

Arquitectura de Computadores (AC)

Cuaderno de prácticas.

Bloque Práctico 0. Entorno de programación

Estudiante (nombre y apellidos): Antonio Martín Ruiz

Grupo de prácticas:2

Fecha de entrega: 9/03/2017

Fecha evaluación en clase: 10/03/2017

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: Se usa para definir el destino del trabajo. El destino del trabajo puede ser una cola, un servidor o una cola en un servidor. En este caso se trata de la cola ac.

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

RESPUESTA: Se puede usar qstat para comprobar si se ha completado la ejecución. Se habrá completado si el proceso aparece en la lista con una C en la columna S (complete en la columna state) o no aparece en la lista.

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

RESPUESTA: Consultado el archivo generado tras la ejecución cuyo nombre cuya extensión empieza por ".e".

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

RESPUESTA: Consultado el archivo generado tras la ejecución cuyo nombre cuya extensión empieza por ".o".

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

RESPUESTA: Porque tenemos un total de 24 procesadores lógicos (2 procesadores físicos x 6 cores cada procesador x 2 threads cada núcleo debido al hyperthreading) y se muestra un saludo por cada thread.

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: Porque ya se indica en el propio script en la línea `#PBS -q ac`

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

RESPUESTA: Lo ejecuta cuatro veces, ya que el bucle lo va ejecutando para la mitad con división entera de threads cada vez desde doce. Por lo tanto en la primera iteración lo ejecuta para 12 y divide este número entre 2, en la segunda para 6, en la tercera para 3, la cuarta para 1 thread y la quinta llegaría al tope del bucle, puesto que $\frac{1}{2} = 0$ con la división entera.

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

RESPUESTA: Por lo anteriormente explicado, en la primera ejecución se usan 12 threads y por tanto se imprimen 12 saludos, en la segunda 6 threads y por lo tanto 6 saludos, en la tercera 3 threads y por lo tanto 3 saludos, y en la cuarta 1 thread y por lo tanto 1 saludo.

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:

The image contains two terminal screenshots. The top screenshot shows a user named 'amrantonio' at 'HAL9000' using 'sftp' to transfer files. It shows the upload of 'script_helloomp.sh' and 'HelloOMP.c' to 'helloomp.e43340' and 'helloomp.e43340'. It also shows the download of 'helloomp.e43351' and 'helloomp.e43351' from 'helloomp.e43351'. The bottom screenshot shows a user named 'Eiestudiante13' at 'atcgird' in a directory named 'hello'. It shows a list of files and directories, including 'script_helloomp.sh', 'HelloOMP', and 'script_helloomp.sh'. It also shows the execution of 'qsub script_helloomp.sh' and the resulting output, which includes a list of student names and IDs.

```

amrantonio@HAL9000: ~
sftp> ls
HelloOMP
sftp> rm script_helloomp.sh
Removing /home/Eiestudiante13/hello/script_helloomp.sh
sftp> ls
HelloOMP
sftp> ll
AC_practica_optLP_2016_2017.pdf AC_seminario0_Torque.pdf cpuinfo(atcgrip) HelloOMP helloomp.e43340 script_helloomp.sh
AC_Practicas_pp_2016_17.pdf BP0_MartinRuizAntonio_Y.odt cpuinfo(ordenaadorpersonal) HelloOMP.c helloomp.e43340
sftp> put script_helloomp.sh
Uploading script_helloomp.sh to /home/Eiestudiante13/hello/script_helloomp.sh
script_helloomp.sh
sftp> ls
HelloOMP
sftp> get
helloomp.e43351 helloomp.o43351 script_helloomp.sh
HelloOMP
sftp> get h
helloomp.e43351 helloomp.o43351
sftp> get helloomp.*
Fetching /home/Eiestudiante13/hello/helloomp.e43351 to helloomp.e43351
/home/Eiestudiante13/hello/helloomp.e43351
100% 376 0.4KB/s 00:00
Fetching /home/Eiestudiante13/hello/helloomp.o43351 to helloomp.o43351
/home/Eiestudiante13/hello/helloomp.o43351
100% 528 0.5KB/s 00:00
sftp> ls
HelloOMP
sftp> rm he
helloomp.e43351 helloomp.o43351
sftp> rm helloomp.*
Removing /home/Eiestudiante13/hello/helloomp.e43351
Removing /home/Eiestudiante13/hello/helloomp.o43351
sftp> rm script_helloomp.sh
Removing /home/Eiestudiante13/hello/script_helloomp.sh
sftp> put script_helloomp.sh
Uploading script_helloomp.sh to /home/Eiestudiante13/hello/script_helloomp.sh
script_helloomp.sh
100% 864 0.8KB/s 00:00
sftp> get he
helloomp.e43356 helloomp.o43356
sftp> get helloomp.*
Fetching /home/Eiestudiante13/hello/helloomp.e43356 to helloomp.e43356
/home/Eiestudiante13/hello/helloomp.e43356
100% 376 0.4KB/s 00:00
Fetching /home/Eiestudiante13/hello/helloomp.o43356 to helloomp.o43356
/home/Eiestudiante13/hello/helloomp.o43356
100% 554 0.5KB/s 00:00
sftp>

Eiestudiante13@atcgird:~/hello
acap21 Ciestudiante26 D2estudiante29 E3estudiante6
acap22 Ciestudiante27 D2estudiante3 E3estudiante7
acap23 Ciestudiante28 D2estudiante30 E3estudiante8
acap24 Ciestudiante29 D2estudiante4 E3estudiante9
acap25 Ciestudiante3 D2estudiante5 efernandez
acap26 Ciestudiante30 D2estudiante6 filleras
acap27 Ciestudiante4 D2estudiante7 fortune
acap28 Ciestudiante5 D2estudiante8 gustavo
acap29 Ciestudiante6 D2estudiante9 jdlaz
acap3 Ciestudiante7 D3estudiante1 Jesus
acap30 Ciestudiante8 D3estudiante10 jherra
acap4 Ciestudiante9 D3estudiante11 jortega
acap5 C2estudiante1 D3estudiante12 jperez
acap6 C2estudiante10 D3estudiante13 lost+found
acap7 C2estudiante11 D3estudiante14 nancia
acap8 C2estudiante12 D3estudiante15 ngarenas
acap9 C2estudiante13 D3estudiante16 oresti
Biestudiante1 C2estudiante14 D3estudiante17 original_fortuno
Biestudiante10 C2estudiante15 D3estudiante18 practicas
Biestudiante11 C2estudiante16 D3estudiante19 probador
Biestudiante12 C2estudiante17 D3estudiante2
Biestudiante13 C2estudiante18 D3estudiante20
[Eiestudiante13@atcgird home]$ cd Eiestudiante13
[Eiestudiante13@atcgird ~]$ ls
hello
[Eiestudiante13@atcgird ~]$ cd hello
[Eiestudiante13@atcgird hello]$ ls
HelloOMP script_helloomp.sh
[Eiestudiante13@atcgird hello]$ qsub
HelloOMP script_helloomp.sh
[Eiestudiante13@atcgird hello]$ qsub script_helloomp.sh
43336.atcgird
[Eiestudiante13@atcgird hello]$ ls
HelloOMP helloomp.e43336 helloomp.o43336 script_helloomp.sh
[Eiestudiante13@atcgird hello]$ qsub script_helloomp.sh
43340.atcgird
[Eiestudiante13@atcgird hello]$ qsub script_helloomp.sh
43351.atcgird
[Eiestudiante13@atcgird hello]$ ls
HelloOMP helloomp.e43351 helloomp.o43351 script_helloomp.sh
[Eiestudiante13@atcgird hello]$ qsub script_helloomp.sh
43356.atcgird
[Eiestudiante13@atcgird hello]$

```

[illegible]

```

amrantonio@HAL9000: ~/DGIIM/2DGIIM/AC/Prácticas/Práctica1
$ ./var/lib/torque/mon_priv/jobs/43356.atcgrid.SC: line 24: /HelloOMP: No such file or directory
amrantonio@HAL9000:~/DGIIM/2DGIIM/AC/Prácticas/Práctica1$ cat helloomp.o43356
Id. usuario del trabajo: Eiestudiante13
Id. del trabajo: 43356.atcgrid
Nombre del trabajo especificado por usuario: helloomp
Nodo que ejecuta qsub: atcgrid
cola: ac
Directorio de trabajo: /home/Eiestudiante13/hello
Nodos asignados al trabajo:
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
atcgrid1
No de threads inicial: 12

Para 12 threads:
Para 6 threads:
Para 3 threads:
Para 1 threads:
amrantonio@HAL9000:~/DGIIM/2DGIIM/AC/Prácticas/Práctica1$
```

Como podemos ver obtenemos un error en el archivo .e debido a que no se especifica el directorio en el que se encuentra el ejecutable. Para cada ejecución del mismo se obtiene el mismo error. Al imprimir la variable podemos ver que este directorio es /home/E1estudiante13/hello.

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: Para obtenerlo he realizado la orden `echo 'cat /proc/cpuinfo' | qsub -q ac`. El contenido del archivo se encuentra en el archivo de salida (.o) generado por la orden.

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: Mi PC tiene 2 cores físicos (core id 0-1) y 4 lógicos (processor 0-3).

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

RESPUESTA: atcgrid tiene 12 cores físicos (6 cpu cores x 2 chips de procesamiento) y 24 lógicos (processor 0-23).

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; v3(i) = v1(i) + v2(i), i=0,...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: La función devuelve el instante de tiempo en el que se llama, en una estructura de datos `timespec` definida en la librería `time.h` de la siguiente manera:

```
struct timespec{
time_t tv_sec //Número de segundos enteros
long tv_nsec //Nanosegundos
}
```

En la variable `ncgt` se almacena la diferencia entre dos instantes de tiempo, cuando se comienza la suma de los vectores y cuando se finaliza, por lo que lo que obtenemos es el tiempo que se tarda en realizar la suma.

- 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++
Bibliotecas	<code>#include <stdlib.h></code> <code>#include <stdio.h></code>	<code>#include <cstdlib></code> <code>#include <iostream></code>
Namespace	No se utiliza	<code>using namespace std;</code>
Reserva de espacio	<code>malloc()</code>	<code>new double []</code>
Liberación de espacio	<code>free()</code>	<code>delete[]</code>
Salida de datos	<code>printf()</code>	<code>cout <<</code>

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:

```
Eiestudiante13@atcgird:~/hello
siblings      : 12
core id       : 10
cpu cores     : 6
apicid        : 53
initial apicid : 53
fpu           : yes
fpu_exception : yes
cpuid level   : 11
wp            : yes
flags         : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe sysca
ll nx pdpe1gb rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni dtes64 monitor ds_cpl vm
x smx est tm2 ssse3 cx16 xtpr pdcm pcd dca sse4_1 sse4_2 popcnt lahf_lm epb tpr_shadow vnml flexpriority ept vpid dtherm ida arat
bugs          :
bogomips      : 4799.87
clflush size  : 64
cache_alignm  : 64
address sizes : 40 bits physical, 48 bits virtual
power managem:

[Eiestudiante13@atcgird hello]$ ls
HelloOMP      helloomp.o43356      STDIN.e43369      STDIN.o43369      SumaVectores.c
helloomp.e43356 script_helloomp.sh  STDIN.e45441      STDIN.o45441
[Eiestudiante13@atcgird hello]$ ls
HelloOMP      helloomp.e43356      helloomp.o43356   script_helloomp.sh SumaVectores.c
[Eiestudiante13@atcgird hello]$ ls
HelloOMP      helloomp.e43356      helloomp.o43356   script_helloomp.sh SumaVectores
[Eiestudiante13@atcgird hello]$ echo 'hello/SumaVectores' | qsub -q ac
45443.atcgird
[Eiestudiante13@atcgird hello]$ qstat
Job ID          Name          User          Time Use S Queue
-----
45443.atcgird   STDIN         Eiestudiante13 00:00:00 C ac
[Eiestudiante13@atcgird hello]$ ls
HelloOMP      helloomp.o43356      STDIN.e45443      SumaVectores
helloomp.e43356 script_helloomp.sh  STDIN.o45443
[Eiestudiante13@atcgird hello]$ cat STDIN.o45443
Faltan no componentes del vector
[Eiestudiante13@atcgird hello]$ echo 'hello/SumaVectores 20' | qsub -q ac
45444.atcgird
[Eiestudiante13@atcgird hello]$ cat STDIN.o45444
Tiempo(seg.):0.000000180 / Tamaño Vectores:20 / V1[0]+V2[0]=V3[0](2.000000+2.000000=4.000000) / /V1[19]+V2[19]=V3[19](3.900000+0.1000
00=4.000000) /
[Eiestudiante13@atcgird hello]$
```

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: En la siguiente captura se aprecia el error que se obtiene y para qué tamaños

[illegible]

Este error es producido por un desbordamiento de la pila, que es donde se encuentran alojados los vectores.

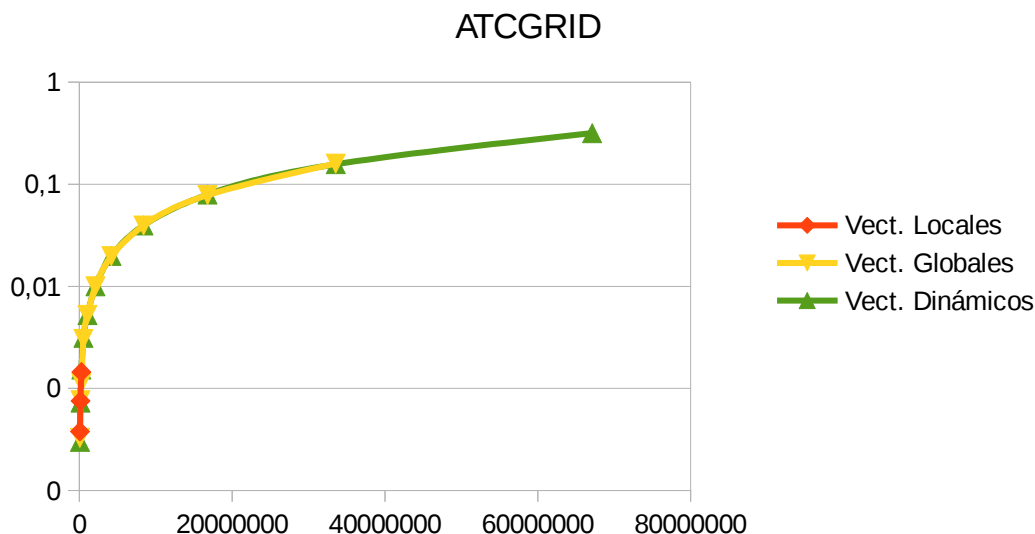
8. Generar los ejecutables del código fuente C para vectores globales y para dinámicos. Genere el ejecutable usando -O2. Ejecutar los dos códigos en atcgrid usando un script como el del Listado 3 (hay que poner en el script el nombre de los ficheros ejecutables generados en este ejercicio) para el mismo 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: No se obtiene error ya que las variables no se almacenan en la pila, sino en el heap y la zona de datos, las cuales toleran tamaños mayores.

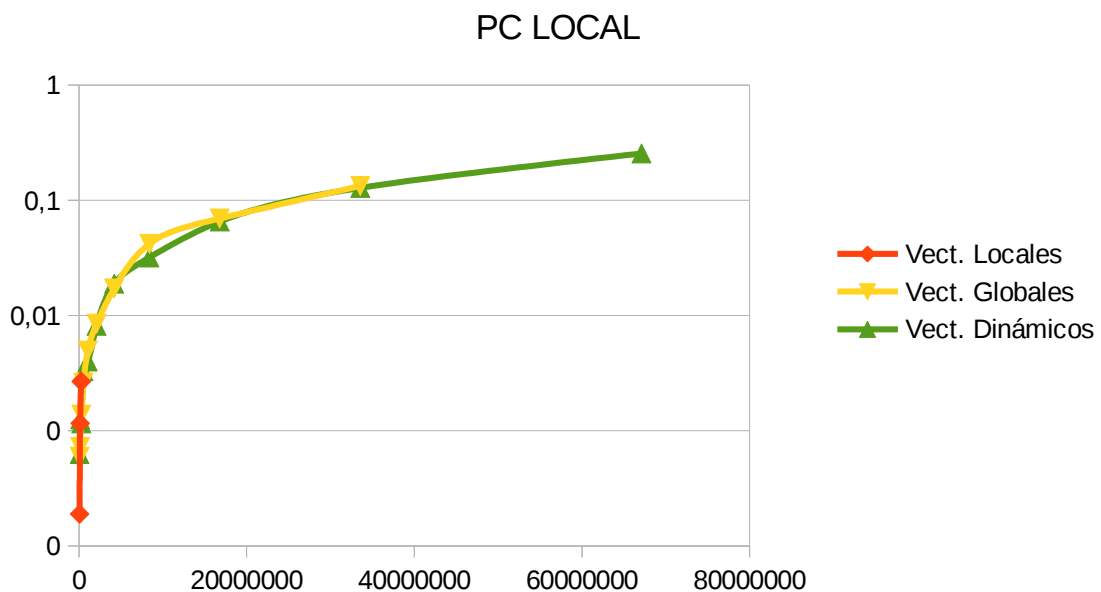
9. Rellenar una tabla como la Tabla 1 para atcgrid y otra para el PC local con los tiempos de ejecución obtenidos en los ejercicios anteriores para el trozo de código que realiza la suma de vectores. En la columna “Bytes de un vector” hay que poner el total de bytes reservado para un vector. Ayudándose 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: Por comodidad he cambiado la tabla que venía dada por la creada en la hoja de cálculo. Los datos son los mismos.

N.º componentes	Bytes/vector	Vect. Locales	Vect. Globales	Vect. Dinámicos
ATCGRID				
65536	524288	0,000377553	0,000327017	0,000299312
131072	1048576	0,000751331	0,00076678	0,00072584
262144	2097152	0,001444985	0,001127927	0,001530293
524288	4194304		0,003058976	0,003124693
1048576	8388608		0,005248386	0,0052086619
2097152	16777216		0,010013525	0,010009522
4194304	33554432		0,019748358	0,019853858
8388608	67108864		0,03939197	0,039543469
16777216	134217728		0,07832079	0,079472318
33554432	268435456		0,159584441	0,157217797
67108864	536870912			0,317581418

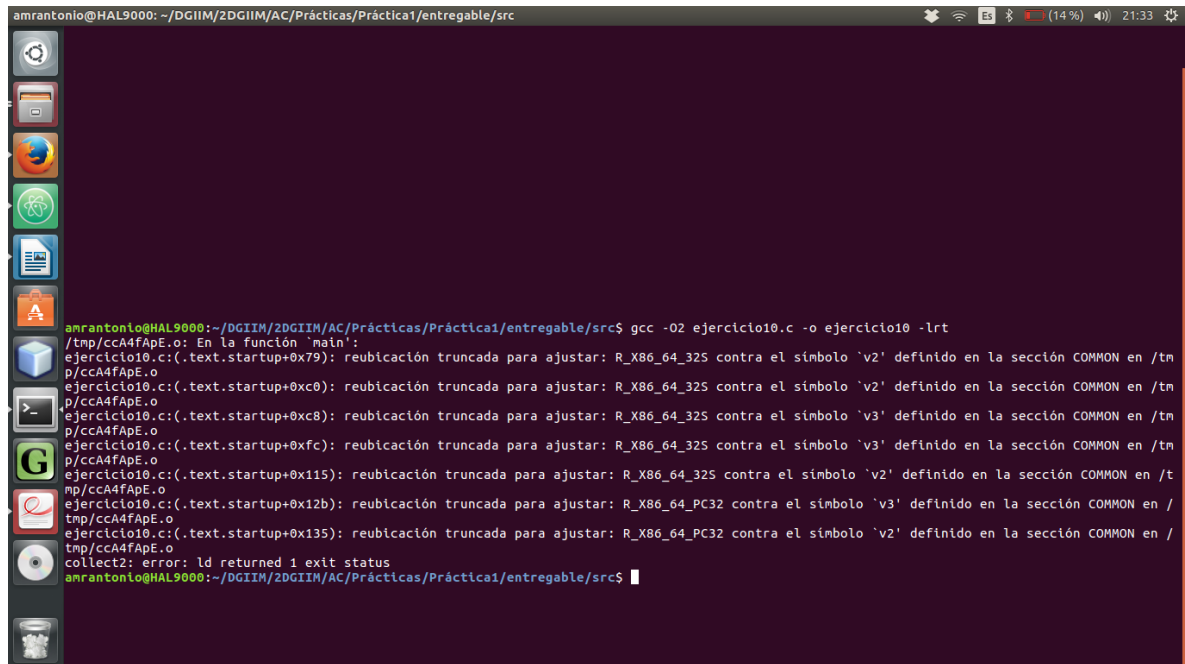


N.º componentes	Bytes/vector	Vect. Locales	Vect. Globales	Vect. Dinámicos
PC LOCAL				
65536	524288	0,000189611	0,000596457	0,000628468
131072	1048576	0,001159934	0,000726018	0,001257487
262144	2097152	0,002679126	0,001382174	0,001156108
524288	4194304		0,002622484	0,003301065
1048576	8388608		0,004970789	0,003996541
2097152	16777216		0,008496987	0,008137907
4194304	33554432		0,017153405	0,01886698
8388608	67108864		0,041946458	0,031856878
16777216	134217728		0,069517418	0,065327979
33554432	268435456		0,133235604	0,128264498
67108864	536870912			0,254480885



10. Modificar el código fuente C para que el límite de los vectores cuando se declaran como variables globales sea igual al máximo número que se puede almacenar en la variable N ($MAX=2^{32}-1$). Generar el ejecutable usando variables globales. ¿Qué ocurre? ¿A qué es debido? Razone además por qué el máximo número que se puede almacenar en N es $2^{32}-1$.

RESPUESTA: Al intentar compilar el programa obtenemos el siguiente error



```
amrantonio@HAL9000: ~/DGIIM/2DGIIM/AC/Prácticas/Práctica1/entregable/src
amrantonio@HAL9000:~/DGIIM/2DGIIM/AC/Prácticas/Práctica1/entregable/src$ gcc -O2 ejercicio10.c -o ejercicio10 -lrt
/tmp/ccA4fApE.o: En la función 'main':
ejercicio10.c:(.text.startup+0x79): reubicación truncada para ajustar: R_X86_64_32S contra el símbolo 'v2' definido en la sección COMMON en /tm
p/ccA4fApE.o
ejercicio10.c:(.text.startup+0xc0): reubicación truncada para ajustar: R_X86_64_32S contra el símbolo 'v2' definido en la sección COMMON en /tm
p/ccA4fApE.o
ejercicio10.c:(.text.startup+0xc8): reubicación truncada para ajustar: R_X86_64_32S contra el símbolo 'v3' definido en la sección COMMON en /tm
p/ccA4fApE.o
ejercicio10.c:(.text.startup+0xfc): reubicación truncada para ajustar: R_X86_64_32S contra el símbolo 'v3' definido en la sección COMMON en /tm
p/ccA4fApE.o
ejercicio10.c:(.text.startup+0x115): reubicación truncada para ajustar: R_X86_64_32S contra el símbolo 'v2' definido en la sección COMMON en /t
mp/ccA4fApE.o
ejercicio10.c:(.text.startup+0x12b): reubicación truncada para ajustar: R_X86_64_PC32 contra el símbolo 'v3' definido en la sección COMMON en /
tmp/ccA4fApE.o
ejercicio10.c:(.text.startup+0x135): reubicación truncada para ajustar: R_X86_64_PC32 contra el símbolo 'v2' definido en la sección COMMON en /
tmp/ccA4fApE.o
collect2: error: ld returned 1 exit status
amrantonio@HAL9000:~/DGIIM/2DGIIM/AC/Prácticas/Práctica1/entregable/src$
```

El error se produce porque excedemos el tamaño permitido para los vectores, por lo que se produciría un desbordamiento, y el compilador lo detecta.

El máximo número que puede almacenar N es ese número ya que es un entero, así que tiene 32 bits, por lo que puede almacenar 2^{32} números distintos, es decir, desde el 0 hasta el $2^{32}-1$.