

## Cluster de prácticas: KAHAN

J. M. Alonso, P. Alonso, F. Alvarruiz, I. Blanquer,  
J. Ibáñez, E. Ramos, J. E. Román

Departament de Sistemes Informàtics i Computació  
Universitat Politècnica de València

Curso 2024/25



1

## Contenido

- 1** Cluster de prácticas: KAHAN
  - Cluster de Prácticas
  - Ejecución de Programas Paralelos

2

## Apartado 1

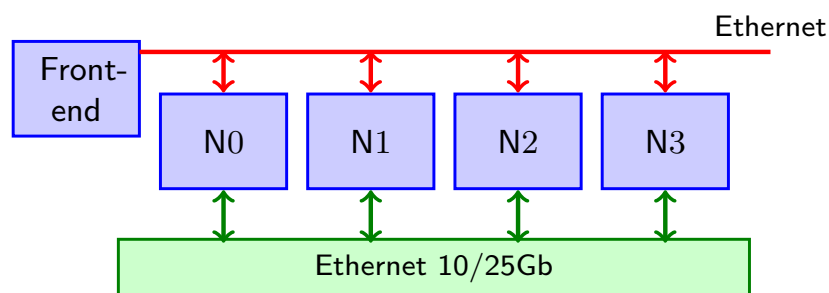
# Cluster de prácticas: KAHAN

- Cluster de Prácticas
- Ejecución de Programas Paralelos

3

## Cluster de Prácticas

Configuración hardware: 4 nodos



Cada nodo:

- 1 procesador AMD EPYC 7551P 32 núcleos (64 virtuales)
- 64GB de memoria RAM
- Disco SSD 240GB
- Ethernet 10/25Gb 2-port 622FLR -SFP28

Agregado: 4 proc, 128 núcleos (256 virtuales), 256 GB

4

## Cluster de Prácticas: Front-End

El nodo cabecera (*front-end*) permite a los usuarios interactuar con el cluster

Conexión:

```
$ ssh -Y -l login@alumno.upv.es kahan.dsic.upv.es
```

Para tareas rutinarias (no lanzar ejecuciones costosas)

- Edición y compilación de los programas
- Ejecuciones cortas para comprobar

Comandos útiles:

- Ficheros/directorios: `cd`, `pwd`, `ls`, `cp`, `mkdir`, `rm`, `mv`, `scp`, `less`, `cat`, `chmod`, `find`
- Procesos: `w`, `kill`, `ps`, `top`
- Editores y otros: `vim`, `emacs`, `pico`, `man`

5

## Cluster de Prácticas: Red

Gigabit Ethernet

- Red auxiliar, sólo para tráfico del S.O. (`ssh`, NFS)

Ethernet 10/25Gb

- Red rápida de baja latencia, ideal para clusters
- Tarjeta Ethernet 10/25Gb 2-port 622FLR -SFP28
- Soporta RDMA, RoCE, iWarp
- Puede funcionar a 25 Gbps

6

## Ejecución de Programas Paralelos

**OpenMP:** ejecutar directamente

Suele ser necesario indicar el número de hilos

```
$ OMP_NUM_THREADS=4 ./prgomp
```

Otra opción es exportar las variables

```
$ OMP_NUM_THREADS=4; OMP_SCHEDULE=dynamic  
$ export OMP_NUM_THREADS OMP_SCHEDULE  
$ ./prgomp
```

---

**MPI:** usar el comando `mpiexec` (o `mpirun`)

Opciones: seleccionar el host, la arquitectura

```
$ mpiexec -n 4 prgmpi <args>  
$ mpiexec -n 6 -host nodo1,nodo2,nodo5 prgmpi
```

7

## Sistemas de Colas

El **sistema de colas** (o **planificador de trabajos** o **gestor de recursos**) es un software que permite usar un cluster de forma compartida entre muchos usuarios

- El usuario puede lanzar “trabajos” normalmente en modo *batch* (no en interactivo) utilizando uno o más nodos
- Un **trabajo** (*job*) es una ejecución particular, con una serie de atributos (nodos, tiempo máximo de ejecución, etc.)
- Se definen políticas de **planificación** de trabajos
- El sistema contabiliza los recursos utilizados (horas)
- Objetivo: maximizar utilización, minimizar espera

Forma de trabajar:

- 1 Se define el trabajo y se lanza a la cola (da un identif.)
- 2 Tras un tiempo de espera, el trabajo se ejecuta
- 3 Al finalizar se recupera la salida producida

8

## Cluster de Prácticas: Colas (1)

Usaremos el sistema de colas SLURM.

### Ejemplo de trabajo jobopenmp.sh

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --time=5:00
#SBATCH --partition=cpa
OMP_NUM_THREADS=3 ./pintegral 1
```

- `--nodes`: número de nodos que se requieren
- `--time`: tiempo de ejecución requerido
- `--partition`: partición empleada en el sistema de colas

Para MPI usar `mpiexec` (no hace falta `-n`)

9

## Cluster de Prácticas: Colas (2)

Para lanzar:

```
$ sbatch jobopenmp.sh
Submitted batch job 728
```

Al terminar se crea en el directorio actual un fichero:  
`slurm-728.out`

Para ver el estado:

```
$ squeue
JOBID PARTITION NAME          USER ST TIME NODES NODELIST
728    cpa      jobopenmp.sh ramos R 0:01  1    kahan01
```

Posibles estados: encolado (PD), ejecutando (R), terminando (CD)

Cancelación de un trabajo: `scancel`

10