

Arquitectura de Computadores (AC)

Cuaderno de prácticas.

Bloque Práctico 0. Entorno de programación

Estudiante (nombre y apellidos): Jose Luis Martínez Ortiz

Grupo de prácticas: C3

Fecha de entrega: 1 – 03 – 2016

Fecha evaluación en clase: 9 – 03 – 2016

Ejercicios basados en los ejemplos del seminario práctico

1. En el primer ejemplo de ejecución en atcgrid usando TORQUE se ejecuta el ejemplo HelloOMP.c usando la siguiente orden: `echo 'hello/HelloOMP' | qsub -q ac`. El resultado de la ejecución de este código en atcgrid se puede ver en el seminario. Conteste a las siguientes preguntas:

- a. ¿Para qué se usa en qsub la opción -q?

RESPUESTA: Para enviarlo a la cola que se indica a continuación

- b. ¿Cómo sabe el usuario que ha terminado la ejecución en atcgrid?

RESPUESTA: Porque con el comando "qstat" no muestra nada y me ha generado 2 ficheros STDIN.oXXX y STDIN.eXXX, donde XXX es el PID del proceso.

- c. ¿Cómo puede saber el usuario si ha habido algún error en la ejecución?

RESPUESTA: Si el tamaño del fichero STDIN.eXXX es mayor que cero.

- d. ¿Cómo ve el usuario el resultado de la ejecución?

RESPUESTA: En el fichero STDIN.oXXX

- e. ¿Por qué en el resultado de la ejecución aparecen 24 saludos "!!!Hello World!!!"?

RESPUESTA: Por que sale un hello World por cada thread del procesador, el procesador es Multicore y multithread, siendo de 6 cores con 2 thread por core * 2 Hyperthreading haciendo un total de 24 threads.

2. En el segundo ejemplo de ejecución en atcgrid usando TORQUE el script script_helloomp.sh usando la siguiente orden: `qsub script_helloomp.sh`. El script ejecuta varias veces el ejecutable del código HelloOMP.c. El resultado de la ejecución de este código en atcgrid se puede ver en el seminario. Conteste a las siguientes preguntas:

- a. ¿Por qué no acompaña a al orden qsub la opción -q en este caso?

RESPUESTA: Por que no hace falta indicarle el destino, al estar indicado en el script

- b. ¿Cuántas veces ejecuta el script el ejecutable HelloOMP en atcgrid? ¿Por qué lo ejecuta ese número de veces?

RESPUESTA: Lo ejecuta 4 veces por que empieza en 12 thread y cada vez que itera divide a la mitad el numero de thread que lanza hasta llegar a 0. Entonces tenemos $12 - 6 - 3 - 1$

- c. ¿Cuántos saludos "!!!Hello World!!!" se imprimen en cada ejecución? (indique el número exacto) ¿Por qué se imprime ese número?

RESPUESTA: 1º ejecución doce veces

2º ejecución seis veces

3º ejecución tres veces.

4º ejecución una vez.

Se imprime ese numero por el numero de thread que ejecuta el script

Adjunto salida del script helloomp.o25250

3. Realizar las siguientes modificaciones en el script “!!!Hello World!!!”:

- Eliminar la variable de entorno \$PBS_O_WORKDIR en el punto en el que aparece.
- Añadir lo necesario para que, cuando se ejecute el script, se imprima la variable de entorno \$PBS_O_WORKDIR.

Ejecutar el script con estas modificaciones. ¿Qué resultados de ejecución se obtienen en este caso? Incorporar en el cuaderno de trabajo volcados de pantalla que muestren estos resultados.

RESPUESTA: Al eliminar la variable \$PBS_O_WORKDIR, que contiene la ruta de trabajo actual, no encuentra el ejecutable HelloOMP y da un error.

Adjunto salidas : helloomp.e25252 y helloomp.o25252

Captura de pantalla : [ejecución](#) [descarga](#).

Resto de ejercicios

4. Incorporar en el fichero .zip que se entregará al profesor el fichero /proc/cpuinfo de alguno de los nodos de atcgrid (atcgrid1, atcgrid2, atcgrid3), y del PC del aula de prácticas o de su PC. Indique qué ha hecho para obtener el contenido de /proc/cpuinfo en atcgrid.

RESPUESTA: [Captura de pantalla](#)

Teniendo en cuenta el contenido de cpuinfo conteste a las siguientes preguntas (justifique las respuestas):

a. ¿Cuántos cores físicos y cuántos cores lógicos tiene el PC del aula de prácticas o su PC?

RESPUESTA: 4 cores fisicos y 8 lógicos. Archivo cpuinfo: cpuinfo_miPC

b. ¿Cuántos cores físicos y cuántos cores lógicos tiene un nodo de atcgrid?

RESPUESTA: 6 cores fisicos y 12 lógicos. Archivo cpuinfo: STDIN.o25258

5. En el Listado 1 se puede ver un código fuente C que calcula la suma de dos vectores y en el Listado 2 una versión con C++:

$v3 = v1 + v2; \quad v3(i) = v1(i) + v2(i), \quad i=0, \dots, N-1$

Los códigos utilizan directivas del compilador para fijar el tipo de variable de los vectores (v1, v2 y v3). En los comentarios que hay al principio de los códigos se indica cómo hay que compilarlos. Los vectores pueden ser:

- Variables locales: descomentando en el código #define VECTOR_LOCAL y comentando #define VECTOR_GLOBAL y #define VECTOR_DYNAMIC
- Variables globales: descomentando #define VECTOR_GLOBAL y comentando #define VECTOR_LOCAL y #define VECTOR_DYNAMIC
- Variables dinámicas: descomentando #define VECTOR_DYNAMIC y comentando #define VECTOR_LOCAL y #define VECTOR_GLOBAL. Si se usan los códigos tal y

como están en Listado 1 y Listado 2, sin hacer ningún cambio, los vectores (v1, v2 y v3) serán variables dinámicas.

Por tanto, se debe definir sólo una de las siguientes constantes: VECTOR_LOCAL, VECTOR_GLOBAL o VECTOR_DYNAMIC.

- a. En los dos códigos (Listado 1 y Listado 2) se utiliza la función `clock_gettime()` para obtener el tiempo de ejecución del trozo de código que calcula la suma de vectores. En el código se imprime la variable `ncgt`, ¿qué contiene esta variable? ¿qué información devuelve exactamente la función `clock_gettime()`? ¿en qué estructura de datos devuelve `clock_gettime()` la información (indicar el tipo de estructura de datos y describir la estructura de datos)?

RESPUESTA:

- b. Escribir en el cuaderno de prácticas las diferencias que hay entre el código fuente C y el código fuente C++ para la suma de vectores.

RESPUESTA:

Descripción diferencia	En C	En C++

6. Generar el ejecutable del código fuente C del Listado 1 para vectores locales (para ello antes de compilar debe descomentar la definición de VECTOR_LOCAL y comentar las definiciones de VECTOR_GLOBAL y VECTOR_DYNAMIC). Ejecutar el código ejecutable resultante en atcgrid usando el la cola TORQUE. Incorporar volcados de pantalla que demuestren la ejecución correcta en atcgrid.

RESPUESTA:

7. Ejecutar en atcgrid el código generado en el apartado anterior usando el script del Listado 3. Generar el ejecutable usando la opción de optimización `-O2` tal y como se indica en el comentario que hay al principio del programa. Ejecutar el código también en su PC local para los mismos tamaños. ¿Se obtiene error para alguno de los tamaños? En caso afirmativo, ¿a qué se debe este error?

RESPUESTA:

8. Generar los ejecutables del código fuente C para vectores globales y para dinámicos. Genere el ejecutable usando `-O2`. Ejecutar los dos códigos en atcgrid usando un script como el del Listado 3 (hay que poner en el script el nombre de los ficheros ejecutables generados en este ejercicio) para el mismo rango de tamaños utilizado en el ejercicio anterior. Ejecutar también los códigos en su PC local. ¿Se obtiene error usando vectores globales o dinámicos? ¿A qué cree que es debido?

RESPUESTA:

9. Rellenar una tabla como la Tabla 1 para atcgrid y otra para el PC local con los tiempos de ejecución obtenidos en los ejercicios anteriores para el trozo de código que realiza la suma de

vectores. En la columna “Bytes de un vector” hay que poner el total de bytes reservado para un vector. Ayudándose de una hoja de cálculo represente en una misma gráfica los tiempos de ejecución obtenidos en atcgrid para vectores locales, globales y dinámicos (eje y) en función del tamaño en bytes de un vector (eje x). Utilice escala logarítmica en el eje de ordenadas (eje y) en todas las gráficas. ¿Hay diferencias en los tiempos de ejecución con vectores locales, globales y dinámicos?

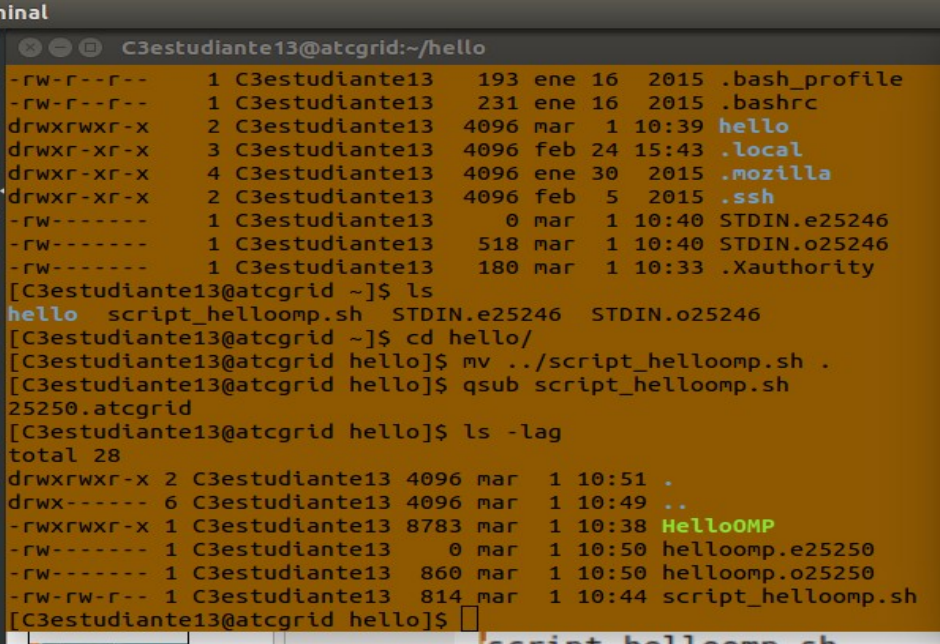
RESPUESTA:

Tabla 1 . Tiempos de ejecución de la suma de vectores para vectores locales, globales y dinámicos

	Nº de Componentes	Bytes de un vector	Tiempo para vect. locales	Tiempo para vect. globales	Tiempo para vect. dinámicos
10	65536				
va	131072				
(1	262144				
R	524288				
	1048576				
R	2097152				
	4194304				
	8388608				
	16777216				
	33554432				
	67108864				

Anexo I : Capturas de pantalla.

Ejecución del script “script_helloomp.sh” en el servidor.



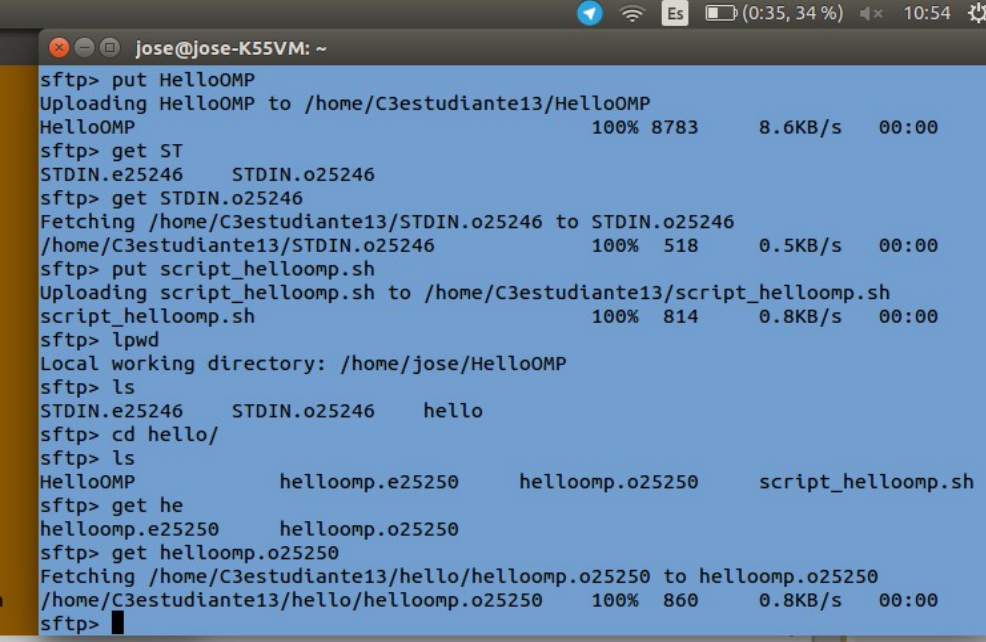
```

C3estudiante13@atcgrid:~/hello
-rw-r--r-- 1 C3estudiante13 193 ene 16 2015 .bash_profile
-rw-r--r-- 1 C3estudiante13 231 ene 16 2015 .bashrc
drwxrwxr-x 2 C3estudiante13 4096 mar 1 10:39 hello
drwxr-xr-x 3 C3estudiante13 4096 feb 24 15:43 .local
drwxr-xr-x 4 C3estudiante13 4096 ene 30 2015 .mozilla
drwxr-xr-x 2 C3estudiante13 4096 feb 5 2015 .ssh
-rw----- 1 C3estudiante13 0 mar 1 10:40 STDIN.e25246
-rw----- 1 C3estudiante13 518 mar 1 10:40 STDIN.o25246
-rw----- 1 C3estudiante13 180 mar 1 10:33 .Xauthority
[C3estudiante13@atcgrid ~]$ ls
hello script_helloomp.sh STDIN.e25246 STDIN.o25246
[C3estudiante13@atcgrid ~]$ cd hello/
[C3estudiante13@atcgrid hello]$ mv ../script_helloomp.sh .
[C3estudiante13@atcgrid hello]$ qsub script_helloomp.sh
25250.atcgrid
[C3estudiante13@atcgrid hello]$ ls -lag
total 28
drwxrwxr-x 2 C3estudiante13 4096 mar 1 10:51 .
drwx----- 6 C3estudiante13 4096 mar 1 10:49 ..
-rwxrwxr-x 1 C3estudiante13 8783 mar 1 10:38 HelloOMP
-rw----- 1 C3estudiante13 0 mar 1 10:50 helloomp.e25250
-rw----- 1 C3estudiante13 860 mar 1 10:50 helloomp.o25250
-rw-rw-r-- 1 C3estudiante13 814 mar 1 10:44 script_helloomp.sh
[C3estudiante13@atcgrid hello]$

```

[Volver](#)

Descarga de la salida del script “script_helloomp.sh”.



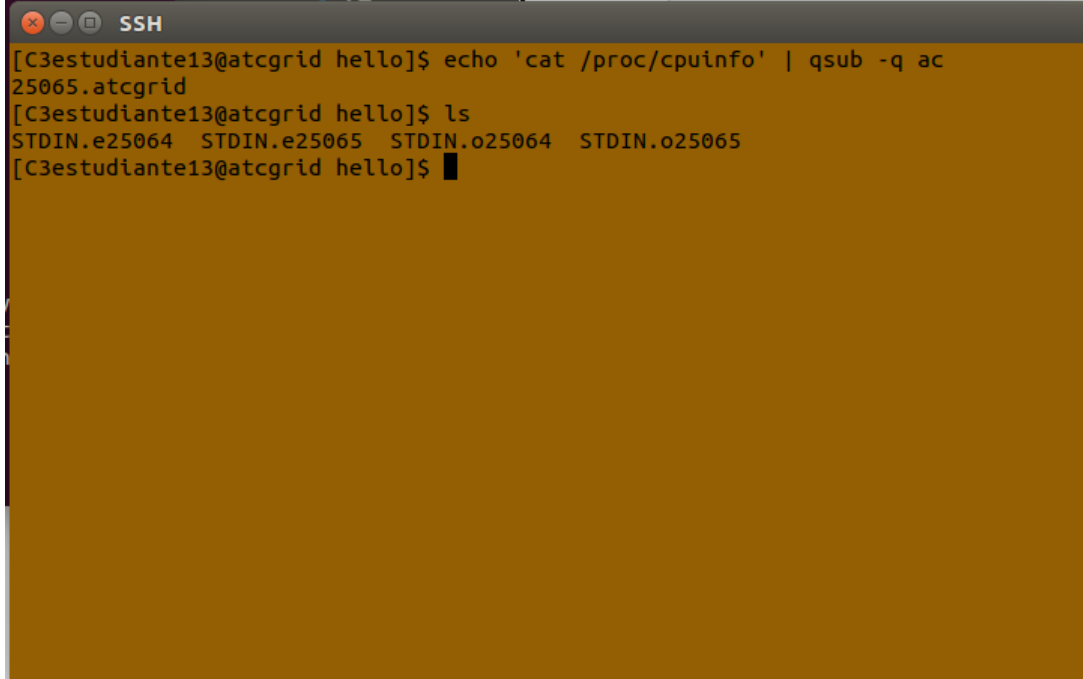
```

jose@jose-K55VM: ~
sftp> put HelloOMP
Uploading HelloOMP to /home/C3estudiante13/HelloOMP
HelloOMP                                100% 8783      8.6KB/s   00:00
sftp> get ST
STDIN.e25246 STDIN.o25246
sftp> get STDIN.o25246
Fetching /home/C3estudiante13/STDIN.o25246 to STDIN.o25246
/home/C3estudiante13/STDIN.o25246      100% 518      0.5KB/s   00:00
sftp> put script_helloomp.sh
Uploading script_helloomp.sh to /home/C3estudiante13/script_helloomp.sh
script_helloomp.sh                    100% 814      0.8KB/s   00:00
sftp> lpwd
Local working directory: /home/jose/HelloOMP
sftp> ls
STDIN.e25246 STDIN.o25246 hello
sftp> cd hello/
sftp> ls
HelloOMP          helloomp.e25250  helloomp.o25250  script_helloomp.sh
sftp> get he
helloomp.e25250  helloomp.o25250
sftp> get helloomp.o25250
Fetching /home/C3estudiante13/hello/helloomp.o25250 to helloomp.o25250
/home/C3estudiante13/hello/helloomp.o25250 100% 860      0.8KB/s   00:00
sftp>

```

[Volver](#)

Como obtener la información del procesador del servidor:



```
SSH
[C3estudiante13@atcgrid hello]$ echo 'cat /proc/cpuinfo' | qsub -q ac
25065.atcgrid
[C3estudiante13@atcgrid hello]$ ls
STDIN.e25064  STDIN.e25065  STDIN.o25064  STDIN.o25065
[C3estudiante13@atcgrid hello]$
```

[Volver.](#)