

Introducción al uso de la infraestructura del NLHPC



Objetivos

- Accediendo al cluster Guacolda-Leftrarú
 - SSH
 - Infraestructura y recursos
- Uso de Slurm
 - Parámetros
 - Uso interactivo y encolado de tareas
 - Uso de comandos informativos
 - Monitoreo de las tareas
- Uso de Módulos
 - Consultando por software y sus versiones
 - Cargando módulos
- Escalamiento
- Ejercicios prácticos

Participación general

Nuestra metodología busca ser dinámica y participativa

- Realizar consultas durante la presentación
- Se realizarán ejercicios en grupo
- En cada ejercicio los usuarios deberán:
 - Participar en la realización de los ejercicios
 - Compartir pantalla de manera grupal
 - Explicar los resultados de los ejercicios
- Se asignarán distintos usuarios para la realización de cada ejercicio

Infraestructura

Nodo Login/debug (gn)

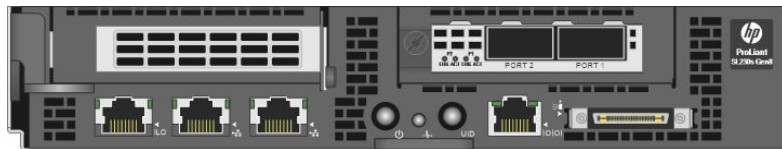


Partición Debug

- 4 Nodos
 - Intel(R) Xeon(R) CPU E5-2660
 - 20 CPUs
 - 59 GB RAM
- Límites de tiempo de ejecución a 30 minutos
- Orientado a acceso y pruebas de compilación
- **Total partición:**
 - 80 CPUs
 - 236 GB RAM

Infraestructura - Leftraru

Nodo Slims (cn)



Partición Slims

- 132 Nodos
 - Intel(R) Xeon E5-2660 v2
 - 20 CPUs
 - 46 GB RAM
- Límites de tiempo de ejecución: 30 días
- Partición **por defecto**
- **Total partición:**
 - 2.640 CPUs
 - 6.072 GB RAM

Infraestructura - Leftrarú

Nodo General (sn)



Partición General

- 48 nodos
 - Intel(R) Xeon Gold 6152
 - 44 cores
 - 187 GB RAM DIMM DDR4
- Límites de tiempo de ejecución: 30 días
- **Total partición:**
 - 2.112 CPUs
 - 8.976 GB RAM

Infraestructura - Leftrarú

Nodo Largemem (fn)



Partición Largemem

- 9 nodos
 - Intel(R) Xeon Gold 6152
 - 44 cores
 - 765 GB RAM
- Destinado para tareas de 192G+ RAM
- Límites de tiempo de ejecución: 30 días
- **Total partición:**
 - 396 CPUs
 - 6.885 GB RAM

Infraestructura - Guacolda

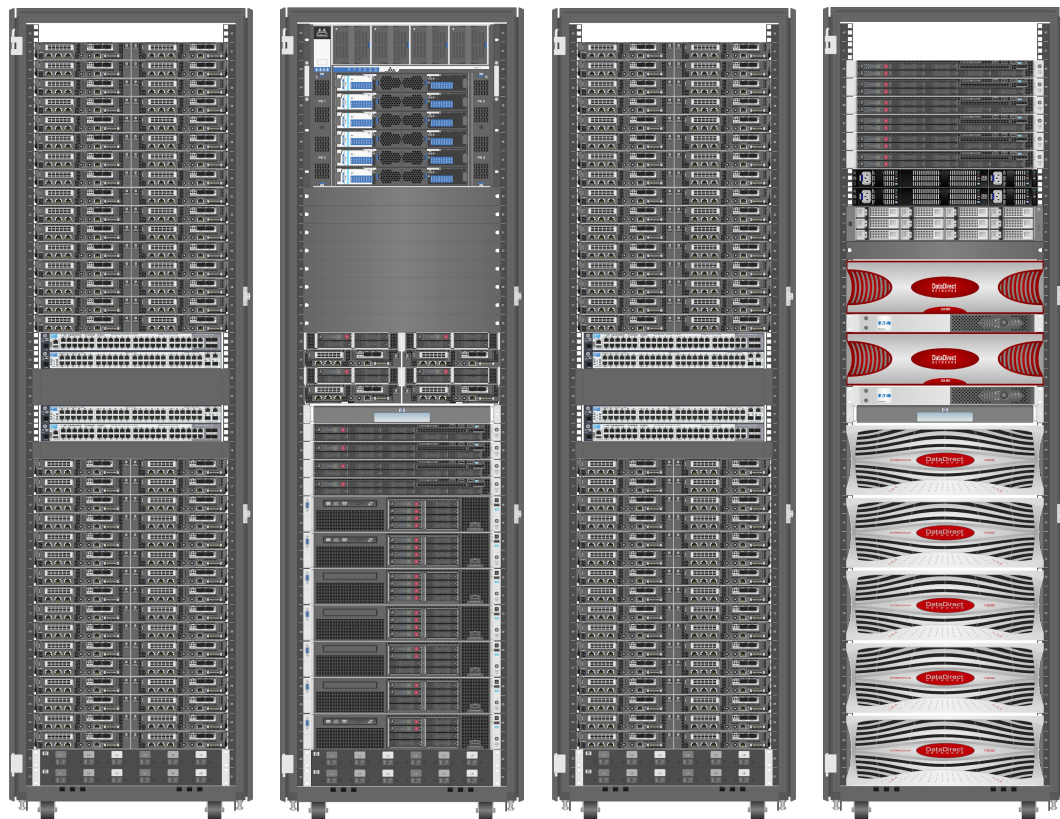
Nodo GPU (gn)



Partición GPU

- 2 Nodos
 - Intel(R) Xeon Gold 6152
 - 44 cores
 - 187 GB RAM
 - 2 NVIDIA Volta V100 cada nodo
 - 16GB
 - 5120 CUDA cores cada una
- Destinado a tareas que requieran uso de GPUs
- Límites de tiempo de ejecución: 30 días
- **Total partición:**
 - 88 CPUs
 - 374 GB RAM
 - 20.480 CUDA cores

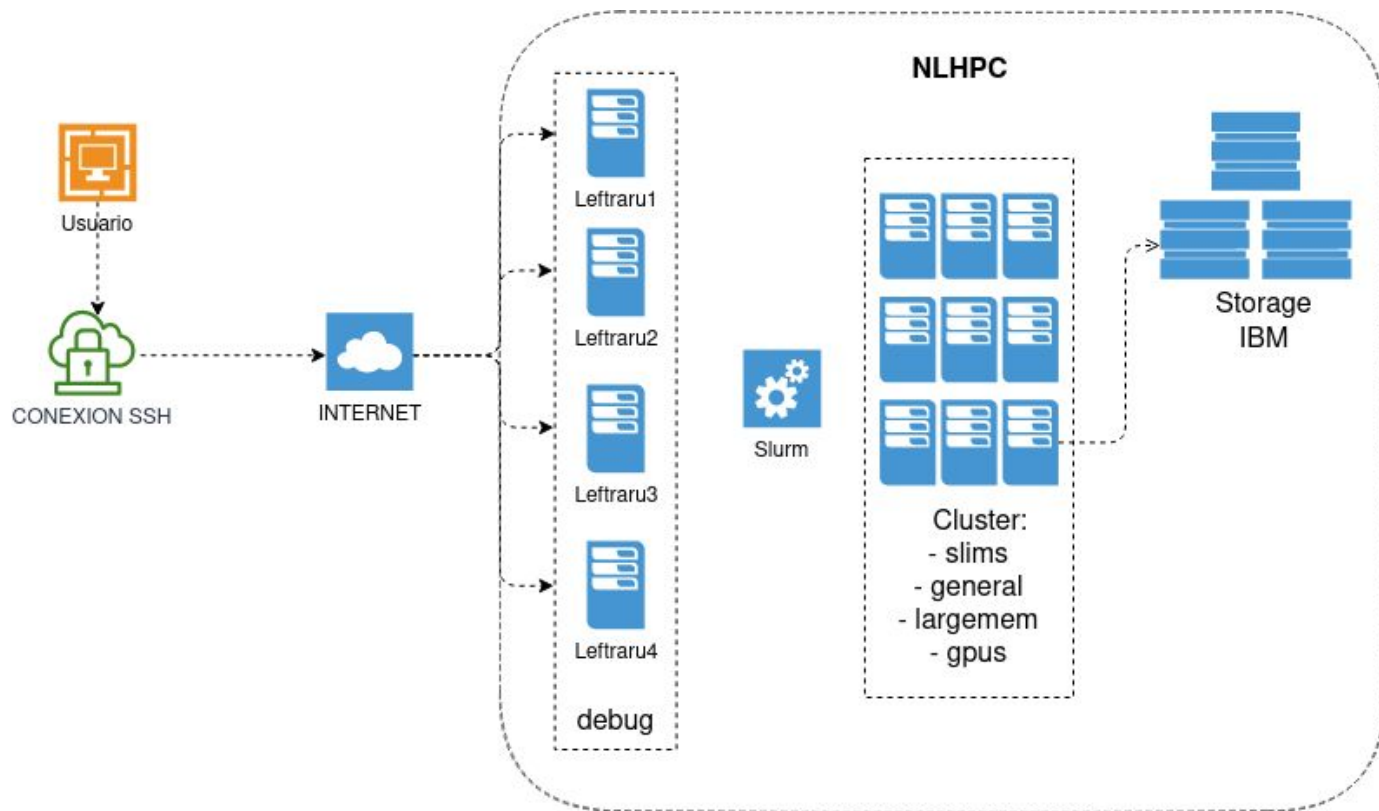
Infraestructura NLHPC



- 266 TFlops
- 5236 cores
- 191 nodos
- 4 PB almacenamiento IBM Spectrum Scale
- Red LAN Infiniband FDR 56Gbps

https://wiki.nlhpc.cl/Hardware_Disponible

Accediendo al Cluster



Llaves SSH

Generar llave SSH

```
[dbowman@HAL ~] ssh-keygen -t ed25519
```

```
Enter file in which to save the key (/home/dbowman/.ssh/id_ed25519)
```

```
Enter passphrase (empty for no passphrase)
```

Copiar llave SSH

```
[dbowman@HAL ~] ssh-copy-id dbowman@leftrararu.nlhpc.cl
```

Acceder al cluster

```
[dbowman@HAL ~] ssh dbowman@leftrararu.nlhpc.cl
```

El uso de llaves SSH ofrece un método de autenticación más seguro.

¿Qué es SLURM



- Gestor de recursos
- Administra los recursos de las particiones de Leftraru y Guacolda.
- Gestiona las tareas en ejecución y en espera en el cluster.
- Reserva recursos compartidos.
- Permite la ejecución de tareas hasta por 30 días

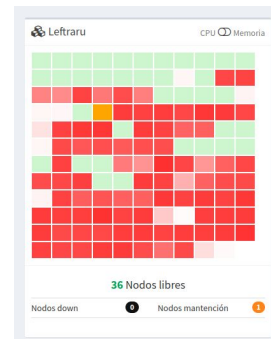


Obteniendo información de las particiones

- ***sinfo***: ver estado de las particiones

```
[root@master2 ~]# sinfo
```

PARTITION	AVAIL	TIMELIMIT	NODES	STATE	NODELIST
slims*	up	infinite	1	drain	cn037
slims*	up	infinite	12	mix	cn[023-024,026,045,072-073,079,087,096,107,129,131]
slims*	up	infinite	63	alloc	cn[019-020,038-044,...108-128,130]
slims*	up	infinite	55	idle	cn[001-018,021-022,...082-083,086,088-090,132]
general	up	infinite	9	mix	sn[002,006,014-016,021,028,030-031]
general	up	infinite	39	alloc	sn[001,003-005,007-013,...032-048]
largemem	up	infinite	2	mix	fn[001,007]
largemem	up	infinite	2	alloc	fn[002,004]
largemem	up	infinite	5	idle	fn[003,005-006,008-009]
gpus	up	infinite	2	mix	gn[001-002]
debug	up	infinite	4	idle	leftraru[1-4]



squeue: listado de tareas en ejecución y pendientes

Monitorear desde la consola:

```
[usuario@leftraru1 ~]$ squeue
```



JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
4400799	slims	example	usuario	R	0:00	1	cn042

sacct: estados de tareas ejecutadas



```
[usuario@leftraru1 ~]$ sacct -X
```

JobID	JobName	Partition	Account	AllocCPUS	State	ExitCode
24118136	14131-DIA+	slims	users	2	RUNNING	0:0
24118147	14132-DIA+	slims	users	2	RUNNING	0:0
24118148	14133-DIA+	slims	users	2	COMPLETED	0:0
24118154	14137-DIA+	slims	users	2	COMPLETED	0:0



Información detallada de job

```
[root@leftraru1 ~]# scontrol -dd show job 9160565
```

```
JobId=9160565 JobName=w.fepec-f-cnt-oo.m2
  UserId=workxwz(11942) GroupId=fisica_cmm(11222) MCS_label=N/A
  Priority=109951 Nice=0 Account=unab QOS=120-30-std
  JobState=RUNNING Reason=None Dependency=(null)
  Requeue=0 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
  RunTime=04:28:03 TimeLimit=3-00:00:00 TimeMin=N/A
  SubmitTime=2017-10-09T11:25:36 EligibleTime=2017-10-09T11:25:36
  StartTime=2017-10-09T15:04:40 EndTime=2017-10-12T15:04:40 Deadline=N/A
  PreemptTime=None SuspendTime=None SecsPreSuspend=0
  Partition=slims AllocNode:Sid=leftraru4:52235
  ReqNodeList=(null) ExcNodeList=(null)
```

Enviar trabajos en SLURM



```
[usuario@leftraru1 ~]$ srun hostname
```

Comando

```
cn028
```

Salida comando

```
[usuario@leftraru1 ~]$ sbatch <job_script>
```

```
Submitted batch job 9142401
```

```
[usuario@leftraru1 ~]$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
9142401	slims	pi_levqu	root	R	0:08	1	cn105

```
[usuario@leftraru1 ~]$ scancel 9142401
```

Cancelar trabajo



Parámetros en SLURM

Parámetro	Uso	Acción
-J	-J mi-tarea	Asigna nombre a la tarea
-p	-p slims	Indica partición a utilizar
-n	-n 1	Nº de procesos
-c	-c 20	CPUs por proceso
--ntasks-per-node	--ntasks-per-node=20	Procesos por nodo
--mem-per-cpu	--mem-per-cpu=2300	Memoria por CPUs
-o	-o salida_%j.out	Log de salida
-e	-e errores_%j.err	Log de salida de errores
-mail-user	-mail-user=user@abc.xyz	Donde se envia info del JOBS
-mail-type	-mail-user=ALL	Tipo de información a enviar



Ejercicio 1

- Ejecute el comando **hostname** en la partición *Slims* con **srun**:
 - Con un único proceso.
 - Con dos procesos iguales.
 - Con dos procesos en distintos nodos.
 - Asignando dos CPU por proceso.
- ¿Qué resultados se han obtenido?
- En la partición *slims*
 - ¿Cuántas CPU puedo reservar por proceso? ¿Por qué ese número?
 - ¿Qué diferencia hay en la partición *general*?
 - ¿Qué ocurre si reservo más CPU de los disponibles?
- ¿Qué ocurre si no especifico la partición en la que quiero ejecutar mi comando?



Ejemplo de script básico SBATCH

Utilizar su editor por consola preferido: nvim, vim, vi, nano

```
#!/bin/bash
#SBATCH -J ejemplo
#SBATCH -p slims
#SBATCH -n 1
#SBATCH -c 1
#SBATCH -o archivo_%j.out
#SBATCH -e archivo_%j.err
#SBATCH --mail-user=foo@example.org
#SBATCH --mail-type=ALL

sleep 10
```

Y ejecutar el script:

```
sbatch test.sh
```

Ejercicio 2

- Crea un *script* de ejecución para lanzarlo con **sbatch**, con las siguientes consideraciones:
 - Utilizar la partición *slims*.
 - Reserva una única CPU.
 - Ejecuta el comando **stress -c 1**
- 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:
 - ¿Cuánto tiempo estará la tarea en ejecución?
 - ¿Qué comando puede utilizar para obtener información acerca de la tarea en ejecución?
 - ¿Cómo puedo cancelar mi tarea?
- Para poder lanzar tareas, debo conocer el uso del Cluster ¿Qué comando me permite conocer el estado de las particiones?



Monitorear Job - *htop*

Puede ingresar a través de ssh a un nodo en donde tenga una tarea en ejecución y ejecutar **htop**

```
1 [|||||100.0%] 6 [ 0.0%] 11 [ 0.0%] 16 [ 0.0%]
2 [ 0.0%] 7 [ 0.0%] 12 [ 0.0%] 17 [ 0.0%]
3 [|||||100.0%] 8 [ 0.0%] 13 [ 0.0%] 18 [ 0.0%]
4 [ 0.0%] 9 [ 0.7%] 14 [ 0.0%] 19 [ 0.0%]
5 [ 0.0%] 10 [ 0.0%] 15 [ 0.0%] 20 [ 0.0%]

Mem[||||| 1.56G/62.7G] Tasks: 44, 39 thr: 3 running
Swp[ 0K/62.5G] Load average: 2.00 2.01 2.05
Uptime: 4 days, 00:36:11

  PID USER      PRI  NI  VIRT   RES   SHR  S  CPU% MEM%   TIME+  Command
18305 nperinet    20   0 19020  3768   892  R 100.0  0.0 2h52:34 ./LL_RK4.x
18335 nperinet    20   0 17104  1584   892  R 100.0  0.0 2h47:30 ./LL_RK4.x
18605 root        20   0 121M   2432  1476  R  0.0  0.0 0:00.05 htop
   1 root        20   0 185M   5068  2384  S  0.0  0.0 0:11.06 /usr/lib/systemd/systemd --switched-root --system --de
  669 root        20   0 112M   2000  1564  S  0.0  0.0 0:00.00 /bin/bash
  671 root        20   0 36816  7236  6908  S  0.0  0.0 0:00.78 /usr/lib/systemd/systemd-journald
  726 root        20   0 43808  2428  1268  S  0.0  0.0 0:00.83 /usr/lib/systemd/systemd-udev
1012 root        16  -4 51188  1620  1236  S  0.0  0.0 0:00.05 /sbin/auditd -n
1002 root        16  -4 51188  1620  1236  S  0.0  0.0 0:00.25 /sbin/auditd -n
1206 root        20   0 448M  10956  6968  S  0.0  0.0 0:00.00 /usr/sbin/NetworkManager --no-daemon
1209 root        20   0 448M  10956  6968  S  0.0  0.0 0:00.09 /usr/sbin/NetworkManager --no-daemon
1139 root        20   0 448M  10956  6968  S  0.0  0.0 0:02.34 /usr/sbin/NetworkManager --no-daemon
1144 avahi        20   0 30220  1564  1300  S  0.0  0.0 0:00.44 avahi-daemon: running [cnf004.local]
1148 dbus        20   0 28824  1772  1352  S  0.0  0.0 0:00.44 /bin/dbus-daemon --system --address=systemd: --nofork
1166 root        20   0 198M  1236   776  S  0.0  0.0 0:00.00 /usr/sbin/gssproxy -D
1167 root        20   0 198M  1236   776  S  0.0  0.0 0:00.00 /usr/sbin/gssproxy -D
1168 root        20   0 198M  1236   776  S  0.0  0.0 0:00.00 /usr/sbin/gssproxy -D
1169 root        20   0 198M  1236   776  S  0.0  0.0 0:00.00 /usr/sbin/gssproxy -D
1170 root        20   0 198M  1236   776  S  0.0  0.0 0:00.00 /usr/sbin/gssproxy -D
1164 root        20   0 198M  1236   776  S  0.0  0.0 0:00.30 /usr/sbin/gssproxy -D

F1Help F2Setup F3Search F4Filter F5Free F6SortBy F7Nice F8Nice F9Kill F10Quit
```

Generador de scripts SBATCH

Seleccione una partición

☒ general ☐ slims ☐ largemem

☐ gpus ☐ debug

Programación paralela

☐ Secuencial / No sé

☒ OpenMP ☐ MPI

☐ MPI-OpenMP

Email *

Nombre de la tarea *

Asignar recursos

N° de procesos

CPUs por proceso

Memoria por CPU (MB)

Job array

☐

Tiempo de ejecución (DD-HH:MM:SS)

- : :

SLURM Script

```
#!/bin/bash
#-----Script SBATCH - NLHPC -----
#SBATCH -J ejercicio
#SBATCH -p slims
#SBATCH -n 1
#SBATCH -c 1
#SBATCH --mem-per-cpu=2300
#SBATCH --mail-user=usuarios@mail.com
#SBATCH --mail-type=ALL
#SBATCH -o ejercicio_%j.out
#SBATCH -e ejercicio_%j.err

#-----Toolchain-----
#-----Modulos-----
#-----Comando-----
```

Copiar script

En el siguiente link podrá crear de forma simple sus scripts SBATCH para ser ejecutados en el cluster.

https://wiki.nlhpc.cl/Generador_Scripts

Ejercicio 3

- Crea un *script* para lanzarlo con **sbatch**, con las siguientes consideraciones:
 - La partición a lanzar es *slims*.
 - Reserva un nodo completo.
 - Ejecuta el comando **stress -c 40 -t 10m**.
- Verifica en qué nodo se está ejecutando tu tarea, accede mediante ssh al nodo y ejecuta htop.
- ¿Cuántos procesos se están ejecutando?
- ¿Cuál es el porcentaje de uso de cada proceso?
- ¿Cómo sería la manera correcta de lanzar la tarea con el fin de que cada proceso se ejecute al 100%?
- Compara y explica el uso de CPU entre el proceso inicial y el último.



Reserva Memoria RAM

Límite en la memoria RAM:

- Por defecto se reserva 1GB de RAM por core reservado
- Si se excede se obtendrá el error: “Exceeded job memory limit”
- Reservar RAM por core usado: **#SBATCH --mem-per-cpu=2300**

systemd-cgtop -m | grep job_id

Path	Tasks	%CPU	Memory	Input/s	Output/s
/	485	4356.4	23.4G	-	-
/slurm	-	-	14.3G	-	-
/slurm/uid_2398	-	-	14.3G	-	-
/slurm/u...8/job_24117526	-	-	14.3G	-	-
/system.slice	-	4354.2	8.4G	-	-

Ejercicio 4

- Crea un *script* para lanzarlo con **sbatch**, con las siguientes consideraciones:
 - La partición a lanzar es *slims*.
 - No asignar memoria RAM
 - Reserva una CPU por proceso.
 - Debe enviar un correo electrónico cuando la tarea cambie de estado.
 - Ejecutar **stress -m 1 --vm-bytes 2048M -t 15m**
- ¿Qué ocurre con la ejecución? ¿Cuál es la razón?
- Modifica el script para ejecutar la tarea.
- ¿Cuántos recursos de RAM está utilizando la tarea?



Sistema de Módulos LMOD

- Permite tener diferentes aplicaciones y versiones de estos en un mismo sistema operativo.
- En el NLHPC usamos **Lmod** (<https://github.com/TACC/Lmod>).
- El actual sistema de módulos está disponible para las distintas arquitecturas de procesador (AVX512, AVX, SSE4.2)

Lmod: Buscar módulo

```
eguerre@leftrar1:/home/eguerre$ ml spider Python
```

Python:

Description:

Python is a programming language that lets you work more quickly and integrate your systems more effectively.

Versions:

Python/2.7.15

Python/3.7.2

Python/3.7.3

Other possible modules matches:

Biopython IPython protobuf-python

To find other possible module matches execute:

```
$ module -r spider '.*Python.*'
```

For detailed information about a specific "Python" module (including how to load the modules) use the module's full name.

For example:

```
$ module spider Python/3.7.3
```

Lmod: Cargar distintas versiones

```
eguerra@leftraru1:/home/eguerra$ m1 Python/3.7.3
eguerra@leftraru1:/home/eguerra$ m1
```

Currently Loaded Modules:

1) GCCcore/8.2.0	4) impi/2019.2.187	7) intel/2019b	10) libreadline/8.0	13) SQLite/3.27.1	16) libffi/3.2.1
2) icc/2019.2.187-GCC-8.2.0-2.31.1	5) imkl/2019.2.187	8) bzip2/1.0.6	11) ncurses/6.1	14) XZ/5.2.4	17) Python/3.7.3
3) ifort/2019.2.187-GCC-8.2.0-2.31.1	6) binutils/2.32	9) zlib/1.2.11	12) Tcl/8.6.9	15) GMP/6.1.2	

```
eguerra@leftraru1:/home/eguerra$ python -V
Python 3.7.3
```

```
eguerra@leftraru1:/home/eguerra$ m1 Python/2.7.15
```

The following have been reloaded with a version change:

1) Python/3.7.3 => Python/2.7.15

```
eguerra@leftraru1:/home/eguerra$ m1
```

Currently Loaded Modules:

1) GCCcore/8.2.0	4) impi/2019.2.187	7) intel/2019b	10) libreadline/8.0	13) SQLite/3.27.1	16) libffi/3.2.1
2) icc/2019.2.187-GCC-8.2.0-2.31.1	5) imkl/2019.2.187	8) bzip2/1.0.6	11) ncurses/6.1	14) XZ/5.2.4	17) Python/2.7.15
3) ifort/2019.2.187-GCC-8.2.0-2.31.1	6) binutils/2.32	9) zlib/1.2.11	12) Tcl/8.6.9	15) GMP/6.1.2	

```
eguerra@leftraru1:/home/eguerra$ python -V
Python 2.7.15
```

Ejercicio 5

- Descarga el siguiente código Python con wget: [n-queens-problem-3.py](#) en tu directorio de trabajo.
- Crea un *script* de para lanzarlo con **sbatch**, con las siguientes consideraciones:
 - Utilizar la partición *slims*.
 - Cada proceso reserva una CPU.
 - Supondremos que cada trabajo reserva 2300 Mb de RAM.
 - Ejecuta el código con la versión de Python/3.9.5

Ver cuota de disco

```
[$USER@leftraru1 ~]# usoDisco
```

Uso de disco del usuario: \$USER

Cuota = 200G

Utilizado = 148.95G

% de utilización = 74.5%

Eficiencia Computacional

- Comportamiento de un programa al ejecutarlo de manera paralela (más de una CPU)
- Un programa puede escalar en un rango de procesadores $[1..n]$
- Se logra la eficiencia cuando la medición se mantiene constante sobre un factor de 0,5
- Esto es importante, ya que un programa no se ejecutará en la mitad de tiempo si se ejecuta en un doble de procesadores.
 - Buscamos hacer un uso **óptimo** y **eficiente** de los recursos

Eficiencia Computacional - Speedup y Eficiencia

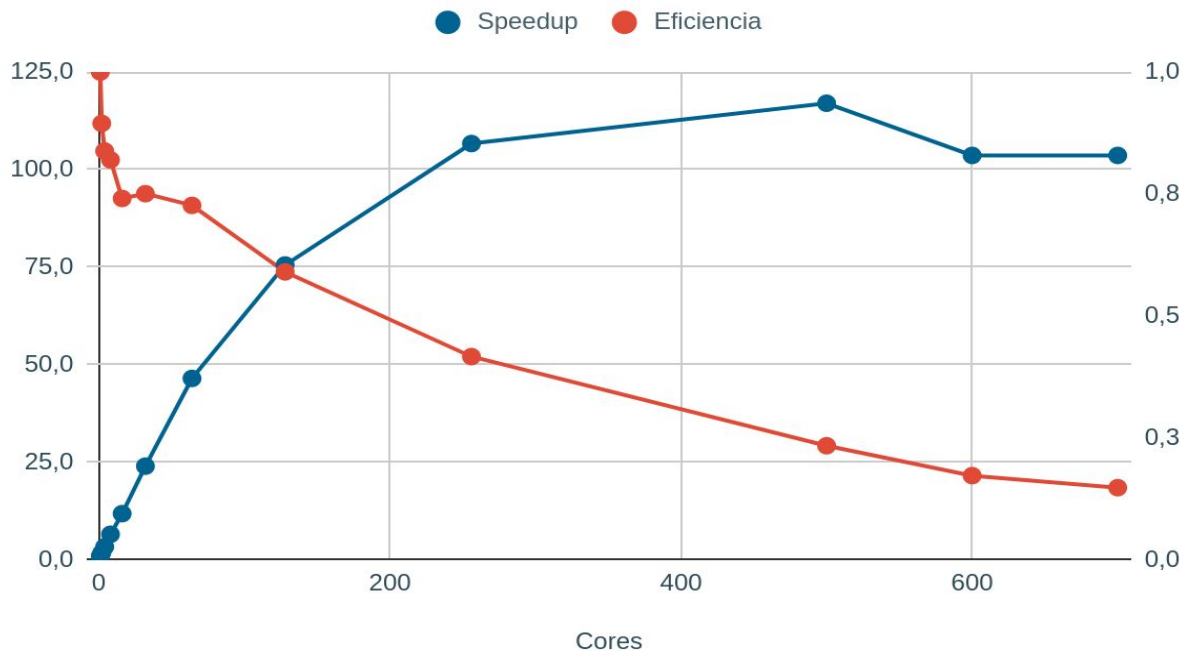
- **SpeedUp** es la métrica que nos indica la ganancia mediante la paralelización:
 - $\text{SpeedUp} = \text{Tiempo Original} / \text{Tiempo Mejora}$
- **Eficiencia** es la métrica del uso de los recursos computacionales
 - $\text{Eficiencia} = \text{SpeedUp} / \text{Número de CPU}$

Procesadores	Tiempo de Ejecución	Speedup	Eficiencia
1	1:00:27	1,0	1,0
2	0:33:47	1,8	0,9
4	0:18:02	3,4	0,8
8	0:09:13	6,6	0,8
16	0:05:06	11,9	0,7
32	0:02:31	24,0	0,8
64	0:01:18	46,5	0,7
128	0:00:48	75,6	0,6
256	0:00:34	106,7	0,4
500	0:00:31	117,0	0,2
600	0:00:35	103,6	0,2
700	0:00:35	103,6	0,1

<https://wiki.nlhpc.cl/Escalamiento>

Eficiencia Computacional - Speedup y Eficiencia - Gráfico

Speedup y Eficiencia



Uso de partición GPUS

- ¿Cómo ejecutar un programa mediante Slurm que necesite utilizar la partición gpus?
 - ¿Qué opciones existen actualmente en el Cluster?
 - ¿Qué módulos se deben cargar para utilizar la partición gpus?

```
#!/bin/bash
#SBATCH -J gpu-example
#SBATCH -p gpus

#SBATCH -n 1
#SBATCH -c 1
#SBATCH --gres=gpu:1
#SBATCH --mem-per-cpu=4250

ml purge
ml fosscuda/2019b
ml NAMD/3.0alpha9
...
```

Enlaces de interés

Los invitamos a visitar nuestra página web:

www.nlhpc.cl

También tenemos una wiki pública con información útil:

https://wiki.nlhpc.cl/Bienvenida_NLHPC

Solicitud de cuentas de usuarios

<https://solicitudes.nlhpc.cl/>

Dashboard

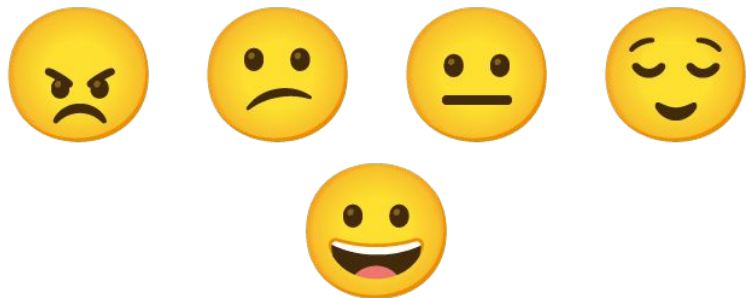
<https://dashboard.nlhpc.cl/>

En caso de dudas, pueden escribirnos a soporte@nlhpc.cl

Queremos saber tu opinión

- Encuesta de satisfacción
- Correo de agradecimiento

¿Cómo evalúa el curso recibido?



¡Gracias por participar!
:)

www.nlhpc.cl
2023

