ACARUS

CAR

Responsable :

María del Carmen Heras Sánchez

Asesores Técnicos :

Daniel Mendoza Camacho

Yessica Vidal Quintanar

http://acarus.uson.mx

Día 4:

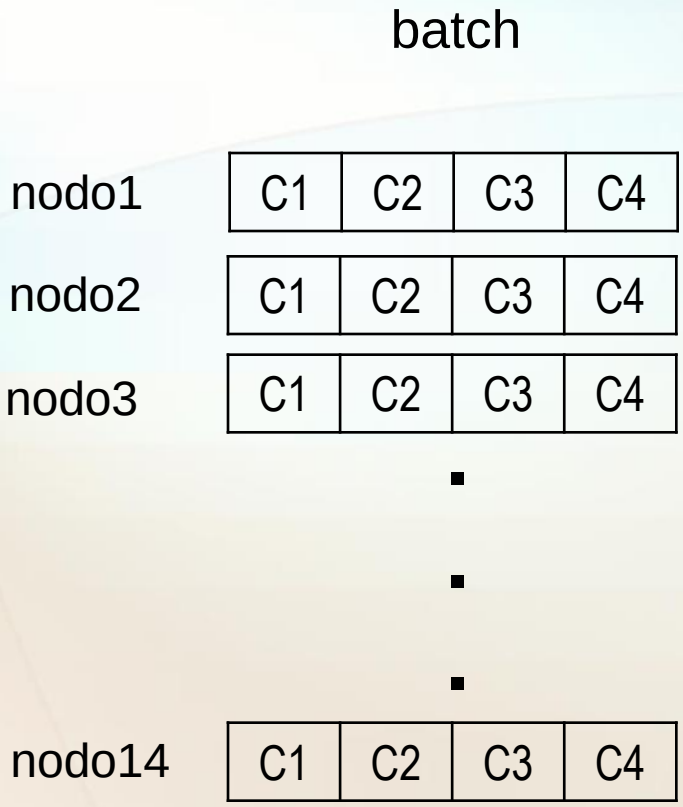
- ✓ **Sistema de Colas**

Antes de comenzar…

Pasos para ejecutar mi programa (job)

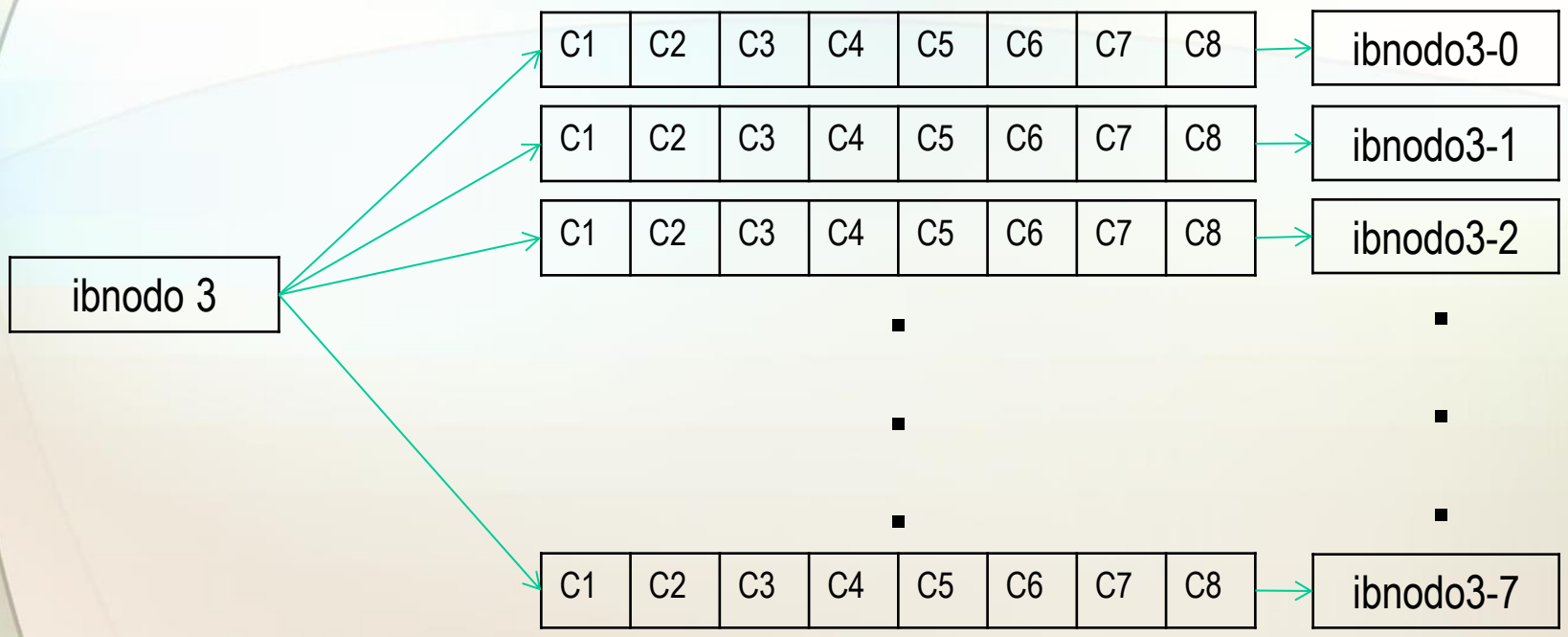
- Crear y compilar nuestro programa en C, Fortran, MPI o seleccionar uno de los programas como Gaussian, Gromacs, etc.
- Si es necesario compilar, utilizar un job interactivo para compilar en alguno de los nodos en los cuales el job se ejecutara.
- Crear un batch script para lanzar el job a ejecución.
- Lanzar el script ya sea utilizando el sistema de colas torque o slurm, dependerá de los recursos necesitados.
- Monitorear el estatus del job
- Ver los resultados.

Recursos choya

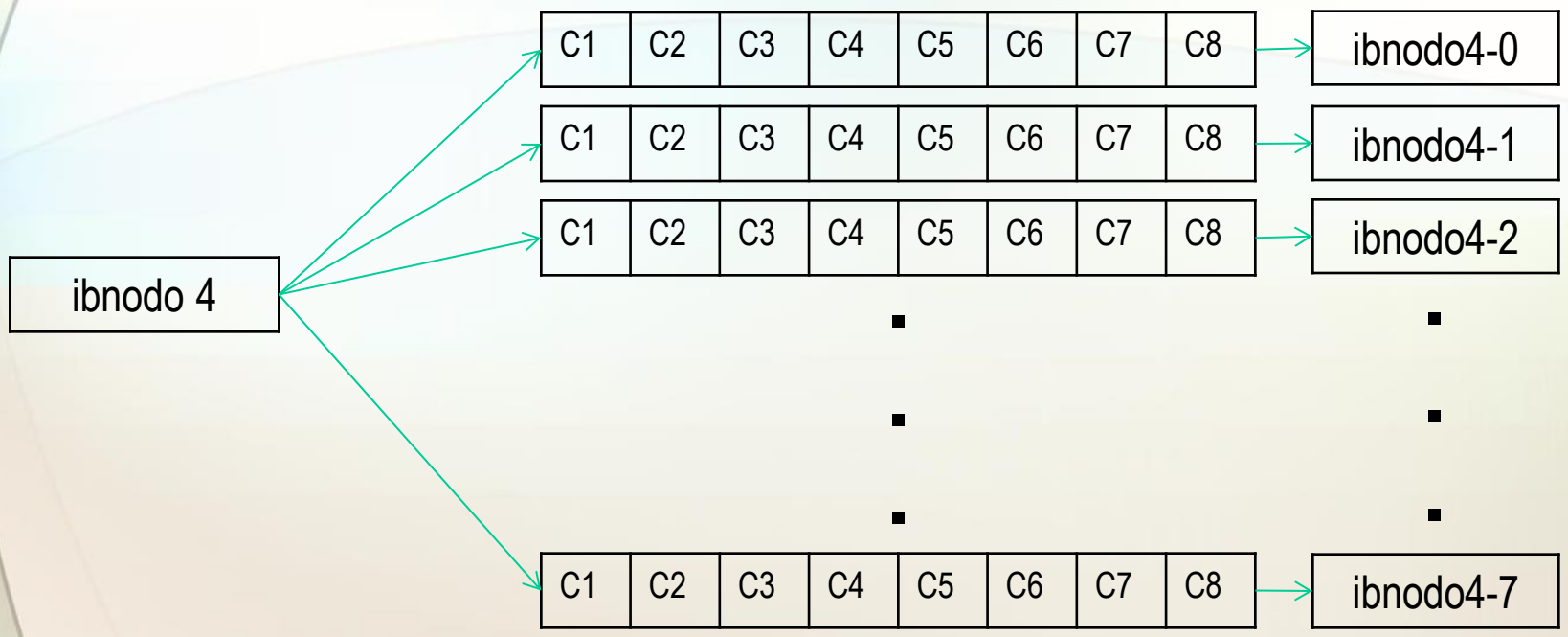


1 cola de ejecución (batch)
14 nodos
4 cores por nodo
1 Gb de RAM por core

Recursos ocotillo



Recursos ocotillo



Recursos ocotillo

q128b

ibnodo3-0	ibnodo4-0
ibnodo3-1	ibnodo4-1
ibnodo3-2	ibnodo4-2
ibnodo3-3	ibnodo4-3
ibnodo3-4	ibnodo4-4
ibnodo3-5	ibnodo4-5
ibnodo3-6	ibnodo4-6
ibnodo3-7	ibnodo4-7

q256b

ibnodo5-0	ibnodo6-0	ibnodo8-0	ibnodo9-0
ibnodo5-1	ibnodo6-1	ibnodo8-1	ibnodo9-1
ibnodo5-2	ibnodo6-2	ibnodo8-2	ibnodo9-2
ibnodo5-3	ibnodo6-3	ibnodo8-3	ibnodo9-3
ibnodo5-4	ibnodo6-4	ibnodo8-4	ibnodo9-4
ibnodo5-5	ibnodo6-5	ibnodo8-5	ibnodo9-5
ibnodo5-6	ibnodo6-6	ibnodo8-6	ibnodo9-6
ibnodo5-7	ibnodo6-7	ibnodo8-7	ibnodo9-7

Recursos ocotillo
CPU

Para utilizarse con el sistema de colas **Torque**

48 ibnodos
8 cores por ibnodo
2 Gb de RAM por core

3 colas de ejecución de hasta 128 cores
    q128b, q128c, q128d

1 colas de ejecución de hasta 256 cores
    q256b

1 cola de ejecución de hasta 512 cores
    q512a

Recursos ocotillo
GPU

Para utilizarse con el sistema de colas **Slurm**

Nodos de procesamiento
2 ibnodos (ibnodo1, ibnodo2)
64 cores por ibnodo
2 Gb de RAM por core

1 partición de 128 cores
gpu

Nodos de visualización
2 nodos (visualización1, visualizacion2)
8 cores por nodo
10 Gb de RAM por core

1 partición de 16 cores
visualizacion

Sistema de Colas

- Torque
- Maui

Es un gestor de recursos distribuidos que proporciona control sobre los trabajos por lotes y nodos de computación distribuida. **Torque** puede integrarse con el sw no comercial calendarizador de Cluster Maui o el comercial Moab para mejorar la utilización global, programación y administración en un clúster.

La ejecución de trabajos se lleva acabo vía scripts los cuales son analizados por el sistema de colas TORQUE para poder identificar el tipo y la cantidad de recursos de cómputo que necesita el trabajo, posteriormente MAUI ejecutará el trabajo en el script en el número de ibnodos solicitados.

En el script se definen las variables de ambiente necesarias, la secuencia de comandos y también es necesario indicarle al sistema de colas varias opciones para ejecutarlo

Comandos Básicos del Sistema de Colas Torque

qsub	Lanza un job
qsub –I	Lanza un job interactivo
qdel job_id	Cancela un job
qstat –a	Muestra el estatus de todos los jobs
qstat –n	Muestra el estatus de los jobs y los ibnodos asignados
qstat –r	Muestra los jobs en ejecución
showq	Muestra los jobs en ejecución, los encolados y los bloqueados
showbf	Muestra la disponibilidad
showbf –S	Muestra la disponibilidad detallada por ibnodo
pbsnodes –a	Muestra el estatus de los ibnodos
checkjob job_id	Muestra información detallada de un job

Opciones para Batch Script de Torque

#PBS –N	Nombre con el que identificara el job
#PBS –q	Nombre de la cola o partición
#PBS –l walltime	Tiempo máximo de ejecución
#PBS –l nodes=#:ppn=#	Número de nodos y cores por nodo solicitados

Script Trabajo Serial – Ocotillo torque

```
#!/bin/bash

#PBS -l nodes=1:ppn=1
#PBS -l walltime=00:30:00
#PBS -q q512a
#PBS -N serial

PBS_SCRATCH=/tmp/${USER}/${PBS_JOBID}
mkdir -p $PBS_SCRATCH
cd ${PBS_O_WORKDIR}

#Carga los Modulos de Open64
module load open64/5.0

#Ejecuta el programa
./hola_mundo_serial-co

#Elimina los archivos temporales
rm -rf $PBS_SCRATCH
```

Script Trabajo Paralelo – Ocotillo torque

```
#!/bin/bash

#PBS -l nodes=1:ppn=4
#PBS -l walltime=00:30:00
#PBS -N paralelo
#PBS -q q512a

export NCPUS=$(wc -l < $PBS_NODEFILE)
cd $PBS_O_WORKDIR

echo "ibnodos esclavos"
cat $PBS_NODEFILE

module load  gnu/openmpi-1.4.3

ENTRADA=./hola_mundo_mpi-c
SALIDA=hola_mundo_mpi.out

mpirun $ENTRADA > $SALIDA
```

Práctica 1

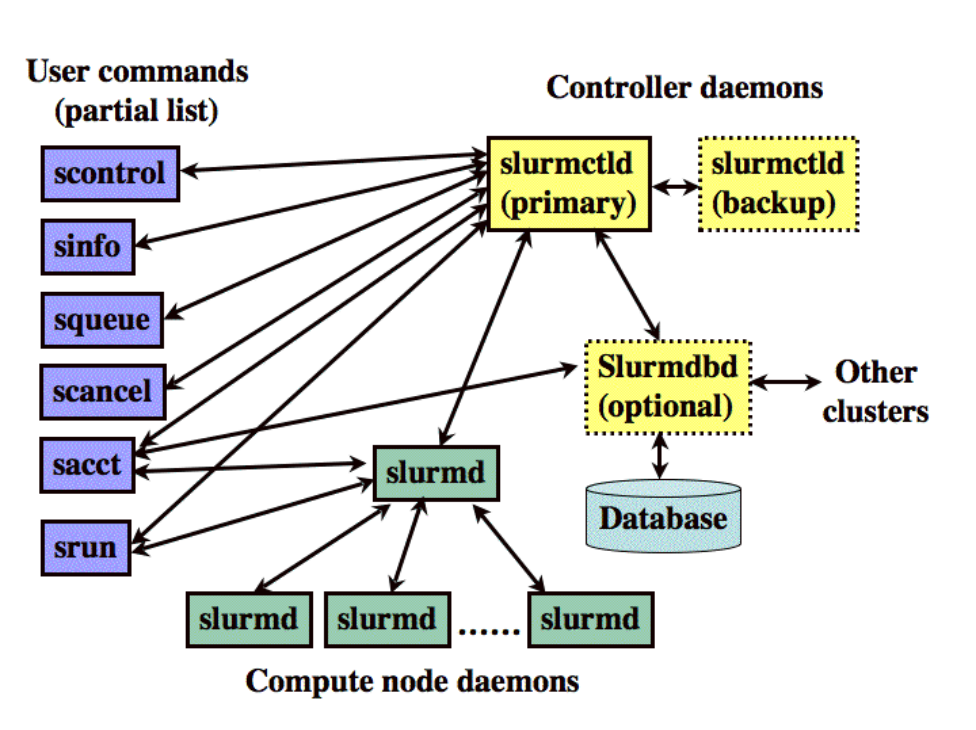
- Crear los dos scripts anteriores en su cuenta utilizando los programas ejecutables que anteriormente habían compilado.
- Lanzar los jobs al sistema de colas con torque utilizando qsub
- Monitorear el status del job con qstat
- Verificar el resultado en los archivos de salida

Sistema de Colas

- Slurm

The Simple Linux Utility for Resource Manager

Es un calendarizador de tareas open source, tolerante a fallos y altamente escalable para sistemas Linux grandes y pequeños



Comandos Básicos Slurm

srun	Ejecutar un comando en nodos de cómputo asignado
sbatch	Lanzar un job utilizando un script
squeue	Mostrar el estado de los jobs encolados
scancel	Cancelar un job
sinfo	Mostrar el estado de los nodos

Opciones para Batch Script de Slurm

--job-name	Nombre con el que identificara el job
--partition	Partición a la cual se lanzara el job
--time	Tiempo máximo de ejecución
--ntasks	Número de tareas
--nodes	Número de nodos
--ntasks-per-node	Numero de tareas por nodo

Script Trabajo Serial – Ocotillo slurm

```
#!/bin/bash

#SBATCH --ntasks=1
#SBATCH --partition=visualizacion
#SBATCH --job-name=serial
#SBATCH --time=00:30:00

./hola_mundo_serial-co
```

Script Trabajo Paralelo – Ocotillo slurm

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --ntasks-per-node=8
#SBATCH --partition=visualizacion
#SBATCH --time=00:30:00
#SBATCH --job-name=mpi

module load gnu/openmpi-1.5.3

mpirun ./hola_mundo_mpi
```

Práctica 2

- Crear los dos scripts anteriores en su cuenta utilizando los programas ejecutables que anteriormente habían compilado.
- Lanzar los jobs al sistema de colas con slurm utilizando sbatch
- Monitorear el status del job con sinfo y squeue
- Verificar el resultado en los archivos de salida

Práctica 3

- Hacer el ejercicio anterior en el cluster choya

Práctica 4

- Correr el ejemplo de multiplicación de matrices con mpi en choya. El archivo fuente se encuentra en /tmp/curso2016/multmat.c en el cluster ocotillo. Hay que copiarlo al cluster choya
- Pueden compilarlo con cualquiera de las opciones disponibles de mpi
  - gnu/mpich-3.1.2 (GCC 4.4.7)
  - gnu/openmpi-1.8.3 (GCC 4.4.7)
  - intel/openmpi-1.8.3 (Intel Parallel Studio XE 2013)