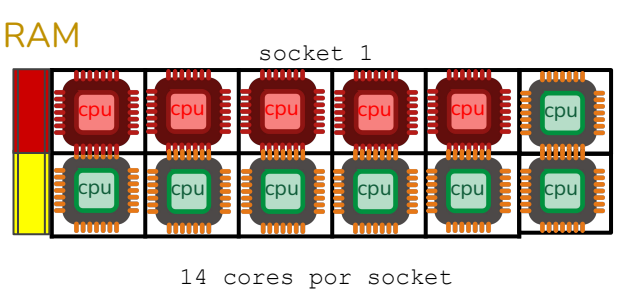


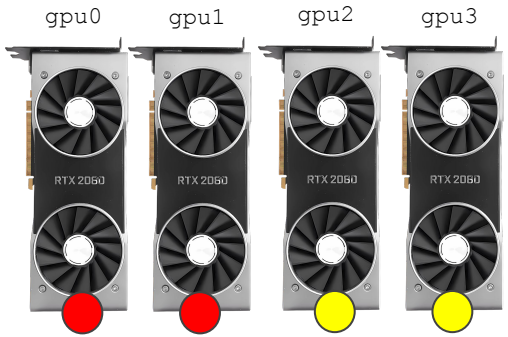


¿Cómo funciona un trabajo?

hermes

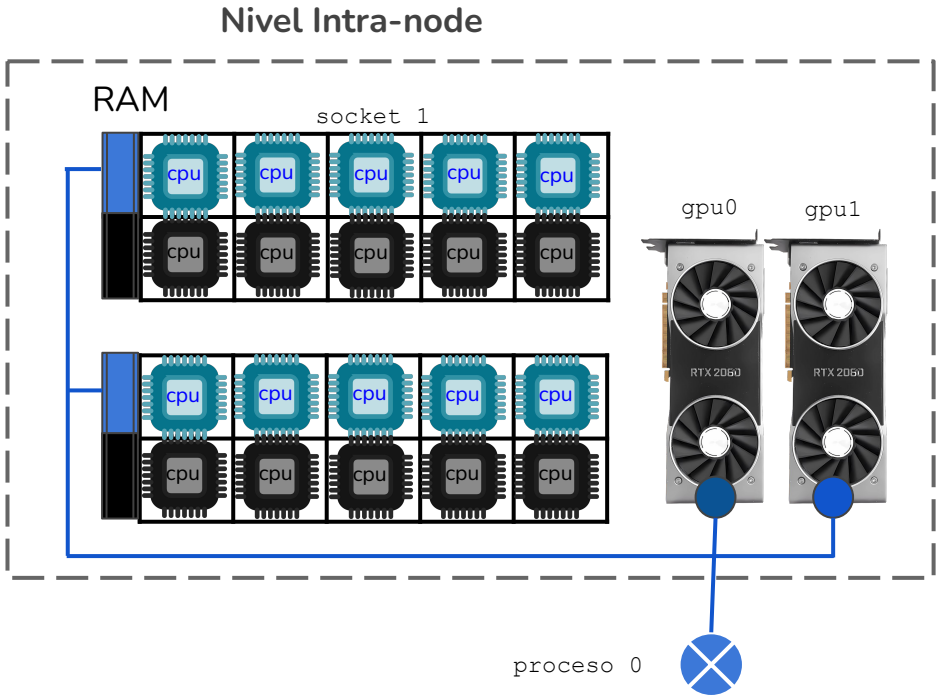
–nodes=1
–gres=gpu:A30:2
–p=hermes
–time=16:00:00
–cores-per-socket=5
~~–ntasks-per-node=2~~
~~–gpus-per-task=1~~



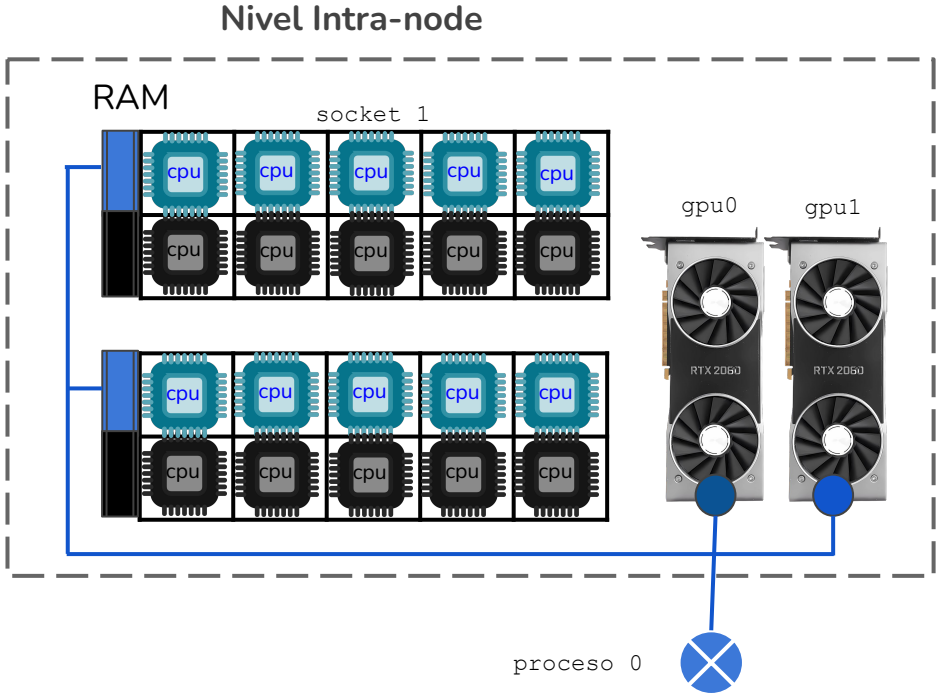


OJO, ES POSIBLE QUE SI, POR EJEMPLO, SE UTILIZA PYTORCH, INTERNAMENTE SE HAGA MULTIPROCESSING. ESTO TAMBIÉN PUEDE OCURRIR CON ACCELERATE.

¿Cómo funciona un trabajo?



¿Cómo funciona un trabajo?

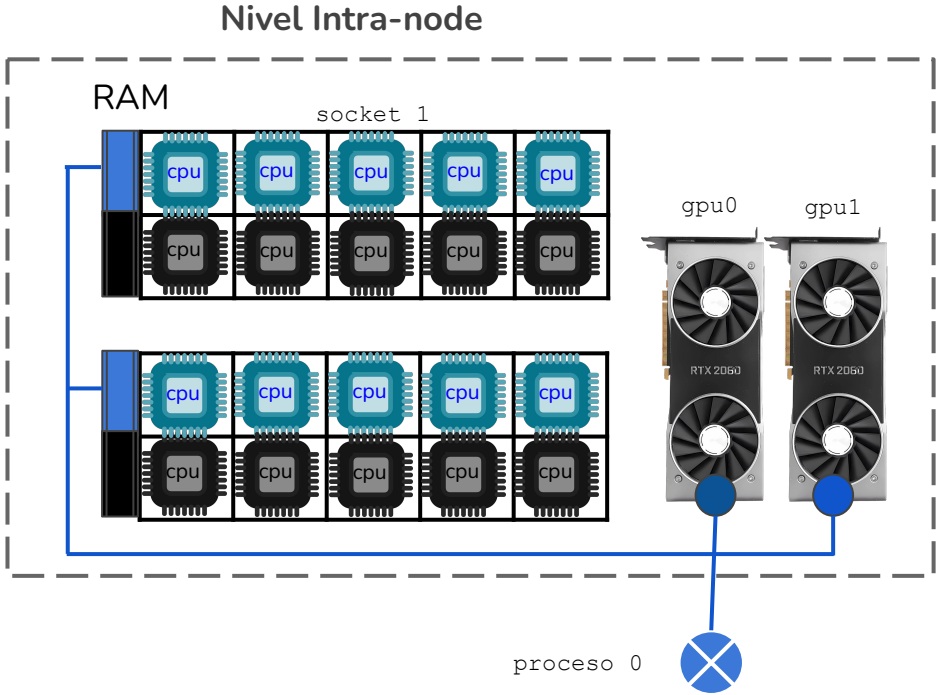


proceso 0

LAUNCH.SH

```
#!/bin/bash
CUDA_VISIBLE_DEVICES=0,1

torchrun
    --standalone
    --nnodes=1
    --nproc-per-node=2
    YOUR_TRAINING_SCRIPT.py (--arg1 ... train
script args...)
```

¿Cómo funciona un trabajo?



proceso 0

LAUNCH.SH

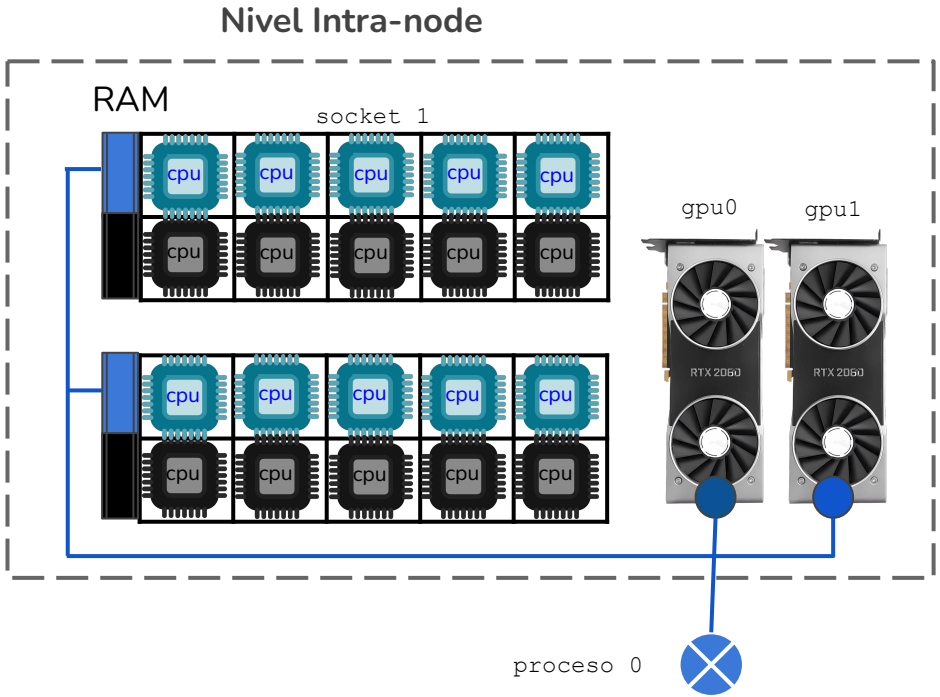
```
#!/bin/bash
CUDA_VISIBLE_DEVICES=0,1

torchrun
    --standalone
    --nnodes=1
    --nproc-per-node=2
    YOUR_TRAINING_SCRIPT.py (--arg1 ... train
script args...)
```

proceso 0,1

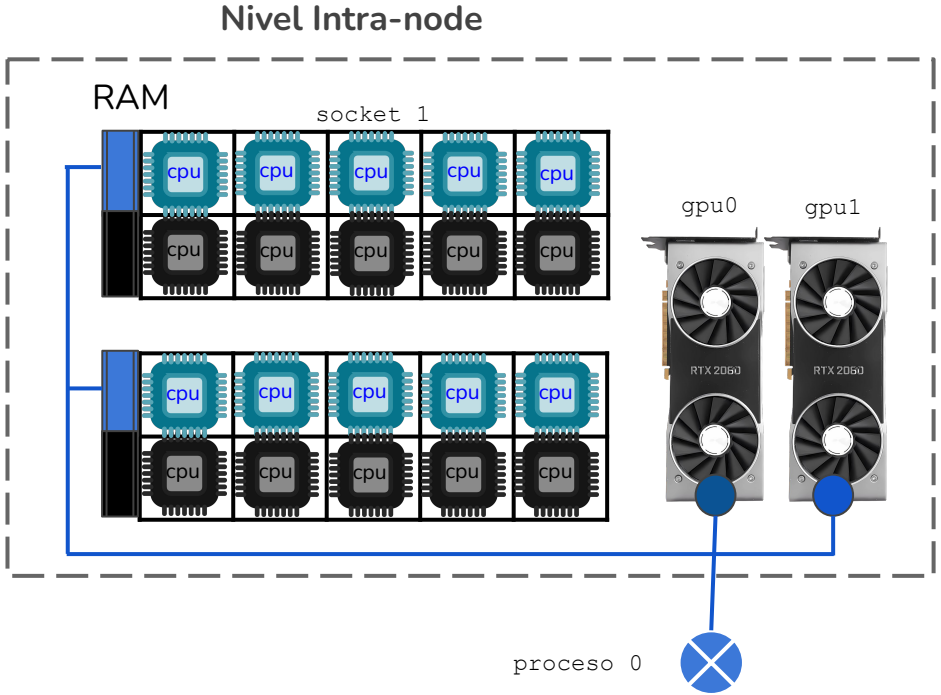
YOUR_TRAINING_SCRIPT.py

```
local_rank = int(os.environ["LOCAL_RANK"])
torch.cuda.set_device(local_rank) #0,1
```

¿Cómo funciona un trabajo?



¿Cómo funciona un trabajo?



proceso 0

LAUNCH.SH

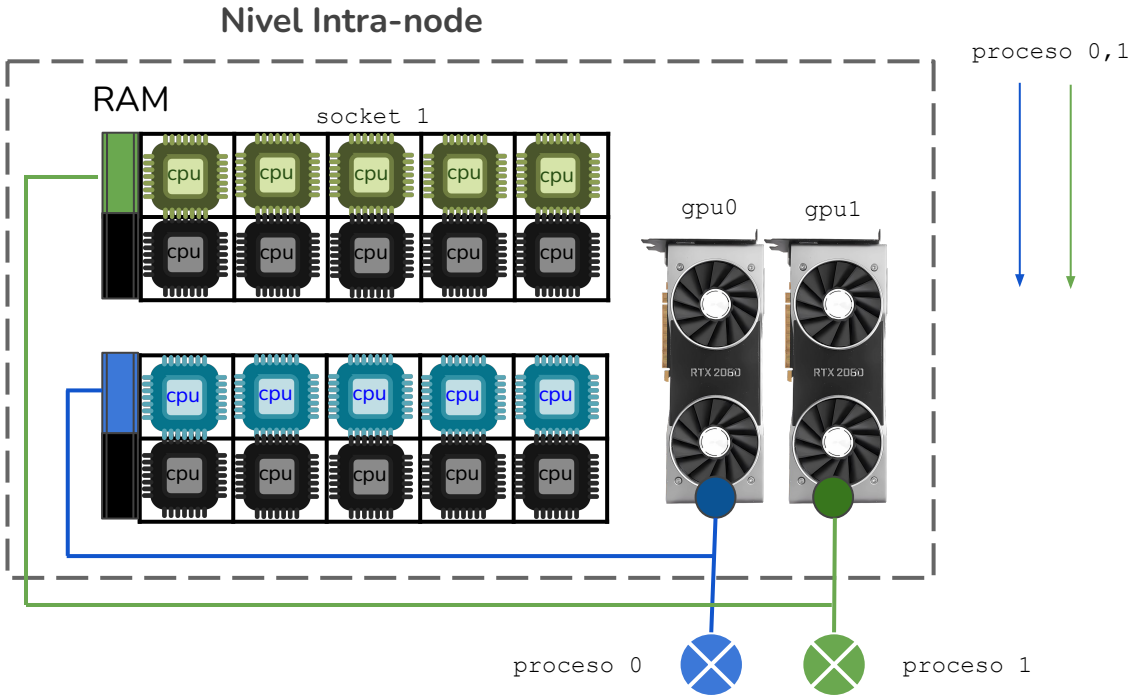
```
#!/bin/bash
CUDA_VISIBLE_DEVICES=0,1

python YOUR_TRAINING_SCRIPT.py (—id $SLURM_PROCID
--arg2 ... train script args...)
```

proceso 0,1

YOUR_TRAINING_SCRIPT.py

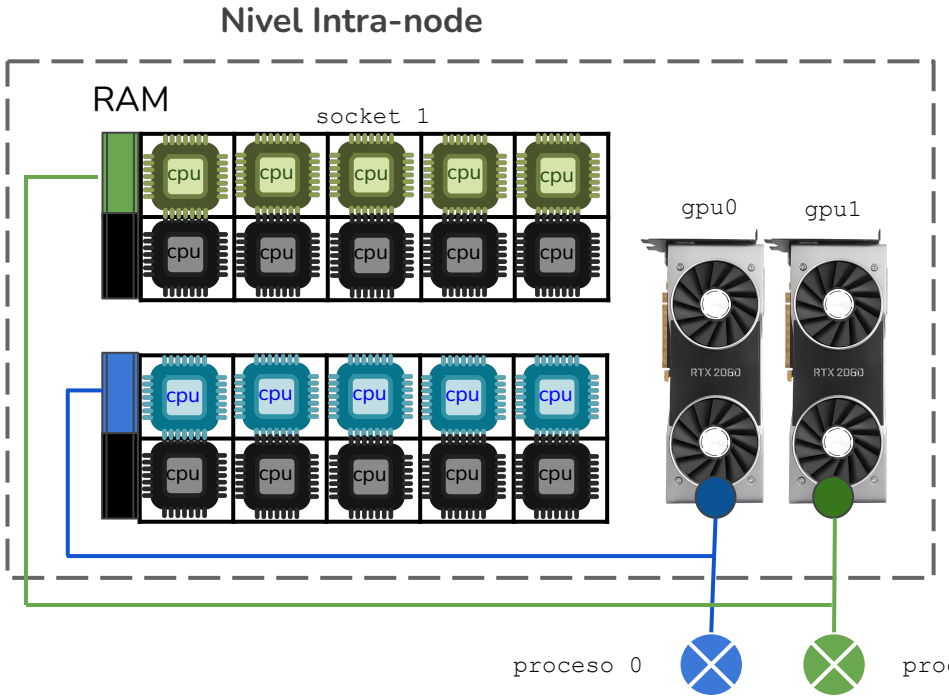
```
mp.spawn(nprocs=2)
local_rank = int(os.environ["LOCAL_RANK"])
torch.cuda.set_device(local_rank) #0,1
```

¿Cómo funciona un trabajo?



LAUNCH.SH

¿Cómo funciona un trabajo?

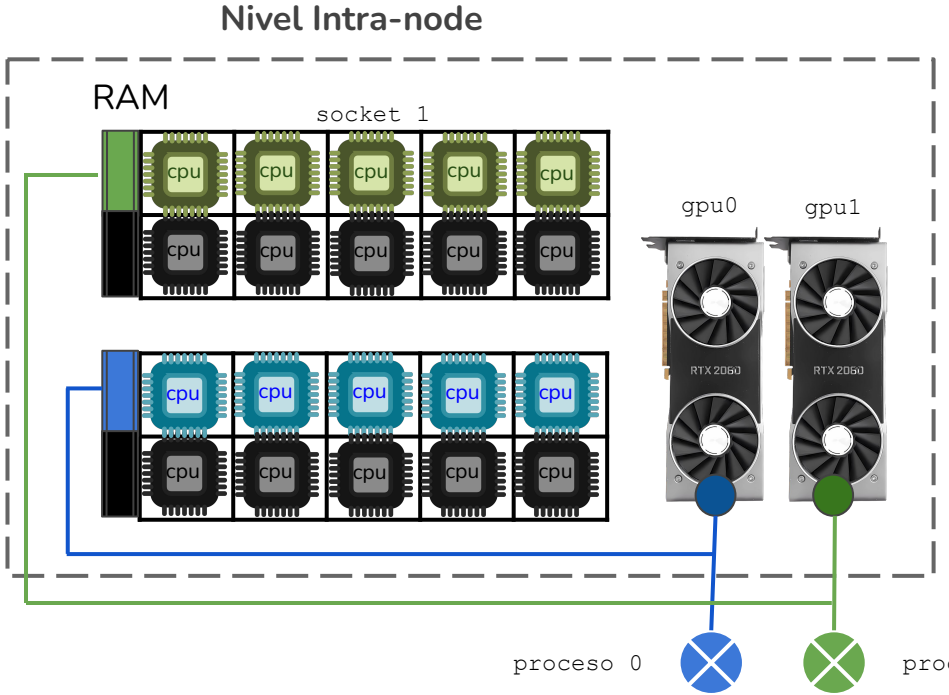


proceso 0,1

LAUNCH.SH

```
#!/bin/bash
CUDA_VISIBLE_DEVICES=0,1

python YOUR_TRAINING_SCRIPT.py (—id $SLURM_PROCID
--arg2 ... train script args...)
```

¿Cómo funciona un trabajo?



proceso 0,1

LAUNCH.SH

```
#!/bin/bash
CUDA_VISIBLE_DEVICES=0,1

python YOUR_TRAINING_SCRIPT.py (—id $SLURM_PROCID
--arg2 ... train script args...)
```

proceso 0,1

YOUR_TRAINING_SCRIPT.py

```
local_rank = int(args.id)
torch.cuda.set_device(local_rank) #0,1
```

¿Cómo funciona un trabajo?

Tres tipos de posibles salidas (principalmente):

- Modo interactivo:

salloc –nodes=1–gpus-per-node=2 etc..
(muestra la salida por pantalla continuamente, si abandonas la sesión, se da por finalizado el trabajo).

- Modo estático:

sbatch –nodes=1 –gpus-per-node=2 ./a.sh
(vuelca la salida en un fichero de texto con nombre JOBID.out).

srun –nodes=1 –gpus-per-node=2 python a.py (vuelca la salida en un fichero de texto con nombre JOBID.out).

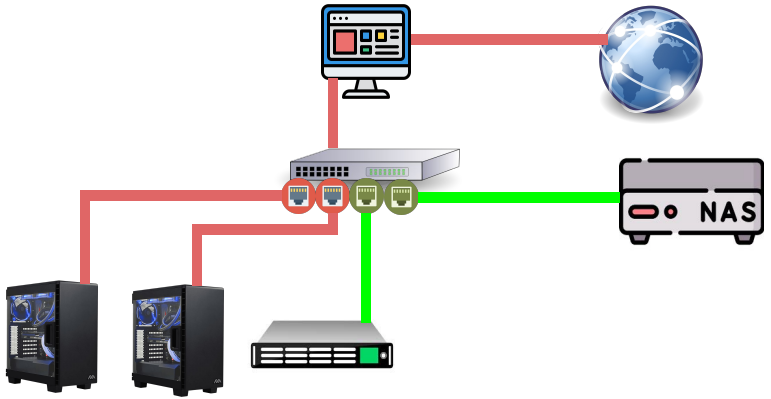
- Modo estático+email:

sbatch –nodes=1 –mail=user etc...

Configuración del clúster

slurm
workload manager

- El acceso a la máquinas no está permitido mediante SSH. Se debe conectar por ssh al nodo login (sirius).
- Todos los datos deben guardarse en /home/$USER/data, o en su defecto en /mnt/data/$USER. Instalaciones, datasets, códigos... TODO.
- Para facilitar la gestión, cada usuario no tiene límite de almacenamiento, pero se controlará.
- Existen diferentes niveles de prioridad de usuario para la cola (más tarde lo vemos).
- Las carpetas de datos son individuales. Se pueden crear carpetas compartidas para grupos de trabajo o proyectos (datos muy pesados y evitar replicados).



–p=A5000

inf-004-gpu-3

inf-004-gpu-4

conexión a 2.5G

–p=A30

hermes

conexión a 10G

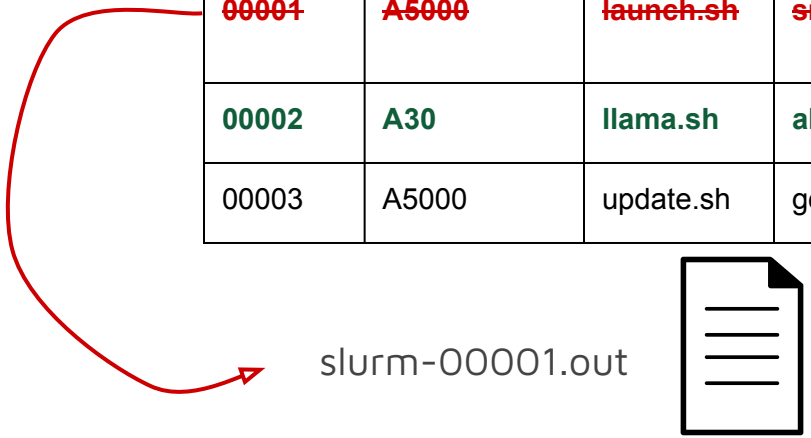
MOTIVOS:

- Heterogeneidad
- Velocidad de transferencia
- ¿¿ Unimos ??

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST (REASON)
00001	A5000	launch.sh	smoreno	R	15:32	2	inf-004-gpu-4, inf-004-gpu-3
00002	A30	llama.sh	alvarory	R	01:33	1	hermes
00003	A5000	update.sh	gestionlsi	PD	0:00	1	(RESOURCES)

slurm
workload manager

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST (REASON)
~~00001~~	~~A5000~~	~~launch.sh~~	~~smoreno~~	~~R~~	~~15:32~~	~~2~~	~~inf-004-gpu-4, inf-004-gpu-3~~
00002	A30	llama.sh	alvarory	R	01:33	1	hermes
00003	A5000	update.sh	gestionlsi	PD	0:00	1	(RESOURCES)



Prioridades del usuario en la cola

PRIORITY	TIME LIMIT
0	24:00:00
1	48:00:00
2	72:00:00

Estados del sistema

Podemos conocer el estado del sistema actualmente.
Comando: **sinfo**.

Estados del sistema

Podemos conocer el estado del sistema actualmente.
Comando: **sinfo**.

PARTITION	AVAIL	TIMELIMIT	NODES	STATE	NODELIST (REASON)
A5000	up	infinite	2	idle	inf-004-gpu-4, inf-004-gpu-3
A30	up	infinite	1	idle	hermes

Flujo de trabajo

Visión global del sistema

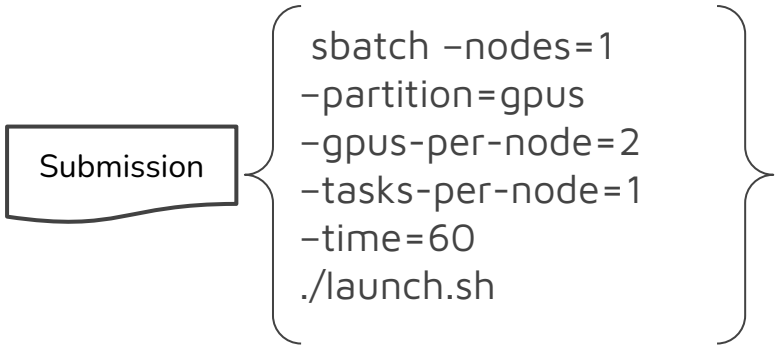
Flujo de ejecución de los trabajos

Visión global del sistema

Flujo de ejecución de los trabajos

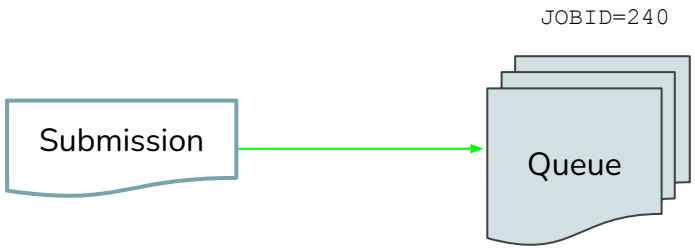
Visión global del sistema

Flujo de ejecución de los trabajos



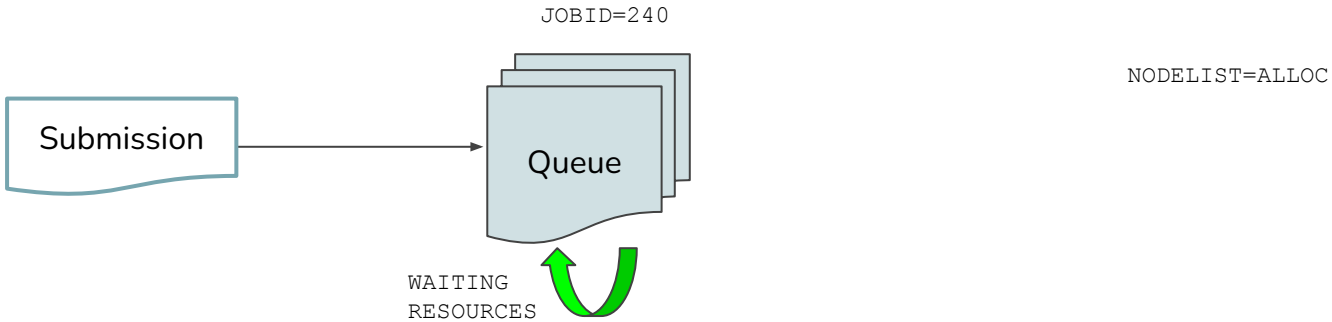
Visión global del sistema

Flujo de ejecución de los trabajos



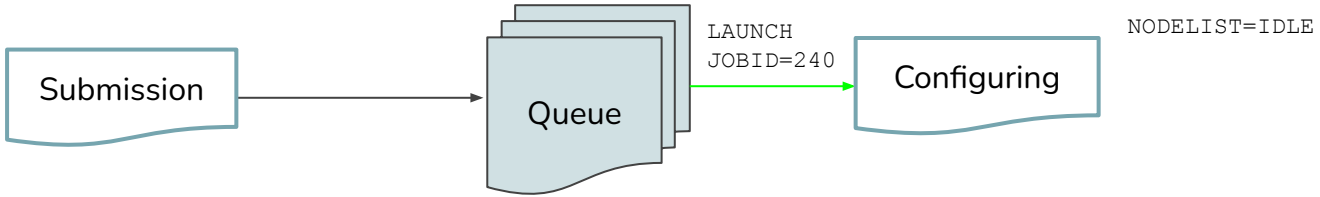
Visión global del sistema

Flujo de ejecución de los trabajos



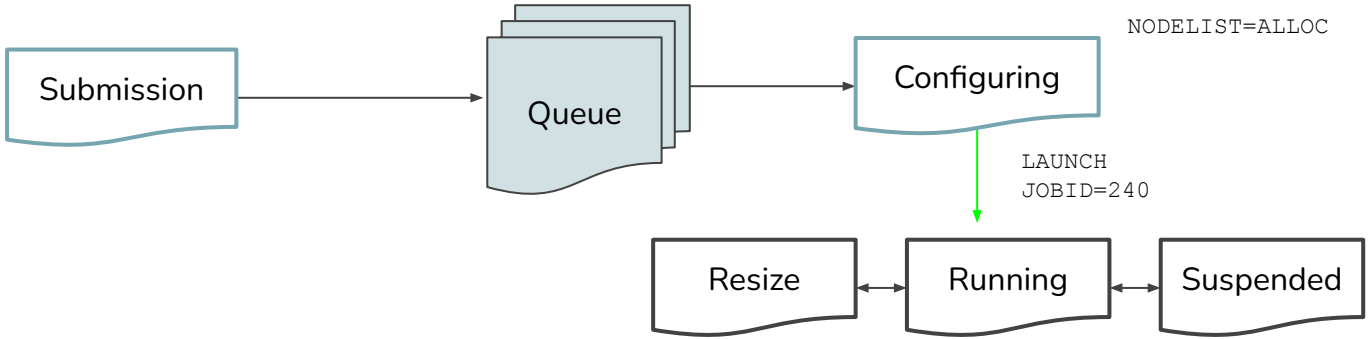
Visión global del sistema

Flujo de ejecución de los trabajos



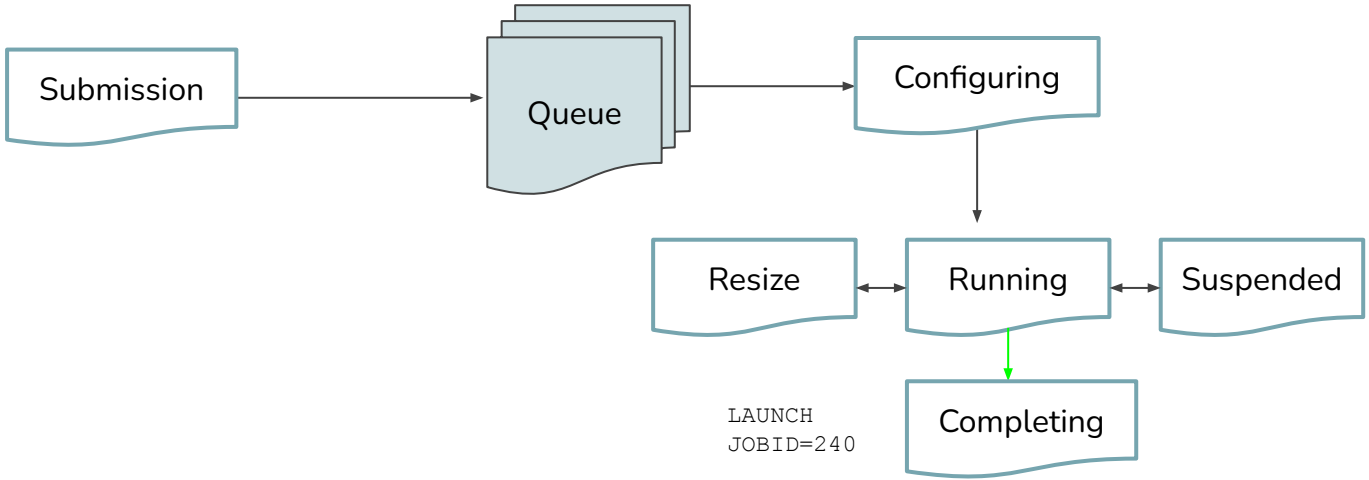
Visión global del sistema

Flujo de ejecución de los trabajos



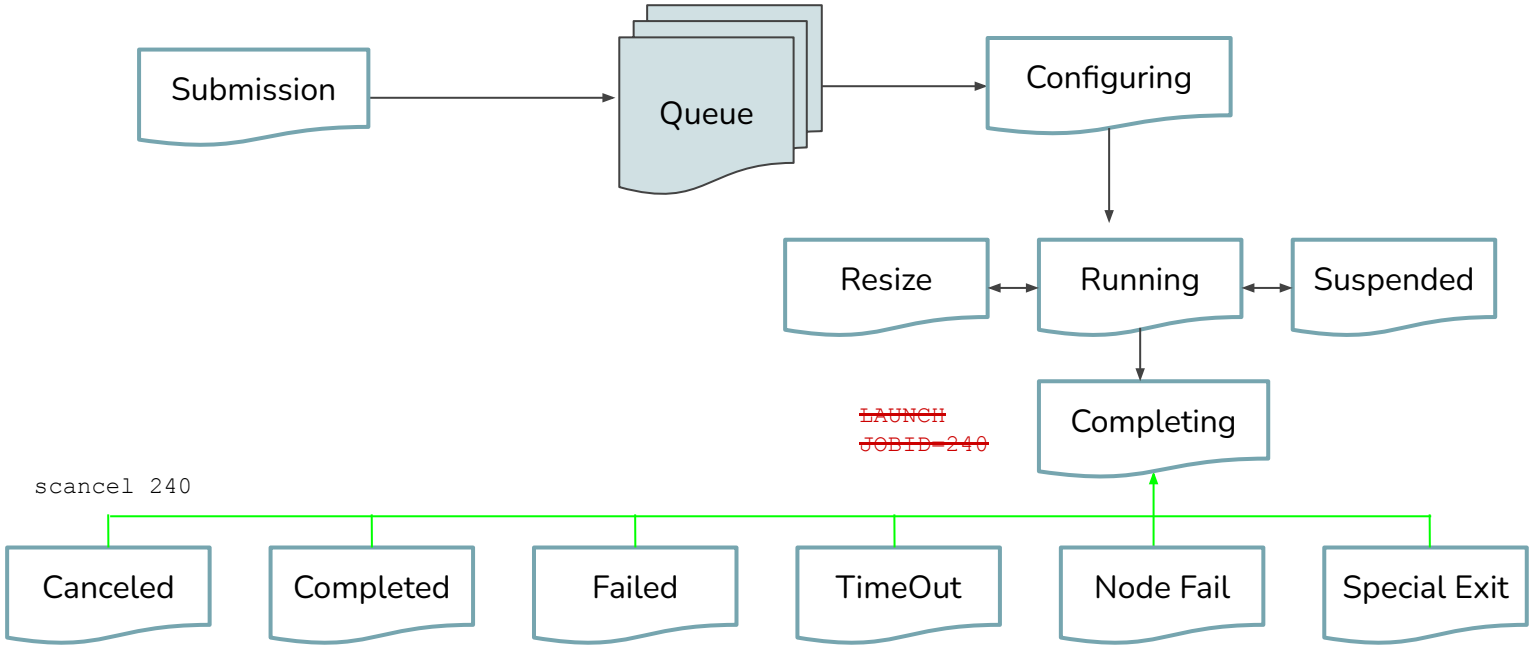
Visión global del sistema

Flujo de ejecución de los trabajos



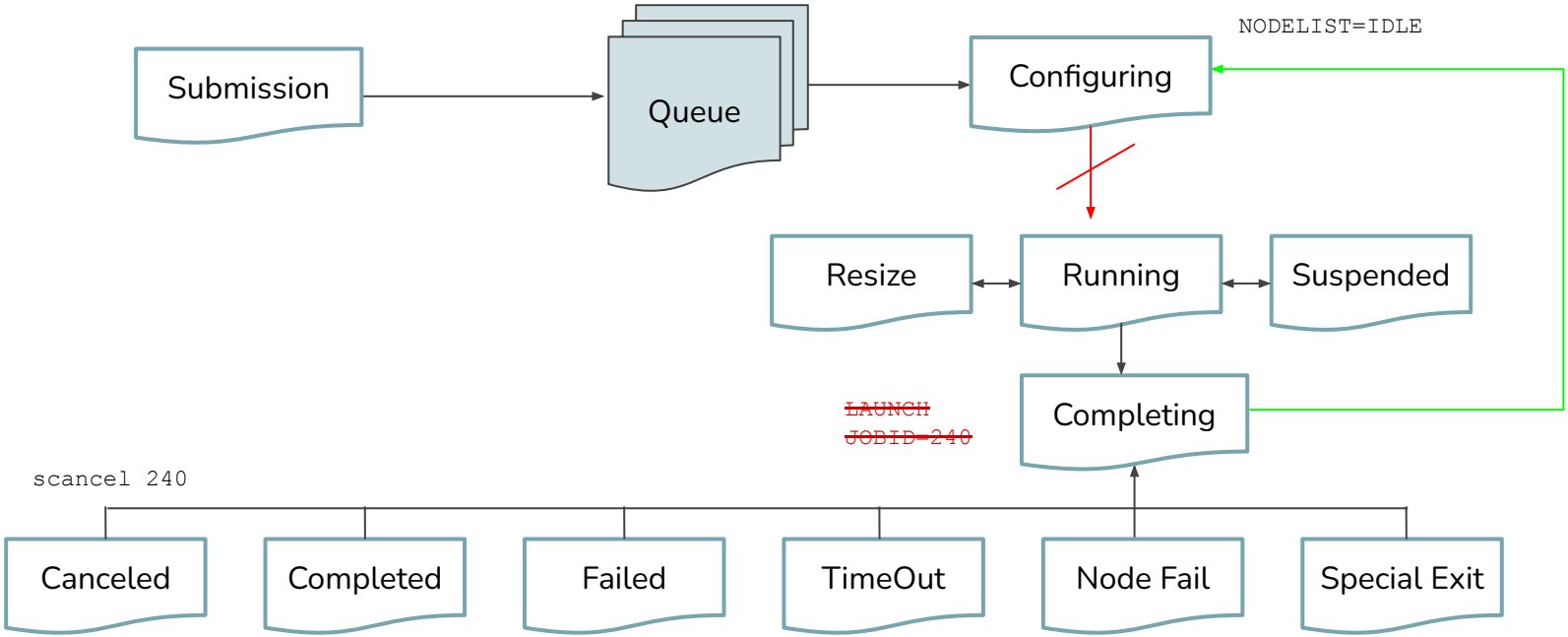
Visión global del sistema

Flujo de ejecución de los trabajos



Visión global del sistema

Flujo de ejecución de los trabajos



Cluster de cómputo y SLURM

https://github.com/smorenospace

Seminario 7 de mayo de 2024