El cluster como herramienta para análisis masivos.

Curso: Métodos Estadísticos y Análiticos de datos Genómicos

Jérôme Verleyen
20 de enero 2010

Índice

- Introducción
- ¿Que es un cluster?
- Cluster del IBt.
- Uso general.
- Ejemplos.

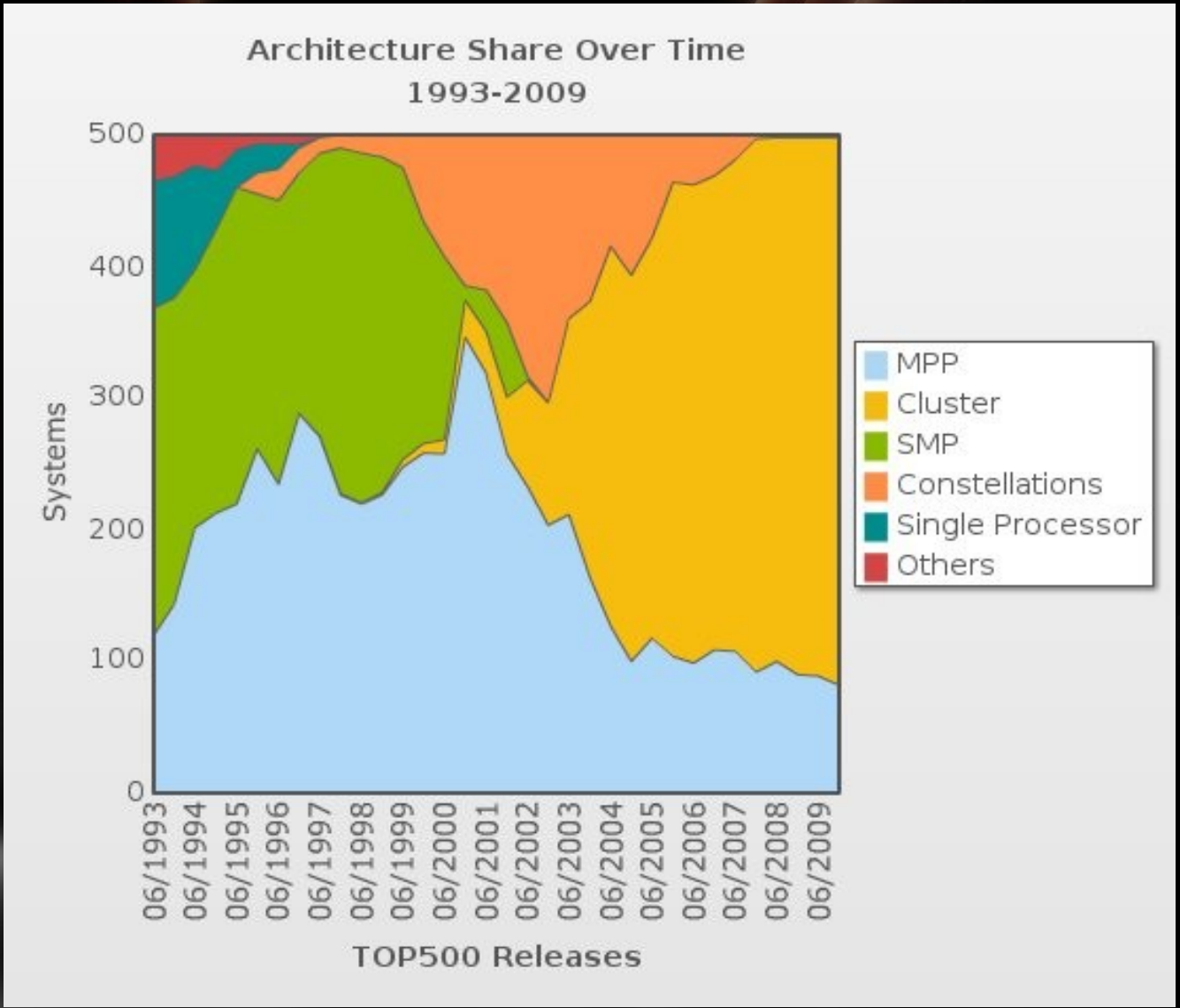
Bioinformática

- Dominio de estudio:
  - Secuencias de ADN, proteínas, ARN
  - Genomas, transcriptomas, metabolomas, regulomas
  - Filogenia,
  - Muchos más.....
- Las necesidades de poder de cálculo aumentan con la cantidad de datos a analizar.
- Muchos estudios necesitan a una “supercomputadora”.

”Supercomputer” (~1960)

- HPC (High-Performance Computing) ~ Años 1960: la empresa CDC (Control Data Corporation) con Seymour Cray.
- Una supercomputadora: al límite tecnológico de poder de cálculo (CPU) y de almacenamiento (I/O).
- Ahora, una simple computadora de oficina equivale a una “supercompomputadora” de los años 1960.

Evolución de la tecnología



Mejores supercomputadoras (www.top500.org)

- 1961: IBM 7030 ; 1,2 Mflops ;Los Alamos National Laboratory, USA
- 1974: CDC : Star-100 ; 100 Mflops ; Lawrence Livermore National Laboratory, USA
- 1983: Cray X-MP4; 820 Mflops.
- 2000: IBM POWER3 ; 4,94 TFlops Lawrence Livermore National Laboratory, USA
- 2009: Cray XT (Amd opteron); 1759 Tflops; Oak Ridge National Laboratory, USA

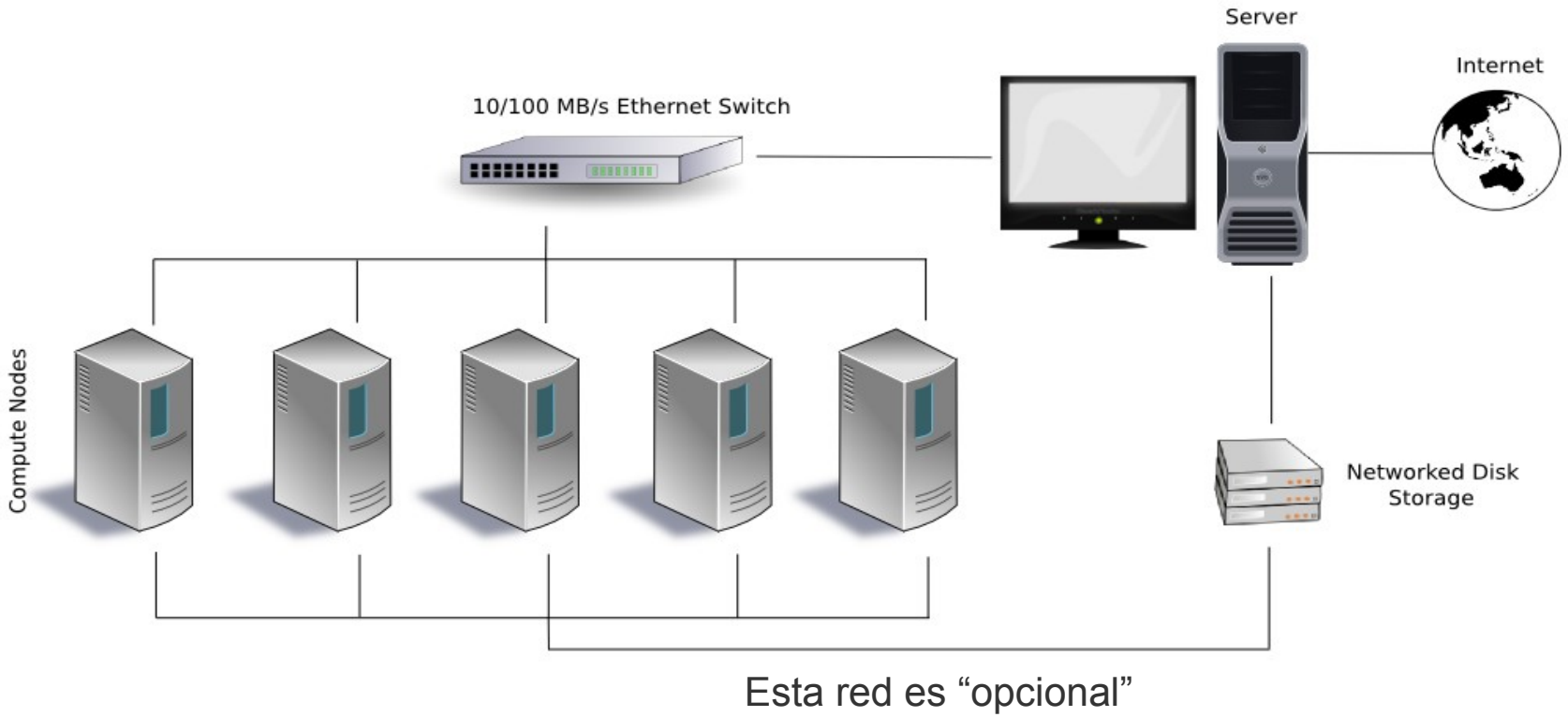
Inicios de Boewulf

- Hasta 1990, las supercomputadoras (MPPs) tenían precios muy elevados.
- Desde 1980, algunos investigadores desarrollan sistemas para estaciones de trabajos :
  - "Task Sharing"
  - Envío de mensajes.
- Sin embargo:
  - Licencias de los sistemas operativos (UNIX).
  - Precios de los componentes (red y máquinas) caros.

Nacimiento de Beowulf

- En los años 1990 ocurrieron dos eventos importantes:
  - 1991: lanzamiento de Linux por Linus Torwald.
  - Computadoras personales alcanzan las estaciones de trabajo.
  - Materiales de red bajan en precios (“Boum” de Internet).
- Donals Becker (NASA) crea el primer cluster “Beowulf” con 16 486 DX4 y red Ethernet.
- Beowulf: “H*éroe de una canción inglesa, que vence a un monstruo Grendel*”

Arquitectura cluster “Beowulf”



Autor: Mukarramahmad

Herramientas de un cluster

- Sistema operativo: Linux en mayoría (>80% Top500).
- Librerías paralelas: “Message Passing Interface” (MPI):
  - protocolo de comunicación : punto a punto, en grupo, definiciones de canales.
  - Disponible en C, C++ y Fortran
  - Diversas implementaciones : OpenMpi, Mpich, Mpich2, Lam/Mpi
  - API en C, fortran para desarrollar aplicaciones paralelas.

Herramientas de un cluster (II)

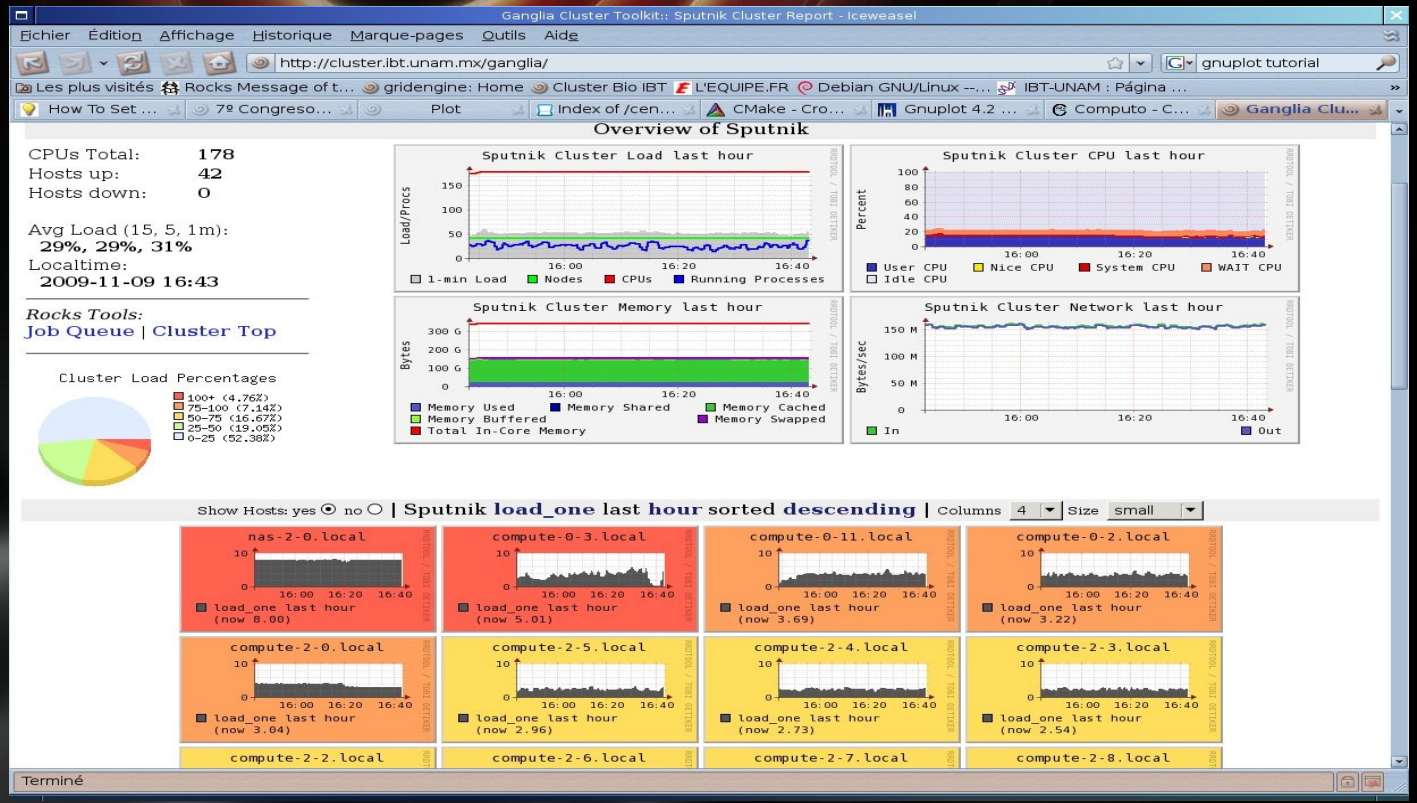
- Gestión de los nodos:
  - Disponibilidad total de los nodos para “todos” los usuarios.
- Uso de un sistema de colas:
  - Determinación de parámetros a tomar en cuenta (Tiempo CPU, RAM, característica especial).
  - A cada nodo se le asigna una o más colas.
  - Definición de política de usuarios (grupos con más prioridad)
  - Noción de “job” : descripción de la tarea a realizar.
- Implementación : Torque, OpenPBS, SGE, Condor.

Herramientas de un cluster (III)

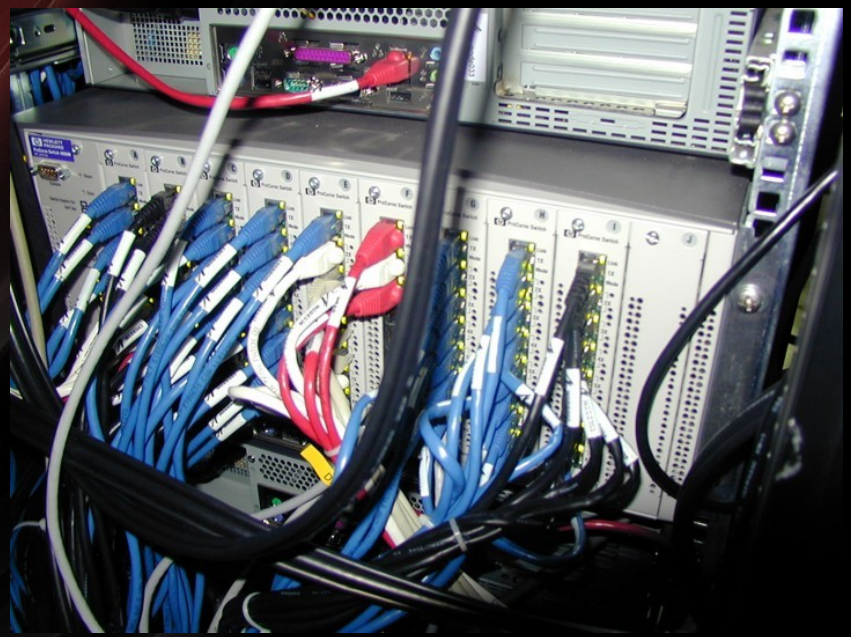
- Sistema de archivo: Depende del tipo de uso
- Sistema de archivo centralizado: NFS
    - en un(os) servidor(es) de disco.
- Sistema de archivos distribuidos: PVFS, PVFS-2, Lustre
    - Los nodos contienen parte del “File System”
    - PVFS-2 integra MPI
    - Usados en cluster de bases de datos por ejemplo.
- Distribución para cluster:
    - Oscar (Redhat y Debian), Rocks (Centos, Solaris)
    - Facilidad de instalación

Herramientas de un cluster (IV)

- Sistema de control de los componentes: Ganglia
  - monitoreo via web.
  - Componente ligero (cpu y red).



Cluster del IBt



Cluster del IBt (II)

- CPUS: Nodos de cálculo a base de AMD 64 bits, doble “dual cores” (35+1), doble “quad cores” (3) y dobe “hexa core” (1): 176 cores disponibles.
- 3 servidores de archivos (20 Tb para secuenciación).
- Red Gigabits ethernet
- Centos 5.3 como Sistema Operativo (distribución “Rocks” para cluster)

Paquetes disponibles

- El “roll” de bioinfomática de la distribución Rocks incluye (no exhaustivo):
  - Blast (ncbi y mpiBlast)
  - Hmmer
  - Gromacs
  - MrBayes
  - Emboss
  - Bioperl

Paquetes disponibles (II)

- Se añadió paquetes no incluidos en el “roll” de bioinfomática (no exhaustivo):
  - NAMD
  - GAPipeline
  - R (2.10)
  - Meme
  - Autodocksuite

Acceso al cluster

- Por el momento, únicamente gracias a una conexión segura SSH
- Host: cluster.ibt.unam.mx
- Para pedir cuenta en el cluster:
  - Email a "clustadm@ibt.unam.mx"
- Se prevee acceso más fácil vía web.

Gestión del cluster

- EL cluster esta controlado via el uso de colas:
  - Trabajos no más de 24 horas : cola “normal”
  - Trabajos de más de 24 horas : cola “lenta”
- El manejador de colas es “Sun Grid Engine”
- Prioridad: los recursos son accesible según prioridad definida así:
  - El que más lo usa, menos tiene prioridad
  - Los jobs < 24 horas tienen más prioridad (posibilidad de suspensión de la cola lenta)
  - Equipos especiales : UUSM tiene equipos prioritarios.

Gestión del cluster (II)

- Se necesita escribir un “job”: descripción del trabajo que se quiere realizar
  - Es un archivo que contiene la descripción de las tareas que se requiere.
  - Se ejecutara en un nodo (no en el master)
  - Necesita pensar al acceso a los datos y donde escribir los resultados (leer y escribir via la red es 100 veces mas lento que en el disco local).
- Para aprender, se puede usar un job “interactivo”: el sistema le da un cpu, para fines de pruebas.

Estados de los jobs

- Los jobs están controlados por SGE. Maneja la noción de “estado” de job, hasta su termino.
- Estados más “vistos”:
  - “qw” : Queue and waiting : determinando los recursos pedidos, la disponibilidad de estos recursos, prioridad.
  - “r” : running
  - “h” hold: suspendido por el usuario o el sistema
  - “E” Error: se generó un error al momento de correrlo.

Monitoreo de los recursos: “qstat”

- Ver las colas disponibles:

```
[jerome@cluster ~]$ qstat -g c
CLUSTER QUEUE                   CQLOAD   USED    RES  AVAIL  TOTAL aoACDS  cdsuE
--------------------------------------------------------------------------------
all.q                             0.25      1      0     71    164      0     92
lenta.q                           0.27      1      0     11     48      0     36
secuencia.q                       0.37      1      0      7     40      0     32
uaem.q                            0.80     48      0      0     48      0      0
```

Monitoreo de jobs

- Ver sus propios trabajos

```
[jerome@cluster ~]$ qstat

job-ID  prior   name       user         state submit/start at     queue
                slots ja-task-ID

-----------------------------------------------------------------------------
-------------------------------

   5768 0.75000 PiCalc-4   jerome       r     11/24/2009 21:17:33 all.q@compute-
0-4.local             4
```

Monitoreo de jobs (II)

- Ver todos los jobs : “qstat -u *”

```
[jerome@cluster ~]$ qstat -u \*

job-ID  prior   name       user         state submit/start at     queue                          slots ja-
task-ID

-----------------------------------------------------------------------------------------------
-----

   5756 0.60001 d298       cmp_uaem     r     11/24/2009 16:12:15 uaem.q@compute-2-1.local           8

   5755 0.70001 r24g573    cmp_uaem     r     11/24/2009 16:28:45 uaem.q@compute-1-10.local         16

   5282 0.70001 c298r24g   cmp_uaem     r     11/10/2009 12:55:48 uaem.q@compute-2-5.local          16

   5291 0.60001 sa1573_15  cmp_uaem     r     11/10/2009 15:27:03 uaem.q@compute-2-7.local           8

   5762 0.50391 dG40_BAFDV siguel       r     11/24/2009 17:42:36 lenta.q@compute-0-3.local          1

   5765 0.71250 QRLOGIN    acobian      r     11/24/2009 18:19:19 all.q@compute-0-7.local            1

   5766 0.51256 my_velvet_ lcollado     r     11/24/2009 19:01:19 secuencia.q@compute-1-14.local     1

   5768 0.73837 PiCalc-4   jerome       r     11/24/2009 21:17:33 all.q@compute-0-4.local            4
```

Job Interactivo

- Una vez en el master del cluster:
  - Usar el comando “qrsh”
  - Un nodo sera proporcionado
  - Ejecución de los programas deseados
  - “exit” para liberar el recurso

Ejemplo de job interactivo

```
[jerome@cluster ~]$ qrsh
[jerome@compute-0-7 ~]$ export BLASTDB=/share/Databases/Blast/db/
[jerome@compute-0-7 ~]$ blastall -p blastn -d est_mouse -i nuc.fasta -o result.txt
[jerome@compute-0-7 ~]$ head result.txt
BLASTN 2.2.20 [Feb-08-2009]


Reference: Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schaffer,
Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
"Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs",  Nucleic Acids Res. 25:3389-3402.

Query= gi|58270915|ref|XM_572614.1| Filobasidiella neoformans
hypothetical protein (CNI04020) mRNA, complete cds
[jerome@compute-0-7 ~]$ exit
[jerome@cluster ~]$
```

¿Como hacer un job?

- Se necesita un script, que describe lo que se desea hacer, por ejemplo: “simple.sh”

```
#!/bin/sh
# request Bourne shell as shell for job
#$ -S /bin/sh
#$ -N PruebaSimple     # Salidas en PruebaSimple.oXXX y PruebaSimple.eXXX

# print date and time
date
# Sleep for 20 seconds
sleep 20
# print date and time again
date
```

“Someter” el job: qsub

- El comando “qsub” permite enviar trabajos a la cola:
  - qsub simple.sh
  - qsub -l lenta long-time-job.sh (para perdir más de 24 horas)
- El comando qsub regresa un “jobId”, el número de su trabajo.
- El comando “qstat” permite verificar el estado de su job

Ejemplo de corrida sencilla

```
[jerome@cluster ~]$ qsub simple.sh
Your job 5785 ("PruebaSimple") has been submitted

[jerome@cluster ~]$ qstat
job-ID  prior   name       user         state submit/start at     queue
  slots ja-task-ID
-----------------------------------------------------------------------------
------------------
 5785 0.00000 PruebaSimp jerome       qw    11/25/2009 10:32:45
    1

[jerome@cluster ~]$ qstat
job-ID  prior   name       user         state submit/start at     queue
  slots ja-task-ID
-----------------------------------------------------------------------------
------------------
 5785 0.51250 PruebaSimp jerome       r     11/25/2009 10:32:45 all.q@compute-0-5.local

[jerome@cluster ~]$ qstat
[jerome@cluster ~]$ ls PruebaSimple.*5785  -l
-rw-r--r-- 1 jerome administrador  0 Nov 25 10:34 PruebaSimple.e5785
-rw-r--r-- 1 jerome administrador 58 Nov 25 10:35 PruebaSimple.o5785
```

Ejemplo mas útil : Blast

- Secuencia en archivo “nuc.fasta”, base de datos nt, y resultados en “results.txt” : “script-blast.sh”:

```
#!/bin/bash

#$ -N Blast


PATH=$PATH:/opt/Bio/ncbi/bin/


export BLASTDB=/share/Databases/Blast/db

blastall -p blastn -d nt -i nuc.fasta -o results.txt
```

Enviar el job de Blast

```
[jerome@cluster ~]$ qsub script-blast.sh
Your job 5786 ("Blast") has been submitted

[jerome@cluster ~]$ qstat -j 5786
==============================================================
job_number:                 5786
exec_file:                  job_scripts/5786
submission_time:            Wed Nov 25 10:41:45 2009
owner:                      jerome
uid:                        1865
group:                      administrador
gid:                        502
sge_o_home:                 /home/jerome
sge_o_log_name:             jerome
sge_o_path:                 /opt/gridengine/bin/lx26-
amd64:/usr/kerberos/bin:/opt/gridengine/bin/lx26-
amd64:/usr/java/latest/bin:/usr/local/bin:/bin:/usr/bin:/opt/Bio/ncbi/bin:/opt/Bio/m
piblast/bin/:/opt/Bio/hmmer/bin:/opt/Bio/EMBOSS/bin:/opt/Bio/clustalw/bin:/opt/Bio/t
coffee/bin:/opt/Bio/phylip/exe:/opt/Bio/mrbayes:/opt/Bio/fasta:/opt/Bio/glimmer/bin:
//opt/Bio/glimmer/scripts:/opt/Bio/gromacs/bin:/opt/Bio/gmap/bin:/opt/Bio/tigr/bin:/
opt/eclipse:/opt/ganglia/bin:/opt/ganglia/sbin:/opt/maven/bin:/opt/openmpi/bin/:/opt
/rocks/bin:/opt/rocks/sbin:/home/jerome/Paquetes/bps/bin
sge_o_shell:                /bin/bash
sge_o_workdir:              /home/jerome
sge_o_host:                 cluster
account:                    sge
```

.../...

Resultado del Blast

```
[jerome@cluster ~]$ ls *5786 -l
-rw-r--r-- 1 jerome administrador 0 Nov 25 10:41 Blast.e5786
-rw-r--r-- 1 jerome administrador 0 Nov 25 10:41 Blast.o5786
[jerome@cluster ~]$ head -30 results.txt
BLASTN 2.2.20 [Feb-08-2009]
   .../...
Query= gi|58270915|ref|XM_572614.1| Filobasidiella neoformans
hypothetical protein (CNI04020) mRNA, complete cds
         (2389 letters)

Database: nt
           9,572,415 sequences; 28,778,988,663 total letters

Searching.....................................................done

                                                                 Score    E
Sequences producing significant alignments:                      (bits) Value

ref|XM_572614.1| Cryptococcus neoformans var. neoformans JEC21 h...  4736   0.0
ref|XM_768838.1| Cryptococcus neoformans var. neoformans B-3501A...  4633   0.0
gb|AE017349.1| Cryptococcus neoformans var. neoformans JEC21 chr...  2426   0.0
ref|XM_002096726.1| Drosophila yakuba GE24873 (Dyak\GE24873), mRNA     46   0.90
ref|XM_001980973.1| Drosophila erecta GG17473 (Dere\GG17473), mRNA     46   0.90
gb|AE014297.2| Drosophila melanogaster chromosome 3R, complete s...    46   0.90
ref|NM_001144554.1| Drosophila melanogaster grain (grn), transcr...    46   0.90
ref|XM_002032053.1| Drosophila sechellia GM26366 (Dsec\GM26366),...    46   0.90
```

Cancelar un job

- Se usa el comando “qdel” para matar un trabajo
- Este puede estar en cualquier estado, pero no acabado :-)
- Se usa el número del jobId para indicar cual job se quiere parar

Ejemplo de matar un job

```
[jerome@cluster ~]$ qstat
job-ID  prior   name       user         state submit/start at     queue
   slots ja-task-ID
-----------------------------------------------------------------------------
--------------------
   5791 0.00000 PruebaSimp jerome       qw    11/25/2009 11:29:37
      1
[jerome@cluster ~]$ qstat
job-ID  prior   name       user         state submit/start at     queue
   slots ja-task-ID
-----------------------------------------------------------------------------
--------------------
   5791 0.51255 PruebaSimp jerome       r     11/25/2009 11:29:47 all.q@compute-1-7.local
      1
[jerome@cluster ~]$ qdel 5791
jerome has registered the job 5791 for deletion
[jerome@cluster ~]$ qstat
[jerome@cluster ~]$
```

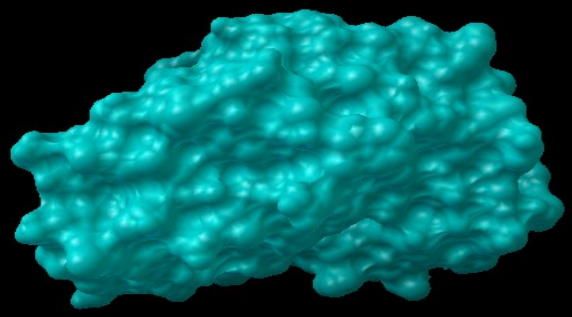
Cluster: para estudios más grandes!

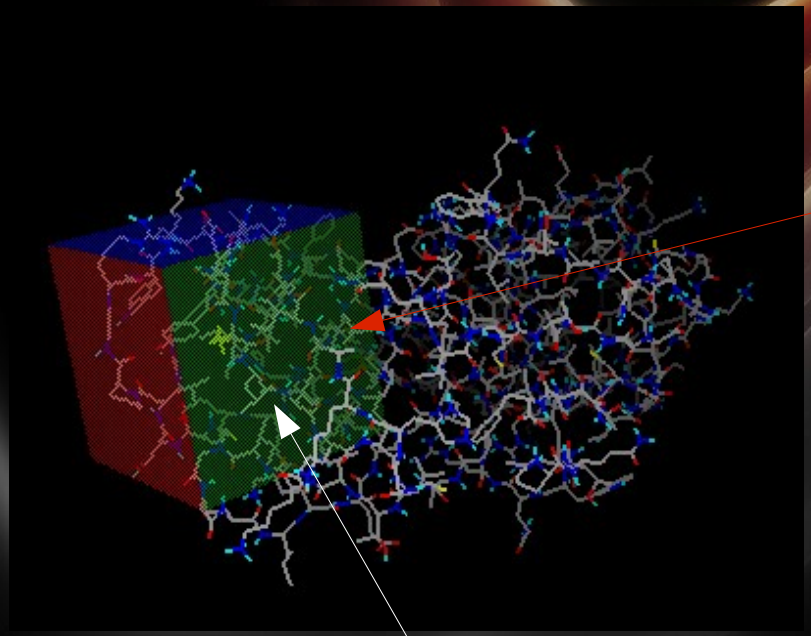
- Hasta ahora ejemplos sencillos. ¿Porque usar el cluster ?
- Un cluster se usa principalmente para:
  - Programas paralelos.
  - Estudios con grandes cantidades de datos similares.
  - Estudios paramétricos.

Estudio de Docking de proteínas

- Caso concreto, Docking de proteínas:
  - Proteína maligna con un sitio activo conocido: hay que anihilarlo
  - Biblioteca completa de ligandos : ZINC
  - Utilizar un programa de docking para analizar las interacciones proteína-ligando y sacar los posibles candidatos
  - Un solo “click” para enviar el estudio.

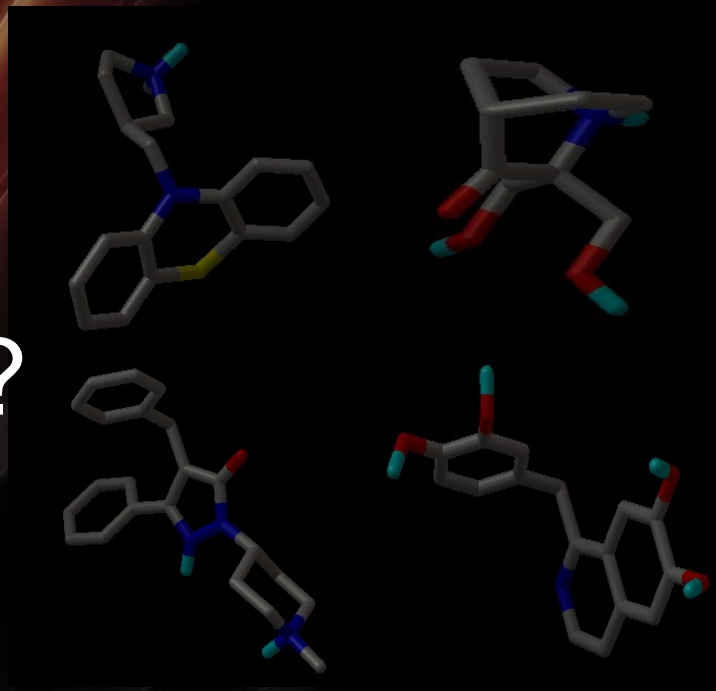
La proteína y los ligandos





Sitio activo

?



Zinc Database

¿Como? : Job Array

- Usaremos un job especial para lograr ese trabajo: “job array”.
- Noción de parámetro : en este caso, el nombre del ligando elegido
- Se maneja una sola referencia de job (fácil de “matar”)
- Cada sub-job podrá correr en cuanto estén disponibles los recursos necesarios. (una sola CPU).

Definición de la lista.

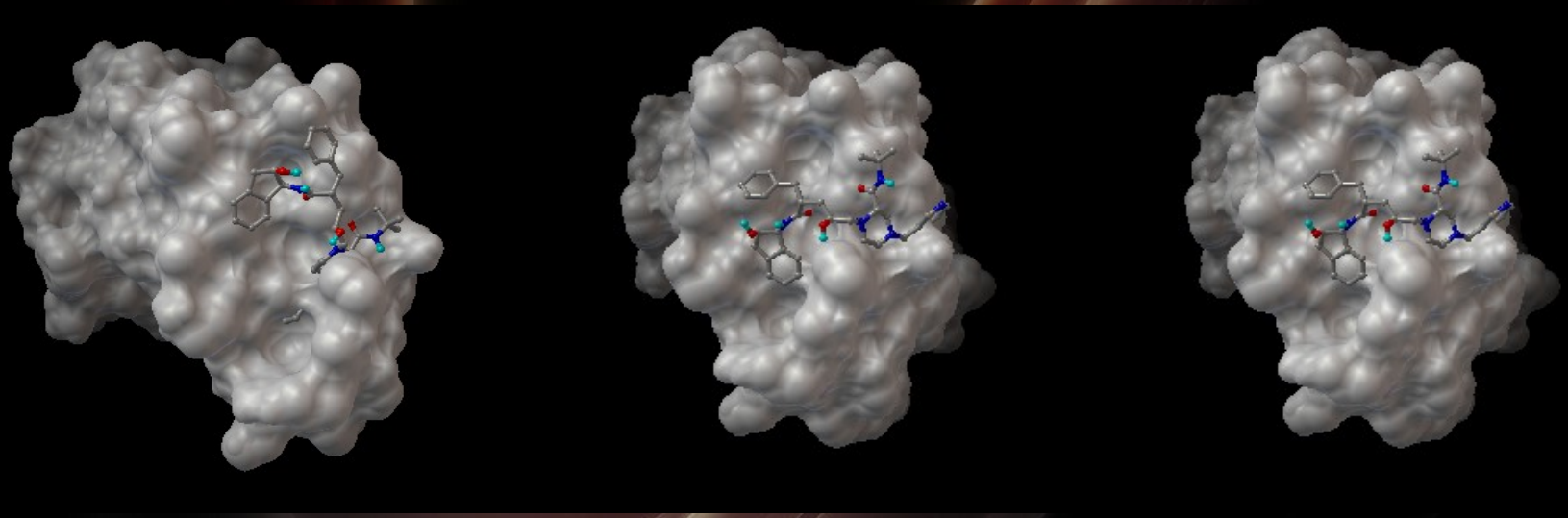
- Lista de los ligandos: un archivo lista-ligandos.txt (ej: 2100 ligandos) :

```
ZINC19737225
ZINC4990323
ZINC982962

../..

ZINC8575396
ZINC20030231
```

Puntos importantes del script

```
#!/bin/bash

#$ -N ScreeningJob

#$ -j y

## En este caso, hay 2100 ligandos que estudiar.
#$ -t 1-2100

LISTLIGAND=~/lista-ligandos

LIGAND=$(cat $LISTLIGAND | head -n $SGE_TASK_ID | tail -n 1)

.../... pre-calculos por compuestos presente en la reja

# Docking
autodock4 -p "$LIGAND"_x1hpv.dpf -l "$LIGAND".dlg
```

Extraer los mejores:



226.9 Kcal/mol

2120.9 Kcal/mol

1690.4 Kcal/mol

Referencias:

- http://www.Top500.org
- http://www.beowulf.org
- http://www.rocksclusters.org
- http://gridengine.sunsource.net
- http://cluster.ibt.unam.mx
- http://zinc.docking.org

¡Gracias!