Aspectos avanzados en el uso del clúster UO

Dando poder a los investigadores

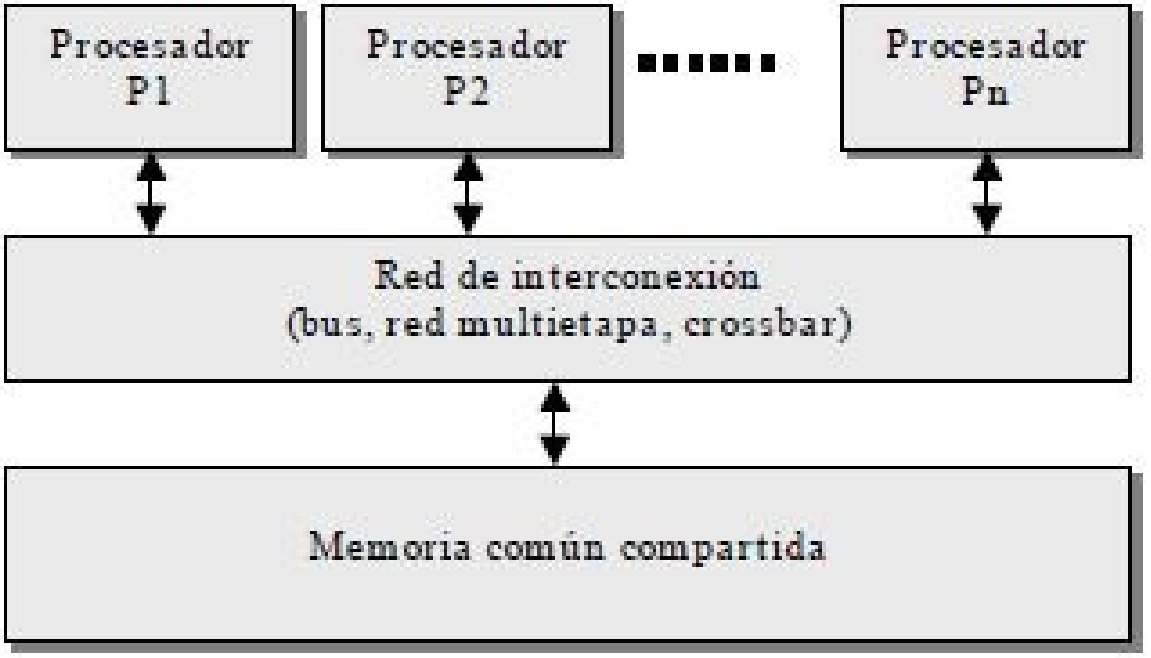
Computación paralela

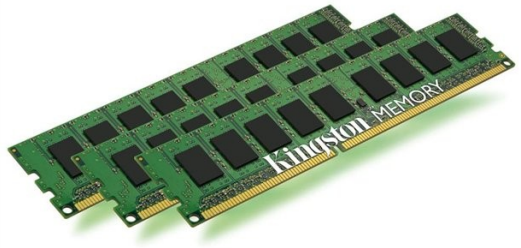
- Antecedentes: una computadora desktop con 1 procesador → CONCURRENCIA
- Una computadora desktop con más de un procesador (2, 4, 8 o 16 procesadores).
- Una computadora masivamente paralela (cientos o miles de procesadores).
- Un clúster de computadoras escalable (cientos o miles de procesadores conectados mediante una red).

Esquemas de memoria para el trabajo en paralelo

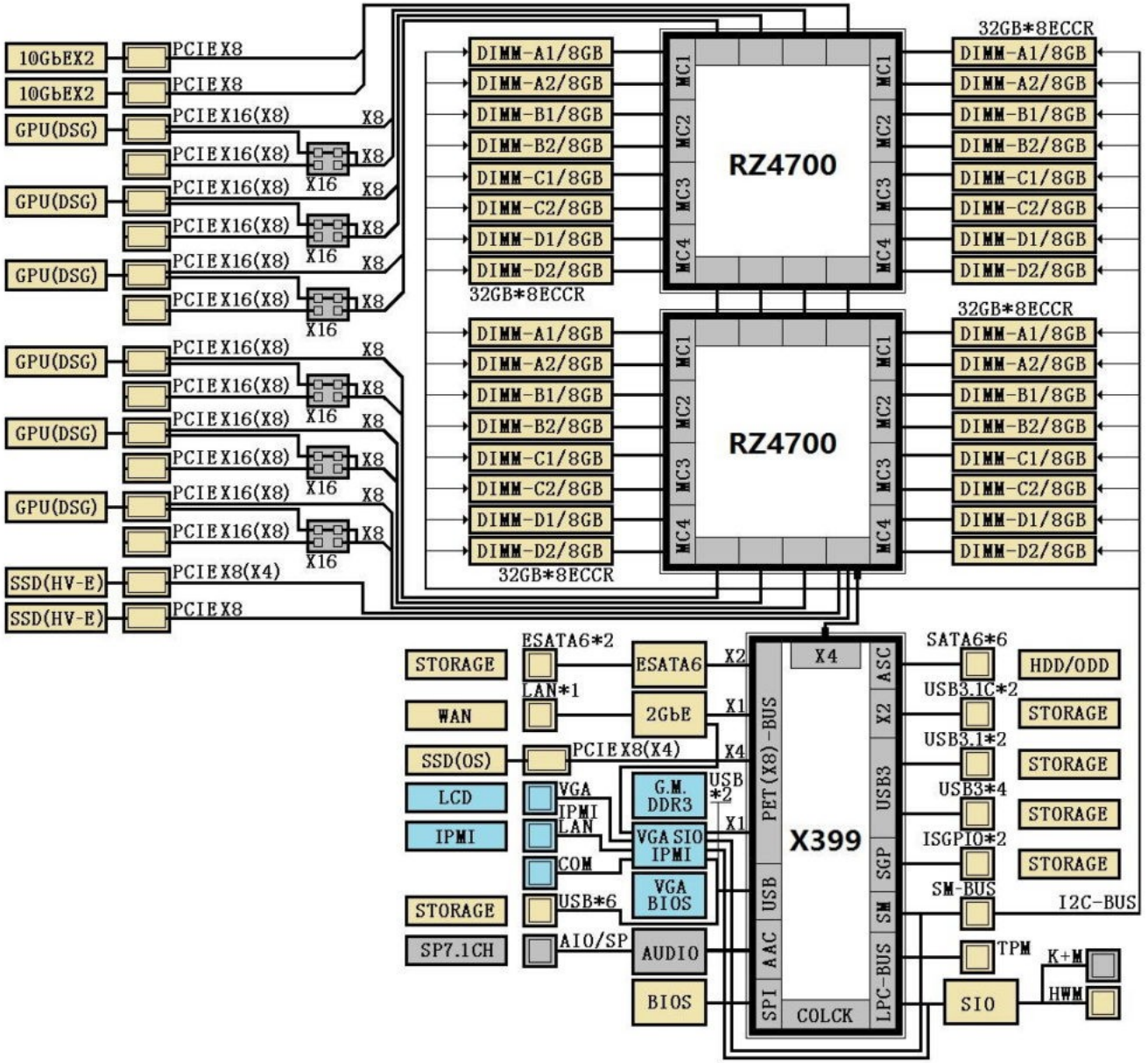
Desktop con varios cores

- Memoria compartida

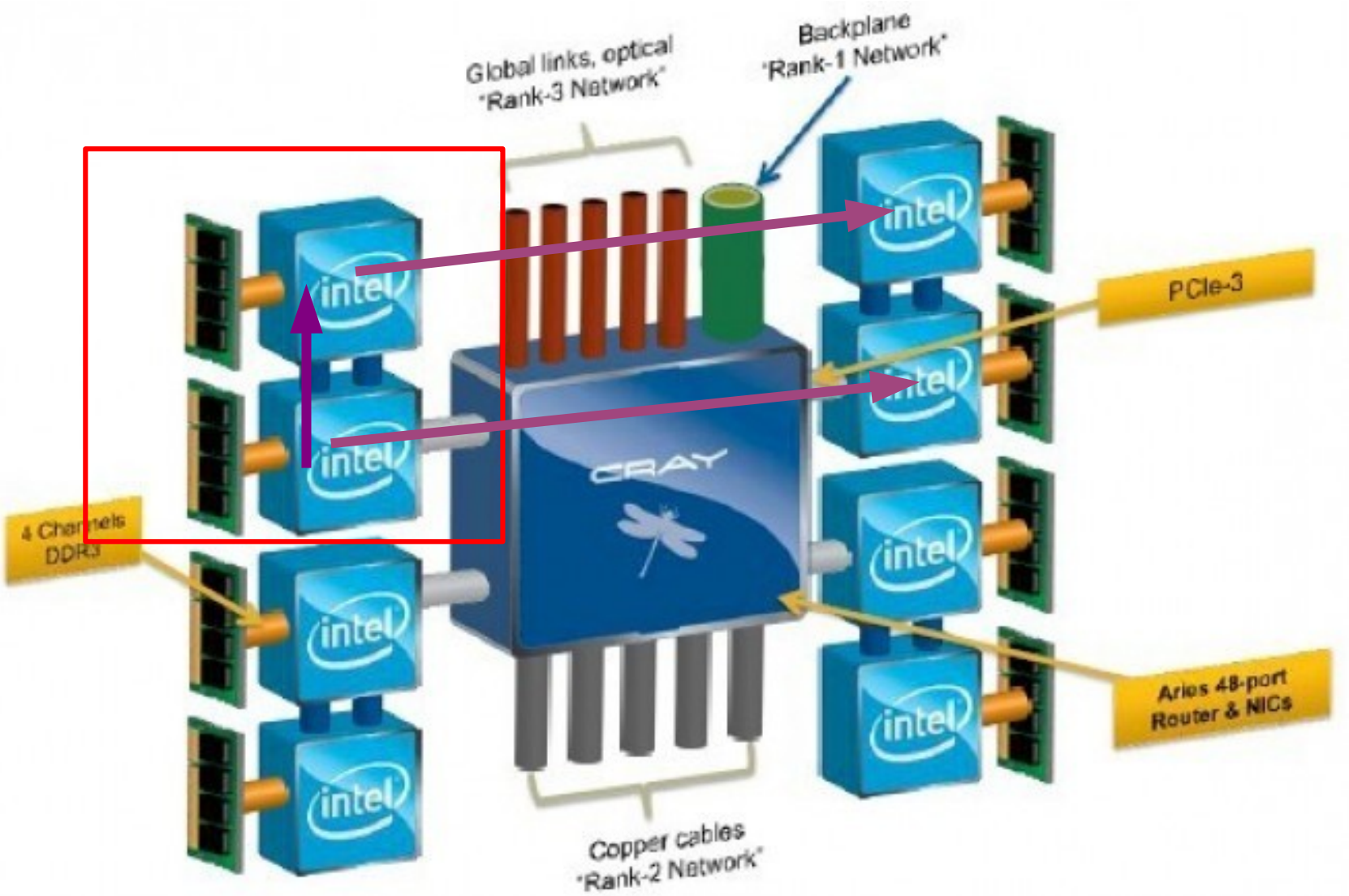




AMD-X399-Chipset-1000...



Multicomputadora

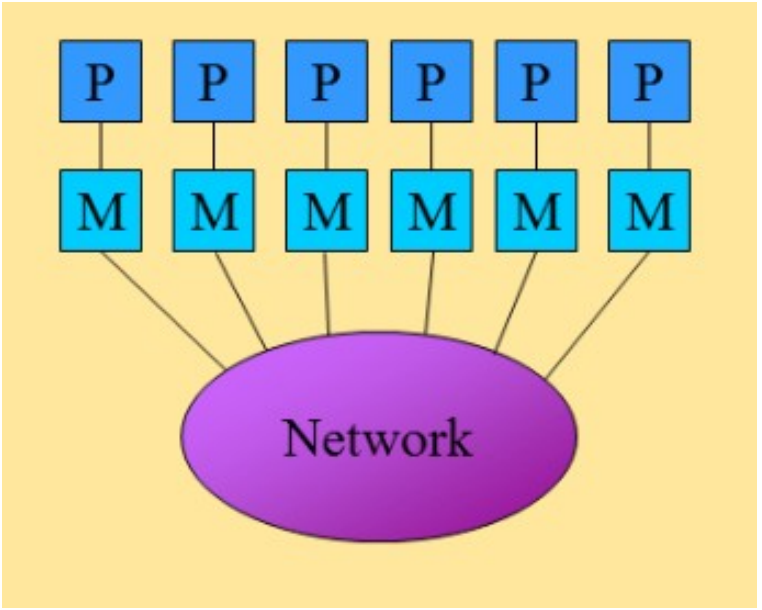


Clústers de computadoras

- Beowulf

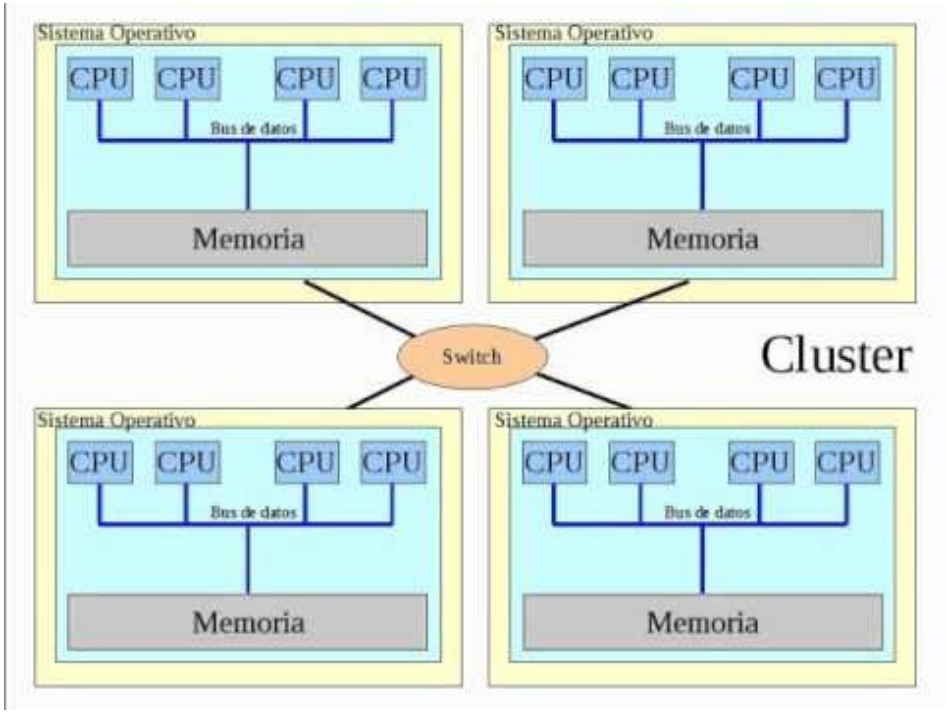
- Especializado

Memoria distribuida



- Cada unidad del clúster tiene su propia memoria, controlada por su procesador.
- La comunicación entre procesadores se realiza mediante mensajes a través de la red.

Memoria compartida y distribuida



- A nivel de nodo está compartida la memoria entre los procesadores locales.
- A nivel de clúster está distribuida y se accede mediante mensajes.

Compartida/distribuida

Clúster de la Universidad de Oriente (HPC-UO)

http://portal.uo.hpc.cu

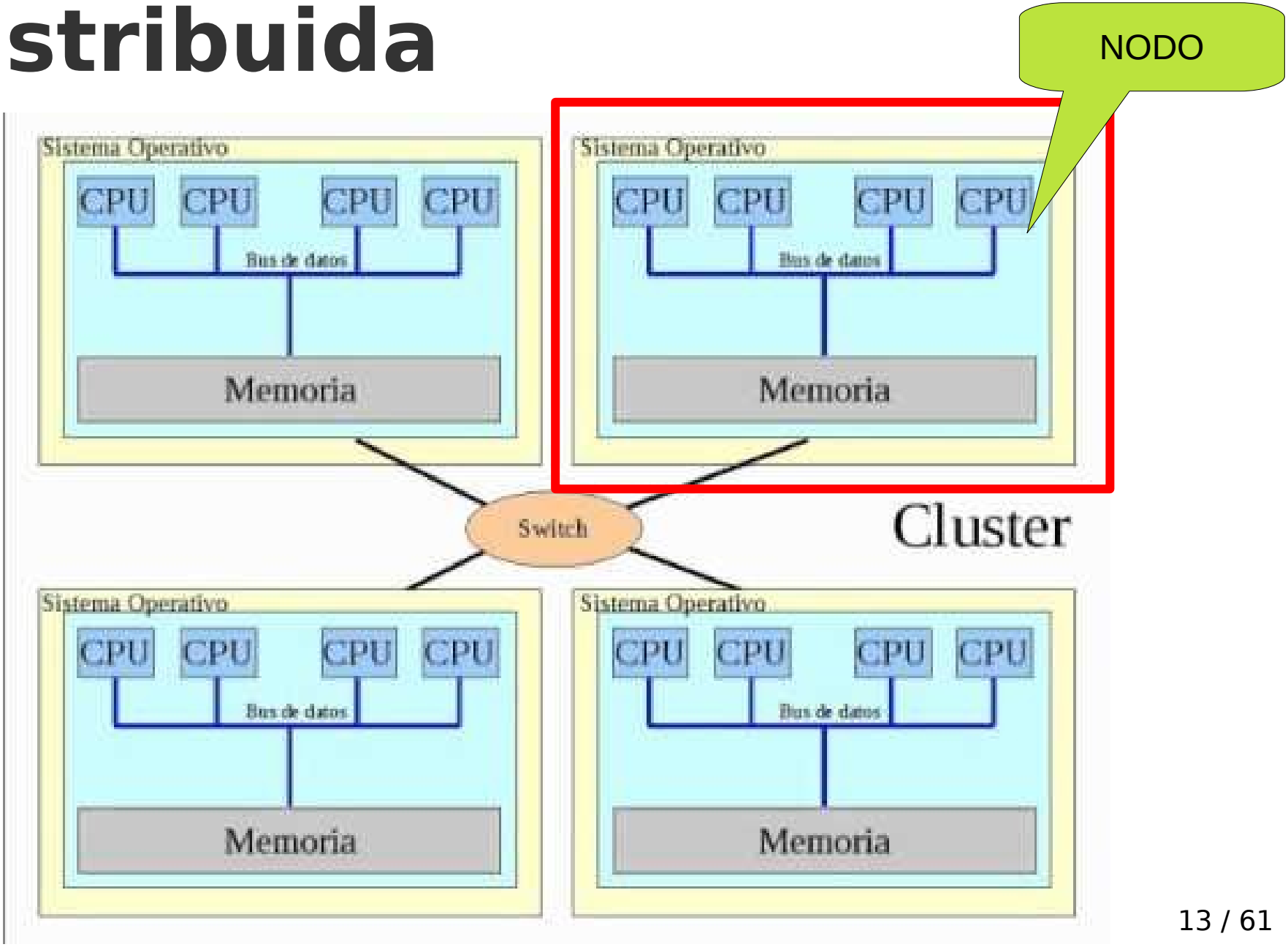
ssh://login.uo.hpc.cu

HPC-UO

Sistema de supercómputo de la Universidad de Oriente

Dando poder a los investigadores

Memoria compartida y distribuida



Capacidad de un nodo

- 32 cores x 2.4 GHz
- 64 GB RAM

- Red Ethernet (1 Gb)
- Red Infiniband (40 Gb)

HPC

Para el servicio de HPC se utilizan 13 nodos para el procesamiento CPU con una arquitectura de Memoria Distribuida y 4 procesamiento CPU con una arquitectura de Memoria Compartida.

Equipo	Modelo	Cantidad	Procesador	Cores	RAM	Conexion
Memoria Compartida	SUN-FIRE 4470 X	4	4 x Intel Xeon X7550 (Gainestown or **Nehalem-EP)**	64	64 GB	2 Ethernet GB
	DELL R720	1	1 x Intel Xeon E5-2609 (Sandy Bridge-EP)	4	16 GB	2 Ethernet GB
Memoria Distribuida	DELL Power Edge C6145	13	4 x AMD Opteron 6136	32	64 GB	1 Ethernet GB 40 InfiniBand GB

El rendimiento teórico del sistema es de 2,1 TFLOPs de operaciones en el cluster de Memoria Distribuida y 3,99 TFLOPs de operaciones en el cluster de Memoria Compartida

Big Data

Nodos	modelo	Cantidad	Procesador	Cores	RAM	Ethernet
DELL Power Edge C6145		1	4 x AMD Opteron 6136	32	64 GB	1 Ethernet GB
		1	2 x AMD Opteron 6136	16	64 GB	1 Ethernet GB

El rendimiento teórico del sistema es de 0,45 TFLOPs de operaciones en CPU

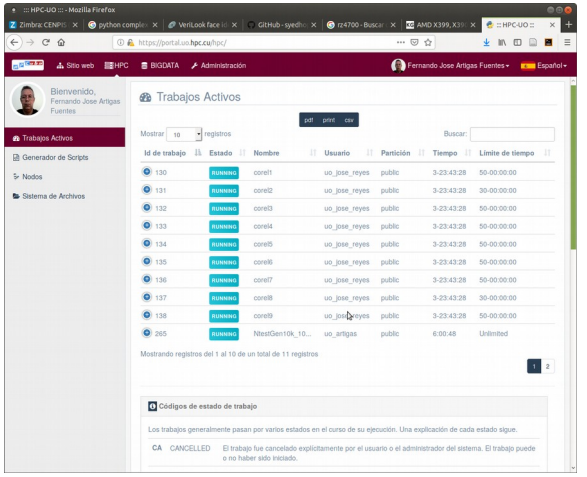
Capacidad total

- 13 nodos

- 416 cores
- 832 GB de RAM
- Almacenamiento disponible para usuarios: 3T

Acceso de los usuarios

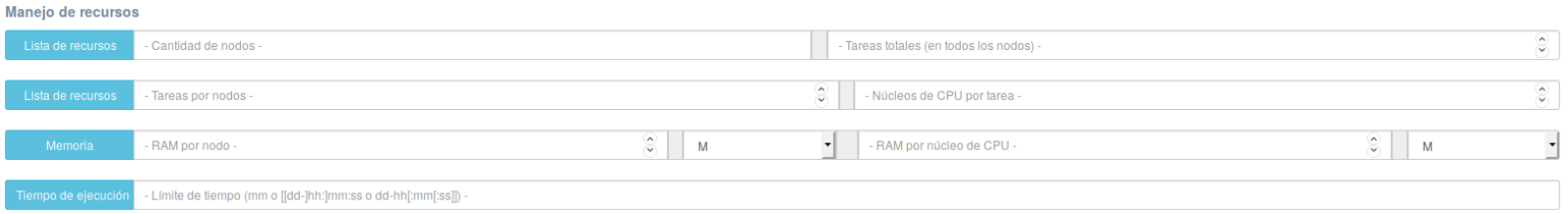
- A través de una conexión ssh + scp (o winscp) para mover información.
- A través de la interfaz web
  - Todo en un único sitio.



- + Scripts, o guiones, que describen los trabajos.

Parámetros del script

- Cantidad de nodos
- Cantidad de memoria por nodo
- Cantidad de cores por nodo
- Tareas por nodo
- Programa a ejecutar, datos, módulos, etc.



Relación entre el esquema de memoria usado por los programas de usuario y la reserva de recursos en el clúster

Procedencia de programas

- Programas de terceros
  - Programas secuenciales, en su mayoría.
  - Programas paralelos para memoria compartida, especializados.
  - ~~Programas paralelos para GPUs.~~
  - Programas paralelos para memoria distribuida, mediante MPI.

AVERIGUAR!!!!!!

Procedencia de programas

- Programas propios, desarrollados por el usuario.
- Programas encargados, desarrollados por programadores del entorno.

- Fácil de averiguar!!!!!!

Mirar la documentación

- ¿Memoria compartida o distribuida?

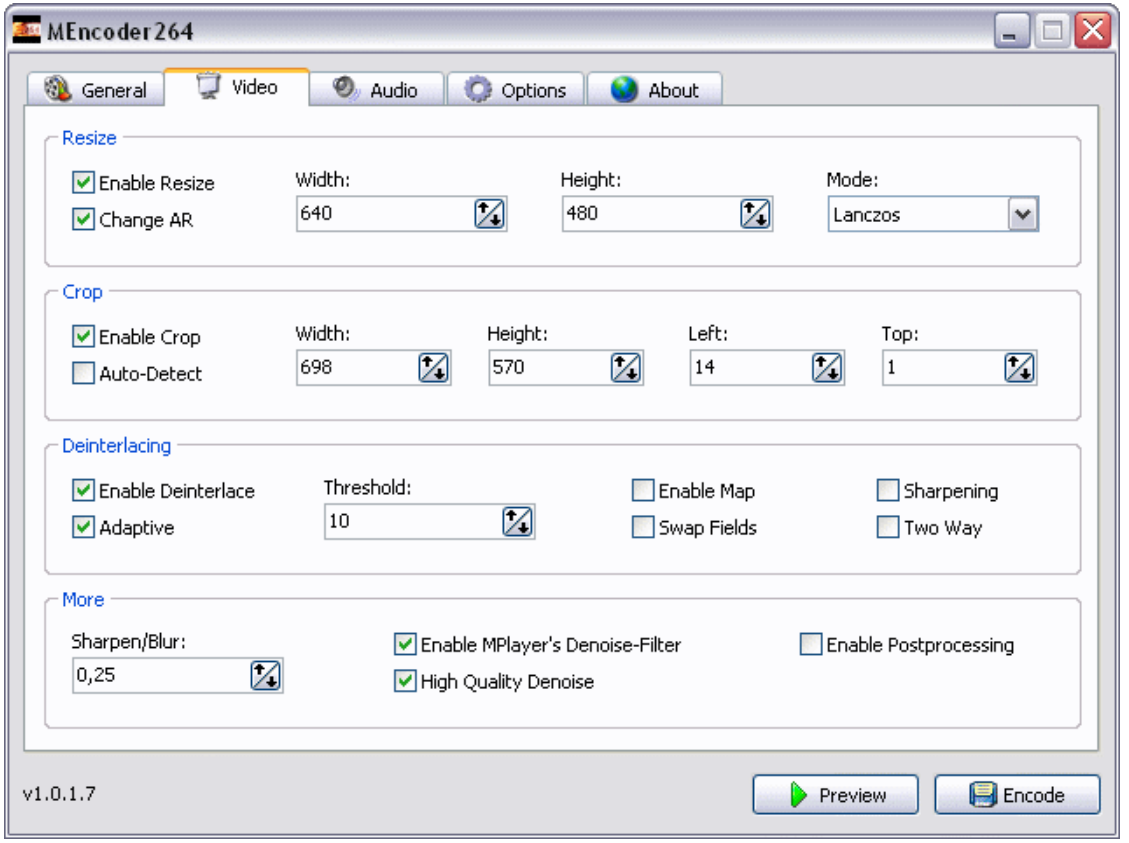
- OpenMP → Memoria compartida (1 nodo)
- OpenMPI → Memoria distribuida (1 o más nodos)

Mirar la documentación

- Requerimientos de memoria
  - 1 GB
  - 2 GB
  - 4 GB

  - ¿No sabe cuál es? Obtener un estimado...

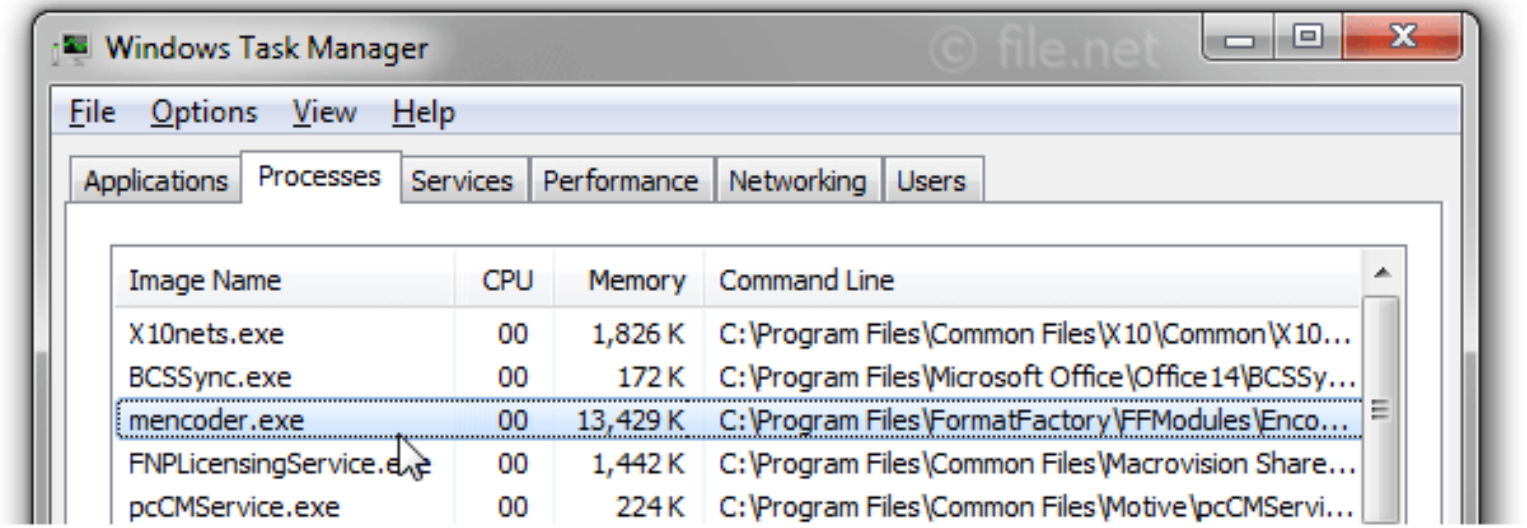
Caso de estudio I
Un programa secuencial
Mencoder



MENCODER

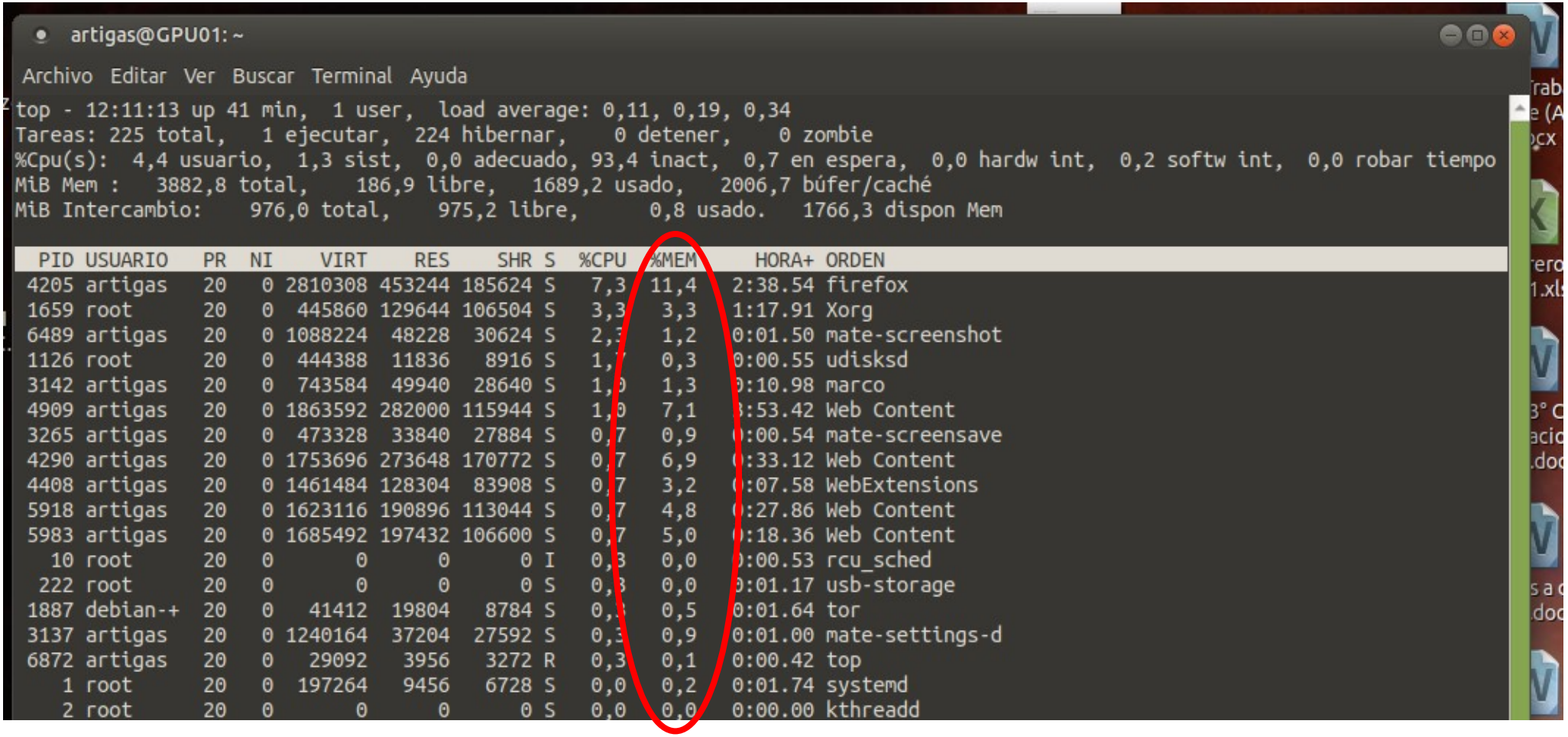
- Programa para convertir de formato ficheros de videos.
- mencoder <fichero de entrada> -ovc xvid -oac mp3lame -oac mp3lame -lameopts abr:br=192 -xvidencopts pass=2:bitrate=-700000 -o <fichero de salida>

Memoria...



- 13.5 MB de memoria RAM
- Es poco, puede ejecutarse con otros programas sobre el mismo nodo → No reservar el nodo en exclusivo.

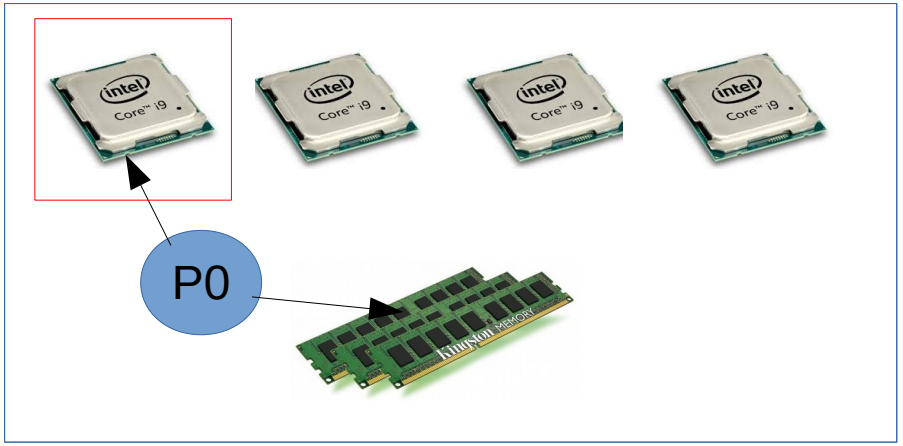
Memoria sobre Linux

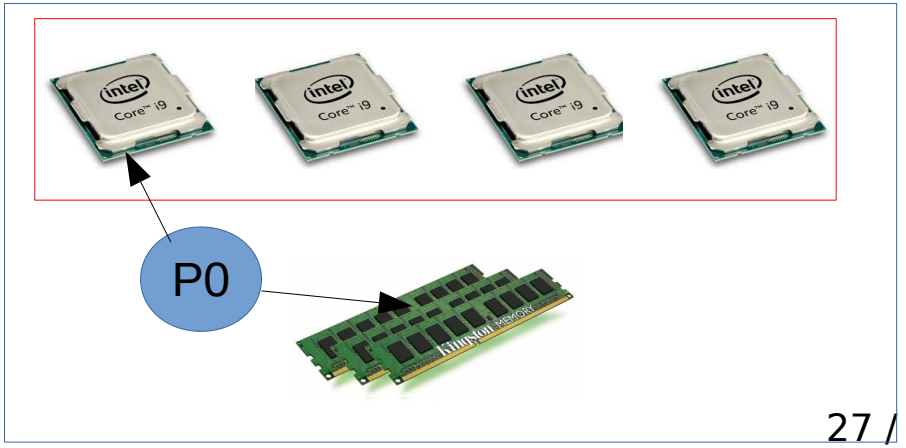


Requerimientos sobre HPC

- Programa secuencial → 1 nodo
- CORES → 1 core

- ¿Por qué no sirve de nada reservar más de un nodo o más de un core en este caso?

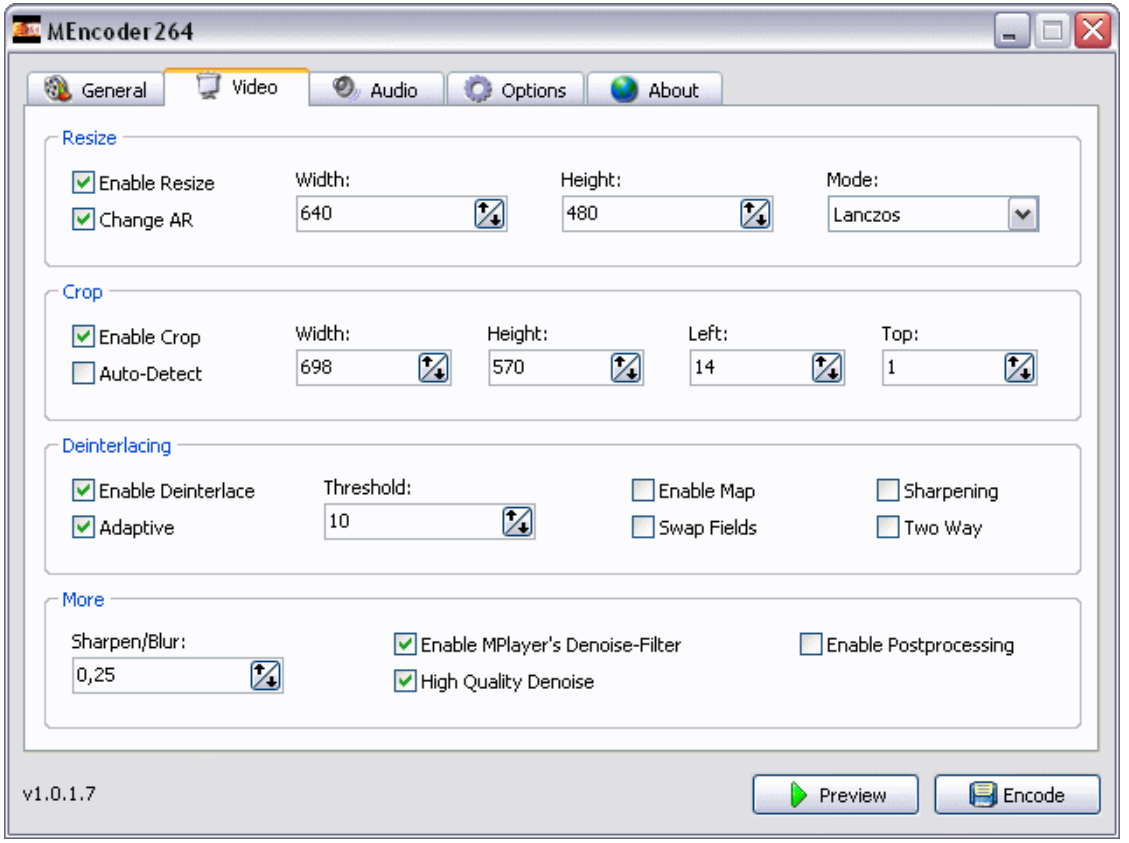




Script ejemplo (I)

```
#!/bin/bash

#SBATCH -p public # partition (queue)

#SBATCH --nodes=1

#SBATCH --ntasks-per-node=1

#SBATCH -o mencoder_%j.out # STDOUT

#SBATCH -e mencoder_%j.err # STDERR

mencoder entrada.mkv -ovc xvid -oac mp3lame -oac
mp3lame -lameopts abr:br=192 -xvidencopts
pass=2:bitrate=-700000 -o salida.avi
```

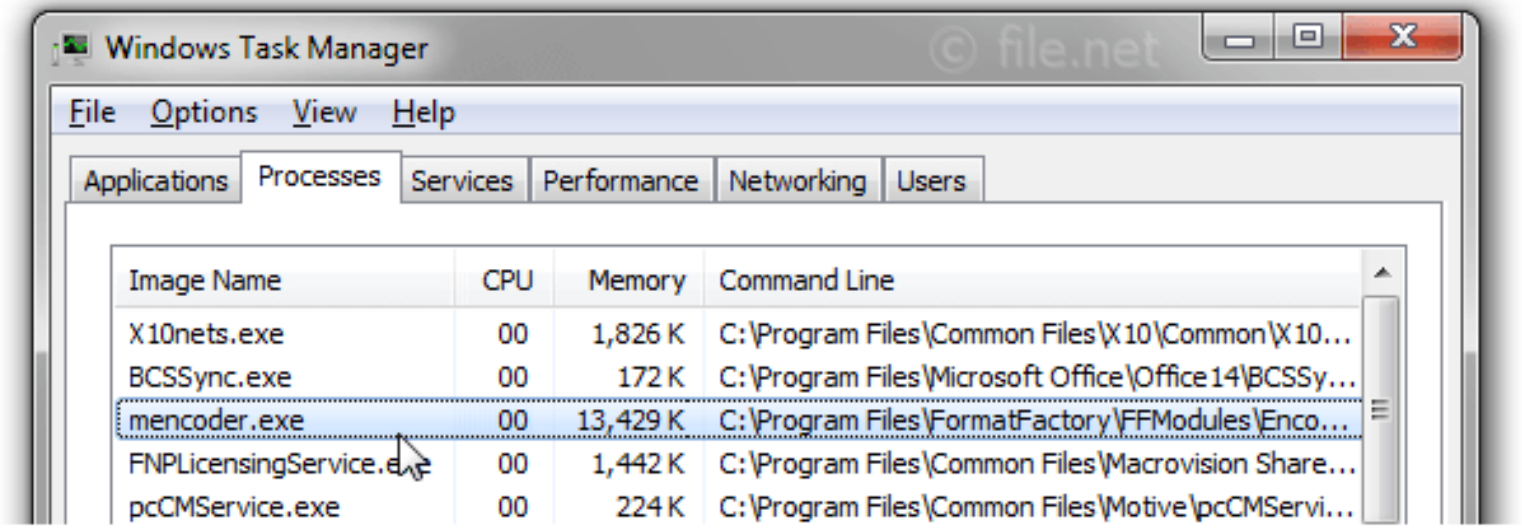
Caso de estudio II
Un programa para memoria compartida
MEMCODER



MENCODER

- Programa para convertir de formato ficheros de videos.
- mencoder <fichero de entrada> -ovc xvid -oac mp3lame -oac mp3lame -lameopts abr:br=192 -xvidencopts pass=2:bitrate=-700000:threads=4 -o <fichero de salida>

Memoria...



- 13.5 MB de memoria RAM
- Es poco, puede ejecutarse con otros programas sobre el mismo nodo → No reservar el nodo en exclusivo.

Requerimientos sobre HPC

- Memoria compartida → 1 nodo
- CORES (hasta 32 disponibles) → 1 hasta 32
  - Para procesar un fichero de entrada con 4 hilos va bien, entonces cores = 4, .
  - Para procesar MUCHOS ficheros de entrada, entonces cores = 32. ¿Reservar el nodo completo?

Script ejemplo (I)

```
#!/bin/bash
#SBATCH -p public # partition (queue)
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=4
#SBATCH -o mencoder_%j.out # STDOUT
#SBATCH -e mencoder_%j.err # STDERR
mencoder entrada.mkv -ovc xvid -oac mp3lame -oac
mp3lame -lameopts abr:br=192 -xvidencopts
pass=2:bitrate=-700000:threads=4 -o salida.avi
```

Siempre = 1

Entre 1 y 32

Debe coincidir con ntasks-per-node

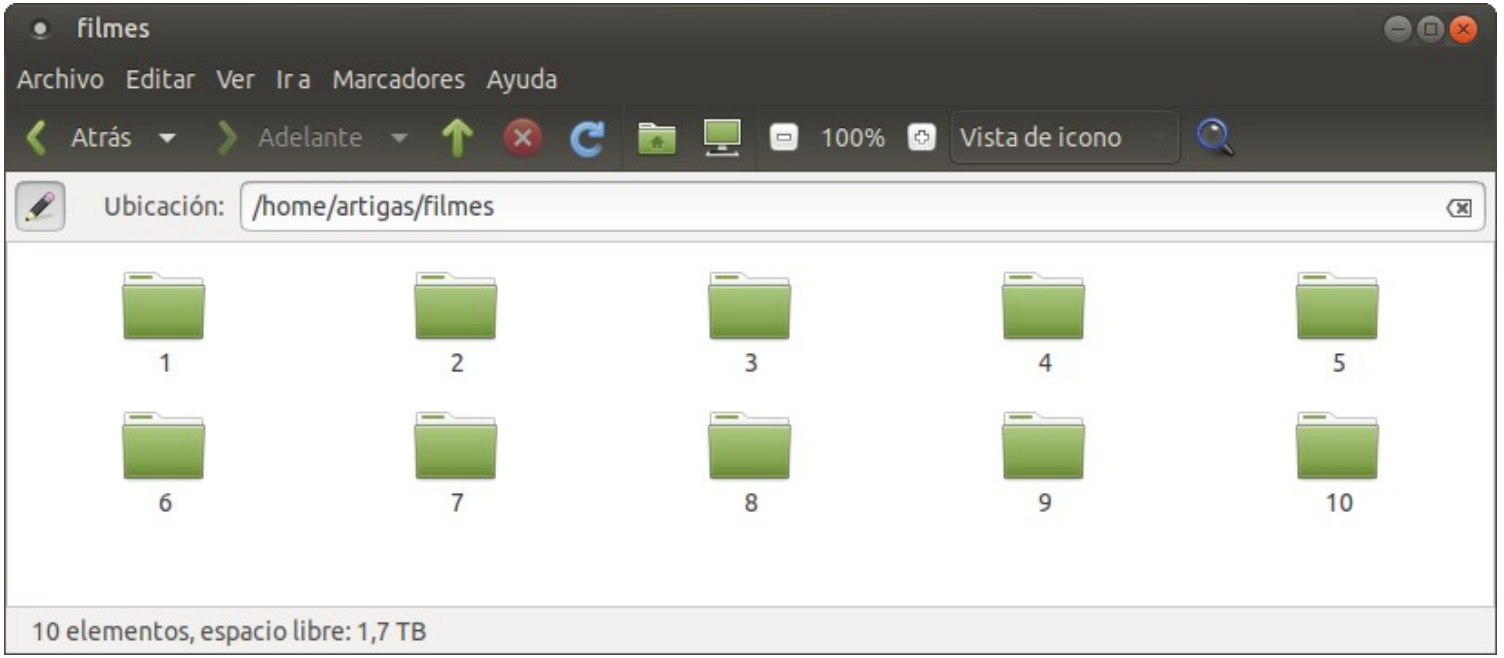
Script ejemplo (II)

```
#!/bin/bash
#SBATCH -p public # partition (queue)
#SBATCH --nodes=1
#SBATCH --exclusive
#SBATCH --ntasks-per-node=32
#SBATCH -o mencoder_%j.out # STDOUT
#SBATCH -e mencoder_%j.err # STDERR
for infile in *.mkv #Suponga que son 100 ficheros a convertir
do
mencoder ${infile} -ovc xvid -oac mp3lame -oac mp3lame -
lameopts abr:br=192 -xvidencopts pass=2:bitrate=-
700000:threads=32 -o ${infile}.avi
done
```

¿Puedo ir más allá?

- Suponga que tiene que procesar 1000 ficheros de video de High Definition (7GB cada uno).
- Sobre un nodo demora una eternidad.
- Solución: usar paralelismo de datos y … un poco de bricolage.

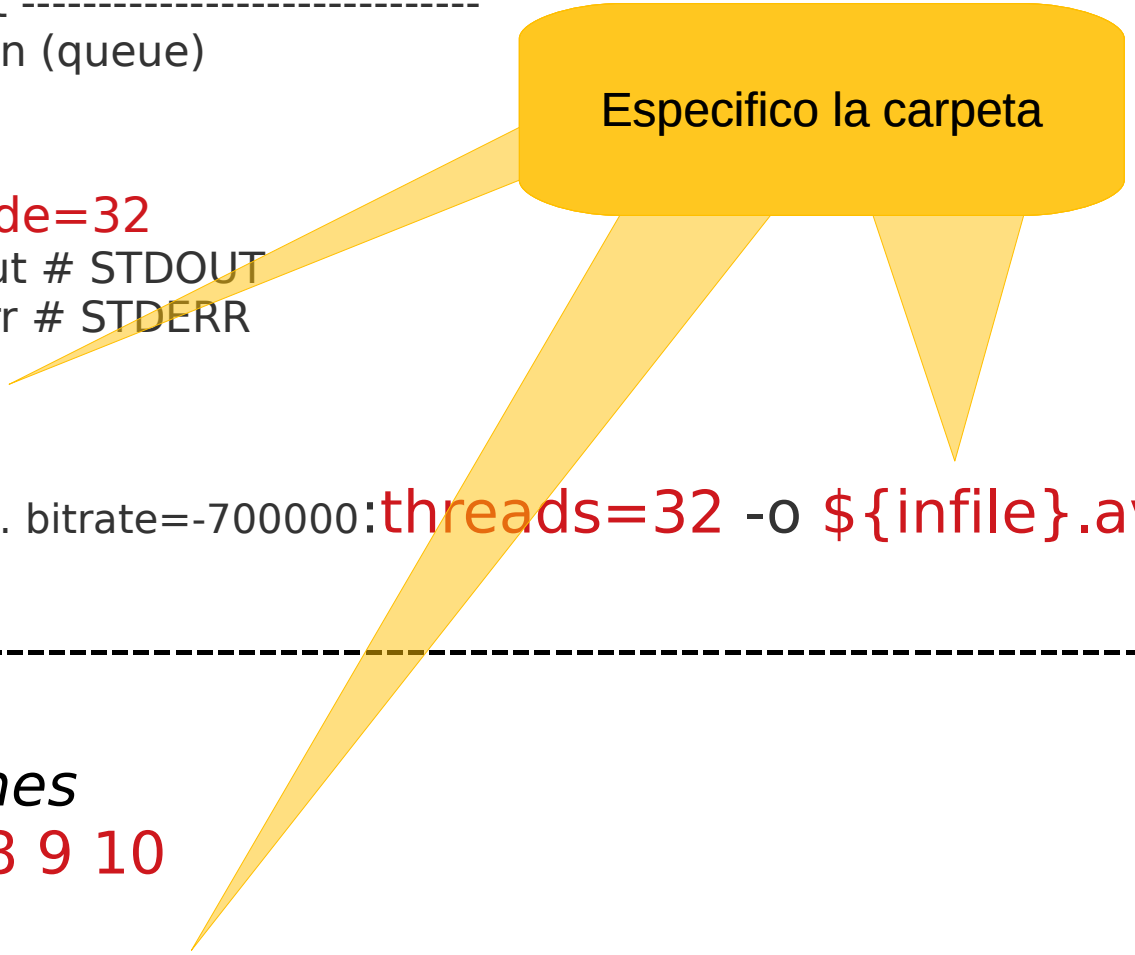
!Puedo ir más allá!



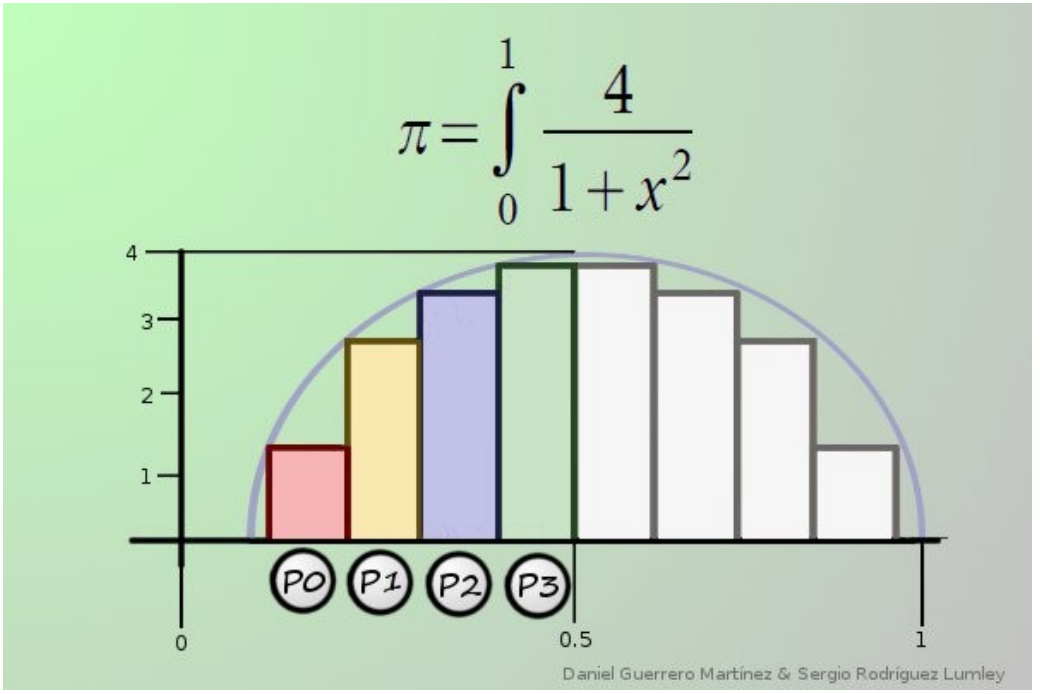
En cada carpeta almaceno, por ejemplo, 100 ficheros de entrada, y ahora tengo ¡10 problemas distintos!

Script ejemplo (III)

```
#!/bin/bash
#---------------- encoder_script ----------------------------
#SBATCH -p public # partition (queue)
#SBATCH --nodes=1
#SBATCH --exclusive
#SBATCH --ntasks-per-node=32
#SBATCH -o mencoder_%j.out # STDOUT
#SBATCH -e mencoder_%j.err # STDERR
for infile in $1/*.mkv
do
mencoder ${infile} ... bitrate=-700000:threads=32 -o ${infile}.avi
done
#-------------------------------------------------------------------

cd /home/artigas/filmes
for i in 1 2 3 4 5 6 7 8 9 10
do
    sbatch encoder_script $i
done
```

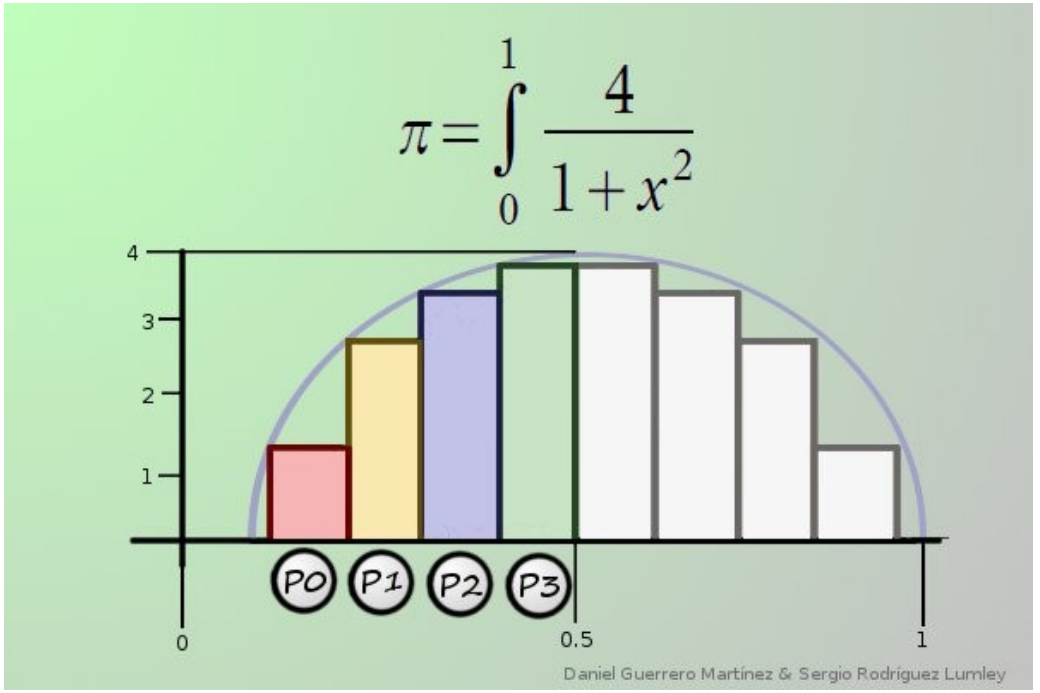


Caso de estudio III
Un programa para memoria compartida
Paso de mensajes con MPI



Cálculo de PI en paralelo

- Esquema puro de memoria distribuida
  - Se usan varios nodos
  - En cada nodo se usa un único procesador



Ejemplo de script para MPI

```
#!/bin/bash
#---------------- encoder_script ----------------------------
#SBATCH -p public # partition (queue)
#SBATCH --nodes=10
#SBATCH –-exclusive
#SBATCH --ntasks-per-node=1
#SBATCH -o pi_%j.out # STDOUT
#SBATCH -e pi_%j.err # STDERR

#Cargar el módulo de MPI
module load OpenMPI

#ejecutar 10 copias del programa, una sobre cada nodo
mpirun -np 10 parallel_mpi
```

Totalmente ineficiente, a menos que pruebe escalabilidad, o cada copia use una cantidad enorme de memoria RAM

Ventajas de MPI

```
#!/bin/bash
#SBATCH -p public # partition (queue)
#SBATCH --nodes=10
#SBATCH ––exclusive
#SBATCH --ntasks-per-node=1
#SBATCH -o pi_%j.out # STDOUT
#SBATCH -e pi_%j.err # STDERR
```

```
#!/bin/bash
#SBATCH -p public # partition (queue)
#SBATCH --nodes=1
#SBATCH ––exclusive
#SBATCH --ntasks-per-node=10
#SBATCH -o pi_%j.out # STDOUT
#SBATCH -e pi_%j.err # STDERR
```

- Funciona lo mismo para memoria compartida que para distribuida.
- La elección dependerá del consumo de memoria y del costo de las comunicaciones.

Ventajas de MPI

```
#!/bin/bash
#SBATCH -p public # partition (queue)
#SBATCH --nodes=10
#SBATCH --ntasks-per-node=1
#SBATCH -o pi_%j.out # STDOUT
#SBATCH -e pi_%j.err # STDERR
```

```
#!/bin/bash
#SBATCH -p public # partition (queue)
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=10
#SBATCH -o pi_%j.out # STDOUT
#SBATCH -e pi_%j.err # STDERR
```

```
#!/bin/bash
#SBATCH -p public # partition (queue)
#SBATCH --nodes=5
#SBATCH --ntasks-per-node=2
#SBATCH -o pi_%j.out # STDOUT
#SBATCH -e pi_%j.err # STDERR
```

- Funciona como un esquema híbrido.

Elección del modelo ...

Para una misma cantidad total de cores

Menor consumo de memoria — Mayor consumo de memoria

Menos nodos
Más cores x nodo

Memoria
compartida

Equilibrio

Esquema híbrido

Más nodos
Menos cores x nodo

Memoria
distribuida

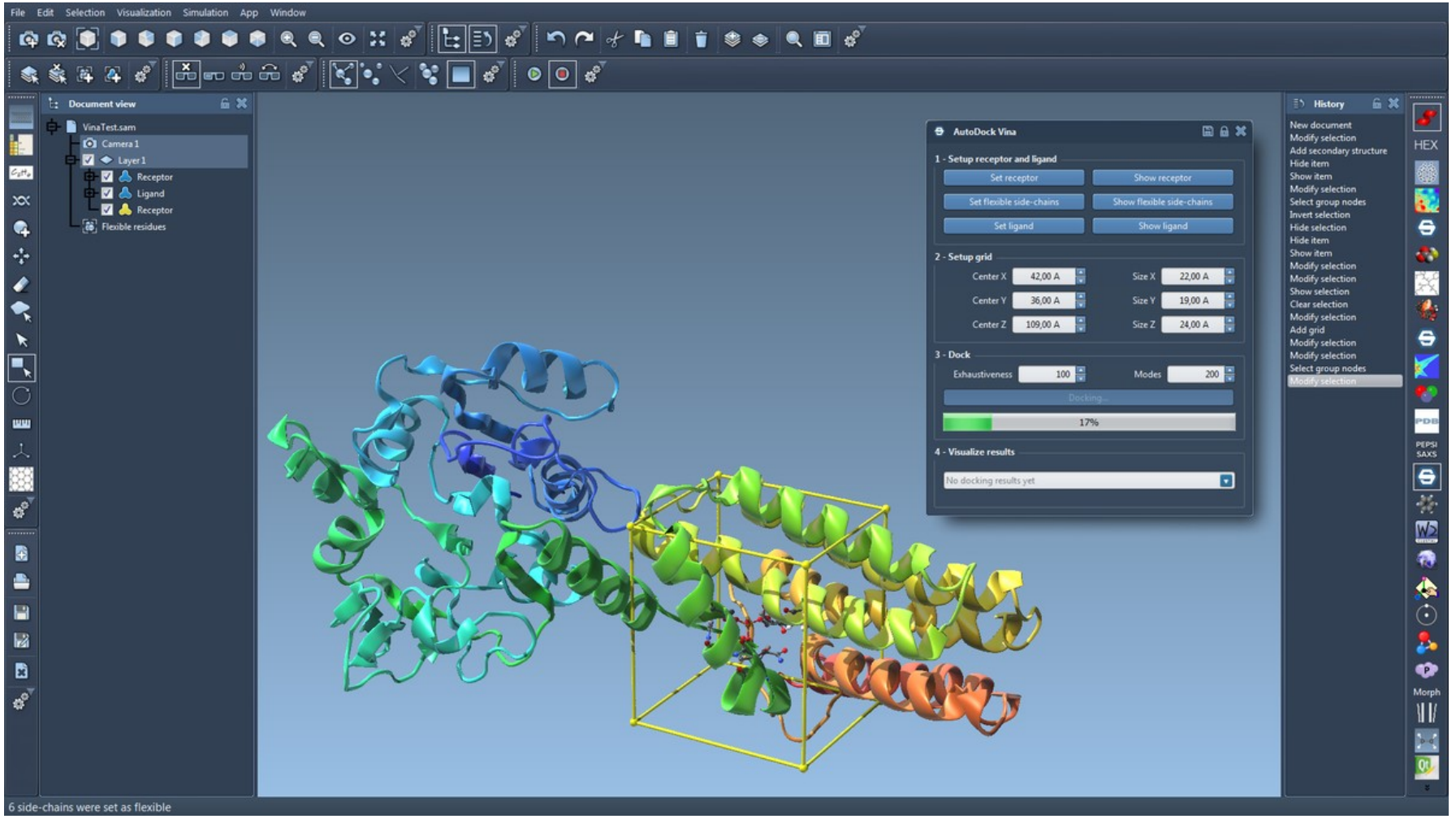
Si necesito más cores

Reservo más nodos y por lo tanto dispongo de más cores y más memoria

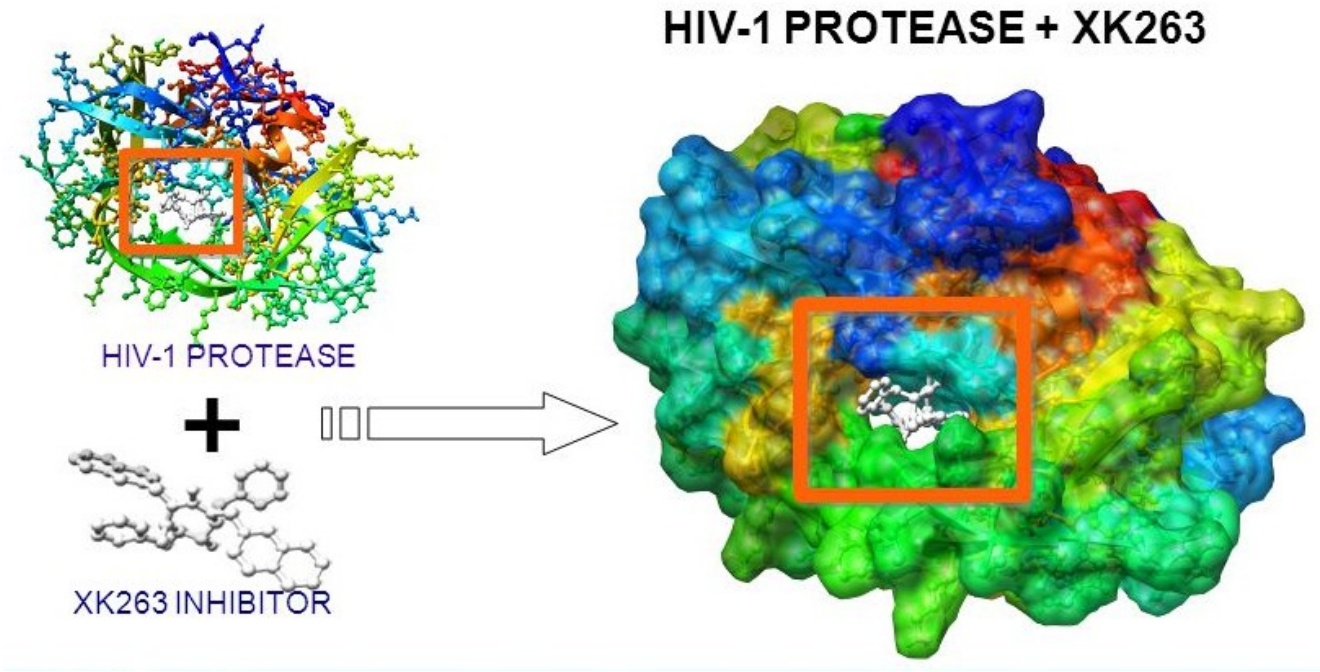
¿Número ideal de nodos?
¿Número ideal de cores?
Ver conferencia 2

Caso de estudio complejo
Autodock
Las 1001 formas de ejecutarlo

Autodock

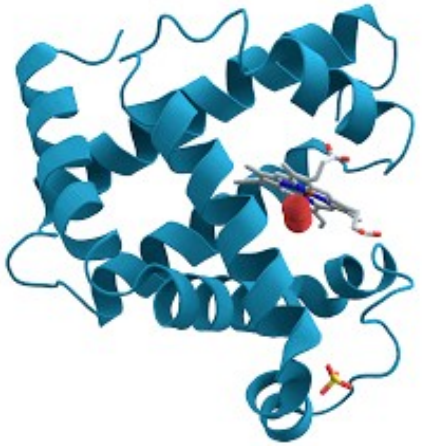

File Edit Selection Visualization Simulation App Window
Document view
VinaTest.sam
Camera 1
Layer 1
Receptor
Ligand
Receptor
Flexible residues
AutoDock Vina
1 - Setup receptor and ligand
Set receptor
Show receptor
Set flexible side-chains
Show flexible side-chains
Set ligand
Show ligand
2 - Setup grid
Center X 42,00 A
Size X 22,00 A
Center Y 36,00 A
Size Y 19,00 A
Center Z 109,00 A
Size Z 24,00 A
3 - Dock
Exhaustiveness 100
Modes 200
Docking...
17%
4 - Visualize results
No docking results yet
History
New document
Modify selection
Add secondary structure
Hide item
Show item
Modify selection
Select group nodes
Invert selection
Hide selection
Hide item
Show item
Modify selection
Modify selection
Show selection
Clear selection
Modify selection
Add grid
Modify selection
Modify selection
Select group nodes
Modify selection
HEX
PDB
PEPSI SAXS
Morph
6 side-chains were set as flexible

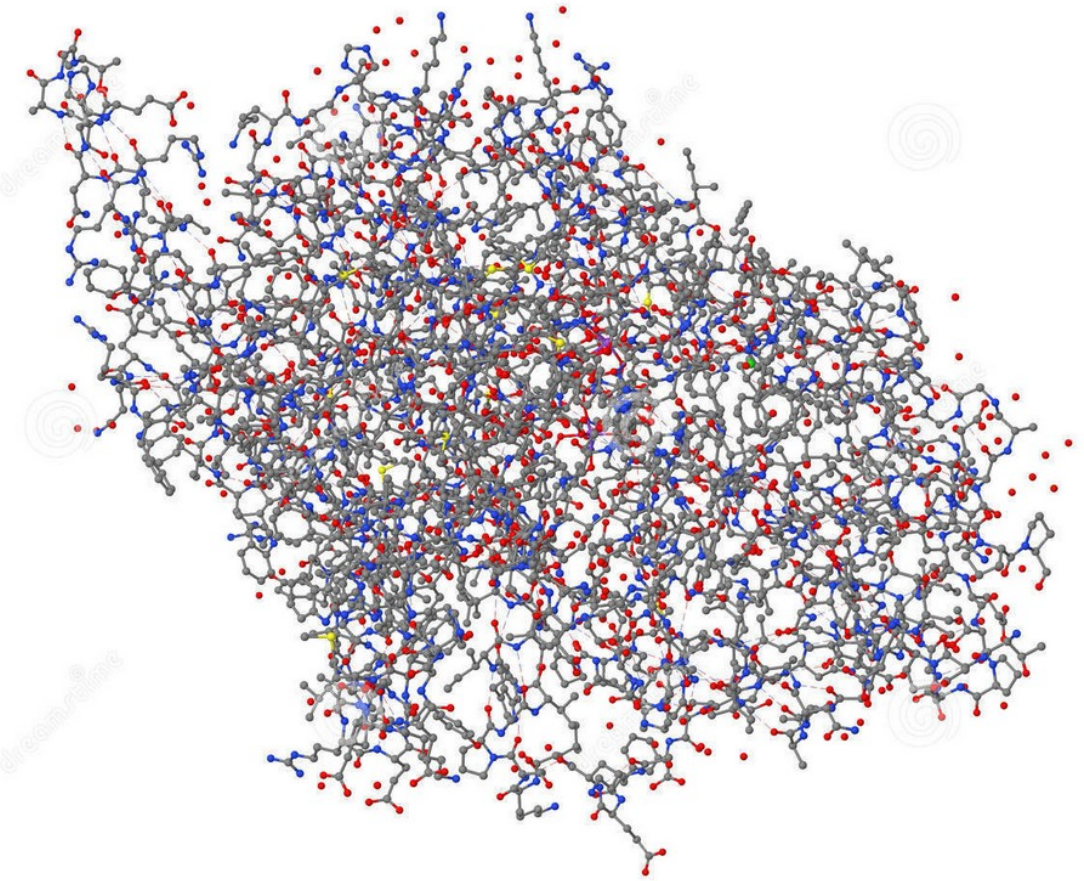
Acoplamiento molecular



En el campo del Modelado molecular, es un método que predice la conformación preferida de una molécula, al estar unida a otra, con el fin de formar un complejo estable. (WIKIPEDIA)

Autodock





Necesidades de capacidad de cómputo y memoria variadas

AutoDock Paralelo

Article

Parallel implementation of AutoDock

June 2007 · Journal of Applied Crystallography 40(3):598-599
DOI: 10.1107/S0021889807011053

Prashant Khodade · R. Prabhu · Nagasuma R Chandra · Show all 5 authors · R. Govindarajan

Overview | Stats | Comments | Citations (29) | References (4) | Related rese

Abstract

Computational docking of ligands to protein structures is a key step in structure-based drug design. Currently, the time required for each docking run is high and thus limits the use of docking in a high-throughput manner, warranting parallelization of docking algorithms. AutoDock, a widely used tool, has been chosen for parallelization. Near-linear increases in speed were observed with 96 processors, reducing the time required for docking ligands to HIV-protease from 81 min, as an example, on a single IBM Power-5 processor (1.65 GHz), to about 1 min on an IBM cluster, with 96 such processors. This implementation would make it feasible to perform virtual ligand screening using AutoDock.

Autodock, 1001 vías de...

Journal List › Springer Open Choice › PMC4801993

Springer JOURNAL OF COMPUTER-AIDED MOLECULAR DESIGN ▸ springer.com

J Comput Aided Mol Des. 2016; 30: 237–249. PMCID: PMC4801993
Published online 2016 Feb 20. doi: 10.1007/s10822-016-9900-9 PMID: 26897747

1001 Ways to run AutoDock Vina for virtual screening

Mohammad Mahdi Jaghoori,✉ Boris Bleijlevens, and Silvia D. Olabarriaga

▸ Author information ▸ Article notes ▸ Copyright and License information Disclaimer

Autodock, 1001 vías de ...

Table 2

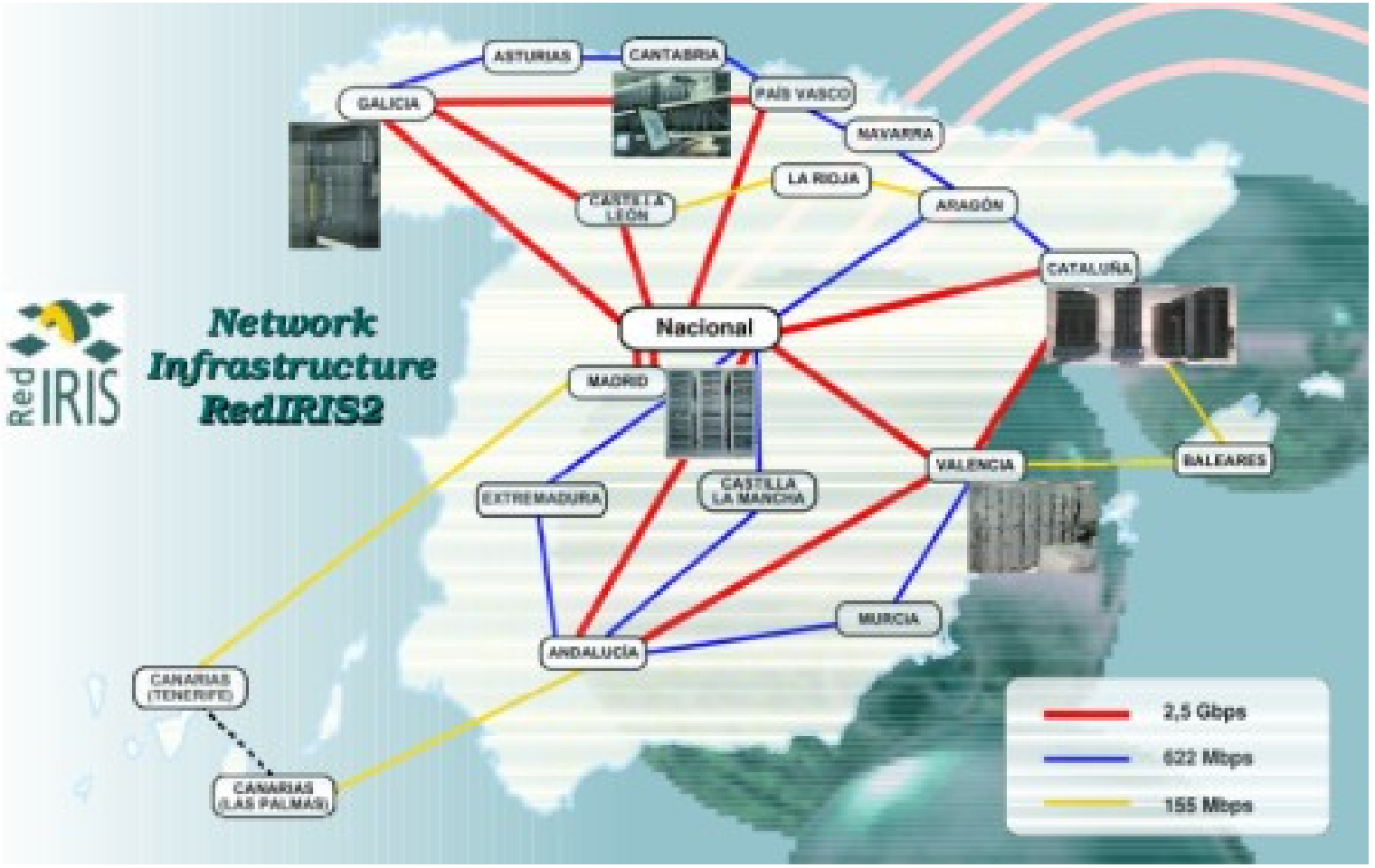
This table shows the four ligand libraries, four infrastructures, and the three docking boxes (FTO and the big and small boxes on NUR77)

	Multi-core	AMC cluster	Hadoop cluster	Grid
Nutra	Small	Small	Small	Small
	Big	Big	Big	Big
	FTO	FTO	FTO	FTO
HMDB	Small	Small	Small	Small
	Big	Big	Big	Big
	FTO	FTO	FTO	FTO
FDA	Small	Small	Small	Small
	Big	Big	Big	Big
	FTO	FTO	FTO	FTO
ZNP	Small	Small	Small	Small
	Big	Big	Big	Big
	FTO	FTO	FTO	FTO

When a name “Small”, “Big” or “FTO” is in black, it shows that we ran VS experiments with that docking box on the corresponding infrastructure and ligand libraries

- Sobre un nodo multicore.
- Sobre Clúster “HPC”
- Sobre un Clúster “Hadoop” (BigData)
- Sobre un GRID

Computación en GRID



Autodock ...

- ¿Qué cantidad de recursos debo reservar?
  - Depende del tamaño del problema a tratar.
  - Pudiera considerar tres tipos de problemas típicos:
    - Pequeño
    - Mediano
    - Grande
  - Ver la conferencia 2.

Resumen

TIPs para uso de nodos

- Programas secuenciales → UN SOLO NODO
- Programas para memoria compartida → UN SOLO NODO
- Programas para memoria distribuida → Varios nodos
- Programas híbridos → Varios nodos

Ver conferencia 2

TIPs para uso de cores

- Programas secuenciales → UN SOLO CORE
- Programas para memoria compartida → varios cores sobre un solo nodo
- Programas para memoria distribuida → Un core sobre varios nodos
- Programas híbridos → varios cores sobre varios nodos

Ver conferencia 2

TIPs para uso de memoria

- Si el programa no tiene requisitos especiales o no los conoce
  - Ejecutar normalmente (...ya explotará)
- Si requiere de mucha memoria
  - → especificar la memoria necesaria en el script (siempre es una buena práctica)
  - → valorar reservar el nodo de modo exclusivo, aunque no use todos los cores.

TIPs para reserva exclusiva

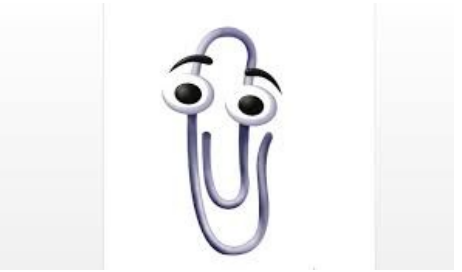
- ¿Cuándo necesito reservar todos los recursos de memoria y potencia de cálculo del nodo?
  - Si el programa tiene requerimientos altos.
  - Si el problema que abordo es complejo y costoso.
- Cuando las interferencias falsean los resultados.
  - Estoy midiendo el tiempo de ejecución y me afectan otros procesos. Ej. acceso al disco local.

Para la conferencia 2

- ¿Cómo determinar el número ideal de nodos y cores a reservar para un caso particular?
- ¿Existen otras dudas en el uso del clúster?

Para el futuro cercano...

- Asistente virtual para el uso del clúster.



FIN

¿Preguntas?

Aspectos avanzados en el uso del clúster UO

¡GRACIAS!