2º curso / 2º cuatr.
Grado Ing. Inform.
Doble Grado Ing. Inform. y Mat.

# Arquitectura de Computadores (AC)

## Cuaderno de prácticas.
## Bloque Práctico 0. Entorno de programación

Estudiante (nombre y apellidos): Paula Ruiz García
Grupo de prácticas: D1
Fecha de entrega: 4 de marzo de 2018
Fecha evaluación en clase:

---

1. Incorpore volcados de pantalla que muestren lo que devuelve lscpu en atcgrid y en su PC.

**CAPTURAS**: PC:

```
[PaulaRuizGarcia paularg98@ei142091:~] 2018-02-26 lunes
$lscpu
Arquitectura:          x86_64
modo(s) de operación de las CPUs:32-bit, 64-bit
Orden de bytes:        Little Endian
CPU(s):                4
On-line CPU(s) list:   0-3
Hilo(s) de procesamiento por núcleo:1
Núcleo(s) por «socket»:4
Socket(s):             1
Modo(s) NUMA:          1
ID de fabricante:      GenuineIntel
Familia de CPU:        6
Modelo:                58
Model name:            Intel(R) Core(TM) i5-3350P CPU @ 3.10GHz
Revisión:             9
CPU MHz:               1600.140
CPU max MHz:           3300,0000
CPU min MHz:           1600,0000
BogoMIPS:              6186.11
Virtualización:       VT-x
Caché L1d:             32K
Caché L1i:             32K
Caché L2:              256K
Caché L3:              6144K
NUMA node0 CPU(s):     0-3
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pg
e mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe sysca
ll nx rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology
 nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx e
st tm2 ssse3 cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic popcnt tsc_deadline_
timer aes xsave avx f16c rdrand lahf_lm epb tpr_shadow vnmi flexpriority e
pt vpid fsgsbase smep erms xsaveopt dtherm ida arat pln pts
```

Atcgrid:

```
[PaulaRuizGarcia paularg98@ei142091:~/Escritorio/Home/AC-Practicas/P0] 201
8-02-26 lunes
$cat STDIN.o61786
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                24
On-line CPU(s) list:   0-23
Thread(s) per core:    2
Core(s) per socket:    6
Socket(s):             2
NUMA node(s):          2
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 44
Model name:            Intel(R) Xeon(R) CPU           E5645  @ 2.40GHz
Stepping:              2
CPU MHz:               1599.964
CPU max MHz:           2401.0000
CPU min MHz:           1600.0000
BogoMIPS:              4800.14
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              12288K
NUMA node0 CPU(s):     0-5,12-17
NUMA node1 CPU(s):     6-11,18-23
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 syscall
 nx pdpe1gb rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xto
pology nonstop_tsc cpuid aperfmperf pni dtes64 monitor ds_cpl vmx smx est
tm2 ssse3 cx16 xtpr pdcm pcid dca sse4_1 sse4_2 popcnt lahf_lm epb pti ret
poline tpr_shadow vnmi flexpriority ept vpid dtherm ida arat
```

Conteste a las siguientes preguntas:

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

**RESPUESTA**: 4 cores físicos y 4 cores lógicos.

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

**RESPUESTA**: 12 cores físicos y 24 cores lógicos.

**2.** 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**:

ncgt contiene la suma de la resta de los segundos antes y después de calcular la suma de los vectores, y la resta de los nanosegundos antes y después de calcular la suma de los vectores.

clock_gettime devuelve una estructura de datos la cual tiene una variable t_time que devuelve segundos y una variable long donde nos da los nanosegundos.

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++ |
|---|---|---|
| A la hora de mostrar en pantalla usamos printf para C y cout para C++ | printf("Faltan nº componentes del vector\n"); | cout << "Faltan nº componentes del vector\n" << endl ; |
| malloc en C se utiliza para que si no hay espacio disponible se devuelve null, mientras que en C++ new genera una excepción si no hay espacio disponible. | v1 = (double*) malloc(N*sizeof(double)); | v1 = new double [N]; |
| En C free libera el espacio reservado a una variable mientras que delete [] lo hace en C++. | free(v1); | delete [] v1; |

**3.** 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). Incorporar volcados de pantalla que demuestren la ejecución correcta en atcgrid o en su PC.

**RESPUESTA**:

Tiempo(seg.):0.000000255 / Tamaño Vectores:5 / V1[0]+V2[0]=V3[0](0.500000+0.500000=1.000000) V1[4]+V2[4]=V3[4](0.900000+0.100000=1.000000) /

```
[PaulaRuizGarcia paularg98@ei142091:~/Escritorio/Home/AC-Practicas/P0] 2018
-02-26 lunes
$cat STDIN.o61857
Tiempo(seg.):0.000000255         / Tamaño Vectores:5    / V1[0]+V2[0]=V3[0]
(0.500000+0.500000=1.000000) V1[4]+V2[4]=V3[4](0.900000+0.100000=1.000000)
/
sftp> put SumaVectores
Uploading SumaVectores to /home/D1estudiante18/suma/SumaVectores
SumaVectores                              100% 8888     8.7KB/s   00:00
sftp> ls
STDIN.e61854      STDIN.o61854      SumaVectores      SumaVectoresC.c
sftp> get STDIN.o61854
Fetching /home/D1estudiante18/suma/STDIN.o61854 to STDIN.o61854
/home/D1estudiante18/suma/STDIN.o61854   100%   34     0.0KB/s   00:00
sftp> ls
STDIN.e61854      STDIN.e61857      STDIN.o61854      STDIN.o61857
SumaVectores      SumaVectoresC.c
sftp> get STDIN.o61857
Fetching /home/D1estudiante18/suma/STDIN.o61857 to STDIN.o61857
/home/D1estudiante18/suma/STDIN.o61857   100%  143     0.1KB/s   00:00
[D1estudiante18@atcgrid ~]$ mkdir suma
[D1estudiante18@atcgrid ~]$ ls
suma
[D1estudiante18@atcgrid ~]$ cd suma
[D1estudiante18@atcgrid suma]$ ls
SumaVectoresC.c
[D1estudiante18@atcgrid suma]$ echo 'suma/SumaVectores' | qsub -q ac
61854.atcgrid
[D1estudiante18@atcgrid suma]$ qstat
Job ID                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
61854.atcgrid              STDIN            D1estudiante18  00:00:00 C ac

[D1estudiante18@atcgrid suma]$ echo 'suma/SumaVectores 5' | qsub -q ac
61857.atcgrid
[D1estudiante18@atcgrid suma]$ qstat
Job ID                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
61857.atcgrid              STDIN            D1estudiante18  00:00:00 C ac

[D1estudiante18@atcgrid suma]$ qstat
```

**4.** 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 para los mismos tamaños. ¿Se obtiene error para alguno de los tamaños? En caso afirmativo, ¿a qué se debe este error? (Incorporar volcados de pantalla)

**RESPUESTA**: Si se obtiene un error Segmentation fault a partir del tamaño 524288 y ese error se debe a que el tamaño de la pila se ha superado.

PC:

```
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-03 Saturday
$bash SumaVectores.sh
Id. usuario del trabajo:
Id. del trabajo:
Nombre del trabajo especificado por usuario:
Nodo que ejecuta qsub:
Directorio en el que se ha ejecutado qsub:
Cola:
Tiempo(seg.):0.000580800         / Tamaño Vectores:65536        / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200
000) V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tiempo(seg.):0.001232400         / Tamaño Vectores:131072       / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.4
00000) V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tiempo(seg.):0.002516300         / Tamaño Vectores:262144       / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.8
00000) V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
SumaVectores.sh: line 20:   183 Segmentation fault      (core dumped) ./SumaVectoresC $N
SumaVectores.sh: line 20:   184 Segmentation fault      (core dumped) ./SumaVectoresC $N
SumaVectores.sh: line 20:   185 Segmentation fault      (core dumped) ./SumaVectoresC $N
SumaVectores.sh: line 20:   186 Segmentation fault      (core dumped) ./SumaVectoresC $N
SumaVectores.sh: line 20:   187 Segmentation fault      (core dumped) ./SumaVectoresC $N
SumaVectores.sh: line 20:   188 Segmentation fault      (core dumped) ./SumaVectoresC $N
SumaVectores.sh: line 20:   189 Segmentation fault      (core dumped) ./SumaVectoresC $N
SumaVectores.sh: line 20:   190 Segmentation fault      (core dumped) ./SumaVectoresC $N
```

Atcgrid:

```
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-03 Saturday
$cat SumaVectoresC_vlocales.o64095
Id. usuario del trabajo: D1estudiante18
Id. del trabajo: 64095.atcgrid
Nombre del trabajo especificado por usuario: SumaVectoresC_vlocales
Nodo que ejecuta qsub: atcgrid
Directorio en el que se ha ejecutado qsub: /home/D1estudiante18
Cola: ac
Tiempo(seg.):0.000440332         / Tamaño Vectores:65536        / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200
000) V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tiempo(seg.):0.000798097         / Tamaño Vectores:131072       / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.4
00000) V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tiempo(seg.):0.001277184         / Tamaño Vectores:262144       / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.8
00000) V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-03 Saturday
$cat SumaVectoresC_vlocales.e64095
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14758 Segmentation fault      (core dumped) ./SumaVectoresC
$N
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14761 Segmentation fault      (core dumped) ./SumaVectoresC
$N
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14764 Segmentation fault      (core dumped) ./SumaVectoresC
$N
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14771 Segmentation fault      (core dumped) ./SumaVectoresC
$N
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14775 Segmentation fault      (core dumped) ./SumaVectoresC
$N
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14778 Segmentation fault      (core dumped) ./SumaVectoresC
$N
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14781 Segmentation fault      (core dumped) ./SumaVectoresC
$N
/var/lib/torque/mom_priv/jobs/64095.atcgrid.SC: line 20: 14784 Segmentation fault      (core dumped) ./SumaVectoresC
$N
```

**5.** 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. ¿Se obtiene error usando vectores globales o dinámicos? ¿A qué cree que es debido? (Incorporar volcados de pantalla)

**RESPUESTA**: Vectores globales – No se obtiene error

PC:

```
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-04 Sunday
$bash SumaVectores.sh
Id. usuario del trabajo:
Id. del trabajo:
Nombre del trabajo especificado por usuario:
Nodo que ejecuta qsub:
Directorio en el que se ha ejecutado qsub:
Cola:
Tiempo(seg.):0.000586000         / Tamaño Vectores:65536        / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200
000) V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tiempo(seg.):0.001215100         / Tamaño Vectores:131072       / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.4
00000) V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tiempo(seg.):0.002383000         / Tamaño Vectores:262144       / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.8
00000) V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tiempo(seg.):0.004290000         / Tamaño Vectores:524288       / V1[0]+V2[0]=V3[0](52428.800000+52428.800000=104857.
600000) V1[524287]+V2[524287]=V3[524287](104857.500000+0.100000=104857.600000) /
Tiempo(seg.):0.008633000         / Tamaño Vectores:1048576      / V1[0]+V2[0]=V3[0](104857.600000+104857.600000=20971
5.200000) V1[1048575]+V2[1048575]=V3[1048575](209715.100000+0.100000=209715.200000) /
Tiempo(seg.):0.017005400         / Tamaño Vectores:2097152      / V1[0]+V2[0]=V3[0](209715.200000+209715.200000=41943
0.400000) V1[2097151]+V2[2097151]=V3[2097151](419430.300000+0.100000=419430.400000) /
Tiempo(seg.):0.032994900         / Tamaño Vectores:4194304      / V1[0]+V2[0]=V3[0](419430.400000+419430.400000=83886
0.800000) V1[4194303]+V2[4194303]=V3[4194303](838860.700000+0.100000=838860.800000) /
Tiempo(seg.):0.072001000         / Tamaño Vectores:8388608      / V1[0]+V2[0]=V3[0](838860.800000+838860.800000=16777
21.600000) V1[8388607]+V2[8388607]=V3[8388607](1677721.500000+0.100000=1677721.600000) /
Tiempo(seg.):0.152327200         / Tamaño Vectores:16777216     / V1[0]+V2[0]=V3[0](1677721.600000+1677721.600000=335
5443.200000) V1[16777215]+V2[16777215]=V3[16777215](3355443.100000+0.100000=3355443.200000) /
Tiempo(seg.):0.291848000         / Tamaño Vectores:33554432     / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=671
0886.400000) V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
Tiempo(seg.):0.596602900         / Tamaño Vectores:67108864     / V1[0]+V2[0]=V3[0](6710886.400000+6710886.400000=134
21772.800000) V1[67108863]+V2[67108863]=V3[67108863](13421772.700000+0.100000=13421772.800000) /
```

Atcgrid:

```
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-04 Sunday
$cat SumaVectoresC_vlocales.e64261
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-04 Sunday
$cat SumaVectoresC_vlocales.o64261
Id. usuario del trabajo: D1estudiante18
Id. del trabajo: 64261.atcgrid
Nombre del trabajo especificado por usuario: SumaVectoresC_vlocales
Nodo que ejecuta qsub: atcgrid
Directorio en el que se ha ejecutado qsub: /home/D1estudiante18
Cola: ac
Tiempo(seg.):0.000408114         / Tamaño Vectores:65536        / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200
000) V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tiempo(seg.):0.000554910         / Tamaño Vectores:131072       / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.4
00000) V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tiempo(seg.):0.001096058         / Tamaño Vectores:262144       / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.8
00000) V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tiempo(seg.):0.003475463         / Tamaño Vectores:524288       / V1[0]+V2[0]=V3[0](52428.800000+52428.800000=104857.
600000) V1[524287]+V2[524287]=V3[524287](104857.500000+0.100000=104857.600000) /
Tiempo(seg.):0.006015977         / Tamaño Vectores:1048576      / V1[0]+V2[0]=V3[0](104857.600000+104857.600000=20971
5.200000) V1[1048575]+V2[1048575]=V3[1048575](209715.100000+0.100000=209715.200000) /
Tiempo(seg.):0.011873775         / Tamaño Vectores:2097152      / V1[0]+V2[0]=V3[0](209715.200000+209715.200000=41943
0.400000) V1[2097151]+V2[2097151]=V3[2097151](419430.300000+0.100000=419430.400000) /
Tiempo(seg.):0.023631107         / Tamaño Vectores:4194304      / V1[0]+V2[0]=V3[0](419430.400000+419430.400000=83886
0.800000) V1[4194303]+V2[4194303]=V3[4194303](838860.700000+0.100000=838860.800000) /
Tiempo(seg.):0.047657070         / Tamaño Vectores:8388608      / V1[0]+V2[0]=V3[0](838860.800000+838860.800000=16777
21.600000) V1[8388607]+V2[8388607]=V3[8388607](1677721.500000+0.100000=1677721.600000) /
Tiempo(seg.):0.090048842         / Tamaño Vectores:16777216     / V1[0]+V2[0]=V3[0](1677721.600000+1677721.600000=335
5443.200000) V1[16777215]+V2[16777215]=V3[16777215](3355443.100000+0.100000=3355443.200000) /
Tiempo(seg.):0.183794201         / Tamaño Vectores:33554432     / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=671
0886.400000) V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
Tiempo(seg.):0.179216410         / Tamaño Vectores:33554432     / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=671
0886.400000) V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
```

Vectores Dinamicos – No se obtiene error

PC:

```
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-04 Sunday
$bash SumaVectores.sh
Id. usuario del trabajo:
Id. del trabajo:
Nombre del trabajo especificado por usuario:
Nodo que ejecuta qsub:
Directorio en el que se ha ejecutado qsub:
Cola:
Tiempo(seg.):0.000499500         / Tamaño Vectores:65536        / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200
000) V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tiempo(seg.):0.001152400         / Tamaño Vectores:131072       / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.4
00000) V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tiempo(seg.):0.002098600         / Tamaño Vectores:262144       / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.8
00000) V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tiempo(seg.):0.004459200         / Tamaño Vectores:524288       / V1[0]+V2[0]=V3[0](52428.800000+52428.800000=104857.
600000) V1[524287]+V2[524287]=V3[524287](104857.500000+0.100000=104857.600000) /
Tiempo(seg.):0.008614500         / Tamaño Vectores:1048576      / V1[0]+V2[0]=V3[0](104857.600000+104857.600000=20971
5.200000) V1[1048575]+V2[1048575]=V3[1048575](209715.100000+0.100000=209715.200000) /
Tiempo(seg.):0.017263800         / Tamaño Vectores:2097152      / V1[0]+V2[0]=V3[0](209715.200000+209715.200000=41943
0.400000) V1[2097151]+V2[2097151]=V3[2097151](419430.300000+0.100000=419430.400000) /
Tiempo(seg.):0.033152700         / Tamaño Vectores:4194304      / V1[0]+V2[0]=V3[0](419430.400000+419430.400000=83886
0.800000) V1[4194303]+V2[4194303]=V3[4194303](838860.700000+0.100000=838860.800000) /
Tiempo(seg.):0.065044000         / Tamaño Vectores:8388608      / V1[0]+V2[0]=V3[0](838860.800000+838860.800000=16777
21.600000) V1[8388607]+V2[8388607]=V3[8388607](1677721.500000+0.100000=1677721.600000) /
Tiempo(seg.):0.129520600         / Tamaño Vectores:16777216     / V1[0]+V2[0]=V3[0](1677721.600000+1677721.600000=335
5443.200000) V1[16777215]+V2[16777215]=V3[16777215](3355443.100000+0.100000=3355443.200000) /
Tiempo(seg.):0.262842300         / Tamaño Vectores:33554432     / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=671
0886.400000) V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
Tiempo(seg.):0.607033800         / Tamaño Vectores:67108864     / V1[0]+V2[0]=V3[0](6710886.400000+6710886.400000=134
21772.800000) V1[67108863]+V2[67108863]=V3[67108863](13421772.700000+0.100000=13421772.800000) /
```

Atcgrid:

```
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-04 Sunday
$cat SumaVectoresC_vlocales.o64265
Id. usuario del trabajo: D1estudiante18
Id. del trabajo: 64265.atcgrid
Nombre del trabajo especificado por usuario: SumaVectoresC_vlocales
Nodo que ejecuta qsub: atcgrid
Directorio en el que se ha ejecutado qsub: /home/D1estudiante18
Cola: ac
Tiempo(seg.):0.000418541         / Tamaño Vectores:65536        / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200
000) V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tiempo(seg.):0.000846559         / Tamaño Vectores:131072       / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.4
00000) V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tiempo(seg.):0.001080770         / Tamaño Vectores:262144       / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.8
00000) V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tiempo(seg.):0.002961303         / Tamaño Vectores:524288       / V1[0]+V2[0]=V3[0](52428.800000+52428.800000=104857.
600000) V1[524287]+V2[524287]=V3[524287](104857.500000+0.100000=104857.600000) /
Tiempo(seg.):0.005988094         / Tamaño Vectores:1048576      / V1[0]+V2[0]=V3[0](104857.600000+104857.600000=20971
5.200000) V1[1048575]+V2[1048575]=V3[1048575](209715.100000+0.100000=209715.200000) /
Tiempo(seg.):0.011145572         / Tamaño Vectores:2097152      / V1[0]+V2[0]=V3[0](209715.200000+209715.200000=41943
0.400000) V1[2097151]+V2[2097151]=V3[2097151](419430.300000+0.100000=419430.400000) /
Tiempo(seg.):0.023627042         / Tamaño Vectores:4194304      / V1[0]+V2[0]=V3[0](419430.400000+419430.400000=83886
0.800000) V1[4194303]+V2[4194303]=V3[4194303](838860.700000+0.100000=838860.800000) /
Tiempo(seg.):0.047373244         / Tamaño Vectores:8388608      / V1[0]+V2[0]=V3[0](838860.800000+838860.800000=16777
21.600000) V1[8388607]+V2[8388607]=V3[8388607](1677721.500000+0.100000=1677721.600000) /
Tiempo(seg.):0.094593114         / Tamaño Vectores:16777216     / V1[0]+V2[0]=V3[0](1677721.600000+1677721.600000=335
5443.200000) V1[16777215]+V2[16777215]=V3[16777215](3355443.100000+0.100000=3355443.200000) /
Tiempo(seg.):0.188039115         / Tamaño Vectores:33554432     / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=671
0886.400000) V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
Tiempo(seg.):0.365949495         / Tamaño Vectores:67108864     / V1[0]+V2[0]=V3[0](6710886.400000+6710886.400000=134
21772.800000) V1[67108863]+V2[67108863]=V3[67108863](13421772.700000+0.100000=13421772.800000) /
```

**6.** Rellenar una tabla como la Tabla 1 para atcgrid y otra para su PC 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 y en su PC para vectores locales, globales y dinámicos (eje y) en función del tamaño en bytes de un vector (los valores de la segunda columna de la tabla, que están en escala logarítmica, deben estar en el eje x). Utilice escala logarítmica en el eje de ordenadas (eje y). ¿Hay diferencias en los tiempos de ejecución?

**RESPUESTA**:

**Tabla 1 .**

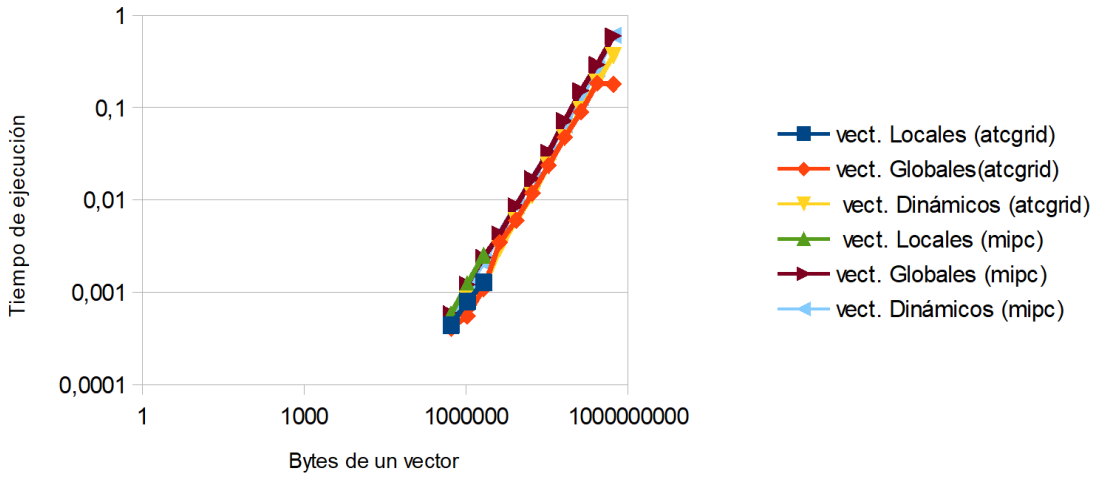
Atcgrid:

| Nº de Componentes | Bytes de un vector | Tiempo para vect. locales | Tiempo para vect. globales | Tiempo para vect. dinámicos |
|---|---|---|---|---|
| 65536 | 524288 | 0.000440332 | 0.000408114 | 0.000418541 |
| 131072 | 1048576 | 0.000798097 | 0.000554910 | 0.000846559 |
| 262144 | 2097152 | 0.001277184 | 0.001096058 | 0.001080770 |
| 524288 | 4194304 | | 0.003475463 | 0.002961303 |
| 1048576 | 8388608 | | 0.006015977 | 0.005988094 |
| 2097152 | 16777216 | | 0.011873775 | 0.011145572 |
| 4194304 | 33554432 | | 0.023631107 | 0.023627042 |
| 8388608 | 67108864 | | 0.047657070 | 0.047373244 |
| 16777216 | 134217728 | | 0.090048842 | 0.094593114 |
| 33554432 | 268435456 | | 0.183794201 | 0.188039115 |
| 67108864 | 536870912 | | 0.179216410 | 0.365949495 |

PC:

| Nº de Componentes | Bytes de un vector | Tiempo para vect. locales | Tiempo para vect. globales | Tiempo para vect. dinámicos |
|---|---|---|---|---|
| 65536 | 524288 | 0.000580800 | 0.000586000 | 0.000499500 |
| 131072 | 1048576 | 0.001232400 | 0.001215100 | 0.001152400 |
| 262144 | 2097152 | 0.002516300 | 0.002383000 | 0.002098600 |
| 524288 | 4194304 | | 0.004290000 | 0.004459200 |
| 1048576 | 8388608 | | 0.008633000 | 0.008614500 |
| 2097152 | 16777216 | | 0.017005400 | 0.017263800 |
| 4194304 | 33554432 | | 0.032994900 | 0.033152700 |
| 8388608 | 67108864 | | 0.072001000 | 0.065044000 |
| 16777216 | 134217728 | | 0.152327200 | 0.129520600 |
| 33554432 | 268435456 | | 0.291848000 | 0.262842300 |
| 67108864 | 536870912 | | 0.596602900 | 0.607033800 |

Grafica:



**7.** 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**: Creo que el error equivale a que el bus de datos al ser un programa de 32 bits se ocuparía entero por lo que eso provocaría el error.

```
[PaulaRuizGarcía root@Pompitas:~/AC/P0] 2018-03-04 Sunday
$gcc -O2 SumaVectoresC.c -o SumaVectoresC -lrt
/tmp/cc6VhpPp.o: In function `main':
SumaVectoresC.c:(.text.startup+0x79): relocation truncated to fit: R_X86_64_32S against symbol `v2' defined in COMMON
 section in /tmp/cc6VhpPp.o
SumaVectoresC.c:(.text.startup+0xc0): relocation truncated to fit: R_X86_64_32S against symbol `v2' defined in COMMON
 section in /tmp/cc6VhpPp.o
SumaVectoresC.c:(.text.startup+0xc8): relocation truncated to fit: R_X86_64_32S against symbol `v3' defined in COMMON
 section in /tmp/cc6VhpPp.o
SumaVectoresC.c:(.text.startup+0xfc): relocation truncated to fit: R_X86_64_32S against symbol `v3' defined in COMMON
 section in /tmp/cc6VhpPp.o
SumaVectoresC.c:(.text.startup+0x115): relocation truncated to fit: R_X86_64_32S against symbol `v2' defined in COMMO
N section in /tmp/cc6VhpPp.o
SumaVectoresC.c:(.text.startup+0x12b): relocation truncated to fit: R_X86_64_PC32 against symbol `v3' defined in COMM
ON section in /tmp/cc6VhpPp.o
SumaVectoresC.c:(.text.startup+0x135): relocation truncated to fit: R_X86_64_PC32 against symbol `v2' defined in COMM
ON section in /tmp/cc6VhpPp.o
collect2: error: ld returned 1 exit status
```

**Listado 1 .** Código C que suma dos vectores

```
/* SumaVectoresC.c
 Suma de dos vectores: v3 = v1 + v2

 Para compilar usar (-lrt: real time library):
       gcc  -O2 SumaVectores.c -o SumaVectores –lrt
 gcc  -O2 –S SumaVectores.c –lrt   //para generar el código ensamblador

 Para ejecutar use: SumaVectoresC longitud
*/

#include <stdlib.h> // biblioteca con funciones atoi(), malloc() y free()
#include <stdio.h> // biblioteca donde se encuentra la función printf()
#include <time.h>      // biblioteca donde se encuentra la función clock_gettime()

//#define PRINTF_ALL     // comentar para quitar el printf ...
                  // que imprime todos los componentes
//Sólo puede estar definida una de las tres constantes VECTOR_ (sólo uno de los ...
//tres defines siguientes puede estar descomentado):
//#define VECTOR_LOCAL    // descomentar para que los vectores sean variables ...
                  // locales (si se supera el tamaño de la pila se ...
                  // generará el error "Violación de Segmento")
//#define VECTOR_GLOBAL// descomentar para que los vectores sean variables ...
                  // globales (su longitud no estará limitada por el ...
                  // tamaño de la pila del programa)
#define VECTOR_DYNAMIC    // descomentar para que los vectores sean variables ...
                  // dinámicas (memoria reutilizable durante la ejecución)
#ifdef VECTOR_GLOBAL
#define MAX 33554432       //=2^25
double v1[MAX], v2[MAX], v3[MAX];
#endif

int main(int argc, char** argv){

  int i;
  struct timespec cgt1,cgt2; double ncgt; //para tiempo de ejecución

  //Leer argumento de entrada (nº de componentes del vector)
  if (argc<2){
    printf("Faltan nº componentes del vector\n");
    exit(-1);
  }

  unsigned int N = atoi(argv[1]); // Máximo N =2^32-1=4294967295 (sizeof(unsigned int) = 4 B)
  #ifdef VECTOR_LOCAL
  double v1[N], v2[N], v3[N];   // Tamaño variable local en tiempo de ejecución ...
                    // disponible en C a partir de actualización C99
  #endif
  #ifdef VECTOR_GLOBAL
  if (N>MAX) N=MAX;
  #endif
  #ifdef VECTOR_DYNAMIC
  double *v1, *v2, *v3;
```

```
  v1 = (double*) malloc(N*sizeof(double));// malloc necesita el tamaño en bytes
  v2 = (double*) malloc(N*sizeof(double)); //si no hay espacio suficiente malloc devuelve NULL
  v3 = (double*) malloc(N*sizeof(double));
    if ( (v1==NULL) || (v2==NULL) || (v3==NULL) ){
    printf("Error en la reserva de espacio para los vectores\n");
    exit(-2);
  }
  #endif

  //Inicializar vectores
  for(i=0; i<N; i++){
    v1[i] = N*0.1+i*0.1; v2[i] = N*0.1-i*0.1; //los valores dependen de N
  }

  clock_gettime(CLOCK_REALTIME,&cgt1);
  //Calcular suma de vectores
  for(i=0; i<N; i++)
    v3[i] = v1[i] + v2[i];

  clock_gettime(CLOCK_REALTIME,&cgt2);
  ncgt=(double) (cgt2.tv_sec-cgt1.tv_sec)+
       (double) ((cgt2.tv_nsec-cgt1.tv_nsec)/(1.e+9));

  //Imprimir resultado de la suma y el tiempo de ejecución
  #ifdef PRINTF_ALL
  printf("Tiempo(seg.):%11.9f\t / Tamaño Vectores:%u\n",ncgt,N);
  for(i=0; i<N; i++)
    printf("/ V1[%d]+V2[%d]=V3[%d](%8.6f+%8.6f=%8.6f) /\n",
           i,i,i,v1[i],v2[i],v3[i]);

  #else
    printf("Tiempo(seg.):%11.9f\t / Tamaño Vectores:%u\t/ V1[0]+V2[0]=V3[0](%8.6f+%8.6f=%8.6f) / /
           V1[%d]+V2[%d]=V3[%d](%8.6f+%8.6f=%8.6f) /\n",
           ncgt,N,v1[0],v2[0],v3[0],N-1,N-1,N-1,v1[N-1],v2[N-1],v3[N-1]);
  #endif

  #ifdef VECTOR_DYNAMIC
  free(v1); // libera el espacio reservado para v1
  free(v2); // libera el espacio reservado para v2
  free(v3); // libera el espacio reservado para v3
  #endif
  return 0;
}
```

**Listado 2 .** Código C++ que suma dos vectores

```
/* SumaVectoresCpp.cpp
 Suma de dos vectores: v3 = v1 + v2

 Para compilar usar (-lrt: real time library):
            g++  -O2 SumaVectoresCpp.cpp -o SumaVectoresCpp -lrt

 Para ejecutar use: SumaVectoresCpp longitud
*/

#include <cstdlib>  // biblioteca con atoi()
#include <iostream> // biblioteca donde se encuentra la función cout
using namespace std;
#include <time.h>      // biblioteca donde se encuentra la función clock_gettime()

//#define COUT_ALL // comentar para quitar el cout ...
               // que imprime todos los componentes
//Sólo puede estar definida una de las tres constantes VECTOR_ (sólo uno de los ...
//tres defines siguientes puede estar descomentado):
//#define VECTOR_LOCAL    // descomentar para que los vectores sean variables ...
                  // locales (si se supera el tamaño de la pila se ...
                  // generará el error "Violación de Segmento")
//#define VECTOR_GLOBAL// descomentar para que los vectores sean variables ...
                  // globales (su longitud no estará limitada por el ...
                  // tamaño de la pila del programa)
#define VECTOR_DYNAMIC    // descomentar para que los vectores sean variables ...
                  // dinámicas (memoria reutilizable durante la ejecución)
#ifdef VECTOR_GLOBAL
#define MAX 33554432      //=2^25
double v1[MAX], v2[MAX], v3[MAX];
#endif

int main(int argc, char** argv){

  struct timespec cgt1,cgt2;   //para tiempo de ejecución

  //Leer argumento de entrada (nº de componentes del vector)
  if (argc<2){
    cout << "Faltan nº componentes del vector\n" << endl ;
    exit(-1);
  }

  unsigned int N = atoi(argv[1]);
  #ifdef VECTOR_LOCAL
  double v1[N], v2[N], v3[N];
  #endif
  #ifdef VECTOR_GLOBAL
  if (N>MAX) N=MAX;
  #endif
  #ifdef VECTOR_DYNAMIC
  double *v1, *v2, *v3;
  v1 = new double [N];    //si no hay espacio suficiente new genera una excepción
  v2 = new double [N];
  v3 = new double [N];
  #endif
```

```
 //Inicializar vectores
 for(int i=0; i<N; i++){
   v1[i] = N*0.1+i*0.1; v2[i] = N*0.1-i*0.1; //los valores dependen de N
 }
  clock_gettime(CLOCK_REALTIME,&cgt1);
 //Calcular suma de vectores
 for(int i=0; i<N; i++)
   v3[i] = v1[i] + v2[i];
 clock_gettime(CLOCK_REALTIME,&cgt2);
 double ncgt=(double) (cgt2.tv_sec-cgt1.tv_sec)+
      (double) ((cgt2.tv_nsec-cgt1.tv_nsec)/(1.e+9));

 //Imprimir resultado de la suma y el tiempo de ejecución
 #ifdef COUT_ALL
 cout << "Tiempo(seg.):" << ncgt << "\t/ Tamaño Vectores:" << N << endl;
 for(int i=0; i<N; i++)
    cout << "/ V1[" << i << "]+V2[" << i << "]=V3" << i << "](" << v1[i] << "+" << v2[i] << "="
    << v3[i] << ") /\t" << endl;
 cout <<"\n"<< endl;
 #else
    cout << "Tiempo(seg.):" << ncgt << "\t/ Tamaño Vectores:" << N << "\t/ V1[0]+V2[0]=V3[0]("
    << v1[0] << "+" << v2[0] << "=" << v3[0] << ") / / V1[" << N-1 << "]+V2[" << N-1 << "]=V3["
    << N-1 << "](" << v1[N-1] << "+" << v2[N-1] << "=" << v3[N-1] << ")/\n" << endl;
 #endif

 #ifdef VECTOR_DYNAMIC
 delete [] v1; // libera el espacio reservado para v1
 delete [] v2; // libera el espacio reservado para v2
 delete [] v3; // libera el espacio reservado para v3
 #endif
 return 0;
}
```

**Listado 3 .** Script para la suma de vectores (SumaVectores.sh). Se supone en el script que el fichero a ejecutar se llama SumaVectorC y que se encuentra en el directorio en el que se ha ejecutado qsub.

```
#!/bin/bash
#Se asigna al trabajo el nombre SumaVectoresC_vlocales
#PBS -N SumaVectoresC_vlocales
#Se asigna al trabajo la cola ac
#PBS -q ac
#Se imprime información del trabajo usando variables de entorno de PBS
echo "Id. usuario del trabajo: $PBS_O_LOGNAME"
echo "Id. del trabajo: $PBS_JOBID"
echo "Nombre  del trabajo especificado por usuario: $PBS_JOBNAME"
echo "Nodo que ejecuta qsub: $PBS_O_HOST"
echo "Directorio en el que se ha ejecutado qsub: $PBS_O_WORKDIR"
echo "Cola: $PBS_QUEUE"
echo "Nodos asignados al trabajo:"
cat $PBS_NODEFILE
#Se ejecuta SumaVectorC, que está en el directorio en el que se ha ejecutado qsub,
#para N potencia de 2 desde 2^16 a 2^26
for ((N=65536;N<67108865;N=N*2))
do
   $PBS_O_WORKDIR/SumaVectoresC $N
done
```