EXAMEN DE PRÁCTICAS AC bp0

Cluster de prácticas (atcgrid): componentes

Nodos de cómputo (atcgrid[1-4]):

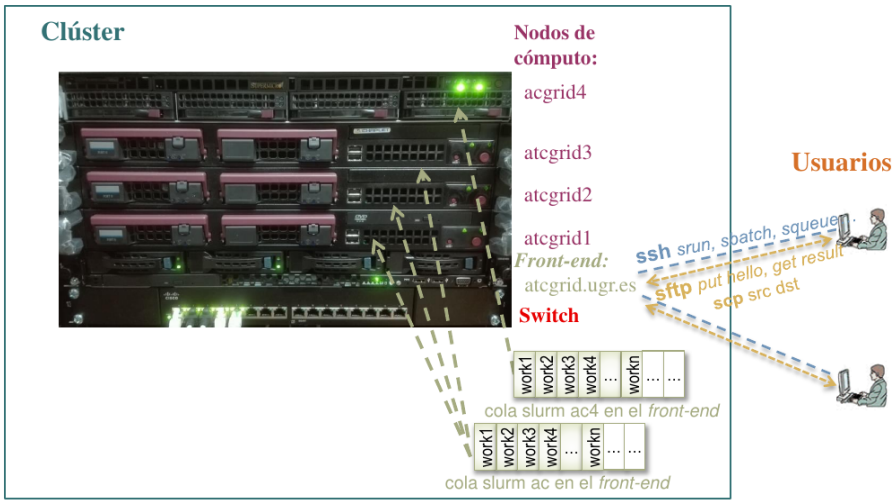
➔ Tres servidores rack SuperMicro SuperServer 6016T-T (atcgrid[1-3])
http://www.supermicro.com/products/system/1U/6016/SYS-6016T-T.cfm

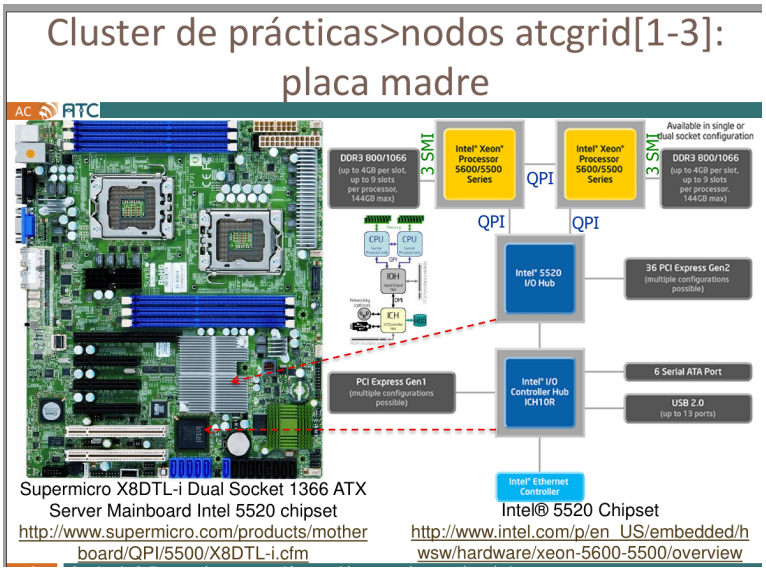
➔ Un servidor rack SuperMicro SYS-6019U-TR4 1U (atcgrid4)
https://www.supermicro.com/products/system/1u/6019/SYS-6019U-TR4.cfm

Cables

Switch

Nodo *front-end* (*host, master*):
Asus RS300-E9-PS4

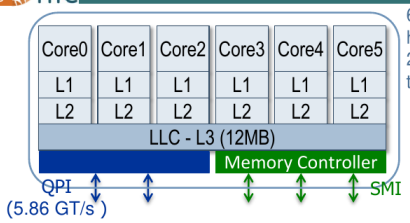




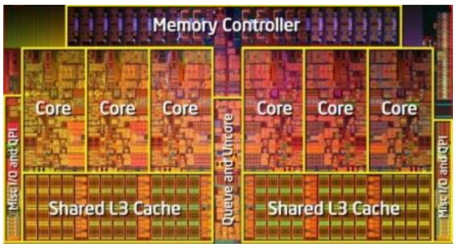
Supermicro X8DTL-i Dual Socket 1366 ATX Server Mainboard Intel 5520 chipset
http://www.supermicro.com/products/motherboard/QPI/5500/X8DTL-i.cfm

Intel® 5520 Chipset
http://www.intel.com/p/en_US/embedded/hwsw/hardware/xeon-5600-5500/overview

Cluster de prácticas>nodos atcgrid[1-3]: chip de procesamiento

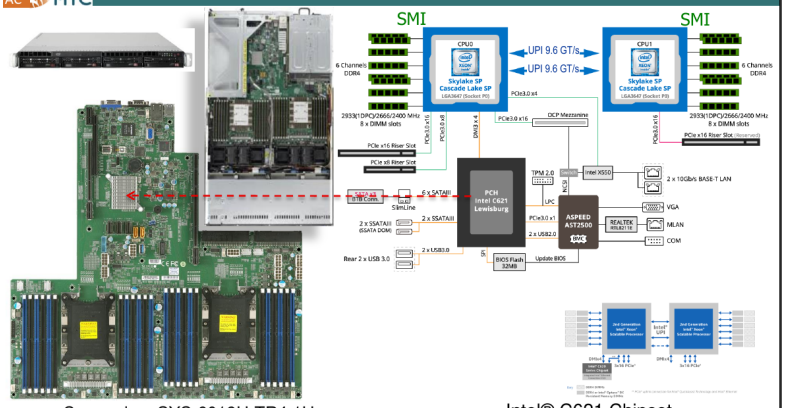


6 cores hyperthreading 2.4 GHz (12 threads)



Intel Xeon E5645 (6 cores/12 threads, 12M L3 Cache compartida, 2.40 GHz cada core, 5.86 GT/s Intel® QPI)
http://ark.intel.com/products/48768?wapkw=(E5645)

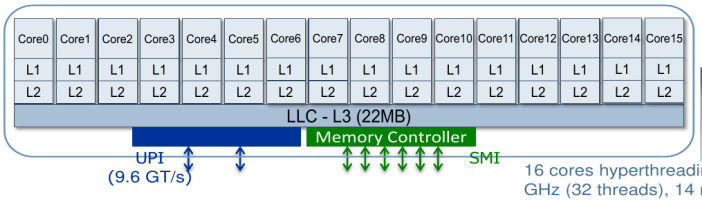
Cluster de prácticas> nodo atcgrid4: placa madre



Supermicro SYS-6019U-TR4 1U
https://www.supermicro.com/en/products/motherboard/X11DPU

Intel® C621 Chipset
Intel® C621 Chipset Product Specifications

Cluster de prácticas (atcgrid4): chip de procesamiento de la CPU



16 cores hyperthreading 2.1 GHz (32 threads), 14 nm

Intel® Xeon® Silver 4216 (16 cores/32 threads, 22MB L3 Cache compartida, 2.10 GHz máxima 3,2 GHz, cada core, 9.6 GT/s Intel® UPI)

Ejemplo con 18 núcleos

Cluster de prácticas (atcgrid): acceso

- Tiene un nodo frontend llamado atcgrid:
- 4 nodos de cómputo:
    - atcgrid1, 2 y 3 poseen 12 cores físicos y 24 lógicos.
        - ac es la cola en slurm.
    - atcgrid4 posee 32 cores físicos y 64 cores lógicos.
        - ac4 es la cola en slurm
- Cada usuario tiene un home en el nodo **front-end** del clúster atcgrid que se puede acceder dentro de la VPN con :
- SSH: **ssh -X username@atcgrid.ugr.es** + **password** (para realizar ejecuciones)
- **sftp (secure file transfer protocol) (**Para cargar y descargar ficheros se usa el cliente / o para transferir archivos)**:**

    $ sftp username@atcgrid.ugr.es + **password**

    $ put nombre_del_fichero

Gestor de carga de trabajo

La ejecución de los siguientes comandos se llevará a cabo en el front-end, con conexión ssh:

COMANDOS:
Slurm

➢ **Slurm** se usa como gestor de colas.

Ejemplos con comandos slurm

❖ **srun -pac -Ac ./hello // srun -p ac4 -A ac lscpu**
srun envía a ejecutar un trabajo a través de una cola slurm directamente, puede ser ejecutable. La ejecución de srun es interactiva.
Sí aparece **-p**, se envía a nodos de la cola especificada con **-p** (un trabajo solo puede usar un nodo de la **cola ac**).

❖ **sbatch -p ac script.sh // sbatch -p ac —wrap "echo Hola" // sbatch -p ac —wrap "./hello" // sbatch -p ac —wrap "echo Hola ; ./hello"**
sbatch envía a **ejecutar** un **script** (en este caso script.sh, "echo Hola", "./hello" y "echo Hola ; ./hello") a través de una **slurm**. La **salida se devuelve**

en un fichero. La ejecución con **srun** es **interactiva**, con **sbatch** es en **segundo plano.** Se recomienda usar **sbatch**.
Internamente en el script debe llamarse a srun ./ejecutable sin más parámetros, se heredan los parámetros usados en **sbatch**.
Si se usa –wrap, permite convertir lo que está entre comillas en un script.

- **squeue**
  Muestra todos los **trabajos** en **ejecución** y los que están **encolados**.

- **scancel jobid**
  Elimina el **trabajo** con identificador **jobid.**

- **sinfo**
  Lista información de las **particiones (colas)** y de los **nodos.**

- **sinfo -p ac -o"%10D %10G %20b %f"**
  Lista todos los **nodos (D), los recursos (G),** y las **características activas (b)** y **disponibles (f)** en la partición especificada (-p) (%[[.]size]type[suffix])

- **sbatch -pac -n1 -c12 script.sh** // **srun -pac -n1 -c12 hellomp**
  Slurm está configurado para asignar recursos a los procesos (llamados tacks en slurm) a nivel de core físico. Esto significa que por defecto slurm asigna un core a un proceso. Para asignar x, se debe usar con **sbatch**/**srun** la opción **—cpus-per-task**=**x** (abreviada de la forma **-cx**).
  Para asegurar que solo se crea un proceso hay que incluir **—ntasks**=**1 (-n1)** en sbatch/srun.

- **sbatch -pac -n1 -c12 —hint**=**nomultithread script.sh** //
  sbatch -pac -n1 -c12 —hint=**nomultithread helloomp**
  En **Slrum**, por defecto, **cpu** se refiere a cores lógicos. Sí no se quieren usar cores lógicos hay que añadir la opción **-hint**=**nomultithread a sbatch**/**srun.**
  Para que con **sbatch** se tenga en cuenta —-hint=nomultithread se debe usar srun dentro del script delante del ejecutable.

- **sbatch -pac -n1 -c12 —exclusive —hint**=**nomultithread script.sh** //
  sbatch -pac -n1 -c12 —exclusive —hint=**nomultithread helloomp**

Para que no se ejecute más de un proceso en un nodo de cómputo atcgrid hay que usar **—exclusive** con sbatch/srun (se recomienda no utilizarlo en los srun dentro de un script).

- Para mandar trabajos se utiliza:
  - **srun -p<cola> -Aac -n1 -c<x> --hint=nomultithread --exclusive ./ejecutable**
  - **sbatch -p<cola> -Aac -n1 -c<x> --hint=nomultithread --exclusive script.sh**
  - **sbatch -p<cola> -Aac -n1 -c<x> --hint=nomultithread --exclusive --wrap "./ejecutable"**
  - -p <cola> indica la partición a utilizar, <cola> puede ser ac o ac4
  - -A indica la cuenta a utilizar, siempre se usa la "ac".
  - -n1 indica que se debe ejecutar en una sola tarea.
  - -c<x> indica que <x> es el número de CPUs a utilizar
  - --hint=nomultithread es para evitar que se utilicen más hebras que CPUs físicos.
  - --exclusive indica que la ejecución no puede compartir nodos con otros trabajos.

- **Variables de entorno de SLURM**

SBATCH --job-name=helloOMP2	Ponerle nombre al trabajo
SBATCH --partition=ac	Partición a utilizar
SBATCH --account=ac	Cuenta a utilizar
$SLURM_JOB_USER	ID del usuario del trabajo
$SLURM_JOBID	ID del trabajo
$SLURM_JOB_NAME	Nombre del trabajo dado por el usuario
$SLURM_SUBMIT_DIR	Directorio donde se ha llamado al script
$SLURM_JOB_PARTITION	Cola
$SLURM_SUBMIT_HOST	Nodo que ejecuta el trabajo, o el host.
$SLURM_JOB_NUM_NODES	Número de nodos asignados al trabajo.
$SLURM_JOB_NODELIST	Nodos asignados al trabajo
SLURM_JOB_CPUS_PER_NODE	CPUs por nodo

Particiones slurm (colas) en atcgrid

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ac*          up       1:00      3   idle atcgrid[1-3]
ac4          up       1:00      1   idle atcgrid4
aapt         up       2:00      3   idle atcgrid[1-3]
acap         up       1:00      3   idle atcgrid[1-3]
```

***** significa que **ac** es la cola utilizada por defecto, es decir, cuando no se usa **-p**

Nota: Utilizar siempre con **sbatch** las opciones **-n1** y **-c**, **—exclusive** y, para usar cores físicos y no lógicos, no olvidar incluir **--hint=nomultithread**. Utilizar siempre con srun, si lo usa fuera de un script, las opciones -n1 y -c y, para usar cores físicos y no lógicos, no olvide incluir --hint=nomultithread. Recordar que **los srun dentro de un *script* heredan las opciones incluidas en el sbatch que se usa para enviar el *script* a la cola slurm**. Se recomienda usar **sbatch en lugar de srun** para **enviar trabajos a ejecutar a través slurm** porque éste último deja bloqueada la ventana hasta que termina la ejecución, mientras que usando **sbatch** la **ejecución** se realiza en **segundo plano**.

EJEMPLOS

Ejemplo hello OpenMP

HelloOMP.c

```c
/* Compilar con:
 gcc -O2 -fopenmp -o HelloOMP HelloOMP.c
*/
#include <stdio.h>
#include <omp.h>

int main(void) {

#pragma omp parallel
  printf("(%d:!!!Hello world!!!)",
          omp_get_thread_num());

return(0);

}
```

Cada thread imprime su identificador. El identificador se obtiene con la función **omp_get_thread_num()**

Script para la ejecución del ejemplo HelloOMP en atcgrid

script_helloomp.sh

```
#!/bin/bash
#Órdenes para el Gestor de carga de trabajo (no intercalar instrucciones del script):
#1. Asigna al trabajo un nombre
#SBATCH --job-name=helloOMP
#2. Asignar el trabajo a una cola (partición)
#SBATCH --partition=ac
#3. Asignar el trabajo a un account
#SBATCH --account=ac
#4. Para que el trabajo no comparta recursos #SBATCH –exclusive
#5. Para que se genere un único proceso del SO que pueda usar un máximo de 12 núcleos
#SBATCH --ntasks 1 --cpus-per-task 12

#Obtener información de las variables del entorno del Gestor de carga de trabajo:
echo "Id. usuario del trabajo: $SLURM_JOB_USER"
echo "Id. del trabajo: $SLURM_JOBID"
echo "Nombre del trabajo especificado por usuario: $SLURM_JOB_NAME"
echo "Directorio de trabajo (en el que se ejecuta el script): $SLURM_SUBMIT_DIR"
echo "Cola: $SLURM_JOB_PARTITION"
echo "Nodo que ejecuta este trabajo:$SLURM_SUBMIT_HOST"
echo "Nº de nodos asignados al trabajo: $SLURM_JOB_NUM_NODES"
echo "Nodos asignados al trabajo: $SLURM_JOB_NODELIST"
echo "Nº CPUs disponibles para el trabajo en el nodo: $SLURM_JOB_CPUS_PER_NODE“

#Instrucciones del script para ejecutar código:
echo -e "\n 1. Ejecución helloOMP una vez sin cambiar nº de threads (valor por defecto):\n"
srun ./HelloOMP
echo -e "\n 2. Ejecución helloOMP varias veces con distinto nº de threads:\n"
for ((P=12;P>0;P=P/2))
do
   #export OMP_NUM_THREADS=$P
   echo -e "\n  - Para $P threads:"
   srun --cpus-per-task=$P --hint nomultithread ./HelloOMP
done
```

Órdenes para Gestor de carga de trabajo

Para imprimir variables del Gestor de carga de trabajo

Instrucciones del script

No olvidar poner **srun** delante del ejecutable

Se puede descomentar `export OMP_NUM_THREADS=$P` y quitar `--cpus-per-task=$P`