Sesión Práctica – Uso de Guacolda-Leftraru

16 de noviembre de 2020

1. Introducción

A lo largo de esta sesión se realizarán ejercicios prácticos que permitirán adquirir conocimientos para usar un sistema gestor de recursos computacionales, tal como es el caso de *SLURM*, el sistema que tenemos instalado en Guacolda-Leftraru.

2. Metodología

La clase se dividirá en varios grupos de trabajo y en cada uno de los grupos habrá un coordinador.
Para cada uno de los ejercicios:

1. El profesor plantea el ejercicio y resuelve posibles dudas de los alumnos.
2. Se divide la clase en grupos por un tiempo prefijado. Este tiempo inicialmente será de 5 minutos. El profesor avisará si el tiempo para resolver un ejercicio concreto es distinto al señalado.
3. Dentro de cada uno de los grupos, el coordinador compartirá pantalla y entre todos resolverán el ejercicio de manera colaborativa. El profesor irá entrando a algunos de los grupos para ver si necesitan algún tipo de apoyo. NOTA: no hace falta anotar los comandos y respuestas de los ejercicios, tan sólo tener clara la respuesta para poder presentarla.
4. Al finalizar un temporizador, todos los grupos volverán a la sala principal.
5. El profesor seleccionará un grupo al azar y su coordinador compartirá pantalla para presentar la resolución del ejercicio. El resto de alumnos podrán intervenir para completar o corregir la solución planteada.

3. Ejercicios

3.1. Ejercicio 1

1. Lanza en la partición *slims* de Leftraru ejecutando el comando `hostname` con `srun`:
   a) Con 1 único proceso.
   b) Con 2 procesos iguales.
   c) Con 2 procesos iguales en distintos nodos.
   d) Lanzando un proceso que tenga dos *cores* reservados.
2. Anota cada uno de los comandos ejecutados y la salida de los mismos.
3. En la partición *slims* ¿cuántos *cores* puedes reservar por proceso? ¿Por qué es ese número? ¿Es igual en la partición *general*? ¿Qué ocurre si reservas más *cores* de los disponibles?
4. ¿Qué ocurre si no especificas la partición al ejecutar?

3.2. Ejercicio 2

1. Crea un *script* de ejecución para lanzarlo con `sbatch`, con las siguientes consideraciones:
   a) La partición a lanzar es *slims*.
   b) Reserva un único *core*.
   c) Ejecuta el comando `stress -c 1`.
2. Anota el *script* que has usado.
3. El comando `stress` sirve para poner a prueba los distintos componentes de un computador. En este ejemplo estamos pidiendo usar una *CPU* (al 100 %) durante un tiempo ilimitado. Ya que no se le ha especificado al comando un tiempo de término, en principio, la tarea no debiera terminar nunca. En relación a esto:
   a) ¿Cuánto tiempo estará la tarea corriendo hasta que sea cancelada por el sistema? ¿Qué comando debes de usar para obtener información sobre un trabajo que ya está corriendo?
   b) ¿Qué comando puedes usar para obtener información acerca de las particiones?
   c) Cancela la tarea. ¿Qué comando/s has usado para cancelarla?

3.3. Ejercicio 3

1. Crea un *script* de ejecución para lanzarlo con `sbatch`, con las siguientes consideraciones:
   a) La partición a lanzar es *slims*.
   b) Reserva un nodo completo.
   c) Ejecuta el comando `stress -c 40 -t 15m`.
2. Anota el *script* que has usado.
3. ¿Cuántos recursos de *CPU* estás usando? Muestra una captura de pantalla que muestre gráficamente lo que ocurre con la carga de *CPU* en el nodo donde se está ejecutando la tarea.
4. Entra al nodo donde se está ejecutando la tarea, ejecuta el comando `htop` e interpreta lo que está pasando.
5. ¿Cuál sería la manera correcta de lanzar la ejecución para usar todos los *cores* disponibles en un nodo? Inserta en el informe una captura de pantalla que muestre gráficamente lo que ocurre con la carga de *CPU* en el nodo donde se está ejecutando la tarea.

3.4. Ejercicio 4

1. Crea un *script* de ejecución para lanzarlo con `sbatch`, con las siguientes consideraciones:
   a) La partición a lanzar es *slims*.
   b) Reserva un único *core*.
   c) Indica que envíe un correo cuando el *job* cambie de estado.
   d) Ejecuta el comando `stress -m 1 --vm-bytes 2048M -t 15m`.
2. Anota el *script* que has usado.
3. ¿Qué está ocurriendo con la ejecución? ¿a qué se debe?
4. Relanza la tarea modificando el *script* para poder ejecutar la tarea. Anota el *script*.
5. ¿Cuántos recursos de *RAM* estás usando? Inserta en el informe una captura de pantalla que muestre gráficamente lo que ocurre con la carga de *RAM* en el nodo donde se está ejecutando la tarea.

3.5. Ejercicio 5

1. Descarga el siguiente código: n-queens-problem-3.py en tu directorio de trabajo.
2. Crea un *script* de ejecución para lanzarlo con `sbatch`, con las siguientes consideraciones:
   a) La partición a lanzar es *slims*.
   b) Cada *job* reserva un único *core*.
   c) Supón que cada proceso ocupa 2400MB de memoria RAM.
   d) Ejecuta el código con la versión de Python 3 más actualizada del sistema .
3. Anota el *script* que has usado.
4. Anota la salida de la ejecución.

3.6. Ejercicio 6

1. Imagina que deseas ejecutar el programa `namd`.
   a) ¿Qué opciones tienes disponibles en la infraestructura del NLHPC?
   b) ¿Qué diferencias hay entre ellas?
   c) Si quieres lanzar usando GPUs, ¿qué módulos debes de cargar?
2. Anota los comandos que has usado.