LABORATORIO NACIONAL DE SUPERCÓMPUTO DEL SURESTE DE MÉXICO

Guía básica de utilización de la Supercomputadora *Cuetlaxcoapan*

Aprobado por: Dr. Manuel Martín Ortiz

Fecha de Publicación: 13/02/2019

5 Edición

Control de cambios

No. de Edición	Fecha	Motivo de la edición
1ra. Edición	12/02/2016	Primer Ejemplar
2da. Edición	15/08/2016	Segundo Ejemplar
3era. Edición	10/02/2017	Tercer Ejemplar
4ta Edición	16/02/2018	Cuarto Ejemplar
5ta Edición	10/01/2019	Revisión anual: Se agregan los scripts

Contenido

Control de cambios 2
1. Introducción 4
2. Distribución General de los Recursos de Cómputo del LNS 4
3. Guía básica de usuario 6
3.1 Estructura básica de un comando del shell 7
3.2 Comandos de gestión de ficheros y directorios 8
3.3 Método de acceso a su cuenta en la supercomputadora 9
3.3.1 Conexión SSH desde Linux o Mac OS X 11
3.3.2 Conexión SSH desde Windows 11
3.4 Procedimiento posterior a la primera conexión con su cuenta en el LNS 17
3.5 Transferencias de archivos 18
3.5.1 Procedimiento para transferencia desde su computadora local hacia la supercomputadora *Cuetlaxcoapan* 18
3.5.2 Procedimiento para transferencia desde la supercomputadora hacia su computadora local 18
3.5.3 Comandos para transferencia de archivos 18
3.6 Edición de archivos en la consola de comandos 24
3.6.1 Programas de edición de texto en terminal 24
4. Chequeo y carga de software disponible en el LNS 26
5. SLURM 28
5.1 Comandos de SLURM 29
5.1.1 Comandos de SLURM esenciales para la ejecución de Jobs en la supercomputadora 30
5.2 Scripts para ejecución de programas en SLURM 31
5.2.1 Script genérico 32
5.2.2 Uso de LAMMPS 32
5.2.3 Uso de Quantum Espresso 33
5.2.4 Uso de SIESTA 33
5.2.5 Uso de GROMACS 34
5.2.6 Uso de Gaussian 34
6. Información adicional 36

1. Introducción

Este documento presenta información básica para establecer una conexión a su cuenta de usuario y utilizar el intérprete de comandos, así como el conjunto de instrucciones y procedimientos para ejecutar de manera apropiada sus tareas de cálculo en la Supercomputadora *Cuetlaxcoapan* del Laboratorio Nacional de Supercómputo (LNS) del Sureste de México.

2. Distribución General de los Recursos de Cómputo del LNS

Es importante que el usuario del LNS conozca la forma en que trabajará en la supercomputadora *Cuetlaxcoapan*, por lo que le sugerimos poner especial atención a la siguiente información:

- Deberá seguir los procedimientos y lineamientos descritos en la siguiente guía para realizar las tareas habituales de uso de la supercomputadora, desde establecer una conexión, enviar tareas para su procesamiento, o realizar transferencias de archivos entre su computadora local y la supercomputadora.
- Por motivos de eficiencia y seguridad, existe una estructura establecida en la supercomputadora (véase la Figura 1) en la que se delimita claramente el espacio en el que trabajará el usuario y los recursos administrados por SLURM (*Simple Linux Utility for Resource Management*), el cual es el sistema encargado de gestionar los procesos y recursos de *cluster* de cómputo del LNS.

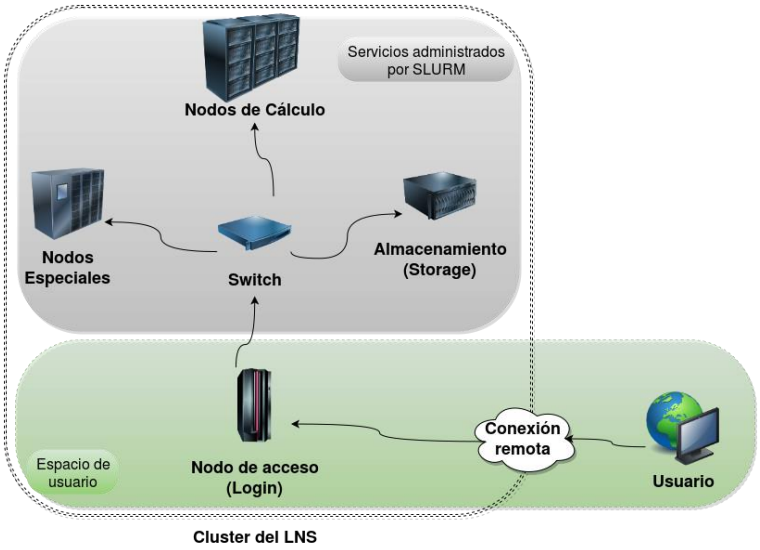

Figura 1: Diagrama de acceso al *cluster* de supercómputo del LNS.

- Para que los recursos sean utilizados de la manera más eficiente, es de gran importancia que respete el espacio de trabajo designado al usuario y que haga uso de la supercomputadora a través del sistema SLURM, ya que esto permitirá balancear adecuadamente los recursos compartidos entre sus tareas y las de otros usuarios.
- La supercomputadora C*uetlaxcoapan* está compuesta por 136 nodos estándares de cálculo, 6 nodos especiales y un sistema de almacenamiento de 520 TB, interconectados en una red Infiniband con topología de estrella mediante un switch central (véase la Fig. 1). Estos recursos son administrados por el sistema SLURM. El acceso a los recursos se lleva a cabo mediante un nodo de gestión (*login*) que constituye el entorno donde los usuarios deberán realizar sus labores habituales de preparación, envío y monitoreo de tareas de cálculo, así como de recuperación de los datos generados.
- Es importante aclarar que no se permite realizar labores de post-procesamiento de datos en el nodo de *login*. El usuario deberá transferir los datos generados a su computadora local para post-procesamiento (visualización, etc.)
- Todos los usuarios tienen la misma prioridad de acceso a los recursos (nodos de cálculo y almacenamiento).
- Cada usuario tiene asignado un espacio de almacenamiento personal (directorio `HOME=/home/cuenta`) donde puede almacenar hasta 50 GB de información. Este espacio de almacenamiento es global (visible en todos los nodos) y está destinado únicamente para scripts de cálculo, aplicaciones desarrolladas por el usuario y aplicaciones comerciales de uso personal. No se permite guardar resultados de cálculos u otro tipo de datos del usuario en el directorio `HOME` (para mayor información al respecto, consultar las políticas de uso del LNS en www.lns.buap.mx).
- Los resultados generados por el usuario deberán almacenarse en el directorio `SCRATCH=/scratch/cuenta`). Este directorio también es global, es decir, visible por todos los nodos de cálculo. Es recomendable respaldar periódicamente su información almacenada en `SCRATCH` ya que el LNS no se hace responsable de la integridad de la misma ante posibles fallas de hardware y software (para mayor información, consultar las políticas de uso del LNS en www.lns.buap.mx).
- La unidad básica de ejecución es un core de procesamiento. Cada nodo estándar de cálculo posee 24 cores (2 sockets Intel Xeon E5-2680 v3 a 2.5 Ghz con 12 cores por

socket) y 128 GB de memoria RAM compartida. Por lo tanto, el tiempo de cálculo se mide en *horas-core*.

- Cada usuario puede utilizar por defecto hasta 240 cores de cálculo de manera simultánea. Estos cores no están ligados a algún nodo específico, es decir, se pueden utilizar de manera indistinta en cualquier nodo de cálculo, ya sea individualmente o en conjunto. Sin embargo, para mayor eficiencia, es recomendable utilizar afinidad entre cores cuando se envían tareas paralelas (véase la sección de SLURM).
- El usuario es responsable de gestionar su tiempo de cálculo. Es decir, debe decidir adecuadamente, con base en criterios de eficiencia, cuántos cores deberá utilizar en cada tarea y por cuánto tiempo deberá correr la tarea (véase la sección de SLURM). Hay que tener en cuenta, por ejemplo, que usando los 240 cores simultáneamente implica gastar 240 horas-core por cada hora de ejecución transcurrida. De esta manera, si su tarea corre durante un día completo (24 horas) gastará 5760 horas-core por día. Es decir, si su proyecto sólo tiene asignadas 50,000 horas-core de cálculo, estas se gastarían completamente en menos de 10 días.

3. Guía básica de usuario

El conocimiento de la línea de comandos de Linux es fundamental para la utilización de la supercomputadora *Cuetlaxcoapan*, tanto para establecer una conexión y acceder a su cuenta, como para enviar tareas de cálculo y manipular de forma remota los datos generados.

Los sistemas operativos Linux y Mac OS X disponen por defecto de una aplicación conocida como *terminal* (véase la Fig. *2*) que abre una sesión en el sistema usando un intérprete de comandos en modo de texto (también conocido como *shell*). La terminal muestra una ventana con el símbolo $ o > seguido de un *prompt* como _ o ▉ que generalmente parpadea para indicarnos que el intérprete está listo para recibir órdenes. El intérprete de comandos constituye entonces la manera más sencilla de interaccionar con la computadora, sobre todo porque esta se utiliza remotamente. El nodo de *login* de la supercomputadora *Cuetlaxcoapan* utiliza RedHat Enterprise Linux versión 6.6 y el intérprete de comandos por defecto es *bash* (www.gnu.org/software/bash). Por lo tanto, todos los comandos que describiremos de aquí en adelante serán ejecutados en *bash*.

Figura 2: La terminal de comandos de Linux o Mac OS X tiene un ícono similar a éste.

Aún cuando su computadora de escritorio utilice un sistema operativo Windows (véase *Conexión SSH desde Windows*) también requiere familiarizarse con el intérprete de comandos de Linux, ya que al conectarse a la supercomputadora *Cuetlaxcoapan* estará utilizando el sistema operativo Linux.

Bash es un intérprete que dispone de la característica de *autocompletado* de comandos, lo cual le resultará extremadamente útil. Utilizando esta capacidad, puede reconocer comandos escribiendo solo las primeras letras y solicitando al intérprete completar el comando presionando la tecla `Tab`. Si hay más de una coincidencia de autocompletado, y esta tecla se presiona más de una vez, *bash* muestra las distintas opciones de completar el comando, o rutas de archivos y directorios y palabras reservadas por el sistema que cumplen con el indicio que escribió, e incluso autocompleta la línea entera si solo existe una opción disponible. Sugerimos que pruebe esta opción y la utilice con regularidad.

3.1 Estructura básica de un comando del shell

Tradicionalmente, la estructura de un comando de *bash* o de algún otro *shell* de Linux como *ksh*, *tcsh*, etc., se asemeja a la siguiente:

```
comando <opciones> <parámetro1> <parámetro2> ...
```

donde las opciones suelen colocarse mediante letras precedidas por un signo de guion corto (-). Los parámetros son argumentos que el comando va a utilizar para desarrollar su función. Por ejemplo, el comando `ls` sin opciones ni argumentos presenta una lista corta de los archivos de un directorio incluyendo subdirectorios, mientras que `ls -l` presenta la misma información, pero ordenada en forma de lista incluyendo los atributos (tamaño,

permisos, etc.) de los archivos y subdirectorios. Si desea información más detallada de un comando específico siempre podrá obtenerla directamente desde la terminal, ejecutando la instrucción `man comando` o `comando --help`.

A continuación, se presentan algunos comandos útiles de Linux junto con una descripción simple de su función y un ejemplo de uso.

3.2 Comandos de gestión de ficheros y directorios

Al establecer una sesión en Linux mediante la línea de comandos, el usuario será posicionado por defecto en el directorio `/home/cuenta`. El usuario posee privilegios para manipular la información en este directorio, es decir, podrá crear, modificar y borrar archivos y subdirectorios. La tabla 1 muestra algunos comandos que son básicos para navegar por el sistema de archivos y directorios, y con ello operar eficientemente en Linux. La figura 3 muestra también ejemplos de ejecución de éstos.

Tabla 1: Comandos básicos de gestión de ficheros y directorios

Comando	Descripción	Ejemplo
`cat file`	Muestra el contenido del archivo `file`	`cat programa.f`
`cd dir`	Cambiar del directorio actual a `dir`	`cd programas`
`ls dir`	Lista el contenido del directorio `dir`	`ls programas`
`rm file`	Elimina el archivo `file`	`rm programa.c`
`cp file1 file2`	Copia el archivo `file1` a `file2`	`cp programa.c programa1.c`
`pwd`	Muestra la ruta del directorio actual	`pwd`
`mkdir dir`	Crea un nuevo directorio	`mkdir resultados`
`rmdir dir`	Elimina directorios	`rmdir pruebas`
`man command`	Despliega el manual de uso de un comando	`man ls`
`touch file`	Crea un archivo vacío	`touch programas/hola.c`
`mv arch dir`	Mueve el archivo `arch` al directorio `dir`	`mv hola.c programas`

`clear`	Limpia la terminal	`Clear`
`du -h dir`	Reporta el tamaño del directorio `dir`	`du -h`
`grep patron arch`	Busca un `patrón` o expresión en un archivo `arch`	`grep OPEN programa.f`

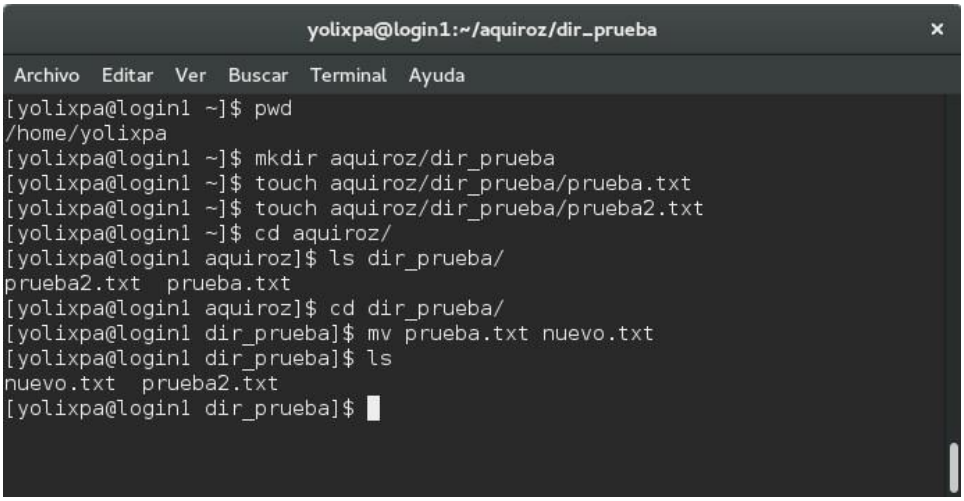

Figura 3: Ejemplos de ejecución de comandos de gestión de ficheros y directorios

3.3 Método de acceso a su cuenta en la supercomputadora

A continuación, se presenta el procedimiento para establecer una conexión desde su computadora local a su cuenta en la supercomputadora Cuetlaxcoapan utilizando SSH (Secure SHell), el cual es un protocolo de comunicación para establecer una sesión remota en una computadora en forma segura usando la línea de comandos. Es la única opción permitida para acceder a la supercomputadora Cuetlaxcoapan.

Por motivos de seguridad, el acceso se lleva a cabo utilizando el servidor ui.buap.mx (con dirección IP 148.228.11.31) como nodo de acceso intermedio, y una vez que haya accedido a éste, deberá hacer una segunda conexión al servidor login del LNS. Ud. tiene una cuenta en ui.buap.mx con el mismo nombre que en la supercomputadora Cuetlaxcoapan e inicialmente con la misma contraseña.

Debido a este método de acceso, es importante que comprenda los roles que toman los equipos de cómputo con los que interactúa con cada conexión SSH que establece,

especialmente al realizar transferencias de archivos (véase Sección 3.5). La figura 4 muestra los roles mencionados a continuación:

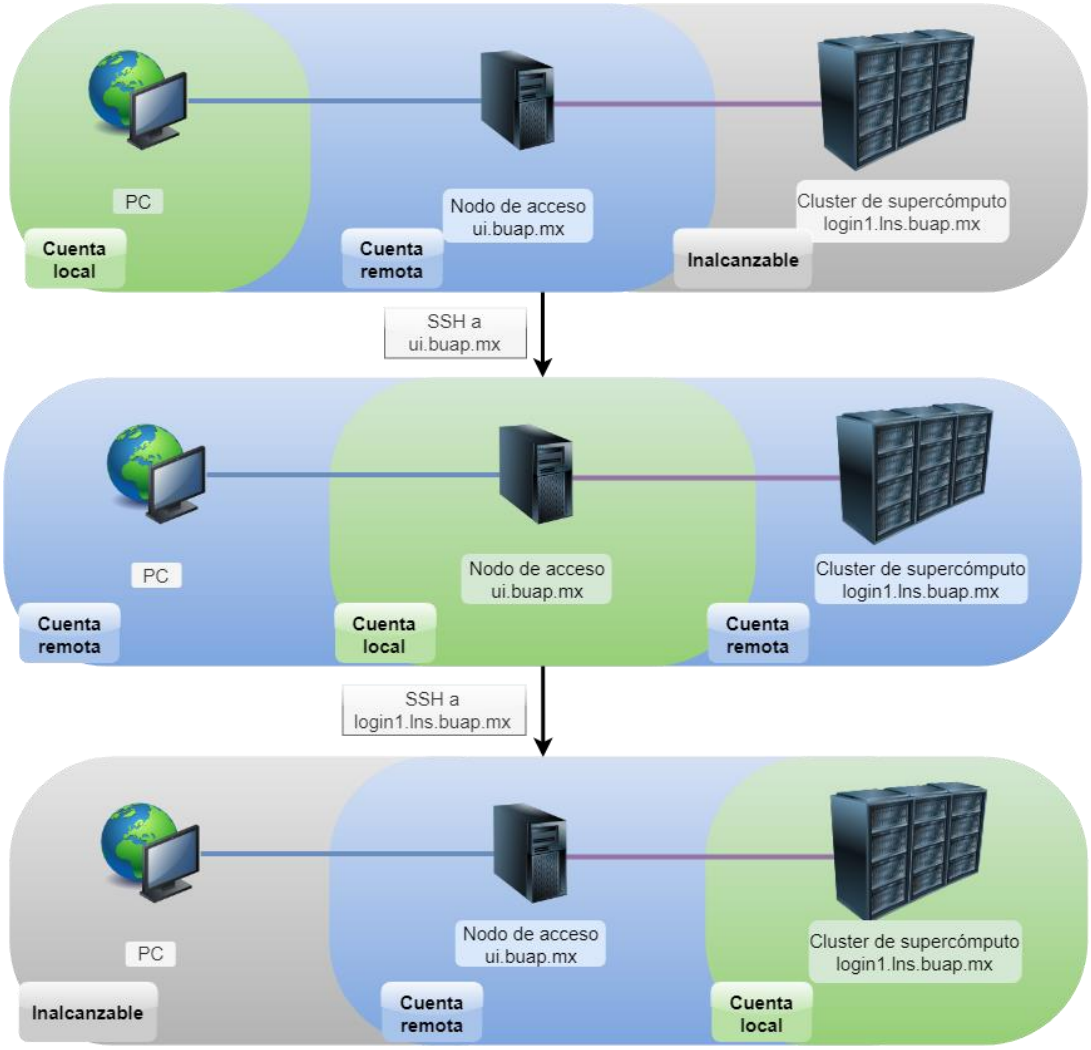

Figura 4: Roles que adquieren sus cuentas con cada conexión SSH que establece

- Entiéndase por *cuenta local* a la cuenta en un equipo de cómputo o *host* con la cual está interactuando en ese momento, independientemente de su ubicación física. Si se encuentra trabajando en la terminal de su computadora de escritorio, pero se encuentra introduciendo comandos en el intérprete de comandos de su cuenta en *ui.buap.mx*, significa que en ese momento su *cuenta local* es *ui.buap.mx.*
- Entiéndase por *cuenta remota* a la cuenta en un equipo de cómputo o *host* a la cual puede conectarse por red desde su actual *cuenta local*. Si se encuentra trabajando en su cuenta de *ui.buap.mx*, es decir, esa es su cuenta local actual, las *cuentas*

remotas a las cuales puede acceder son su cuenta en *login1* y su cuenta en su computadora de escritorio.

- Siempre puede corroborar el equipo o *host* al cuál se encuentra conectado verificando el *prompt* de su terminal, el cual tiene una estructura similar a la siguiente:

```
[cuenta@host carpeta_activa] $
```

Por ejemplo:

```
[yolixpa@login1 Documentos]$
```

También puede verificar el equipo mediante el comando `hostname`:

```
[yolixpa@login1 ~]$ hostname
login1.lns.buap.mx
```

Por último, es importante aclarar que después de su primer inicio de sesión tanto en `ui.buap.mx` como en `login1`, deberá introducir una nueva contraseña de complejidad apropiada para sus conexiones subsecuentes (para mayor información con respecto a la fortaleza de su contraseña, consultar las políticas de uso del LNS en www.lns.buap.mx).

3.3.1 Conexión SSH desde Windows

El sistema operativo Windows no posee una línea de comandos compatible con Linux, sistema operativo con el que cuentan los servidores del cúster de supercómputo. Por esta razón, debemos utilizar una aplicación especial para establecer una conexión SSH hacia una maquina Linux.

Por practicidad, le recomendamos ampliamente que utilice el programa MobaXterm, el cual no sólo es un cliente SSH, sino que también ofrece una interfaz de comandos equivalente a la de los sistemas Unix (Linux o MacOSX), así como un servidor de ventanas X (*X server*) necesario para la interacción desde Windows con programas que requieren una interfaz gráfica (es decir, que no es posible operarlos mediante la línea de comandos) de sistemas Unix. Puede descargar su versión gratuita desde su sitio web https://mobaxterm.mobatek.net/download.html.

Existen otras aplicaciones con las que puede establecer una conexión SSH desde Windows, desde las simples como PuTTY (http://www.putty.org), hasta las que incluyen diversas funciones adicionales como Bitvise SSH Client (https://www.bitvise.com/ssh-client-

download) la cual proporciona también una interfaz gráfica para la transferencia de archivos vía SSH. Puede elegir la que mayores facilidades le ofrezca.

Las siguientes instrucciones corresponden a la configuración de MobaXterm en su equipo Windows ya que, debido a que ofrece una interfaz de comandos similar a Unix, podrá seguir los mismos pasos para establecer una conexión SSH desde Linux o Mac OS X, lo cual le facilitará en gran medida su interacción con el *cluster* de supercómputo.

- Descargue MobaXterm en su sistema Windows desde el sitio web https://mobaxterm.mobatek.net/download.html. Le recomendamos la edición gratuita en su edición portable, aunque puede utilizar también el instalador.
- Al ejecutar MobaXterm aparecerá el botón para abrir una nueva terminal (véase Figura 5). Es posible que Windows le pida mediante una ventana emergente que confirme si desea darle a MobaXterm permisos de comunicación con internet, a lo cual deberá presionar el botón de “Permitir”.
- Para facilitar su trabajo, le recomendamos que cambie los valores del directorio *home* de MobaXterm En el panel de herramientas superior debe encontrar y abrir el Menú *Settings -> Configuration*. Haga click en el icono de folder amarillo que se encuentra en el cuadro de texto de *Persistent home directory*. Busque la carpeta principal de su usuario de Windows y selecciónela. El valor que aparecerá en el cuadro de texto de *persistent home directory* será _ProfileDir_. Haga click en el botón OK para guardar su nueva configuración y cerrar la ventana. MobaXterm le pedirá que reinicie el programa para ver reflejados los cambios.

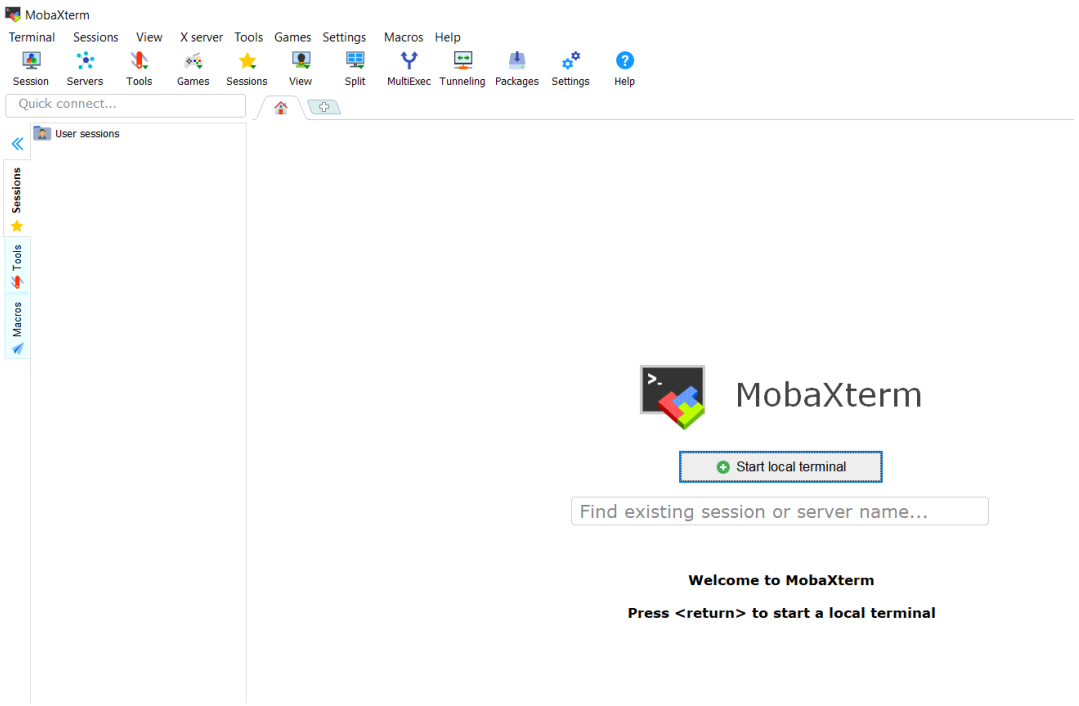

Figura 5: Pantalla principal de MobaXterm. En la parte superior se encuentra el menú de Settings. Al centro el botón para iniciar una nueva terminal.

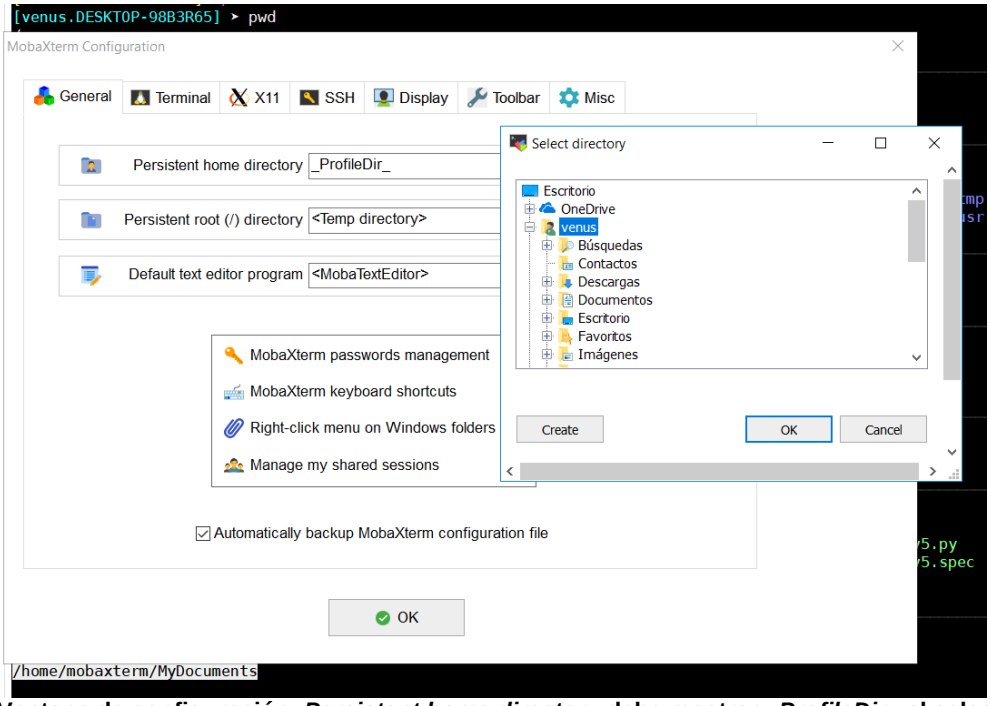

Figura 6: Ventana de configuración. *Persistent home directory* debe mostrar *_ProfileDir_* al seleccionar el directorio de su usuario en Windows. Puede identificarlo fácilmente al notar que contiene carpetas como Documentos o Escritorio.

- Una vez que haya realizado los pasos anteriores, podrá ejecutar en la terminal de MobaXterm una gran cantidad de comandos de estilo Unix (véase figura 7), incluyendo el comando *ssh*, por lo que, a partir de este punto, deberá referirse a las instrucciones de la sección 3.3.2: Conexión SSH desde Linux o Mac OS X para establecer la comunicación con el LNS.

Figura 7: Terminal de comandos de MobaXterm. Trabaja con un intérprete de comandos de tipo Unix, con lo que podrá apegarse a las instrucciones de dichos sistemas operativos

3.3.2 Conexión SSH desde Linux o Mac OS X

El acceso a la supercomputadora desde su computadora local usando Linux o Mac OS X (o de Windows si se encuentra utilizando una aplicación como MobaXterm) se realiza de manera sencilla abriendo primero la terminal y conectándose luego a `ui.buap.mx` (con dirección IP `148.228.11.31`) mediante la orden:

```
ssh cuenta@ui.buap.mx
```

o bien

```
ssh -l cuenta ui.buap.mx
```

También puede acceder al mismo servidor utilizando la dirección IP en lugar del nombre de servidor, es decir, mediante la orden:

```
ssh cuenta@148.228.11.31
```

Tome en cuenta que muchos comandos en Linux ofrecen la flexibilidad de ser ejecutados con distintas opciones y parámetros para realizar la misma acción. Puede utilizar el formato que le parezca más conveniente.

Al establecer la conexión por primera vez, es posible que ssh genere primero una clave de reconocimiento de la maquina remota (`ui.buap.mx` o `148.228.11.31`) y le solicite aceptarla tecleando `yes` en la terminal.

A continuación, ssh solicita la contraseña de la cuenta:

```
cuenta@ui.buap.mx's password:
```

o bien

```
cuenta@148.228.11.31's password:
```

Si es la primera vez que se conecta a *ui*, el sistema le solicitará que introduzca una nueva contraseña. Es un paso que realizará por única ocasión. Después de escribirla, le pedirá confirmarla para hacer el cambio de manera exitosa:

```
You are required to change your password immediately
Changing password for cuenta.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
```

Al introducir la contraseña correctamente tendrá acceso a la línea de comandos en su `cuenta` en `ui.buap.mx`:

```
Last login: Fri Feb 12 00:00:00 2016 from 195.221.47.54
[cuenta@ui ~]$
```

Desde el servidor `ui.buap.mx` deberá conectarse al nodo de acceso (*login*) de la supercomputadora *Cuetlaxcoapan* realizando las mismas acciones anteriores, pero indicando una nueva dirección de destino de conexión, mediante la orden:

```
ssh cuenta@192.168.170.213
```

Al igual que en el caso de la conexión a `ui.buap.mx`, *ssh* generará la primera vez que establezca una conexión una clave de reconocimiento para `192.168.170.213`, y le solicitará aceptarla tecleando `yes` en la pantalla.

A continuación, le pedirá la contraseña de su cuenta en `192.168.170.213` (dirección IP del nodo de acceso de la supercomputadora `login1`):

```
cuenta@192.168.170.213's password:
```

De la misma manera que con `ui`, si es la primera vez que se conecta a `login`, el sistema le solicitará por única ocasión que introduzca una nueva contraseña. Después de escribirla, le pedirá confirmarla para hacer el cambio de manera exitosa:

```
You are required to change your password immediately
Changing password for cuenta.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
```

Al introducir la contraseña correctamente aparecerá la línea de comando de bash y el *prompt* en el nodo de *login* de la supercomputadora Cuetlaxcoapan (véase la Figura 8).

A continuación, podrá empezar a utilizar la supercomputadora.

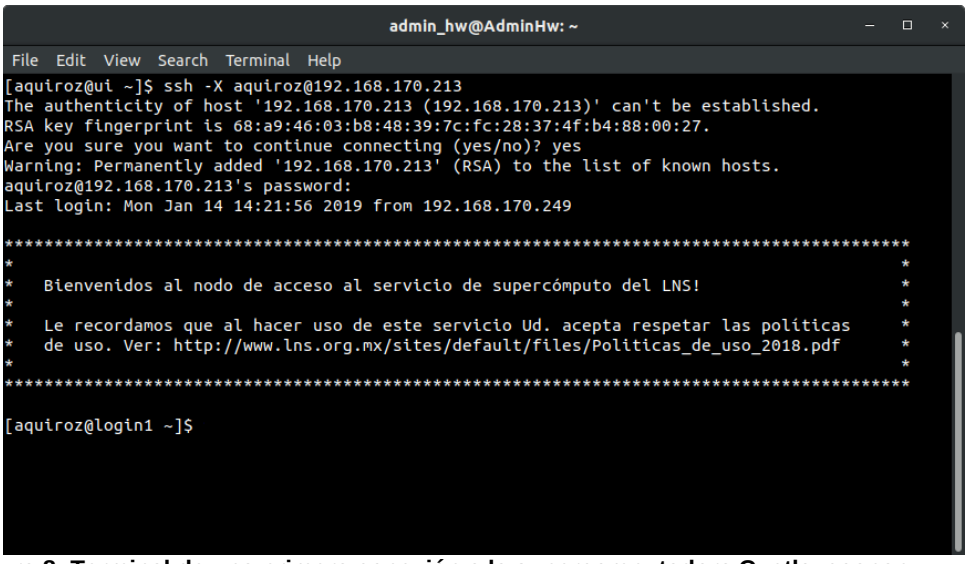

Figura 8: Terminal de una primera conexión a la supercomputadora Cuetlaxcoapan

3.4 Procedimiento posterior a la primera conexión con su cuenta en el LNS

Es indispensable que, después de realizar la primera conexión SSH a su cuenta la supercomputadora *Cuetlaxcoapan*, realice las siguientes tareas:

a) Cambie la contraseña con la que accedió a su cuenta por una más segura. Recomendamos que utilice un mínimo de 8 caracteres y que incluya una combinación de letras, números y símbolos (!"#$%&/()=). El cambio de contraseña se solicitará de manera obligatoria justo después de establecer conexión por primera vez con los servidores *UI* y *login* (como se indica en el procedimiento de la Sección 3.3.2), por lo que no necesita ejecutar ningún comando adicional para conseguirlo. Si más tarde desea volver a cambiar su contraseña, siempre puede ejecutar el comando:

```
passwd
```

y enseguida aparecerán en pantalla las opciones para ingresar su vieja contraseña, la nueva y la confirmación de la misma.

b) Genere las claves de SSH de los nodos de cálculo para activar el acceso desde el nodo de *login*. Esta tarea es necesaria para que el gestor de tareas del *cluster* de supercómputo pueda ejecutar adecuadamente sus trabajos en cualquier nodo de cómputo. Es posible realizar toda la configuración necesaria ejecutando la siguiente instrucción:

```
/software/LNS/txt/gen_claves_ssh.sh
```

3.5 Transferencias de archivos

Es importante tener en cuenta que al utilizar una conexión intermedia a `ui.buap.mx`, la transferencia de archivos entre su computadora local a la supercomputadora *Cuetlaxcoapan* no se puede realizar directamente, por lo que tendrá que realizar la transferencia de datos mediante los procedimientos descritos a continuación.

3.5.1 Procedimiento para transferencia desde su computadora local hacia la supercomputadora *Cuetlaxcoapan*

1. Desde su computadora local copie sus archivos a su cuenta en `ui.buap.mx` utilizando los comandos de transferencia de archivos que se mencionan en la *sección 3.5.3*: *Comandos para transferencia de archivos.*
2. Conectarse a su cuenta en `ui.buap.mx` usando SSH, tal como se ha descrito en la sección 3.3.
3. Transferir los archivos a su cuenta en `192.168.170.213` (nodo de *login* de la supercomputadora) usando los comandos que se mencionan en la *sección 3.5.3*: *Comandos para transferencia de archivos*.

3.5.2 Procedimiento para transferencia desde la supercomputadora hacia su computadora local

1. Conectarse a su cuenta en la supercomputadora usando SSH, tal como se ha descrito en la sección 3.3.
2. Transferir los archivos a su cuenta en `ui.buap.mx` utilizando los comandos de transferencia de archivos que se mencionan en la sección *Comandos para transferencia de archivos en la sección 3.5.3*.
3. Transferir los archivos desde `ui.buap.mx` a su computadora local usando los comandos de transferencia de archivos que se mencionan en la sección *Comandos para transferencia de archivos en la sección 3.5.3.*

3.5.3 Comandos para transferencia de archivos

A continuación, se describen algunos comandos que puede utilizar para realizar transferencia de archivos.

3.5.3.1 SFTP (SSH File Transfer Protocol)

sftp es un comando interactivo para transferir archivos entre dos computadoras usando el protocolo SSH en la línea de comandos. Para ello, primero se establece una conexión desde la computadora local a la computadora remota mediante el comando:

```
sftp usuario@nombre_de_computadora_remota
```

o bien

```
sftp usuario@ip_de_computadora_remota
```

Por ejemplo:

```
sftp yolixpa@192.168.170.213
```

Enseguida se conectará a la computadora remota y el servidor le pedirá que ingrese la contraseña de la cuenta a la que se desea acceder. Al establecer la conexión, le dará acceso al directorio del usuario en la computadora remota. En la línea de comandos aparecerán:

```
Connecting to 192.168.170.213
Using username "yolixpa".
yolixpa@ui.buap.mx's password:

Remote working directory is /home/yolixpa
sftp>
```

A continuación, podemos introducir comandos de SFTP para realizar labores como cambio de directorio y transferencia o recuperación de archivos:

- Para verificar el directorio actual de trabajo en la cuenta remota:

```
sftp> pwd
Remote directory is /home/yolixpa
```

- Para ver el contenido del directorio remoto:

```
sftp> ls
Documents   Downloads   bin   tmp   programas
```

- Para cambiar de directorio remoto:

```
sftp> cd programas
```

- Para transferir un archivo a la cuenta remota:

```
sftp> put file
Uploading file to /home/yolixpa/programas/file
```

- Para recuperar un archivo remoto a la cuenta local:

```
sftp> get file
Fetching /home/yolixpa/programas/file to file
```

- Se puede obtener ayuda interactiva sobre los comandos de SFTP mediante la orden:

```
sftp> help
Available commands:
bye                Quit sftp
exit         Quit sftp
cd path      Change remote directory to "path"
ls           Display remote directory listing
rm path      Delete remote file "path"
etc.
```

- Para salir del intérprete de comandes de *sftp* escriba el comando:

```
sftp> exit
```

3.5.3.2 SCP (Secure Copy)

SCP copia archivos entre dos computadoras remotas de manera no interactiva usando el protocolo SSH.

Tome en cuenta que, al hacer conexiones SSH, el servidor le pedirá la contraseña de su cuenta remota con cada ejecución del comando SCP, a menos que tenga configurado un acceso remoto mediante llaves SSH.

La sintaxis de un comando *scp* es la siguiente:

```
scp [options] source destination
```

donde `source` es la ruta y archivo desea copiar, y `destination` es la ruta y el nombre con el que se guardará la copia. En caso de que la fuente o destino sea en una computadora remota, se debe agregar también la cuenta y dirección del equipo que participará en la copia.

Para enviar un archivo desde la computadora local a un directorio en la computadora remota:

```
scp local_file user@remote_host:path/file
```

donde `local_file` es el archivo que desea copiar desde su cuenta local, `remote_host` es el nombre o dirección IP del servidor remoto, `path` representa la ruta del archivo en formato Unix, y `file` el nombre y extensión con el que se guardará en el equipo remoto.

Por ejemplo:

```
scp local_file yolixpa@192.168.170.213:/home/yolixpa/file
```

Para recuperar un archivo remoto:

```
scp user@remote_host:path/file local_path/local_file
```

donde `remote_host` es el nombre o dirección IP del servidor remoto, `path` representa la ruta del archivo en el equipo remoto en formato Unix, `file` es el nombre y extensión del archivo a copiar, `local_path` es la ruta local en formato Unix en la que desea guardar la copia, y `local_file` es el nombre y extensión con el que guardará el archivo en su cuenta local.

SCP posee la capacidad de transferir el contenido completo de directorios mediante la opción “-r“, por ejemplo:

```
scp -r user@remote_host:dir local_dir/
```

donde `remote_host` es el nombre o dirección IP del servidor remoto, `dir` representa la ruta del directorio en el equipo remoto en formato Unix que se desea copiar, y

`local_dir` es la ruta y nombre del directorio con el que guardará la copia en su cuenta local.

Para mayor información sobre SCP se puede solicitar el manual del comando mediante la orden:

```
man scp
```

3.5.3.3 rsync

Es un comando utilizado para copiar y sincronizar archivos desde o hacia un sistema remoto; utiliza un sistema de compresión y descompresión de datos al momento de enviar o recibir archivos y, si ha realizado la copia de un archivo previamente, al actualizar su copia es capaz de transferir únicamente los cambios entre el archivo origen y destino; por lo anterior se considera la opción más rápida entre de los comandos de transferencia tradicionales, como `scp`. Recomendamos que utilice `rsync` para sus transferencias.

Antes de utilizar `rsync`, asegúrese de tenerlo instalado en su sistema. Para verificar esto puede simplemente escribir el comando

```
rsync --version
```

en su terminal; si el programa despliega el nombre y versión de `rsync` instalada, puede continuar.

La sintaxis de un comando `rsync` es la siguiente:

```
rsync [opciones] fuente destino
```

donde `fuente` es la ruta y nombre del archivo o directorio que quiere transferir; `destino` es la dirección de la computadora a la que desea acceder; y en `opciones` se establecen distintos modos de ejecución del comando rsync, siendo las más interesantes: `-v` que ofrece información detallada del proceso, `-a` que permite una copia de archivos y directorios junto con sus propiedades y rutas originales, y `-z` que habilita las opciones de compresión al transferir. Para combinar las 3 opciones al mismo tiempo, la sintaxis puede ser `-azv`.

El uso del comando `rsync` es similar al descrito en la sección 3.5.3.2 para el comando SCP, tanto para la transferencia de un archivo desde su cuenta del *cluster* de supercómputo hacia su computadora como a la inversa.

Para enviar un archivo desde la computadora local a un directorio en la computadora remota utilice:

```
rsync -avz user@remote_host:path/file local_path/local_file
```

donde `remote_host` es el nombre o dirección IP del servidor remoto, `path` representa la ruta del archivo en el equipo remoto en formato Unix, `file` es el nombre y extensión del archivo a copiar, `local_path` es la ruta local en formato Unix en la que desea guardar la copia, y `local_file` es el nombre y extensión con el que guardará el archivo en su cuenta local.

Por ejemplo:

```
rsync -avz yolixpa@192.168.170.213:ruta/archivo rutalocal/archivo
```

Recuerde agregar la ruta y nombre completos del archivo, incluyendo la extensión del mismo.

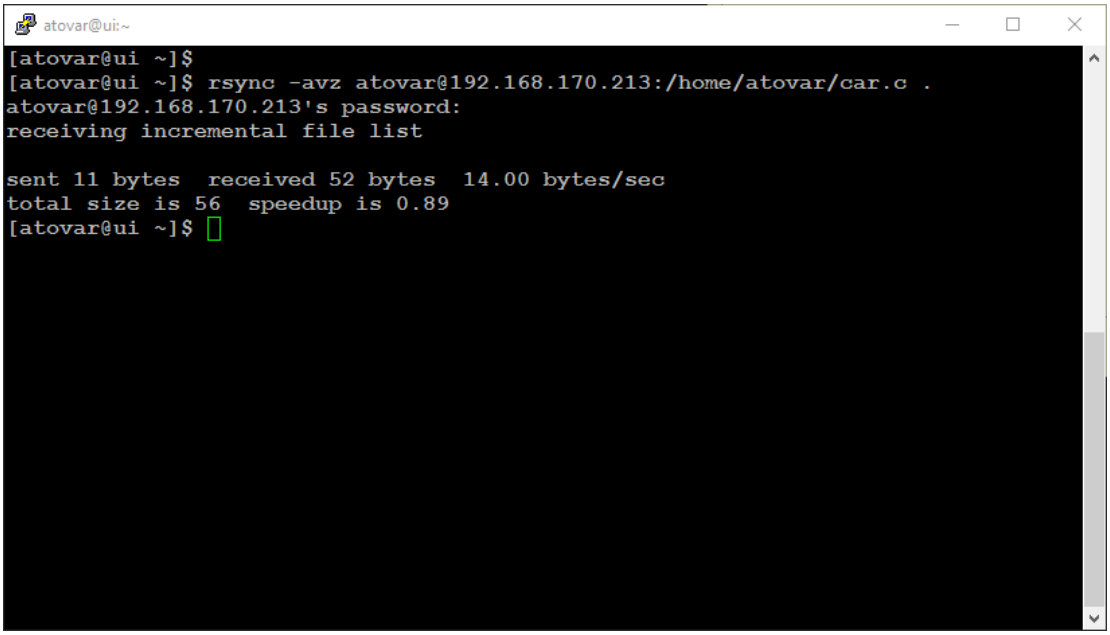

Figura 9: Ejemplo de transferencia de archivos utilizando rsync.

Tome en cuenta que, al igual que con el comando SCP, el servidor le pedirá la contraseña de su cuenta remota con cada ejecución del comando *rsync*, a menos que tenga configurado un acceso remoto mediante llaves SSH.

3.6 Edición de archivos en la consola de comandos

Para realizar modificaciones menores en los archivos en su cuenta, existen diversas opciones de software a su disposición. Programas como vi, vim y nano son comunes y muy utilizados para realizar dichas ediciones de texto en archivos. Recomendamos nuevamente que busque más información sobre tales editores para sacar el máximo provecho a sus funciones.

3.6.1 Programas de edición de texto en terminal

A continuación, se describen las generalidades de los editores antes nombrados:

3.6.1.1 vi

Es un editor de texto incluido en todos los sistemas Unix, es decir, tanto Linux como Mac Os. Para lanzarlo, utilice simplemente el comando:

vi [archivo]

donde archivo es el nombre y, si no se encuentra el archivo en el directorio de trabajo actual, la ruta del archivo que deseamos editar con vi.

Tiene dos modos principales de operación, el modo de *edición o inserción de texto* y el *modo de comandos*. El primero sirve, como su nombre lo indica, para ingresar datos de texto hacia el documento; el segundo por su parte permite la escritura de comandos cortos que editen de manera eficaz el contenido del texto en el archivo.

Existen modos de edición adicionales llamados *modos visuales*, en los cuales es posible seleccionar de manera gráfica el texto para manipularlo. Con ellos, puede seleccionar el texto por caracteres, por líneas o por bloques.

En la tabla 2 encontrará algunos de los comandos más utilizados en *vi,* así como los comandos para acceder a los múltiples modos de operación.

Existe una inmensa variedad de comandos disponibles para el editor *vi*. Reiteramos la utilidad de investigar más acerca de los mismos. En internet puede encontrar comandos avanzados de vi.

Tabla 2: Comandos básicos de editores vi y vim

Modo de edición		Modo de comandos	
Comando	Utilidad	Comando	Utilidad
`a, i`	inicia modo de edición (variantes)	`<ESC>`	Entra al modo de comandos
`o, O`	inicia edición insertando líneas	`:w nombre`	Escribe en el archivo "nombre"
`v`	Inicia el modo visual (por caracteres)	`y`	En modo visual, copia el texto seleccionado
`V,Shift+v`	Inicia el modo visual por líneas	`p`	En modo visual, pega el texto copiado
`Ctrl+v`	Inicia el modo visual por bloques	`:wq`	Escribe el archivo y sale de vi
		`:q!`	Sale del editor sin guardar

3.6.1.2 vim

Es un editor de texto sucesor de *vi* que incorpora características extras en la edición de texto, sobre todo para edición de código en varios de los lenguajes de programación más comunes. Usuarios con suficiente práctica pueden aprovechar múltiples características adicionales, como editar múltiples archivos en pantalla dividida, formateo de color y resaltado de texto, por mencionar algunos ejemplos. Se instala fácilmente a través del manejador de paquetes de su sistema operativo Unix, es decir, tanto Linux como Mac Os pueden ocuparlo. Para lanzarlo, utilice simplemente el comando:

```
vim [archivo]
```

donde `archivo` es el nombre (y la ruta si no se encuentra el archivo en el directorio de trabajo actual) que deseamos editar con `vim`.

Al igual que `vi`, tiene dos modos principales de operación, el modo de *edición o inserción de texto* y el *modo de comandos*. El primero sirve, como su nombre lo indica, para ingresar datos de texto hacia el documento; el segundo por su parte permite la escritura de comandos cortos que editen de manera eficaz el contenido del texto en el archivo.

También cuenta con los *modos visuales* de edición, en los cuales es posible seleccionar el texto por caracteres, por líneas o por bloques para manipularlo.

Los comandos que utiliza en `vi`, también puede utilizarlos en `vim` (véase tabla 2). Puede encontrar comandos avanzados de vim fácilmente en la red.

3.6.1.3 nano

Es un editor de texto básico pero eficiente, utilizado en sistemas Unix, es decir, tanto Linux como Mac Os pueden ocuparlo. Se instala fácilmente a través del manejador de paquetes de su sistema operativo.

Para ocupar *nano* utilice el comando:

```
nano [archivo]
```

donde `archivo` es el nombre (y la ruta si no se encuentra el archivo en el directorio de trabajo actual) que deseamos editar con `nano`.

Este editor de texto, a diferencia de `vi` *y* `vim`, no tiene un modo de comandos, pues en todo momento se encuentra en lo que denominamos en dichos editores como *modo de edición*. La forma en que se introducen los comandos es mediante la combinación de teclas `Ctrl+[letra/tecla(s)]`, siendo los comandos más utilizados los de copiado `(Ctrl+Shift+C)`, pegado `(Ctrl+Shift+V)` y cierre del editor `(Ctrl+X)`. En la parte inferior de la pantalla del editor puede encontrar otros de los comandos más utilizados. Si desea buscar aún más comandos para `nano`, sugerimos consultar ayuda en la red.

4. Chequeo y carga de software disponible en el LNS

Por motivos de seguridad y rendimiento, antes ejecutar programas en el *cluster* de supercómputo requiere indicar qué herramientas de software va a utilizar. Para ello, una vez establecida la conexión con su cuenta en el LNS, se deben cargar como módulos complementarios cualquiera de las herramientas y programas que el Laboratorio tiene a su disponibilidad. Este paso se debe realizar cada que inicie sesión en su cuenta.

Si utiliza los *scripts* proporcionados por esta guía en la sección 5.2: *Scripts para ejecución de programas en SLURM*, ya habrá cargado los módulos necesarios para ejecutar el programa que desea. Sin embargo, si requiere ejecutar algún otro programa que no se encuentra detallado en dicha sección, probablemente deberá cargar más módulos en su sesión.

Los comandos que necesitará para realizar dicha tarea son los siguientes:

- Para verificar los módulos disponibles para el usuario y el directorio que los contiene es:

  `module avail`

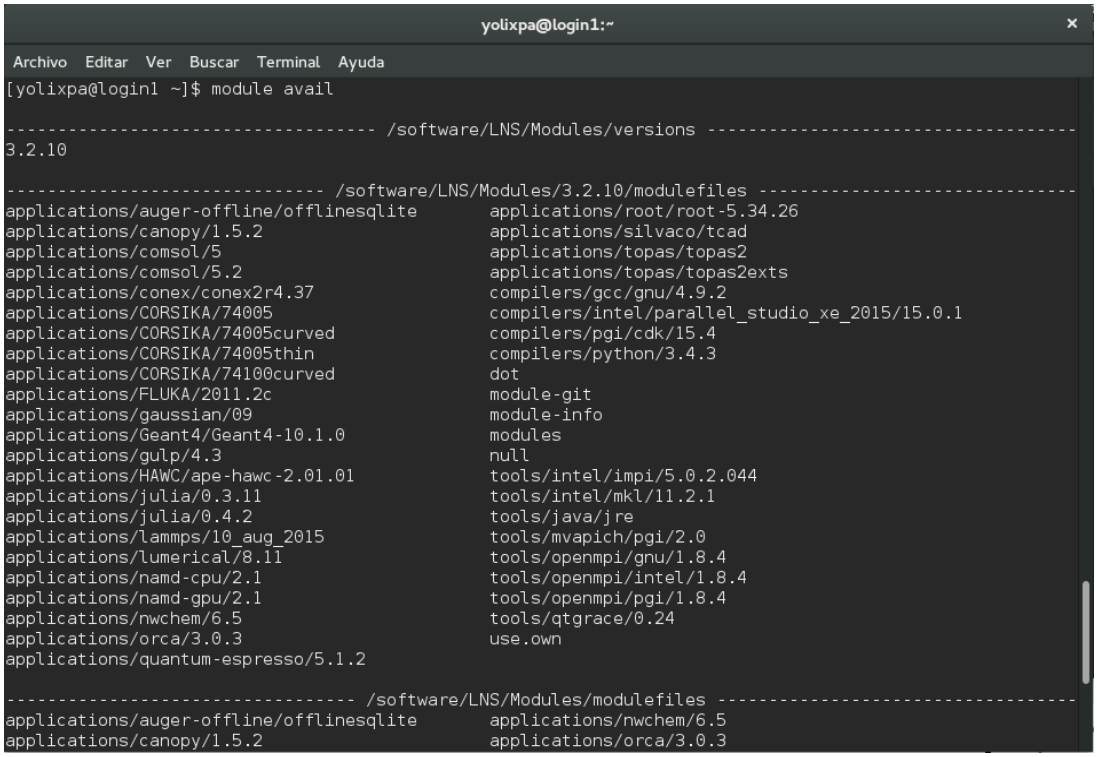

Figura 10: Ejemplo de ejecución del comando *module avail*

- Para cargar un módulo específico use:

  `module load [directorio]`

  donde `directorio` es la ruta y versión del módulo que desea cargar. La obtiene cuando ejecuta el comando `module avail`

- Para verificar los módulos que ya tiene cargados en su sesión:

  `module list`

- Para volver a inhabilitar alguno de los módulos que ya tiene cargados en su sesión:

  `module unload [directorio]`

  donde `directorio` es la ruta y versión del módulo que desea deshabilitar.

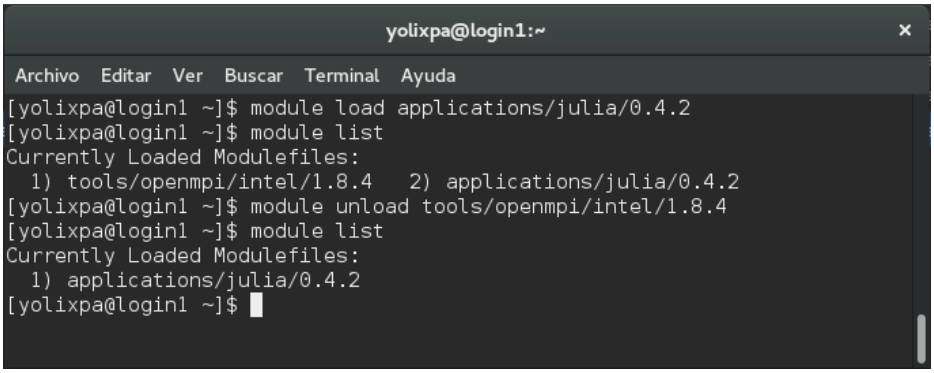

Figura 11: Ejemplos de ejecución de comandos de uso de módulos

5. SLURM

Como su nombre lo indica (Simple Linux Utility for Resource Management), SLURM es un administrador *open-source* de carga de trabajos diseñado para *clusters* Linux de cualquier dimensión. Es utilizado en varias de las computadoras más grandes del mundo.

Provee tres funciones claves:

1. Asigna a los usuarios un acceso exclusivo y/o no exclusivo a recursos (nodos computacionales) por un periodo de tiempo determinado de tal forma que puedan realizar sus tareas.
2. Proporciona un marco para iniciar, ejecutar y monitorear el trabajo (usualmente una tarea paralela) en un set de nodos asignados.
3. Modera la contención de recursos gestionando una cola de trabajos en espera.

Aunque existen otros gestores de carga de trabajos, SLURM es único en varios aspectos:

Escalabilidad: Está diseñado para operar en un *cluster* heterogéneo con decenas de millones de procesadores.

Desempeño: Puede aceptar 1000 peticiones de trabajo por segundo y ejecutar 500 tareas simples por segundo (dependiendo del hardware y la configuración del sistema).

Gratuito y de Código Abierto: Su código fuente está disponible bajo la Licencia Pública General de GNU (GNU General Public License).

Portabilidad: Escrito en C con un motor de configuración de GNU. Si bien escrita inicialmente para Linux, SLURM ha sido portado a una amplia variedad de sistemas.

Gestión de Energía: Las tareas pueden especificar la frecuencia de CPU deseada y el costo energético por tarea es registrado. Recursos desocupados pueden ser apagados hasta requerirlos.

Tolerancia a fallos: Es muy tolerante a los fallos del sistema, incluyendo errores en la ejecución de funciones de control de nodos.

Flexibilidad: Existe un mecanismo de plugins para soportar diversas interconexiones, mecanismos de autenticación, planificadores, etc. Estos plugins están documentados y son bastante sencillas para el usuario final para que pueda entender las fuentes y agregar funcionalidad.

Tareas Dimensionables: Las tareas pueden crecer y reducirse a demanda. Las peticiones de trabajo pueden especificar rangos de tamaño y límite de tiempo.

Trabajos de estado: Tareas de estado en ejecución en el nivel de tareas individuales para ayudar a identificar los desequilibrios de carga y otras anomalías.

5.1 Comandos de SLURM

Los usuarios pueden interactuar con SLURM utilizando diversos comandos de terminal. Los más importantes para los usuarios del *cluster* son los siguientes:

- **sbatch**: Se utiliza para enviar un script de trabajo para su posterior ejecución. El script contendrá típicamente uno o más comandos *srun* para lanzar tareas paralelas.
- **srun**: Se utiliza para enviar un trabajo para su ejecución o iniciar pasos de trabajo en tiempo real. *srun* tiene una amplia variedad de opciones para especificar los requisitos de los recursos, incluyendo: mínimo y máximo número de nodos, conteo de procesadores, y características específicas de nodos a usar (cuanta memoria,

espacio en disco, ciertas características requeridas, etc.). Un trabajo puede contener varios pasos de trabajo que se ejecutan de forma secuencial o en paralelo en nodos independientes o compartidos dentro de la asignación de nodos del trabajo.

- **squeue**: Reporta el estado de los trabajos o pasos de trabajo. Cuenta con una amplia variedad de filtrado, clasificación y opciones de formato. Por defecto, reporta los trabajos que se están ejecutando en orden de prioridad y luego los trabajos pendientes ordenados también por prioridad.
- **scancel**: Se utiliza para cancelar un paso de trabajo o un trabajo pendiente o en ejecución. También puede ser usado para enviar una señal arbitraria a todos los procesos asociados con un trabajo en ejecución o paso de trabajo.

A continuación, se mencionan comandos más avanzados de SLURM en caso de que requiera funciones más avanzadas para gestionar sus trabajos:

- **sinfo**: Informa el estado de las particiones y los nodos gestionados por SLURM. Cuenta con una amplia variedad de opciones de filtrado, clasificación y formato.
- **sattach**: Se utiliza para vincular las entradas, salidas y errores estándar a un trabajo o paso de trabajo en ejecución. Es posible vincular y desvincularlos de trabajos múltiples veces.
- **sbcast**: Se utiliza para transferir un archivo desde un disco local a el disco local en los nodos asignados a un trabajo. Esto puede ser utilizado para ocupar eficazmente los nodos de cómputo sin disco o proporcionar un mejor rendimiento con respecto a un sistema de archivos compartidos.

Si desea información detallada del uso de cualquiera de los demonios, comandos y funciones de APIs de SLURM puede obtenerla mediante el comando `man` (por ejemplo: `man squeue`). También puede ejecutar un comando con las opciones `--help` o `--usage` para desplegar una lista con las opciones disponibles y los parámetros adecuados para su ejecución (por ejemplo: `sbatch --help`). Nótese que todos los comandos distinguen entre mayúsculas y minúsculas. Es posible ejecutar estos comandos en cualquier nodo del *cluster*.

5.1.1 Comandos de SLURM esenciales para la ejecución de Jobs en la

supercomputadora

A continuación, se presentan el procedimiento para hacer uso de los comandos esenciales de SLURM mediante el cual los usuarios del LNS podrán ejecutar sus programas en la supercomputadora:

- Es necesario crear un archivo denominado *script* o *job script* (véase la *sección 5.2: Scripts para ejecución de programas en SLURM*) con los detalles de su cálculo para después enviarlo al sistema de colas SLURM mediante la orden:
  `sbatch job_script`

  donde `job_script` es el nombre de su archivo creado.

- Para monitorear sus tareas (*jobs*) en SLURM se realiza con el comando
  `squeue -u user_name`

  donde `user_name` es el nombre de usuario con el que está registrado en el sistema.

- Para cancelar alguna tarea se hace con el comando
  `scancel job_id`

  donde `job_id` es el identificador que aparece en la salida de `squeue`

5.2 Scripts para ejecución de programas en SLURM

Un *script* es un pequeño archivo que contiene comandos para que un intérprete pueda ejecutarlos. En el caso de los scripts para SLURM, incluye tanto comandos para terminal que ya conoce como directivas para que SLURM configure la forma en que sus cálculos se ejecutarán en el *cluster* de supercómputo. Para hacer un *job script*, simplemente abra su editor de texto preferido, agregue los comandos e instrucciones necesarias y guarde el archivo sin una extensión específica. Para ejecutar un script, utilice el comando `sbatch`.

Es sumamente importante que mande a ejecutar sus cálculos a través de un *script* de SLURM, ya que sólo de esta manera podrá hacer uso del poder de procesamiento de la supercomputadora del LNS.

En esta sección podrá encontrar ejemplos de scripts para ejecutar algunos de los programas LAMMPS, Quantum Espresso, SIESTA, GROMACS y Gaussian, disponibles en el LNS, los cuales ya han sido probados por usuarios reales y utilizados para la carga de tareas en la cola de trabajos (*queue*) de SLURM.

5.2.1 Script genérico

Si entre las sugerencias disponibles no encuentra el script para un programa específico, puede utilizar la siguiente estructura genérica:

```
#!/bin/bash
#SBATCH -J test        # job name
#SBATCH -o test.o%j    # output and error file name (%j expands to jobID)
#SBATCH -n 1           # number of MPI tasks (cores) requested
#SBATCH --ntasks-per-node=1   # task (cores) per node (maximum 24)
#SBATCH -p comp               # SLURM queue (partition)
#SBATCH -t 01:00:00           # run time (hh:mm:ss)
echo $SLURM_JOB_ID
echo $SLURM_JOB_NAME
echo $SLURM_JOB_NUM_NODES
# Load your modules here
module load compilers/intel/parallel_studio_xe_2015/15.0.1
module load tools/intel/impi/5.0.2.044

# Run your task here
mpirun -genv I_MPI_FABRICS shm:ofa YOUR_TASK
```

donde solo queda por especificar la cantidad de nodos y *cores* que utilizará para su cálculo, los módulos de software que utiliza y la ejecución final de su código en la última línea.

5.2.2 Uso de LAMMPS

El siguiente ejemplo es un *job script* para correr LAMMPS en 48 cores (2 nodos) del *cluster* de supercómputo.

```
#!/bin/bash
#SBATCH -J lmp      # job name
#SBATCH -o lmp.o%j  # output and error file name (%j expands to jobID)
#SBATCH -n 48       # total number of MPI tasks (cores) requested
#SBATCH --ntasks-per-node=24  # task (cores) per node (maximum 24)
#SBATCH -p comp               # SLURM queue (partition)
#SBATCH -t 24:00:00           # run time (hh:mm:ss)

# Load Intel Parallel Studio
module load compilers/intel/parallel_studio_xe_2015/15.0.1
module load tools/intel/mkl/11.2.1
```

```
module load tools/intel/impi/5.0.2.044
module load applications/lammps/10_aug_2015

# Run the LAMMPS task
mpirun -genv I_MPI_FABRICS shm:ofa $LMP < input > output
```

A partir de este *script*, solo queda sustituir en la última línea los nombres de los archivos de entrada (`input`) y salida (`output`) de su cálculo, y de ser necesario, el número de *cores* y tiempo de ejecución requeridos.

5.2.3 Uso de Quantum Espresso

A continuación, se presenta un ejemplo de *job script* para correr Quantum Espresso usando 4 nodos de cálculo con 24 cores cada uno (96 cores en total):

```
#!/bin/bash
#SBATCH -J pw          # job name
#SBATCH -o pw.o%j      # output and error file name (%j expands to jobID)
#SBATCH -n 96  # total number of MPI tasks requested (maximum 240)
#SBATCH --ntasks-per-node=24  # number of tasks per node (maximum 24)
#SBATCH -p comp                # SLURM queue (partition)
#SBATCH -t 24:00:00            # run time (hh:mm:ss)

# Load Intel Parallel Studio
module load compilers/intel/parallel_studio_xe_2015/15.0.1
module load tools/intel/mkl/11.2.1
module load tools/intel/impi/5.0.2.044
module load applications/quantum-espresso/5.1.2

# run the Quantum Espresso executable
mpirun -np 96 -genv I_MPI_FABRICS shm:ofa $PW -i input > output
```

A partir de este *script*, solo queda sustituir en la última línea los nombres de los archivos de entrada (`input`) y salida (`output`) de su cálculo, y de ser necesario, el número de *cores* y tiempo de ejecución requeridos.

5.2.4 Uso de SIESTA

El siguiente ejemplo es un *job script* para correr SIESTA en 24 cores (1 nodo) del *cluster* de supercómputo.

```
#!/bin/bash
#SBATCH -J siesta      # job name
#SBATCH -o siesta.o%j  # output and error file name (%j expands to jobID)
#SBATCH -n 24          # total number of MPI tasks requested
#SBATCH --ntasks-per-node=24  # number of tasks per node (maximum 24)
#SBATCH -p comp                # SLURM queue (partition)
#SBATCH -t 02:00:00            # run time (hh:mm:ss)
```

```
# Load Intel Parallel Studio
module load compilers/intel/parallel_studio_xe_2015/15.0.1
module load tools/intel/mkl/11.2.1
module load tools/intel/impi/5.0.2.044

# Run the parallel siesta executable
mpirun -np 24 -genv I_MPI_FABRICS shm:ofa /software/LNS/siesta
/3.2/bin/siesta < input.fdf
```

A partir de este *script*, solo queda sustituir en la última línea los nombres de los archivos de entrada (`input`) y salida (`output`) de su cálculo, y de ser necesario, el número de *cores* y tiempo de ejecución requeridos.

5.2.5 Uso de GROMACS

A continuación, se presenta un ejemplo de *job script* para correr GROMACS usando 4 nodos de cálculo con 24 cores cada uno (96 cores en total):

```
#!/bin/bash
#SBATCH -J gromacs     # job name
#SBATCH -o gromacs.o%j # output and error file name (%j expands to jobID)
#SBATCH -n 96          # total number of mpi tasks requested
#SBATCH --ntasks-per-node=24  # number of tasks per node
#SBATCH -p comp               # SLURM queue (partition)
#SBATCH -t 120:00:00          # run time (hh:mm:ss)
module load compilers/intel/parallel_studio_xe_2015/15.0.1
module load tools/intel/mkl/11.2.1
module load tools/intel/impi/5.0.2.044
source /software/LNS/gromacs/5.0.4/bin/GMXRC.bash

# run the GROMACS task
mpirun -genv I_MPI_FABRICS shm:ofa -np 96 mdrun_mpi -deffnm min.tpr
```

A partir de este *script*, solo queda sustituir en la última línea los nombres de los archivos de entrada (input) y salida (output) de su cálculo, y de ser necesario, el número de *cores* y tiempo de ejecución requeridos.

5.2.6 Uso de Gaussian

Para utilizar Gaussian de manera óptima es necesario considerar el hardware donde se va a correr. Esto es para determinar qué modo de ejecución (memoria compartida o memoria distribuida o ambos) es más conveniente para el problema que se está tratando y para estimar los requerimientos de memoria, almacenamiento y tiempo de cálculo. Por lo tanto, es importante conocer el escalamiento del método de estructura electrónica que se utiliza en función del número total de funciones base.

Es bien sabido, por ejemplo, que DFT tiene un escalamiento de memoria del orden de 3N^2 bytes, donde N es el número de funciones base. Asimismo, DFT tiene un escalamiento de espacio de disco (para almacenar las integrales) del orden de N^2 bytes. Tradicionalmente, la compañía Gaussian Inc. no ofrece información sobre el funcionamiento interno del código Gaussian para determinar con más certeza estos requerimientos, así que es necesario usar la experiencia para estimarlos.

En el caso específico del hardware del LNS, cada nodo de cálculo posee 24 cores de ejecución y 128 GB de memoria (que para fines prácticos se puede usar al 85%, o sea, del orden de 110 GB). Es importante no sobrepasar este monto de memoria. Por lo tanto, hay que declarar siempre la memoria a utilizar en el archivo de entrada de Gaussian (opción %mem) teniendo en cuenta la regla $3N^2$.

A continuación, se presenta un ejemplo de *job script* para correr Gaussian para procesar una molécula del orden de 60 átomos usando el modo de ejecución mixto (memoria distribuida + memoria compartida) utilizando 5 nodos de cálculo con 12 cores cada uno (60 cores en total) usando mediante el sistema de comunicación TCP Linda:

```
#!/bin/bash
#SBATCH -J gaussian     # job name
#SBATCH -o gaussian.o%j # output and error file name (%j expands to jobID)
#SBATCH -n 60           # total number of multicore tasks requested
#SBATCH --ntasks-per-node=12    # number of tasks per node (maximum 24)
#SBATCH -p comp                 # SLURM queue (partition)
#SBATCH -t 72:00:00             # run time (hh:mm:ss)

# Initialize Gaussian environment variables

module load applications/gaussian/09
ulimit -c 0
ulimit -d hard
ulimit -f hard
ulimit -l hard
ulimit -m hard
ulimit -n hard
ulimit -s hard
ulimit -t hard
ulimit -u hard
export GAUSS_SCRDIR=/home/$USER/tmp/$SLURM_JOBID
mkdir -p $GAUSS_SCRDIR

# Specify input and output files

INPUT=input.com
OUTPUT=output.out
```

```
# Append the list of Linda workers at the beginning of input file

nodelist=`scontrol show hostname $SLURM_NODE_LIST | tr '\n' ','`
nodelist=${nodelist%,}
sed "/LindaWorkers/d" $INPUT > ${INPUT}.tmp
sed -i "1i %UseSSH" ${INPUT}.tmp
sed -i "1i %LindaWorkers=$nodelist" ${INPUT}.tmp
sed -i "1i %NProcShared=12" ${INPUT}.tmp

# Run Gaussian
$g09root/g09/g09 < ${INPUT}.tmp > ${OUTPUT}

# Clean out garbage
rm -rf ${INPUT}.tmp
```

A partir de este *script*, solo queda especificar los archivos de entrada y salida en las líneas `INPUT=input.com` y `OUTPUT=output.out` con los nombres de archivo deseados para cálculo, y de ser necesario, el número de *cores* y tiempo de ejecución solicitados.

Además, para este ejemplo, es importante que no incluya línea que contenga la orden `%NProcShared` o `%LindaWorkers` de su fichero de entrada de Gaussian; y que en la línea de especificación de memoria coloque un valor apropiado para su cálculo, digamos para este ejemplo, de `%mem=32gb`.

6. Información adicional

En caso de requerir más información relativa a los tópicos descritos en esta guía, es posible encontrar gran cantidad de recursos en la red y/o en otras guías proporcionadas por el LNS. Si desea obtener ayuda personalizada con respecto al manejo de su cuenta o la ejecución de sus cálculos, también puede dirigirse al correo de atención al usuario lns@correo.buap.mx