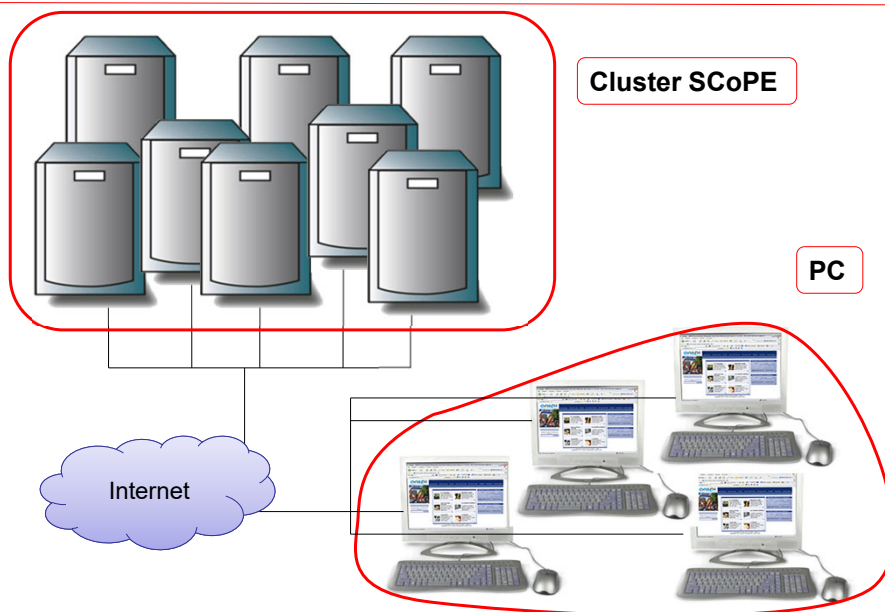


Introduzione all'architettura parallela messa a disposizione dal *Progetto SCoPE*

1

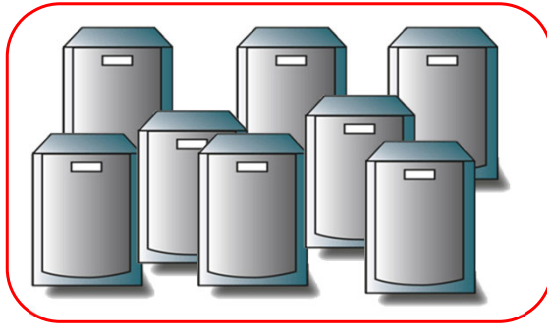
A disposizione



G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

2

A disposizione



Cluster SCoPE

Ogni nodo è un **server Dell PowerEdge M600** ciascuna dotato di:

- due processori quad core Intel Xeon E5410@2.33GHz (Architettura a 64 bit)
- 8 Gb di RAM
- 2 dischi SATA da 80GB in configurazione RAID1
- due schede Gigabit Ethernet configurate in bonding

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

3

Come accedere al cluster

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

4

Accesso al cluster

Da un qualsiasi PC dove sia disponibile il protocollo ssh da riga di comando (Linux, Unix...)

ssh login@ui-studenti.scope.unina.it

(la password all'inizio è uguale alla login, ma al primo accesso verrà chiesto di cambiarla)

Se siete su Windows l'operazione è simile ma vi è utile un client ssh, ed eventualmente un client sftp/scp



G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

5

Accesso al cluster

E' necessario un client SSH.

Se siete su una piattaforma Windows, va sicuramente bene l'emulatore di terminale PuTTY, scaricabile gratuitamente da <http://www.putty.org/>

host name ui-studenti.scope.unina.it
port 22
connection type SSH
login personale, fornita a lezione

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

6

Accesso al cluster

E' necessario un client SSH.

Se siete su una piattaforma Windows, va sicuramente bene il programma WinSCP, scaricabile gratuitamente da <https://winscp.net/eng/download.php>

Per spostare i file da e verso il cluster, può essere utile un client SFTP/FTP/SCP.

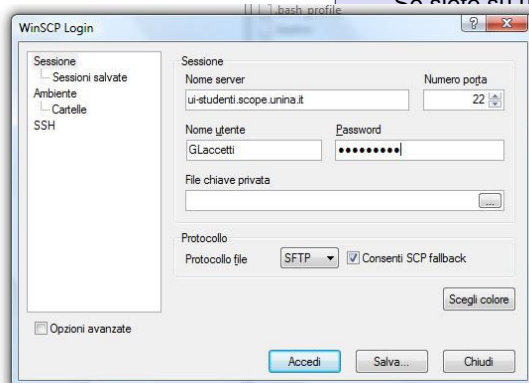
host name ui-studenti.scope.unina.it
port 22
connection type SSH
login personale, fornita a lezione

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

7

Accesso al cluster

E' necessario un client SSH.



Se siete su una piattaforma Windows, va bene il programma WinSCP, scaricabile gratuitamente da <https://winscp.net/eng/download.php>

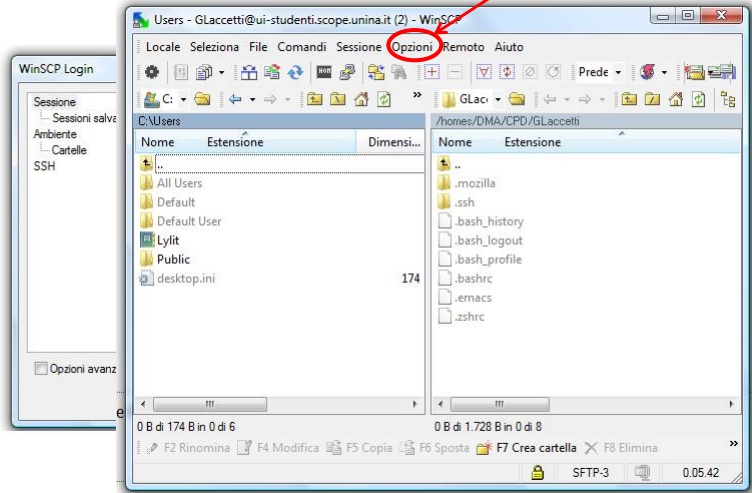
host name ui-studenti.scope.unina.it
port 22
connection type SSH
login personale, fornita a lezione

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

8

Accesso al cluster

E' necessario un client SSH.

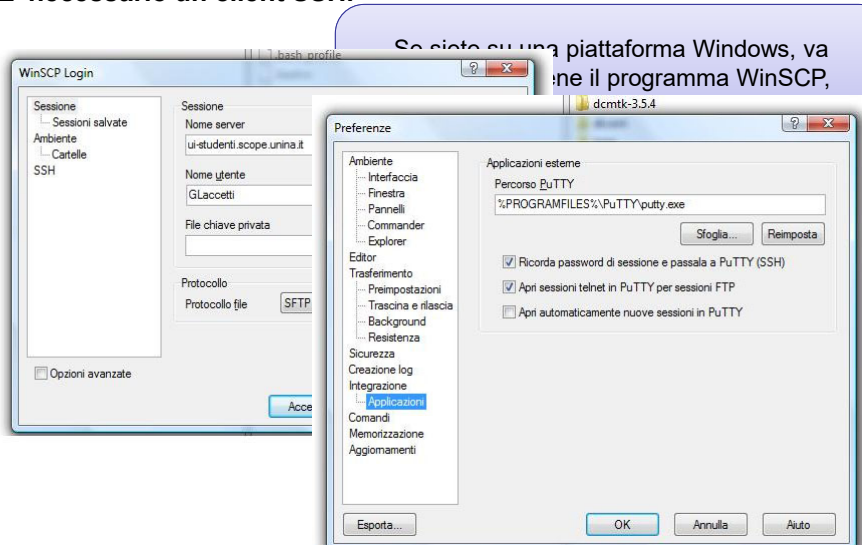


G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

9

Accesso al cluster

E' necessario un client SSH.



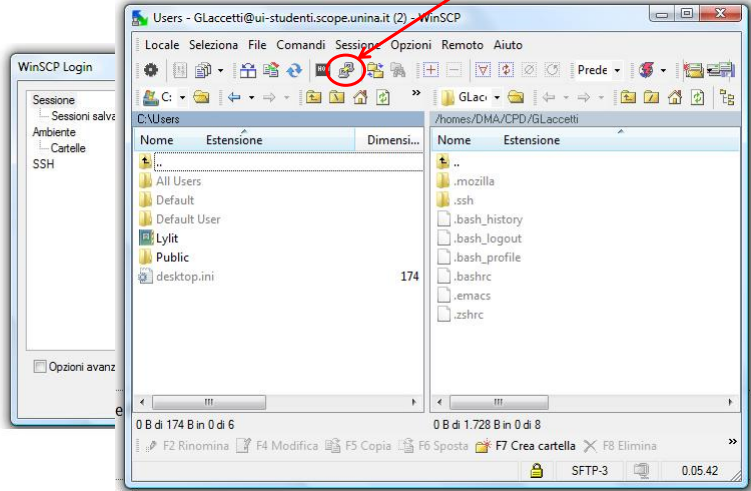
G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

10

Accesso al cluster

E' necessario un client SSH.

Sessione PuTTY



G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

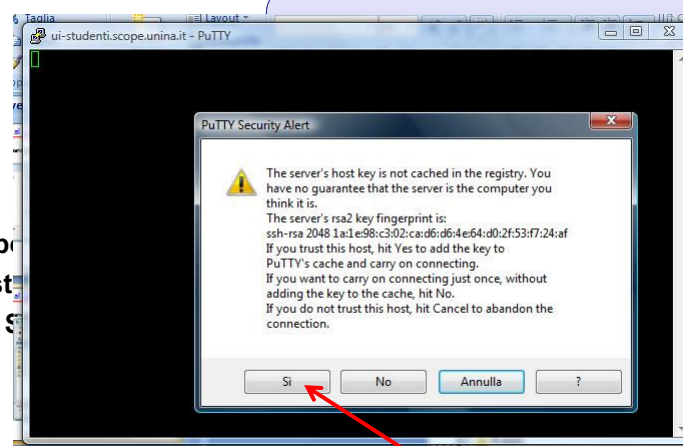
11

Accesso al cluster

E' necessario un client SSH.

Sessione PuTTY

Per sp
il clust
client S



G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

12



ATTENZIONE!!!!
Al primo accesso vi verrà chiesto di
cambiare la password!

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

13

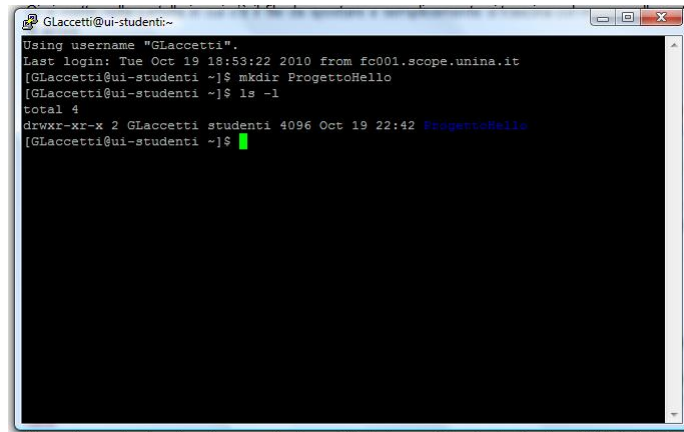
Come lavorare sul cluster

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

14

Scrivere un programma

Nella propria home directory, lo studente crea una cartella per il progetto che sta realizzando, per contenere file sorgente, eseguibile,...



```
GLaccetti@ui-studenti:~  
Using username "GLaccetti".  
Last login: Tue Oct 19 18:53:22 2010 from fc001.scope.unina.it  
[GLaccetti@ui-studenti ~]$ mkdir ProgettoHello  
[GLaccetti@ui-studenti ~]$ ls -l  
total 4  
drwxr-xr-x 2 GLaccetti studenti 4096 Oct 19 22:42 ProgettoHello  
[GLaccetti@ui-studenti ~]$
```

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

15

Scrivere un programma

Quindi, deve scrivere il file sorgente: File C, che si serva della libreria MPI. Utilizzeremo il compilatore dell'implementazione OPENMPI.

Il file sorgente deve essere scritto con un editor Linux (vi, nano,...) oppure con un editor in ambiente Windows che non formatti il testo (es. notepad).

Se è stato scritto in Windows, una volta portato il file sul cluster, va eseguito il comando

dos2unix NomeFileScrittoSuWindows.c

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

16

Scrivere un programma

Quindi, deve scrivere il file sorgente: File C, che si serva della libreria MPI. Utilizzeremo il compilatore dell'implementazione OPENMPI.

Il file sorgente deve essere scritto con un editor Linux (vi, nano,...) oppure con un editor in ambiente Windows che non formatti il testo (es. notepad).

Se è stato scritto in Windows, una volta portato il file sul cluster, va eseguito il comando

dos2unix NomeFileScrittoSuWindows.c

Le applicazioni parallele saranno eseguite sui nodi computazionali esclusivamente in modalità batch tramite il sistema **PBS** (Portable Batch System).

Per compilare ed eseguire il nostro file .c (sottomettere un job) sulle macchine del cluster, si deve creare un file .pbs, cioè un **job-script** che contiene un elenco di direttive.

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

17

Scrivere un programma

Quindi, deve scrivere il file sorgente: File C, che si serva della libreria MPI. Utilizzeremo il compilatore dell'implementazione OPENMPI.

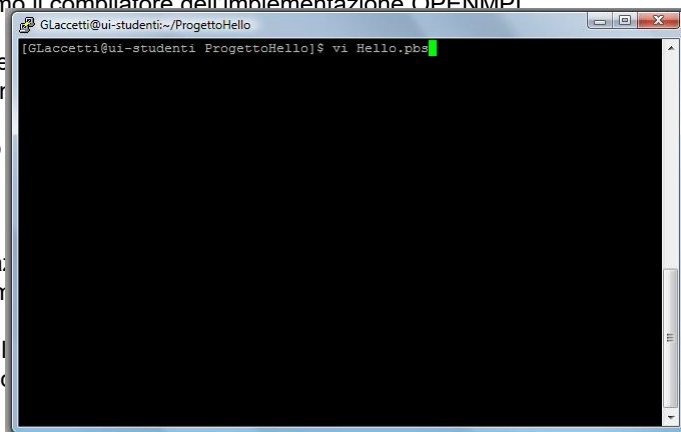
Il file sorgente deve essere scritto con un editor in ambiente Linux (vi, nano,...) oppure con un editor in ambiente Windows che non formatti il testo (es. notepad).

Se è stato scritto in Windows, una volta portato il file sul cluster, va eseguito il comando

dos2unix NomeFileScrittoSuWindows.c

Le applicazioni parallele saranno eseguite sui nodi computazionali esclusivamente in modalità batch tramite il sistema **PBS** (Portable Batch System).

Per compilare ed eseguire il nostro file .c (sottomettere un job) sulle macchine del cluster, si deve creare un file .pbs, cioè un **job-script** che contiene un elenco di direttive.



G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

18

Eseguire un programma: esempio di job-script

```
#!/bin/bash

#####
# #
# The PBS directives #
# #
#####
#PBS -q studenti
#PBS -l nodes=8
#PBS -N hello
#PBS -o hello.out
#PBS -e hello.err
#####
# -q coda su cui va eseguito il job #

# -l numero di nodi richiesti #
# -N nome job(stesso del file pbs) #
# -o, -e nome files contenente l'output #
#####
# #
# qualche informazione sul job #
# #
#####

NCPU=`wc -l < $PBS_NODEFILE`
echo -----
echo ' This job is allocated on '${NCPU}' cpu(s)'
echo 'Job is running on node(s): '
cat $PBS_NODEFILE
```

Job-script che compila
ed esegue Hello.c

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

19

Eseguire un programma: esempio di job-script

```
#!/bin/bash

#####
# #
# The PBS directives #
# #
#####
#PBS -q studenti
#PBS -l nodes=8
#PBS -N Hello
#PBS -o Hello.out
#PBS -e Hello.err
#####
# -q coda su cui va eseguito il job #

# -l numero di nodi richiesti #
# -N nome job(stesso del file pbs) #
# -o, -e nome files contenente l'output #
#####
# #
# qualche informazione sul job #
# #
#####

NCPU=`wc -l < $PBS_NODEFILE`
echo -----
echo ' This job is allocated on '${NCPU}' cpu(s)'
echo 'Job is running on node(s): '
cat $PBS_NODEFILE
```

La prima riga deve
essere sempre questa

Le righe che iniziano con #
sono commenti.
Le righe che iniziano con
#PBS sono direttive.

In rosso le informazioni
tipiche di questa
particolare sottomissione

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

20

Eseguire un programma: esempio di job-script

```
PBS_O_WORKDIR=$PBS_O_HOME/ProgettoHello
echo -----
echo PBS: qsub is running on $PBS_O_HOST
echo PBS: originating queue is $PBS_O_QUEUE
echo PBS: executing queue is $PBS_QUEUE
echo PBS: working directory is $PBS_O_WORKDIR
echo PBS: execution mode is $PBS_ENVIRONMENT
echo PBS: job identifier is $PBS_JOBID
echo PBS: job name is $PBS_JOBNAME
echo PBS: node file is $PBS_NODEFILE
echo PBS: current home directory is $PBS_O_HOME
echo PBS: PATH = $PBS_O_PATH
echo -----
echo "Esegui/usr/lib64/openmpi/1.4-gcc/bin/mpicc -o $PBS_O_WORKDIR/Hello $PBS_O_WORKDIR/Hello.c"
/usr/lib64/openmpi/1.4-gcc/bin/mpicc -o $PBS_O_WORKDIR/Hello $PBS_O_WORKDIR/Hello.c

echo -----
echo "Esegui:/usr/lib64/openmpi/1.4-gcc/bin/-machinefile $PBS_NODEFILE -np $NCPU $PBS_O_WORKDIR/Hello"
/usr/lib64/openmpi/1.4-gcc/bin/mpirun -machinefile $PBS_NODEFILE -np $NCPU $PBS_O_WORKDIR/Hello
```

Stampa informazioni

Rivedremo lo script
nella prossima
lezione!

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

Eseguire un programma

Una volta che si è scritto il file PBS

```
Glacetti@ui-student-0-ProgettoHello
```

```
#  
# Qualcune informazioni sul job  
#  
#####  
  
NCPUF= wc -l < $PBS_NODEFILE  
echo -----  
echo " This job is allocated on '$(NCPUF)' CPUs"  
echo " Job is running on node(s) : "  
cat $PBS_NODEFILE  
  
PBS_O_WORKDIR=$PBS_O_HOME/HELLO  
echo -----  
echo PBS: qsub is running on $PBS_O_HOST  
echo PBS: originating queue is $PBS_O_QUEUE  
echo PBS: executing queue is $PBS_O_QUEUE  
echo PBS: working directory is $PBS_O_WORKDIR  
echo PBS: execution mode is $PBS_ENVIRONMENT  
echo PBS: job identifier is $PBS_JOBID  
echo PBS: job name is $PBS_JOBNAME  
echo PBS: node file is $PBS_NODEFILE  
echo PBS: current home directory is $PBS_O_HOME  
echo PBS: PATH = $PBS_O_PATH  
echo -----  
echo "Eseguito /usr/lib64/openmpi/1.7.7-gcc/bin/mpicc -o $PBS_O_WORKDIR/hello $PBS_O_WORKDIR/Hello.c"  
/usr/lib64/openmpi/1.7.7-gcc/bin/mpicc -o $PBS_O_WORKDIR/hello $PBS_O_WORKDIR/Hello.c  
  
echo "Eseguito /usr/lib64/openmpi/1.7.7-gcc/bin/mpixecx -machinefile $PBS_MACHINEFILE & -np $NCPUF $PBS_O_WORKDIR/Hello"  
/usr/lib64/openmpi/1.7.7-gcc/bin/mpixecx -machinefile $PBS_MACHINEFILE .np $NCPUF $PBS_O_WORKDIR/Hello  
  
%wG
```

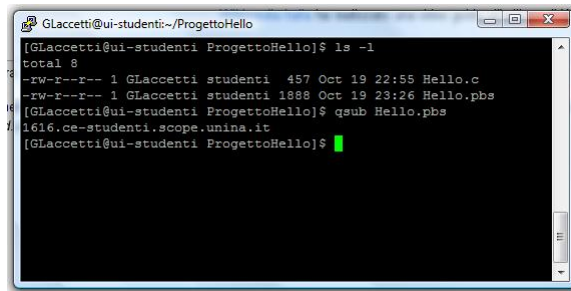
G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

Eeguire un programma

Una volta che si è scritto il file PBS, e si è quindi preparata l'esecuzione parallela, si lancia con il comando

qsub Hello.pbs

qsub è in grado di interpretare le direttive contenute nel PBS



```
GLaccetti@ui-studenti:~/ProgettoHello$ ls -l
total 8
-rw-r--r-- 1 GLaccetti studenti 457 Oct 19 22:55 Hello.c
-rw-r--r-- 1 GLaccetti studenti 1888 Oct 19 23:26 Hello.pbs
[GLaccetti@ui-studenti:~/ProgettoHello]$ qsub Hello.pbs
1616.ce-studenti.scope.unina.it
[GLaccetti@ui-studenti:~/ProgettoHello]$
```

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

23

Eeguire un programma

Una volta che si è scritto il file PBS, e si è quindi preparata l'esecuzione parallela, si lancia con il comando

qsub Hello.pbs

qsub è in grado di interpretare le direttive contenute nel PBS

bisognerà aspettare che il job venga gestito dal sistema (è in coda con altri job) e che termini la sua esecuzione.

A questo punto sarà possibile visualizzare l'output e l'error con i comandi:

cat Hello.err

cat Hello.out

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

24

Eseguire un programma

Per visualizzare i job:

qstat

Per visualizzare la coda ed il loro stato

(E=eseguito,R=in esecuzione, Q= è stato accodato):

qstat -q

Per eliminare un job dalla coda:

qdel jobid

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

25

Disponibilità del cluster

L'accesso al cluster vi sarà possibile solo nei seguenti orari:

?

L'indirizzo a cui inoltrare qualsiasi segnalazione riguardo il cluster, che vi invito a fare REPENTINAMENTE qualora riscontraste qualcosa che non va.

L'indirizzo è: contactcenter@unina.it (cc valeria.mele@unina.it)
scrivete dettagliando bene il problema e i tempi in cui si è verificato .

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

26

Prime credenziali sul cluster

Le login per accedere al cluster sono contenute nel materiale del corso nel file **PDC_Login_Cluster.pdf** nella colonna LOGIN

Attualmente, per ciascuno degli account, la password coincide con il CODICE FISCALE comunicato.

Al primo accesso al sistema verrà richiesto di modificarla.

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

27

Accesso al cluster



ATTENZIONE!!!!
Al primo accesso vi verrà chiesto di
cambiare la password!

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24

28

FINE LEZIONE

G. Laccetti - Parallel and Distributed Computing - a.a. 2023/24