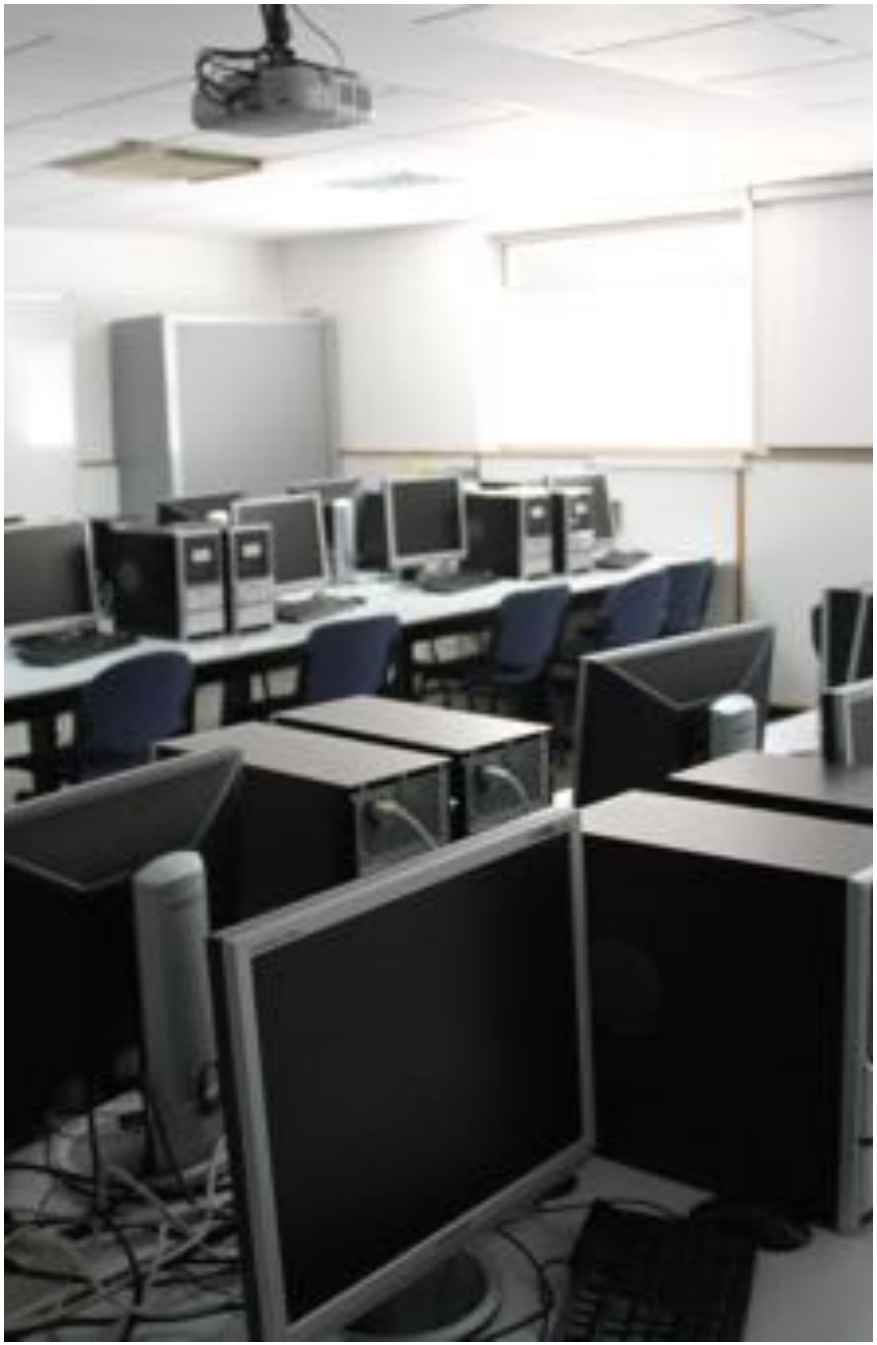
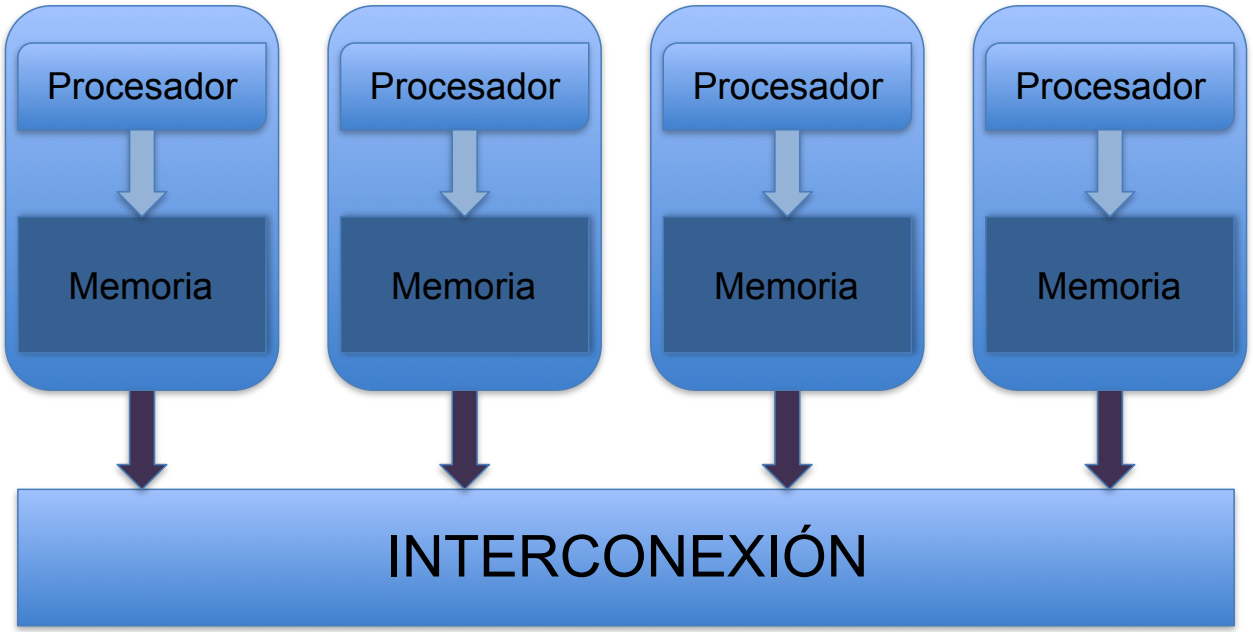
Desarrollo de Software en Arquitecturas Paralelas

Prácticas

Cluster
Red Ethernet

L01



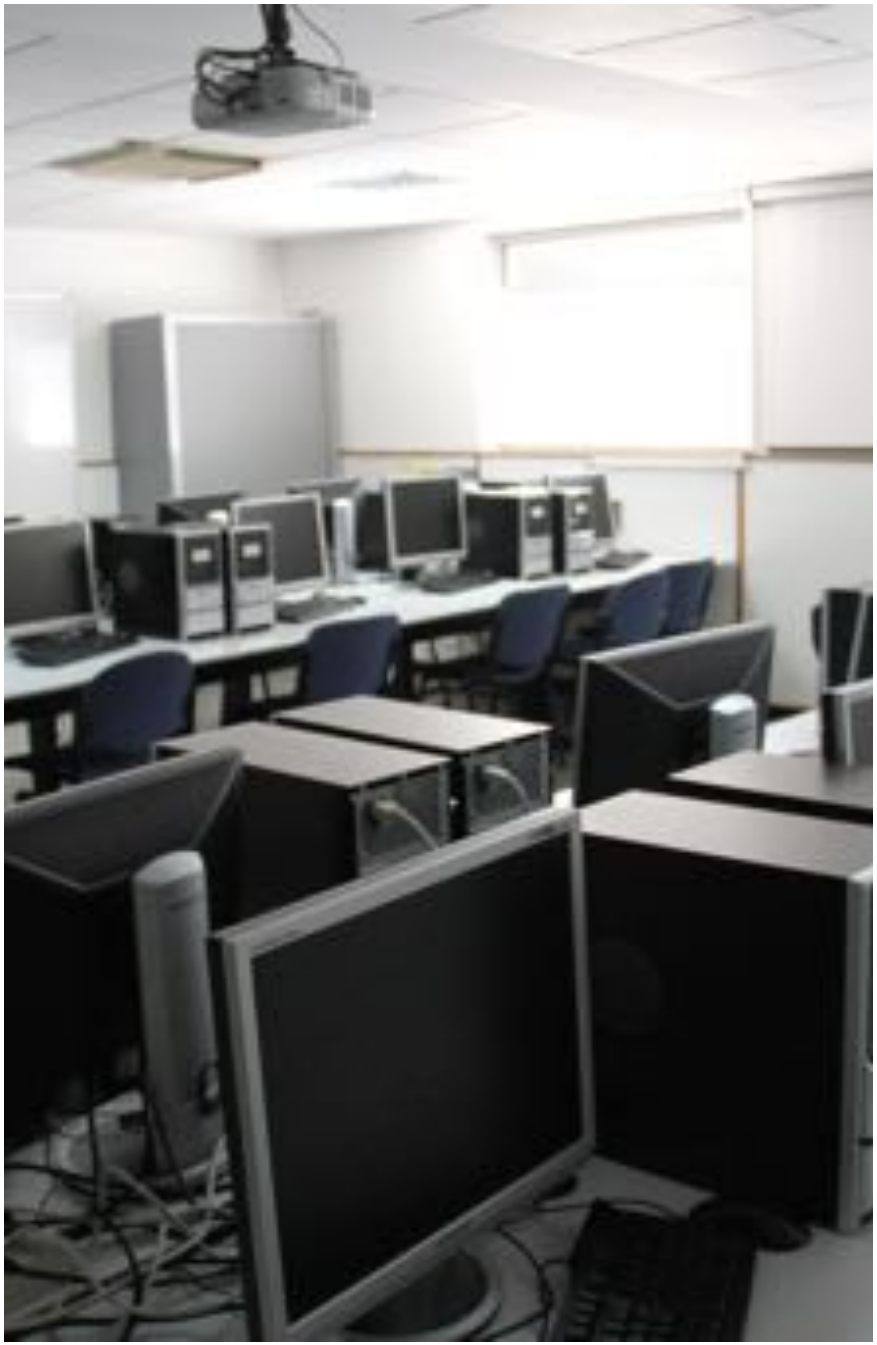


Sistema de memoria distribuida

Interconexión: Red ethernet

Intel Pentium CPU G3220 3GHz

24 machines
# of Cores 2



L01

- Todo el alumnado debe disponer de una cuenta propia que servirá para todas las prácticas que ha de realizar. El nombre de la cuenta se corresponde con el **usuario EPS** y el **mismo password**.

- Al entrar un usuario, detecta si es de la asignatura DSAP o no lo es.

- Todos los alumnos comparten el mismo /home, pero cada uno tiene su propio espacio, en el mismo, pero independiente de los demás.

- Al cerrar sesión (o reiniciar o apagar), **no se borra nada** del directorio $HOME del servidor (**sólo en L01 y algunas máquinas del laboratorio libre**).

MPICH Release 3.2

- mpich-3.2-6-x86_64
- openssh

MPICH Release 3.2 mpichversion

- arnal@clL01-1:~$ mpichversion
  MPICH Version: 3.2
  MPICH Release date: Wed Nov 11 22:06:48 CST 2015
  MPICH Device: ch3:nemesis
  MPICH configure: --build=x86_64-linux-gnu --prefix=/usr --includedir=$
  {prefix}/include --mandir=${
  prefix}/share/man --infodir=${prefix}/share/info --sysconfdir=/etc --localstatedir=/
  var --disable-silent-
  rules --libdir=${prefix}/lib/x86_64-linux-gnu --libexecdir=${prefix}/lib/x86_64-
  linux-gnu --disable-maint
  ainer-mode --disable-dependency-tracking --enable-shared --prefix=/usr --enable-
  fortran=all --disable-rpa
  th --disable-wrapper-rpath --sysconfdir=/etc/mpich --libdir=/usr/lib/x86_64-linux-
  gnu --includedir=/usr/i
  nclude/mpich --docdir=/usr/share/doc/mpich --with-hwloc-prefix=system --enable-
  checkpointing --with-hydra
  -ckpointlib=blcr CPPFLAGS= CFLAGS= CXXFLAGS= FFLAGS= FCFLAGS=
  MPICH CC: gcc -g -O2 -fstack-protector-strong -Wformat -Werror=format-
  security -O2
  MPICH CXX: g++ -g -O2 -fstack-protector-strong -Wformat -Werror=format-
  security -O2
  MPICH F77:   gfortran -g -O2 -fstack-protector-strong -O2
  MPICH FC: gfortran -g -O2 -fstack-protector-strong -O2

MPICH Release 3.2
Configuración de ssh sin contraseña para la comunicación entre nodos

- Generar un par de llaves RSA para el usuario:

arnal@clL01-1:~$ ssh-keygen -t rsa

- Añadir esta llave a las llaves autorizadas:

arnal@clL01-1:~$ cd .ssh

arnal@clL01-1:~/.ssh$ cat id_rsa.pub >> authorized_keys

mpi_hola_mundo.c

```c
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
    // Inicializamos el entorno MPI
    MPI_Init(NULL, NULL);

    // Obtenemos el numero de procesos
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);

    // Obtenemos el rango de los procesos
    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

    // Obtenemos el nombre del procesador
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);

    // Imprimimos un mensaje de hola mundo
    printf("Hola mundo desde el procesador %s, rango %d"
           " de %d procesos\n",
           processor_name, world_rank, world_size);

    // Finalizamos el entorno MPI.
    MPI_Finalize();
}
```

Compilación

- La compilación es similar a la compilación con gcc. Se pueden usar los mismos parámetros.
    - Compilar un programa con MPI en C:

```
mpicc -o mpi_hola_mundo mpi_hola_mundo.c
```

    - Compilar un programa con MPI en C++:
        - Se puede utilizar mpiCC o mpicxx

Compilación

```
mpicc -o mpi_hola_mundo mpi_hola_mundo.c
```

Ejecución

Ejecución de un trabajo de n procesos en tu maquina **local**

```
mpiexec -np 4 mpi_hola_mundo
```

ó

```
mpirun -np 4 mpi_hola_mundo
```

```
Hola mundo desde el procesador clL01-1, rango 0 de 4 procesos
Hola mundo desde el procesador clL01-1, rango 1 de 4 procesos
Hola mundo desde el procesador clL01-1, rango 2 de 4 procesos
Hola mundo desde el procesador clL01-1, rango 3 de 4 procesos
```

Ejecución

Ejecución de un trabajo de n procesos en **varios nodos**

```
mpiexec -machinefile machinefile -np 4 mpi_hola_mundo
```

```
Hola mundo desde el procesador clL01-1, rango 0 de 4 procesos
Hola mundo desde el procesador clL01-2, rango 1 de 4 procesos
Hola mundo desde el procesador clL01-3, rango 2 de 4 procesos
Hola mundo desde el procesador clL01-4, rango 3 de 4 procesos
```

Ejecución

- El fichero “**machinefile**” es de la forma

```
clL01-1

clL01-2

clL01-3

clL01-4

clL01-5

clL01-6
```

- Donde “`clL01-1`”, “`clL01-2`”, “`clL01-3`”, “`clL01-4`”, “`clL01-5`”, “`clL01-6`” son los nombres de las máquinas donde se quiere ejecutar el programa.

- En el laboratorio L01, los nombres de las máquinas son: “clL01-1”, “clL01-2”, …,“clL01-24”.

mpi_hello_world.c

```c
#include <mpi.h>
#include <stdio.h>
#include <utmpx.h>

int main(int argc, char** argv) {
    int sched_getcpu();
    // Initialize the MPI environment
    MPI_Init(NULL, NULL);

    // Get the number of processes
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);

    // Get the rank of the process
    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

    // Get the name of the processor
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);

    // Print off a hello world message
    printf("Hello world from core %d  of processor %s, rank %d"
           " out of %d processes\n",
           sched_getcpu(), processor_name, world_rank, world_size);
    // Finalize the MPI environment.
    MPI_Finalize();
}
```

Si queremos ver el core sobre el que está ejecutándose el proceso en linux se dispone de **sched_getcpu().**
Dicha opción no esta disponible en mac os.

Ejecución

`mpiexec -machinefile machinefile -np 2 mpi_hola_mundo`

Al ejecutar en dos procesos, aunque se utilice la opción

-machinefile

mpi trabajará en local, es decir lanzará los dos procesos en la misma maquina pues se trata de un multicore. Si queremos utilizar menos procesos por nodo podemos utilizar la opción

-npernode

Por ejemplo si queremos ejecutar dos procesos, cada uno en una máquina distinta:

mpirun -machinefile machinefile -np 2 -npernode 1 ejecutable

Ejecución

mpiexec -machinefile machinefile -np 3 **-npernode 1** mpi_hello_world

Hello world from core 0 of processor **clL01-3**, rank 2 out of 3 processes

Hello world from core 1 of processor **clL01-1**, rank 0 out of 3 processes

Hello world from core 0 of processor **clL01-2**, rank 1 out of 3 processes

mpiexec -machinefile machinefile -np 3 mpi_hello_world

Hello world from core 0 of processor clL01-1, rank 1 out of 3 processes

Hello world from core 1 of processor clL01-1, rank 0 out of 3 processes

Hello world from core 0 of processor clL01-2, rank 2 out of 3 processes

Para comprobar el número de cores, en linux, del procesador:

```
usuario@clL01-3:~$ nproc --all

2

usuario@clL01-3:~$ lscpu | grep 'CPU(s)'

CPU(s):                2

On-line CPU(s) list:   0,1

NUMA node0 CPU(s):     0,1
```

Ejecución

mpiexec -host clL01-1,clL01-2 -np 2 mpi_hello_world

Hello world from core 0 of processor clL01-2, rank 1 out of 2 processes
Hello world from core 0 of processor clL01-1, rank 0 out of 2 processes

mpiexec -np 2 mpi_hello_world

Hello world from core 0 of processor clL01-3, rank 0 out of 2 processes
Hello world from core 1 of processor clL01-3, rank 1 out of 2 processes