El procedimiento de uso de la plataforma es el siguiente:

1. Ingresar al servidor toctoc: nom_user@toctoc.sc3.uis.edu.co
2. Saltar al frontend de guane:

ssh guane

Este servidor es el que te permite enviar trabajos al clúster. Con el comando **sinfo** puede ver la información de las particiones del clúster y los servidores disponibles para lanzar trabajos.

```
[latorresn@guane ~]$ sinfo
PARTITION          AVAIL  TIMELIMIT  NODES  STATE NODELIST
normal*               up   infinite      1    mix guane03
normal*               up   infinite      3  alloc guane[01-02,04]
normal*               up   infinite     10   idle guane[05-06,09-16]
guane_16_cores        up   infinite      1    mix guane03
guane_16_cores        up   infinite      2   idle guane[05-06]
guane_24_cores        up   infinite      3  alloc guane[01-02,04]
guane_24_cores        up   infinite      8   idle guane[09-16]
guane_nvidia_m2050    up   infinite      1    mix guane03
guane_nvidia_m2050    up   infinite      3  alloc guane[01-02,04]
guane_nvidia_m2050    up   infinite      2   idle guane[05-06]
guane_nvidia_m2075    up   infinite      8   idle guane[09-16]
Viz                   up   infinite      1  alloc yaje
[latorresn@guane ~]$
```

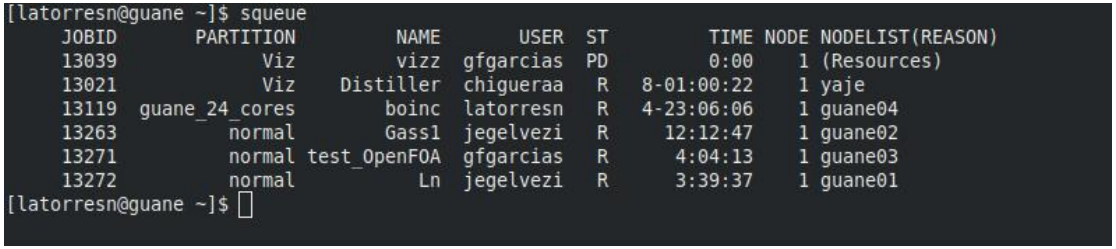
Las particiones son las agrupaciones de los servidores según sus características. La partición **normal** agrupa todos los servidores, **guane_16_cores** a los servidores de 16 cores y **guane_24_cores** los de 24 cores, las otras particiones especifican el tipo de tarjetas GPU que tienen los servidores pero por lo general es para usos muy particulares. El * de la partición **normal** indica que es la partición por defecto, entonces si no especificas en el script a donde quieres lanzar el trabajo, esta es la que se usara, sin discriminar si se usan 16 o 24 cores.

En los estados aparecen varios de ellos: **idle** quiere decir que el servidore(s) están disponibles, **alloc** que están reservados y ejecutando trabajos, **mix** quiere decir que esta realizando algún trabajo pero aún tiene recursos disponibles, esto sucede cuando un usuario reserva por ejemplo 4 cores de los 16 que tiene ese servidor y

quedan 12 disponibles para realizar otras tareas. Pueden aparecer otros estados como **down** que quiere decir que el servidor esta caído por algún motivo y el estado **drain** que por lo general quiere decir que se encuentra en mantenimiento.

Los servidores de guane están numerados desde guane01 a guane16 y en las particiones se muestran repetidos estos nombres debido a que son agrupaciones por características y no por servidores.

El comando **squeue** te muestra el estado de los trabajos en ejecución dentro del clúster:



Por ejemplo, en la cuenta latorresn se esta corriendo el trabajo con **ID 13119**. Este id permite monitorear el trabajo entre otros. Si por algún motivo se quiere cancelar debe ejecutar el comando **scancel** jobid, que para este caso sería **scancel 13119** (Solo los usuarios dueños del trabajo pueden cancelar sus trabajo - un usuario no puede cancelar el trabajo de otro).

Los otros campos son la partición en donde se ejecuta el trabajo, el nombre del trabajo (Este se lo das cuando se crea el script), el usuario que lanzo ese trabajo, el tiempo de ejecución, número de servidores usados y el nombre de los servidores. En **ST (State)** se ve el estado de la tarea, **R** es Running, **PD** es Pending; muchas veces este estado es porque se ha excedido el máximo de recursos permitidos por usuario (4 servidores) o el servidor o partición seleccionada esta en uso por otro usuario. En este ejemplo puedes ver que el trabajo con **ID 13039** esta **PD** por Recursos. Al observar la partición **Viz** (Esta solo tiene un servidor y se usa para entrenamiento de Redes Neuronales) esta en uso, por lo que el usuario **chiguera** con trabajo con **ID 13021** y por tanto el usuario **gfgarcias** no puede usar este servidor hasta que el recurso este disponible.

Para ver el software disponible en el clúster debe teclear el comando: **module av**

Se debe tener en cuenta que dos de estos módulos contienen una versión de Anaconda y por tanto contienen enviroments dentro:

- Analytics/Anaconda/python3
- Bioinformatics/Bioconda/python3

Para cargar un modulo se debe ejecutar el siguiente comando:

module load nom_module

Por ejemplo, para cargar el modulo de GROMACS:

module load Chemistry/gromacs/2018.8_GPU

Si fuera un modulo de Anaconda se deben cargar el enviroment que se requiera, por ejemplo:

module load Analytics/Anaconda/python3
source acitvate tensorflow2_env

Una vez cargado el modulo y el enviroment (cuando se habla de módulos de conda) se puede hacer uso de la aplicación.

3. Crear el script de ejecución de trabajos sobre el clúster. Este script debe ser un archivo creado con algún editor como nano o vi (vim).
   - ➢ nano run.sbatch
   - ➢ Escribir el siguiente contenido (Se debe tener en cuenta que cambia con respecto a la aplicación que se quiera lanzar):

```
#!/bin/bash
#SBATCH --partition=guane_24_cores # Nombre de la partición seleccionada - las que muestra el comando sinfo
#SBATCH --job-name=beast           # Nombre que se le quiere dar al trabajo a lanzar
#SBATCH --time=01:00:00            # Este campo es opcional y establece el tiempo que durara la ejecución.
#SBATCH --nodes=3                  # Número de servidores requeridos. Por lo general es uno solo al menos que el
                                   # programa realice trabajos paralelos en múltiples servidores
#SBATCH --mem=23000                # Cantidad de memoria RAM necesaria en el proceso. Campo opcional y suele
                                   # usarse cuando las aplicaciones tienen en su configuración el establecimiento
                                   # de uso de memoria. Se establece en MB - Cada servidor tiene máximo 100 GB.
#SBATCH --ntasks-per-node=24       # Define el número de tareas a ejecutar por servidor. Se recomienda que para
                                   # un nodo de 24 se seleccione máximo 24 y para el de 16 máximo 16, pueden
                                   # ser menos. Si se sabe que su aplicación uso solo un core, debe escribirse el número 1.
#SBATCH --gres=gpu:4               # Si la aplicación a ejecutar requiere el uso de GPUs debe agregarse este campo
                                   # y especificar el número de tarjetas a usar. Cada servidor contiene 8 tarjetas
```

```
                                        # NVIDIA pero se recomienda usar un máximo de 4 por trabajo.
#SBATCH --output=beast.%j.out          # Nombre del archivo donde se guardara la salida de la ejecución de la aplicación.
                                        # El parámetro %j le agrega al nombre del archivo el JOBID asignado a la tarea.
#SBATCH --error=beast.%j.err           # Nombre del archivo de error. Este contiene la información de los errores
                                        # producidos en la ejecución de la aplicación.
##SBATCH --mail-type=ALL               # Slurm tiene la opción de informar por correo los estados del trabajo. BEGIN, END, FAIL, REQUEUE, ALL
##SBATCH --mail-user=name@andes.edu    # Email a donde se solicita enviar los estados de la tarea.


# NOTA: Para comentar uno de los parámetros de SLURM se debe agregar otro # (En otras palabras, el inicio de linea debe quedar con doble ##)
#=======================================

# load modules and run
module load Chemistry/gromacs/2018.8_GPU

# A partir de aquí se escribe la ejecución normal de al aplicación como se muestra a continuación. En este ejemplo se hace uso de GPUs.
mpirun gmx_mpi mdrun -s gromacs.tpr -ntomp 1 -nb gpu
```

- Para más información de los parámetros de este archivo puedes ingresar a https://slurm.schedmd.com/sbatch.html

4. Lanzar la tarea sobre el cluster. Para esto se ejecuta el comando sbatch:
   sbatch run.sh

   Una vez lanzado el trabajo se mostrara la información del lanzamiento del trabajo. Para ver más detalles de esto se escribe el comando **squeue**

```
[latorresn@guane Example]$ sbatch run.sh
Submitted batch job 13276
[latorresn@guane Example]$ squeue
     JOBID       PARTITION         NAME       USER  ST         TIME NODE NODELIST(REASON)
     13039             Viz         vizz  gfgarcias  PD         0:00    1 (Resources)
     13021             Viz    Distiller  chigueraa   R   8-02:49:45    1 yaje
     13119  guane_24_cores        boinc  latorresn   R   5-00:55:29    1 guane04
     13263          normal        Gass1  jegelvezi   R     14:02:10    1 guane02
     13271          normal test_OpenFOA  gfgarcias   R      5:53:36    1 guane03
     13272          normal           Ln  jegelvezi   R      5:29:00    1 guane01
     13276  guane_24_cores      gromacs  latorresn   R         0:03    3 guane[09-11]
[latorresn@guane Example]$
```

En este ejemplo se muestra que el trabajo fue lanzado con **ID 13276** en los servidores **guane09 a guane11 (3 Servidores)**

Si requieres monitorear el uso de los recursos por parte de la tarea que se lanza, puede ingresar al servidor en el que se esta ejecutando. Para esto se usa el comando **ssh guaneXX** donde **XX** es el número correspondiente al servidor, por ejemplo **ssh guane09**. Una vez en este servidor puede ejecutar el comando **htop** y ver el uso de los recursos consumidos. Solo se puede saltar a servidores en donde se está ejecutando trabajos, si intentaran saltar por ejemplo al servidor **guane04** el sistema informará que no tiene trabajos en ejecución allí y por tanto, no puede ingresar. Para salir de los servidores y desconectarse puede usar el comando **exit** o solo cerrar la terminal. Se recomienda usar el comando para que los comandos que usen queden almacenados en el historial de linux.

Existen otros comando de uso de la plataforma, pero este es el proceso básico para lanzar trabajos sobre el clúster. Es importante esperar que la tarea lleve mas de 10 segundos de ejecución, por defecto hemos configurado SLURM para que los primeros 5 segundos sea tiempo de sincronización y después de este si se ejecute el trabajo.