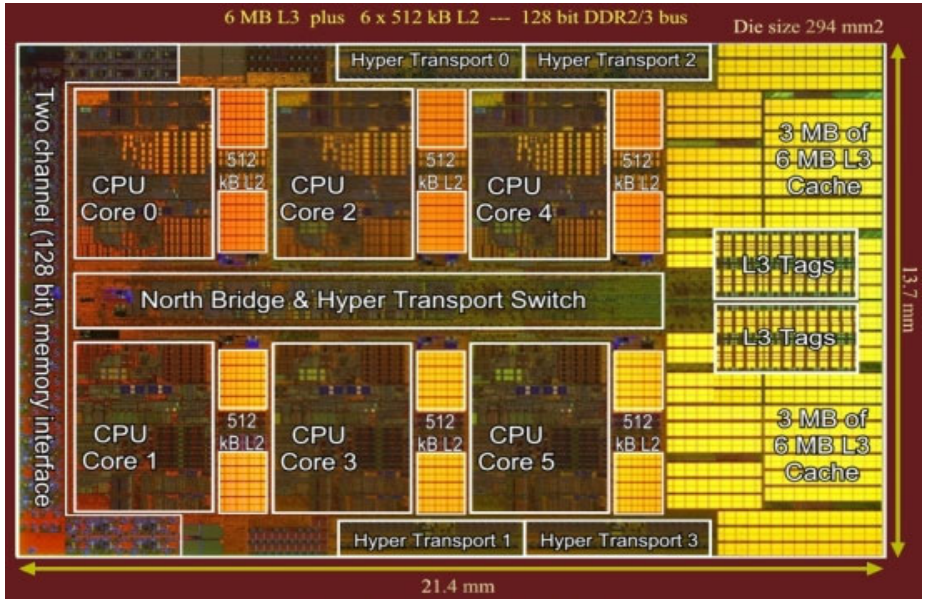
Uso del clúster de docencia
Arquitectura de Ordenadores

Clúster de Ordenadores

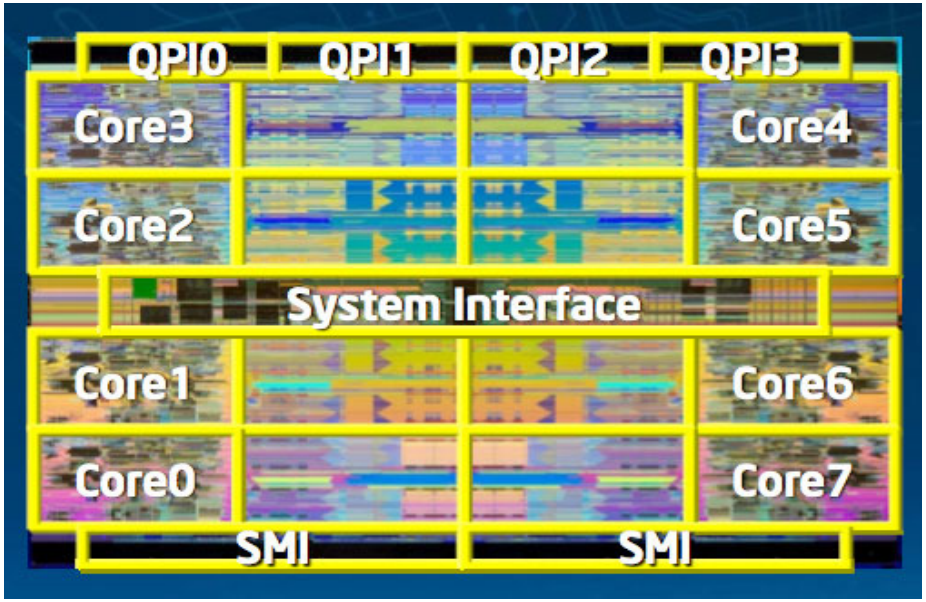
Cada procesador/core es una "máquina" diferente (memoria distribuida)

CPUs Multi-Core

Procesador AMD 6/12 cores



Procesador Intel 8 cores



Todos los procesadores/cores de la máquina conforman una "única máquina" (memoria compartida)

Equipos en el laboratorio

- **Ordenadores**
  - Linux
  - Tienen instalado OpenMP
  - Los usaremos para el desarrollo

- **Clúster**
  - Linux Centos 6.3
  - Tienen instalado OpenMP y SGE
  - Los usaremos para pruebas
  - Diferente número de procesadores, red de comunicaciones, etc.

Clúster Labomat (IP 150.244.56.153)

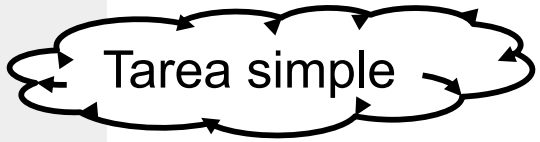
- **25 equipos – 224 cores – 400 GB**
- **Frontend: Equipo con dos procesadores AMD 8-core y 16 GB de RAM**
- **4 equipos con dos procesadores Intel quad-core y 12 GB de RAM.**
  - Tienen habilitado el Hyper-Threading por lo que el total de cores por equipo es 16.
- **4 equipos con dos procesadores AMD six-core y 12 GB de RAM.**
  - En total cada equipo tiene 12 cores.
- **12 equipos “virtuales” con 8 cores (Intel i7) y 24 GB de RAM**

Introducción a SGE

- **El Sun Grid Engine (SGE) maneja y controla todos las tareas que se ejecutan en el clúster, incluyendo el balanceo de las diferentes tareas entre las máquinas disponibles**
  - Se asegura de que todas las tareas se ejecutan en máquinas con suficiente memoria y número de CPUs para ejecutar la tarea
- **Una tarea SGE es simplemente un script Unix que ejecuta uno o varios comandos**
  - En muchos casos sigue la misma secuencia de comandos que se escriben en una shell antes de realizar la tarea real
  - Posteriormente se envía la tarea o script a SGE junto con la lista de requisitos (uso de memoria, número de CPUs, tiempo de ejecución, etc.) y SGE encontrará la máquina o máquinas necesarias para ejecutar la tarea

Introducción a SGE

```
#----------------------------Start program.job-----------------------
#!/bin/bash

# The name of the job, can be whatever makes sense to you
#$ -N ProgramName_EstimatedHoursToFinish_MiscComment

# The job should be placed into the queue 'all.q'.
#$ -q default

# Redirect output stream to this file.
#$ -o sge_output.dat

# Redirect error stream to this file.
#$ -e sge_error.dat

# The batchsystem should use the current directory as working directory.
# Both files (output.dat and error.dat) will be placed in the current
# directory. The batchsystem assumes to find the executable in this directory.
#$ -cwd

# This is my email address for notifications. I want to have all notifications
# at the master node of this cluster.
#$ -M username@domain.name

# Send me an email when the job is finished.
#$ -m e

# This is the file to be executed.
echo $PATH

time genesis ./CLIMB9.g > ./sge.out
#----------------------------End program.sge-----------------------
```



Script simple para SGE

```
#!/bin/bash
cd /home/usuario/ejemplo
myprog
```

- SGE elegirá la máquina para esta tarea y entonces la tarea cambiará de directorio y ejecutará el programa "myprog" en la máquina seleccionada
- En este caso no se especifican ficheros de entrada y salida
  - Se asume que cualquier fichero de entrada se debe encontrar en el directorio de ejecución o que el programa sabe donde encontrarlo
  - Del mismo modo, los ficheros de salida se generarán en el mismo directorio
- La primera línea "# !", identifica el tipo de shell que se usará como interprete de comandos - en este caso es la shell "bash"
  - El uso de "# !" en la primera línea es una convención estándar de Linux
- Otras líneas que empiezan por "#" son líneas de comentarios y serán ignoradas durante la ejecución del script

Envío de tareas a SGE

- **Para enviar una tarea a SGE se usa el comando 'qsub' y el nombre del script:**

```
% qsub simple.q
```

  donde 'simple.q' es el nombre del fichero que incluye los comandos anteriores

- **Cuando se envía una tarea es posible indicar ciertos parámetros que indican como debe ejecutarse:**

```
% qsub -o simple.out simple.q
```

  - El parámetro '-o simple.out' indica que cualquier salida debe ser redireccionada al fichero 'simple.out'
  - SGE provee un número de parámetros que pueden ser usados para identificar como la tarea se debe ejecutar y que tipo de recursos son necesarios para que la tarea se ejecute correctamente

Envío de tareas a SGE

- **Se pueden añadir los parámetros directamente al script:**

```
#!/bin/bash
#
#$ -S /bin/bash
#$ -cwd
#$ -o simple.out
#$ -j y
cd /home/usuario/ejemplo
./myprog
```

  - El parámetro '-cwd' le dice a SGE que ejecute la tarea en el mismo directorio desde el que se ejecutó el comando qsub - por ejemplo, SGE se moverá al directorio de trabajo antes de ejecutar el script de la tarea
  - El parámetro '-o' se emplea para dirigir la salida a un fichero
  - El parámetro '-S' es otro modo de indicar a SGE el tipo de shell (tcsh, bash, or sh).
  - El parámetro '-j y' es un modo de decir al SGE que combine la salida estándar-error con la salida estándar, de modo que toda la salida del programa irá al fichero 'simple.out'

Envío de tareas a SGE

- Una vez enviada una tarea mediante 'qsub' a SGE, es posible conocer el estado de la tarea mediante el comando 'qstat', que además nos muestra información de la tarea:

```
% qstat
job-ID prior name user state submit/start at queue slots ja-task-ID
-----------------------------------------------------------------------------
8 0.55500 simple jose r 03/15/2010 11:35:45 all.q@compute-0-6.local 8
```

- Y eliminar una tarea con ‘qdel’:

```
% qdel 8
jose has deleted job 8
```

- Como se puede apreciar, el parámetro que se pasa a 'qdel' es el ID de la tarea

Envío de tareas a SGE

- **IMPORTANTE: Con 'qstat' solo vemos nuestras tareas y, por tanto, si nuestra tarea no se ejecuta, puede ser que haya tareas de otros usuarios ejecutándose en el cluster**
  - El estado ‘qw’ indicaría que está esperando
  - El estado ‘r’ indicaría que está en ejecución
  - Si aparece en el estado la letra ‘E’ indicaría error

```
% qstat
job-ID prior name user state submit/start at queue slots ja-task-ID
-----------------------------------------------------------------------------
8 0.55500 run_mpi_ri jose r 03/15/2010 11:35:45 all.q@compute-0-6.local 8
9 0.00000 run.sh jose qw 03/15/2010 11:35:45 4
```

  - La tarea 8 se está ejecutando en 8 procesadores ya que su estado (state) es 'r'
  - La tarea 9 llamada 'run.sh' está esperando en la cola ya que el estado de la misma (state) es 'qw’. Tendrá que esperar a que haya recursos disponibles

Envío de tareas a SGE

- **IMPORTANTE: En algunas ocasiones, puede ser que alguna de las tareas se quede indefinidamente esperando porque no hay recursos para ejecutarla, o que se haya producido algún error durante la ejecución y se quede en estado de ejecución para siempre. En estos casos, debemos hacer uso del comando 'qdel', que nos permite eliminar una tarea**

```
% qsub -pe openmp 180 run.sh
Your job 14 ("run.sh") has been submitted

% qstat
job-ID prior name user state submit/start at queue slots ja-task-ID
-----------------------------------------------------------------
14 0.60500 run.sh jose qw 03/15/2010 11:37:27 180
```

  - Ningún equipo tiene 128 cores, la tarea se quedará indefinidamente en espera.

Envío de tareas a SGE

- **IMPORTANTE: Aunque una tarea se quede indefinidamente en espera, el resto de las tareas, siempre que haya recursos disponibles, podrán ejecutarse. Solo en el caso de que la tarea se quede en ejecución de forma indefinida, lo que supone bloquear los recursos empleados por la tarea, otras tareas no podrán ejecutarse. Existen mecanismos en SGE para eliminar de forma automática este tipo de tareas (limitando el tiempo de ejecución), pero actualmente no están habilitados, por lo que es importante estar pendiente del tiempo de ejecución de la tarea y eliminarla cuando creamos que tarda demasiado**

Uso de colas en SGE

- **En el cluster Labomat, SGE ha sido configurado con varias colas que representan diferentes "configuraciones" para diferentes tipos de pruebas**
- **Las colas disponibles en el cluster Labomat son:**
  - *all.q*: Incluye a todos los equipos del cluster (cola por defecto). Es la configuración estándar. El máximo número de cores que puedes solicitar es 16
  - *amd*.q: Incluye solo equipos AMD. El número máximo de cores que puedes solicitar es 12
  - *intel.q*: Incluye solo equipos Intel. El número máximo de cores que puedes solicitar es 16
  - *mv.q*: Incluye solo los equipos sobre máquina virtual. El número máximo de cores que puedes solicitar es 8

Uso de colas en SGE

- **Para utilizar una cola particular solo es necesario indicarlo durante la llamada a "qsub" mediante el parámetro '-q' (también se puede añadir dentro del script):**

  ```
  % qsub -q amd.q simple.sh
  ```

  – En este caso ejecutamos el script simple.sh' usando la cola 'amd.q'

- **Si hacemos "qstat" podemos comprobar que la tarea se está ejecutando en la cola 'amd.q' como indica el campo queue:**

```
% qstat
job-ID prior name user state submit/start at queue slots ja-task-ID
-----------------------------------------------------------------------------
92 0.55500 simple.sh pepe r 03/30/2010 15:05:53 amd.q@compute-0-5.local 2
```

PRACTICA 3

Usar hwloc en el clúster mediante SGE

- Para comprobar la topología de los equipos del cluster mediante hwloc (lstopo) se puede usar este script:

```
#!/bin/bash
#
#$ -S /bin/bash
#$ -cwd
#$ -o salida.out
#$ -j y

# Anadir hwloc al path
export PATH=$PATH:/share/apps/tools/hwloc/bin/
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/share/apps/tools/hwloc/lib64/
# Ejecutar lstopo con salida en formato png (genera un fichero con la imagen)
lstopo --output-format png > figure.png
```

Usar hwloc en el clúster mediante SGE

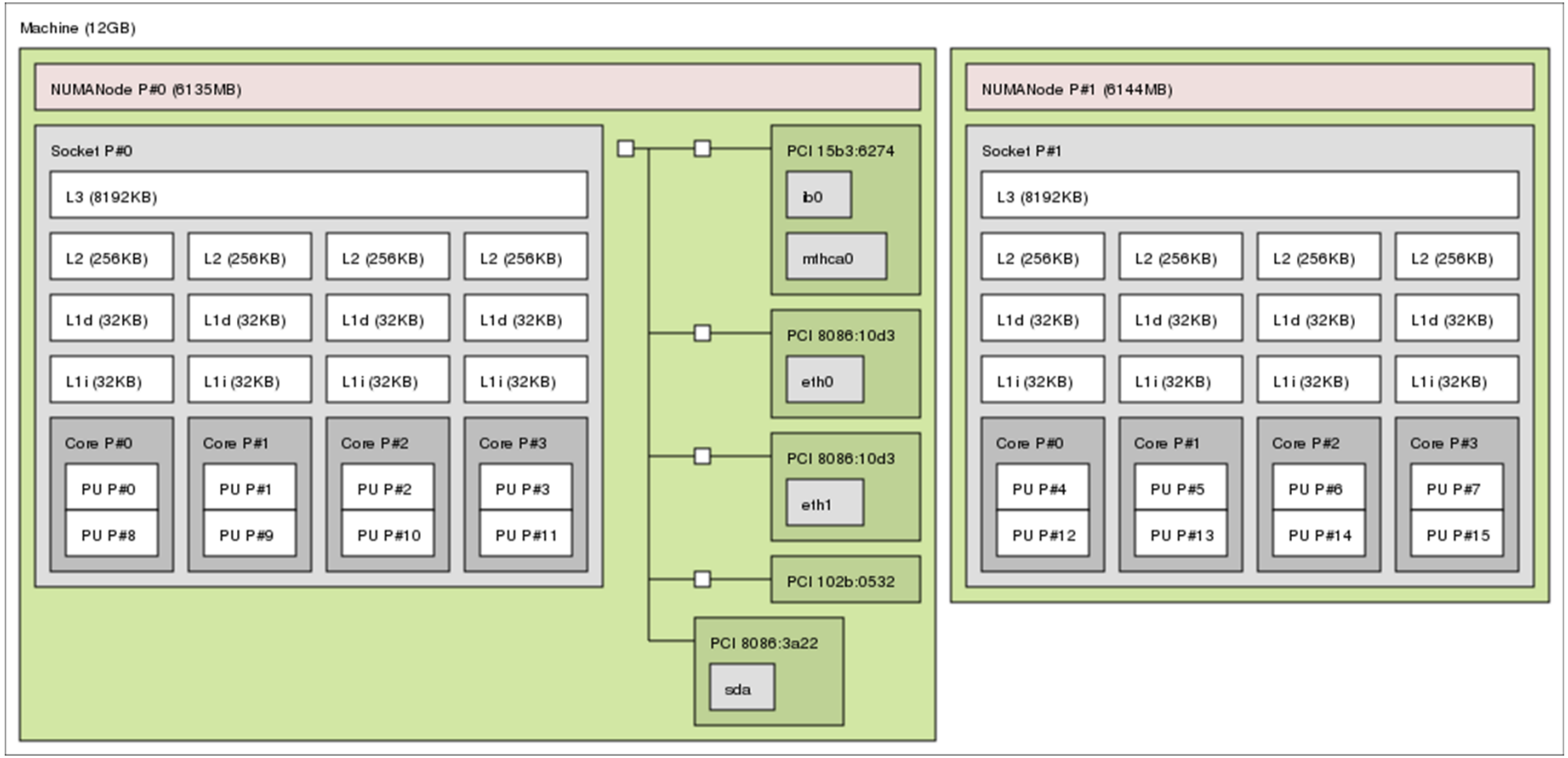
- Y se envía mediante ‘qsub’:

```
% qsub lstopo_cluster.sh
Your job 90 ("lstopo_cluster.sh") has been submitted
```

- Si no se especifica la cola, se obtendrá la topología del equipo seleccionado por SGE

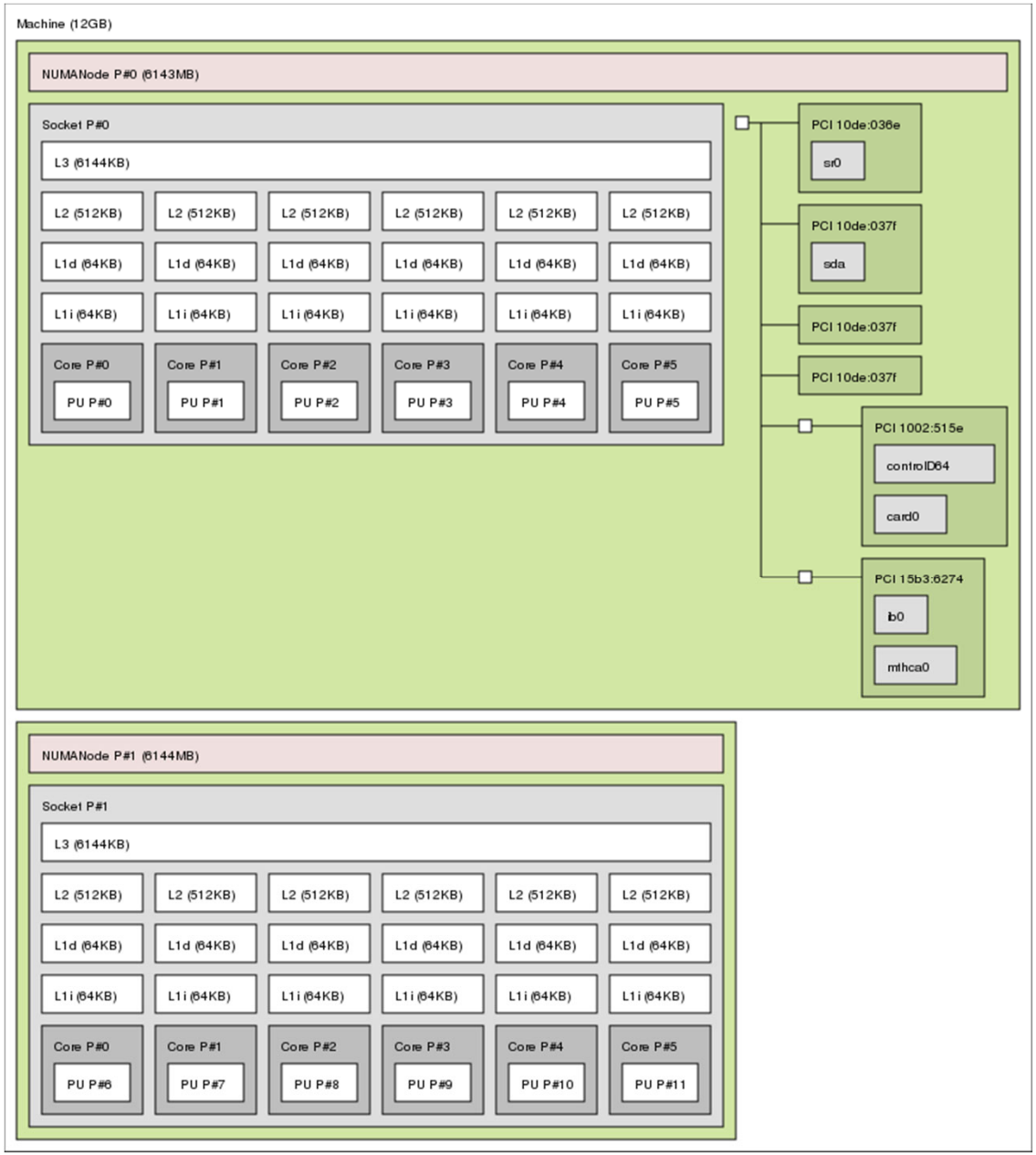
- IMPORTANTE: Hay que usar el mismo tipo de equipo para la ejecución de las prácticas (Intel, AMD, MV), para interpretar adecuadamente los resultados y compararlos.

Usar hwloc en el clúster mediante SGE



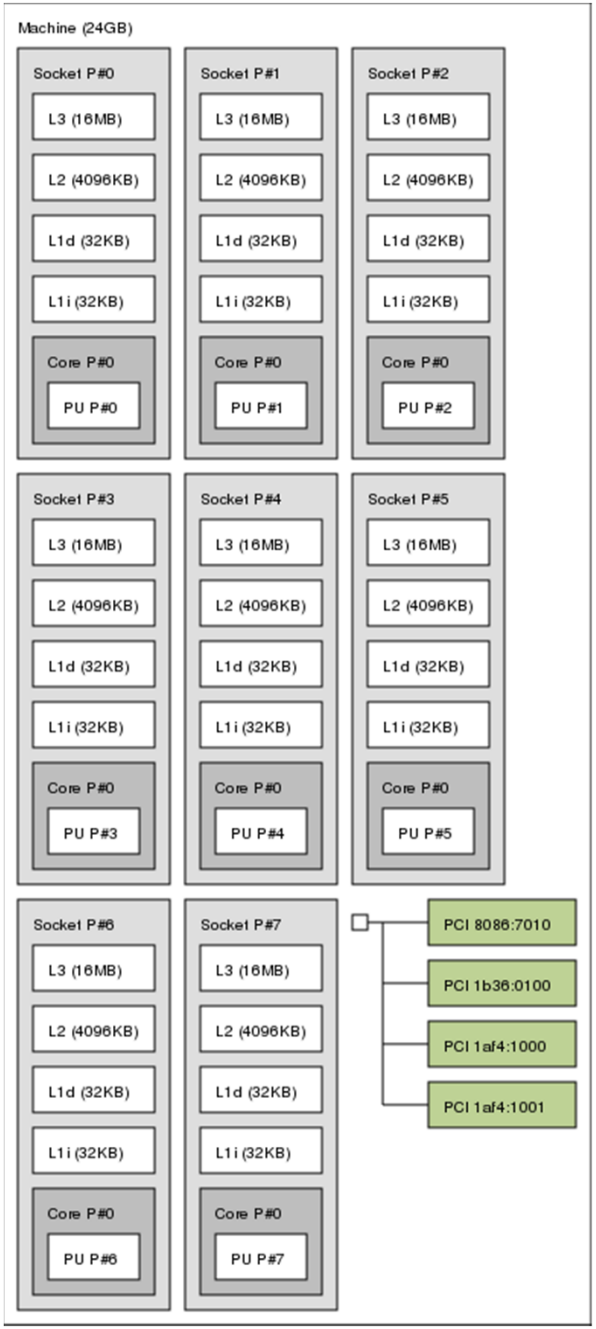

```
# qsub -q intel.q lstopo_cluster.sh
```

Usar hwloc en el clúster mediante SGE




```
# qsub -q amd.q lstopo_cluster.sh
```

Usar hwloc en el clúster mediante SGE




```
# qsub -q mv.q lstopo_cluster.sh
```

Usar Valgrind y Gnuplot en el clúster mediante SGE

- **Para lanzar tareas con valgrind y gnuplot es necesario añadir al path ambos programas:**

```
#!/bin/bash
#
#$ -S /bin/bash
#$ -cwd
#$ -o salida.out
#$ -j y

# Anadir valgrind y gnuplot al path
export PATH=$PATH:/share/apps/tools/valgrind/bin:/share/apps/tools/gnuplot/bin
# Indicar ruta librerías valgrind
export VALGRIND_LIB=/share/apps/tools/valgrind/lib/valgrind
# A partir de aqui tu código…
```

Usar Valgrind y Gnuplot en el clúster mediante SGE

- **Script para lanzar valgrind:**

```
#!/bin/bash
#
#$ -S /bin/bash
#$ -cwd
#$ -o salida.out
#$ -j y

# Anadir valgrind y gnuplot al path
export PATH=$PATH:/share/apps/tools/valgrind/bin:/share/apps/tools/gnuplot/bin
# Indicar ruta librerías valgrind
export VALGRIND_LIB=/share/apps/tools/valgrind/lib/valgrind

# Ejecutamos valgrind
valgrind --tool=cachegrind --cachegrind-out-file=out.dat ./fast 100
cg_annotate out.dat > salida_fast.txt
```

Usar Valgrind y Gnuplot en el clúster mediante SGE

- **Un script para lanzar vuestros scripts “tal cual” en el cluster:**

```
#!/bin/bash
#
#$ -S /bin/bash
#$ -cwd
#$ -o salida.out
#$ -j y

# Anadir valgrind y gnuplot al path
export PATH=$PATH:/share/apps/tools/valgrind/bin:/share/apps/tools/gnuplot/bin
# Indicar ruta librerías valgrind
export VALGRIND_LIB=/share/apps/tools/valgrind/lib/valgrind

# Pasamos el nombre del script como parámetro
echo “Ejecutando script $1…”
echo “”
source $1
```

PRACTICA 4

Enviando tareas OpenMP a SGE

- **Para lanzar tareas OpenMP es necesario utilizar el entorno paralelo 'openmp'.**
  - Se asegura que todos los procesos se ejecutan en la misma máquina, requisito imprescindible para que OpenMP funcione correctamente
  - El entorno paralelo 'openmp' indica a SGE que debe buscar una máquina que tenga disponibles el mismo número de procesadores/cores como procesos se indiquen a la hora de lanzar la tarea

Enviando tareas OpenMP a SGE

- **Un requisito importante a la hora de lanzar tareas OpenMP es conocer de antemano el número de cores disponibles en las máquinas del cluster**
  - Por ejemplo, en el cluster Labomat el número máximo de procesos que se pueden lanzar en una tarea OpenMP es 16, ya que es el número máximo de cores disponibles en los equipos con más cores
    - Intel con dos procesadores → 8 cores por procesador con Hyper-Threading * 2 procesadores = 16 cores
    - AMD con dos procesadores → 6 cores por procesador * 2 procesadores = 12 cores
    - Máquinas virtuales → 8 cores

Enviando tareas OpenMP a SGE

- **Un ejemplo de script para OpenMP es el siguiente:**

```
#!/bin/bash
#
#$ -cwd
#$ -j y
#$ -S /bin/bash
#$ -pe openmp 2
#
export OMP_NUM_THREADS=$NSLOTS
./my_prog
```

  - El número de procesos solicitados (NSLOTS) se asigna a la variable de entorno OMP_NUM_THREADS, que será usada por OpenMP

Enviando tareas OpenMP a SGE

```
% qsub -pe openmp 4 run_openmp.sh
% qstat
job-ID prior name user state submit/start at queue slots ja-task-ID
-----------------------------------------------------------------------
94 0.00000 run_openmp ivangm qw 03/30/2010 15:25:25 4
```

- Al lanzar el script 'run_openmp.sh' se han solicitado 4 procesos usando el entorno paralelo 'openmp'