

2º curso / 2º cuatr.

Grado Ing. Inform.

Arquitectura de Computadores (AC)

Cuaderno de prácticas.

Bloque Práctico 3. Programación paralela III: Interacción con el entorno en OpenMP

Estudiante (nombre y apellidos): Javier Ramirez Pulido

Grupo de prácticas: 2C

Fecha de entrega:

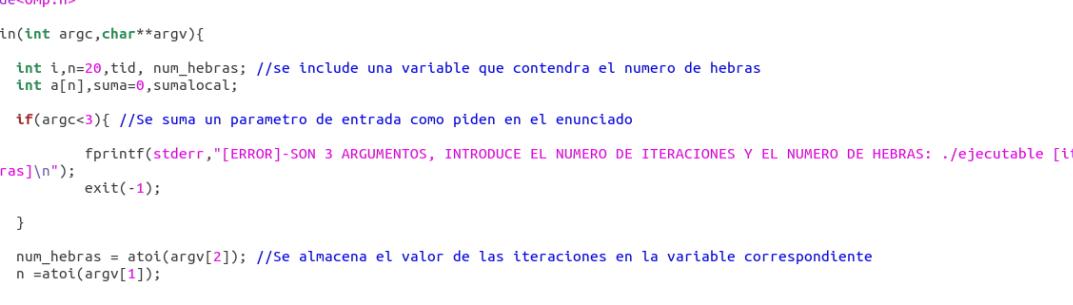
Fecha evaluación en clase:

Antes de comenzar a realizar el trabajo de este cuaderno consultar el fichero con las normas de prácticas que se encuentra en SWAD

Ejercicios basados en los ejemplos del seminario práctico

1. Usar la cláusula `num_threads(x)` en el ejemplo del seminario `if_clause.c`, y añadir un parámetro de entrada al programa que fije el valor `x` que se va a usar en la cláusula. Incorporar en el cuaderno de trabajo de esta práctica volcados de pantalla con ejemplos de ejecución que ilustren la funcionalidad de esta cláusula y explicar por qué lo ilustran.

CAPTURA CÓDIGO FUENTE: if-clauseModificado.c



```
#include<stdio.h>
#include<stdlib.h>
#include<omp.h>

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

    int i,n=20,tid, num_hebras; //se incluye una variable que contendra el numero de hebras
    int a[n],suma=0,sumalocal;

    if(argc<3){ //Se suma un parametro de entrada como piden en el enunciado

        fprintf(stderr,"[ERROR]-SON 3 ARGUMENTOS, INTRODUCE EL NUMERO DE ITERACIONES Y EL NUMERO DE HEBRAS: ./ejecutable [iteraciones]
[nºhebras]\n");
        exit(-1);
    }

    num_hebras = atoi(argv[2]); //Se almacena el valor de las iteraciones en la variable correspondiente
    n =atoi(argv[1]);

    if(n>20)n=20;

    for(i=0;i<n;i++){
        a[i]=i;
    }

    #pragma omp parallel num_threads(num_hebras) if(n>4) default(none) private(sumalocal,tid) shared(a,suma,n)//se incluye
num_threads(x)
    {
        sumalocal=0;
        tid=omp_get_thread_num();

        #pragma omp for private(i) schedule(static) nowait
        for(i=0;i<n;i++)
        {
            sumalocal +=a [i];
            printf(" *thread %d suma de %d thread %d sumalocal=%d\n",tid,i,sumalocal);
        }
    }
}
```

```

Actividades Editor de textos dom 11:25
if-clauseModificado.c
~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer1
Guarda
Abrir
if(argc<3){ //Se suma un parametro de entrada como piden en el enunciado
    fprintf(stderr,[ERROR]-SON 3 ARGUMENTOS, INTRODUCE EL NUMERO DE ITERACIONES Y EL NUMERO DE HEBRAS: ./ejecutable [iteraciones]
[nºhebras]\n");
    exit(-1);
}

num_hebras = atoi(argv[2]); //Se almacena el valor de las iteraciones en la variable correspondiente
n =atoi(argv[1]);

if(n>20)n=20;

for(i=0;i<n;i++){
    a[i]=i;
}

#pragma omp parallel num_threads(num_hebras) if(n>4) default(none) private(sumalocal,tid) shared(a,suma,n)//se incluye
num_threads(x)
{
    sumalocal=0;
    tid=omp_get_thread_num();

    #pragma omp for private(i) schedule(static) nowait
    for(i=0;i<n;i++)
    {
        sumalocal += a[i];
        printf(" thread %d suma de a[%d]=%d sumalocal=%d \n",tid,i,a[i],sumalocal);
    }
    #pragma omp atomic
    suma +=sumalocal;
    #pragma omp barrier
    #pragma omp master
    printf("thread master=%d imprime suma=%d\n",tid,suma);
}

```

C Anchura del tabulador: 8 ▾ Ln 27, Col 52 ▾ INS

CAPTURAS DE PANTALLA:

```

Actividades Terminal dom 11:24
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~ x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido@dom abr 19 gcc -fopenmp if-clauseModificado.c -o if-clause
JavierRamirezPulido@dom abr 19 ./if-clause 20 4
thread 2 suma de a[10]=10 sumalocal=10
thread 2 suma de a[11]=11 sumalocal=21
thread 2 suma de a[12]=12 sumalocal=33
thread 2 suma de a[13]=13 sumalocal=46
thread 2 suma de a[14]=14 sumalocal=60
thread 3 suma de a[15]=15 sumalocal=15
thread 3 suma de a[16]=16 sumalocal=31
thread 3 suma de a[17]=17 sumalocal=48
thread 3 suma de a[18]=18 sumalocal=66
thread 3 suma de a[19]=19 sumalocal=85
thread 1 suma de a[5]=5 sumalocal=5
thread 1 suma de a[6]=6 sumalocal=11
thread 1 suma de a[7]=7 sumalocal=18
thread 1 suma de a[8]=8 sumalocal=26
thread 1 suma de a[9]=9 sumalocal=35
thread 0 suma de a[0]=0 sumalocal=0
thread 0 suma de a[1]=1 sumalocal=1
thread 0 suma de a[2]=2 sumalocal=3
thread 0 suma de a[3]=3 sumalocal=6
thread 0 suma de a[4]=4 sumalocal=10
thread master=0 imprime suma=190
JavierRamirezPulido@dom abr 19 ./if-clause 10 5
thread 1 suma de a[2]=2 sumalocal=2
thread 1 suma de a[3]=3 sumalocal=5
thread 2 suma de a[4]=4 sumalocal=4
thread 2 suma de a[5]=5 sumalocal=9
thread 0 suma de a[0]=0 sumalocal=0
thread 0 suma de a[1]=1 sumalocal=1
thread 3 suma de a[6]=6 sumalocal=6
thread 3 suma de a[7]=7 sumalocal=13
thread 4 suma de a[8]=8 sumalocal=8
thread 4 suma de a[9]=9 sumalocal=17
thread master=0 imprime suma=45
JavierRamirezPulido@dom abr 19

```

En la primera ejecución de ambas, se observa que la primera es ejecutada por 4 hebras (0, 1, 2 y 3) especificadas como uno de los argumentos. En la segunda, se le introducen menos iteraciones, pero más hebras (5 en este caso) y provoca que cada una realice muchas menos iteraciones que en anterior. Del número de hebras dependerá el número de iteraciones de cada una.

```

Actividades Terminal dom 11:24
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~ x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido>dom abr 19 gcc -fopenmp if-clauseModificado.c -o if-clause
JavierRamirezPulido>dom abr 19 ./if-clause 20 4
thread 2 suma de a[10]=10 sumalocal=10
thread 2 suma de a[11]=11 sumalocal=21
thread 2 suma de a[12]=12 sumalocal=33
thread 2 suma de a[13]=13 sumalocal=46
thread 2 suma de a[14]=14 sumalocal=60
thread 3 suma de a[15]=15 sumalocal=15
thread 3 suma de a[16]=16 sumalocal=31
thread 3 suma de a[17]=17 sumalocal=48
thread 3 suma de a[18]=18 sumalocal=66
thread 3 suma de a[19]=19 sumalocal=85
thread 1 suma de a[5]=5 sumalocal=5
thread 1 suma de a[6]=6 sumalocal=11
thread 1 suma de a[7]=7 sumalocal=18
thread 1 suma de a[8]=8 sumalocal=26
thread 1 suma de a[9]=9 sumalocal=35
thread 0 suma de a[0]=0 sumalocal=0
thread 0 suma de a[1]=1 sumalocal=1
thread 0 suma de a[2]=2 sumalocal=3
thread 0 suma de a[3]=3 sumalocal=6
thread 0 suma de a[4]=4 sumalocal=10
thread master=0 imprime suma=190
JavierRamirezPulido>dom abr 19 []

```



```

JavierRamirezPulido>lun abr 20 ./if-clause 2 5
thread 0 suma de a[0]=0 sumalocal=0
thread 0 suma de a[1]=1 sumalocal=1
thread master=0 imprime suma=1
JavierRamirezPulido>lun abr 20 []

```

RESPUESTA:

Como vemos, para un número de iteraciones menor que 4 ocurre que todas las hebras que ejecutan las iteraciones son la misma, la hebra 0.

2. (a) Rellenar la Tabla 1 (se debe poner en la tabla el id del *thread* que ejecuta cada iteración) ejecutando los ejemplos del seminario *schedule-clause.c*, *scheduled-clause.c* y *scheduleg-clause.c* con dos *threads* (0,1) y unas entradas de:

- iteraciones: 16 (0,...15)
- chunck= 1, 2 y 4

Tabla 1 .

Tabla schedule. En la segunda fila, 1, 2 4 representan el tamaño del chunk (consulte seminario)

Iteración	schedule-clause.c			schedule-claused.c			schedule-clauseg.c		
	1	2	4	1	2	4	1	2	4
0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0
2	0	1	0	0	1	0	0	0	0
3	1	1	0	0	1	0	0	0	0
4	0	0	1	0	0	1	0	0	0
5	1	0	1	0	0	1	0	0	0
6	0	1	1	0	0	1	0	0	0
7				0	0	1	0	0	0
8				0	0	0	1	1	1
9				0	0	0	1	1	1
10				0	0	0	1	1	1
11				0	0	0	1	1	1
12				0	0	0	0	0	0
13				0	0	0	0	0	0
14				0	0	0	0	0	0
15				0	0	0	0	0	0

Schedule

```

Actividades Terminal dom 12:56
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido>dom abr 19 ./schedule-clause 1
thread 0 suma a[0] suma=0
thread 0 suma a[2] suma=2
thread 0 suma a[4] suma=6
thread 0 suma a[6] suma=12
thread 1 suma a[1] suma=1
thread 1 suma a[3] suma=4
thread 1 suma a[5] suma=9
Fuera de 'parallel for' suma=12
JavierRamirezPulido>dom abr 19 ./schedule-clause 2
thread 0 suma a[0] suma=0
thread 0 suma a[1] suma=1
thread 0 suma a[4] suma=5
thread 0 suma a[5] suma=10
thread 1 suma a[2] suma=2
thread 1 suma a[3] suma=5
thread 1 suma a[6] suma=11
Fuera de 'parallel for' suma=11
JavierRamirezPulido>dom abr 19 ./schedule-clause 4
thread 0 suma a[0] suma=0
thread 0 suma a[1] suma=1
thread 0 suma a[2] suma=3
thread 0 suma a[3] suma=6
thread 1 suma a[4] suma=4
thread 1 suma a[5] suma=9
thread 1 suma a[6] suma=15
Fuera de 'parallel for' suma=15
JavierRamirezPulido>dom abr 19

```

Scheduled

```

Actividades Terminal dom 12:58
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido>dom abr 19 ./scheduled-clause 16 1
thread 0 suma a[0]=0 suma=0
thread 0 suma a[2]=2 suma=2
thread 0 suma a[3]=3 suma=5
thread 0 suma a[4]=4 suma=9
thread 0 suma a[5]=5 suma=14
thread 0 suma a[6]=6 suma=20
thread 0 suma a[7]=7 suma=27
thread 0 suma a[8]=8 suma=35
thread 0 suma a[9]=9 suma=44
thread 0 suma a[10]=10 suma=54
thread 0 suma a[11]=11 suma=65
thread 0 suma a[12]=12 suma=77
thread 0 suma a[13]=13 suma=90
thread 0 suma a[14]=14 suma=104
thread 0 suma a[15]=15 suma=119
thread 1 suma a[1]=1 suma=1
Fuera de 'parallel for' suma=119
JavierRamirezPulido>dom abr 19 ./scheduled-clause 16 2
thread 0 suma a[0]=0 suma=0
thread 0 suma a[1]=1 suma=1
thread 0 suma a[4]=4 suma=5
thread 0 suma a[5]=5 suma=10
thread 0 suma a[6]=6 suma=16
thread 0 suma a[7]=7 suma=23
thread 0 suma a[8]=8 suma=31
thread 0 suma a[9]=9 suma=40
thread 0 suma a[10]=10 suma=50
thread 0 suma a[11]=11 suma=61
thread 0 suma a[12]=12 suma=73
thread 0 suma a[13]=13 suma=86
thread 0 suma a[14]=14 suma=100
thread 0 suma a[15]=15 suma=115
thread 1 suma a[2]=2 suma=2
thread 1 suma a[3]=3 suma=5
Fuera de 'parallel for' suma=115
Actividades Terminal dom 12:59
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido>dom abr 19 ./scheduled-clause 16 4
thread 0 suma a[0]=0 suma=0
thread 0 suma a[1]=1 suma=1
thread 0 suma a[2]=2 suma=3
thread 0 suma a[3]=3 suma=6
thread 0 suma a[8]=8 suma=14
thread 0 suma a[9]=9 suma=23
thread 0 suma a[10]=10 suma=33
thread 0 suma a[11]=11 suma=44
thread 0 suma a[12]=12 suma=56
thread 0 suma a[13]=13 suma=69
thread 0 suma a[14]=14 suma=83
thread 0 suma a[15]=15 suma=98
thread 1 suma a[4]=4 suma=4
thread 1 suma a[5]=5 suma=9
thread 1 suma a[6]=6 suma=15
thread 1 suma a[7]=7 suma=22
Fuera de 'parallel for' suma=98
JavierRamirezPulido>dom abr 19 []

```

Scheduleg

The image shows a Linux desktop environment with two terminal windows open. Both terminals are running the same command: `dom abr 19 ./scheduleleg-clause 16 1`. The output of the first terminal window is as follows:

```
JavierRamirezPulido>dom abr 19 ./scheduleleg-clause 16 1
thread 0 suma a[0]=0 suma=0
thread 0 suma a[1]=1 suma=1
thread 0 suma a[2]=2 suma=3
thread 0 suma a[3]=3 suma=6
thread 0 suma a[4]=4 suma=10
thread 0 suma a[5]=5 suma=15
thread 0 suma a[6]=6 suma=21
thread 0 suma a[7]=7 suma=28
thread 0 suma a[12]=12 suma=40
thread 0 suma a[13]=13 suma=53
thread 0 suma a[14]=14 suma=67
thread 0 suma a[15]=15 suma=82
thread 1 suma a[8]=8 suma=8
thread 1 suma a[9]=9 suma=17
thread 1 suma a[10]=10 suma=27
thread 1 suma a[11]=11 suma=38
Fuera de 'parallel for' suma=82
```

The output of the second terminal window is identical:

```
JavierRamirezPulido>dom abr 19 ./scheduleleg-clause 16 2
thread 0 suma a[0]=0 suma=0
thread 0 suma a[1]=1 suma=1
thread 0 suma a[2]=2 suma=3
thread 0 suma a[3]=3 suma=6
thread 0 suma a[4]=4 suma=10
thread 0 suma a[5]=5 suma=15
thread 0 suma a[6]=6 suma=21
thread 0 suma a[7]=7 suma=28
thread 0 suma a[12]=12 suma=40
thread 0 suma a[13]=13 suma=53
thread 0 suma a[14]=14 suma=67
thread 0 suma a[15]=15 suma=82
thread 1 suma a[8]=8 suma=8
thread 1 suma a[9]=9 suma=17
thread 1 suma a[10]=10 suma=27
thread 1 suma a[11]=11 suma=38
Fuera de 'parallel for' suma=82
```

The desktop environment includes a dock with various icons and a system tray at the top.

(b) Rellenar otra tabla como la de la figura pero esta vez usando cuatro *threads* (0,1,2,3).

Tabla 2 .

Tabla schedule. En la segunda fila, 1, 2 4 representan el tamaño del chunk (consulte seminario)

Iteración	schedule-clause.c			schedule-clause.d.c			schedule-clauseg.c		
	1	2	4	1	2	4	1	2	4
0	0	0	0	0	0	1	1	2	1
1	1	0	0	2	0	1	1	2	1
2	2	1	0	3	2	1	1	2	1
3	3	1	0	0	2	1	1	2	1
4	0	2	1	0	1	0	0	1	1
5	1	2	1	0	1	0	0	1	1
6	2	3	1	0	3	0	0	1	1
7				0	3	0	2	0	1
8				0	0	2	2	0	1
9				0	0	2	2	0	1
10				0	0	2	3	2	1
11				0	0	2	3	2	1
12				0	0	3	0	3	1
13				0	0	3	0	3	1
14				0	0	3	0	3	1
15				0	0	3	0	3	1

Escriba en el cuaderno de prácticas las diferencias en el comportamiento de *schedule()* con static, dynamic y guided.

CAPTURAS DE PANTALLA:

Schedule

```

Actividades Terminal dom 13:11
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3

JavierRamirezPulido>dom abr 19 ./schedule-clause 1
thread 1 suma a[1] suma=1
thread 1 suma a[5] suma=6
thread 2 suma a[2] suma=2
thread 2 suma a[6] suma=8
thread 3 suma a[3] suma=3
thread 0 suma a[0] suma=0
thread 0 suma a[4] suma=4
Fuera de 'parallel for' suma=8
JavierRamirezPulido>dom abr 19 ./schedule-clause 2
thread 1 suma a[2] suma=2
thread 1 suma a[3] suma=5
thread 3 suma a[6] suma=6
thread 2 suma a[4] suma=4
thread 2 suma a[5] suma=9
thread 0 suma a[0] suma=0
thread 0 suma a[1] suma=1
Fuera de 'parallel for' suma=6
JavierRamirezPulido>dom abr 19 ./schedule-clause 4
thread 1 suma a[4] suma=4
thread 1 suma a[5] suma=9
thread 1 suma a[6] suma=15
thread 0 suma a[0] suma=0
thread 0 suma a[1] suma=1
thread 0 suma a[2] suma=3
thread 0 suma a[3] suma=6
Fuera de 'parallel for' suma=15
JavierRamirezPulido>dom abr 19 []

```

Scheduled

```

Actividades Terminal dom 13:12
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3

JavierRamirezPulido>dom abr 19 ./scheduled-clause 16 1
thread 0 suma a[0]=0 suma=0
thread 0 suma a[3]=3 suma=3
thread 0 suma a[4]=4 suma=7
thread 0 suma a[5]=5 suma=12
thread 0 suma a[6]=6 suma=18
thread 0 suma a[7]=7 suma=25
thread 0 suma a[8]=8 suma=33
thread 0 suma a[9]=9 suma=42
thread 0 suma a[10]=10 suma=52
thread 0 suma a[11]=11 suma=63
thread 0 suma a[12]=12 suma=75
thread 0 suma a[13]=13 suma=88
thread 0 suma a[14]=14 suma=102
thread 0 suma a[15]=15 suma=117
thread 2 suma a[1]=1 suma=1
thread 3 suma a[2]=2 suma=2
Fuera de 'parallel for' suma=117
JavierRamirezPulido>dom abr 19 ./scheduled-clause 16 2
thread 0 suma a[0]=0 suma=0
thread 0 suma a[1]=1 suma=1
thread 0 suma a[8]=8 suma=9
thread 0 suma a[9]=9 suma=18
thread 0 suma a[10]=10 suma=28
thread 0 suma a[11]=11 suma=39
thread 0 suma a[12]=12 suma=51
thread 0 suma a[13]=13 suma=64
thread 0 suma a[14]=14 suma=78
thread 0 suma a[15]=15 suma=93
thread 1 suma a[4]=4 suma=4
thread 1 suma a[5]=5 suma=9
thread 3 suma a[6]=6 suma=6
thread 2 suma a[2]=2 suma=2
thread 2 suma a[3]=3 suma=5
thread 3 suma a[7]=7 suma=13
Fuera de 'parallel for' suma=93

```

```

Actividades Terminal dom 13:14
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3

JavierRamirezPulido>dom abr 19 ./scheduled-clause 16 4
thread 0 suma a[4]=4 suma=4
thread 0 suma a[5]=5 suma=9
thread 0 suma a[6]=6 suma=15
thread 0 suma a[7]=7 suma=22
thread 1 suma a[0]=0 suma=0
thread 1 suma a[1]=1 suma=1
thread 1 suma a[2]=2 suma=3
thread 1 suma a[3]=3 suma=6
thread 2 suma a[8]=8 suma=8
thread 2 suma a[9]=9 suma=17
thread 2 suma a[10]=10 suma=27
thread 2 suma a[11]=11 suma=38
thread 3 suma a[12]=12 suma=12
thread 3 suma a[13]=13 suma=25
thread 3 suma a[14]=14 suma=39
thread 3 suma a[15]=15 suma=54
Fuera de 'parallel for' suma=54
JavierRamirezPulido>dom abr 19

```

Scheduleg

```

Actividades Terminal dom 13:17
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3

JavierRamirezPulido>dom abr 19 ./scheduleg-clause 16 1
thread 0 suma a[4]=4 suma=4
thread 0 suma a[5]=5 suma=9
thread 0 suma a[6]=6 suma=15
thread 0 suma a[12]=12 suma=27
thread 0 suma a[13]=13 suma=40
thread 0 suma a[14]=14 suma=54
thread 0 suma a[15]=15 suma=69
thread 2 suma a[7]=7 suma=7
thread 2 suma a[8]=8 suma=15
thread 2 suma a[9]=9 suma=24
thread 3 suma a[10]=10 suma=10
thread 1 suma a[0]=0 suma=0
thread 1 suma a[1]=1 suma=1
thread 1 suma a[2]=2 suma=3
thread 1 suma a[3]=3 suma=6
thread 3 suma a[11]=11 suma=21
Fuera de 'parallel for' suma=69
JavierRamirezPulido>dom abr 19 ./scheduleg-clause 16 2
thread 2 suma a[0]=0 suma=0
thread 2 suma a[1]=1 suma=1
thread 2 suma a[2]=2 suma=3
thread 2 suma a[3]=3 suma=6
thread 2 suma a[10]=10 suma=16
thread 1 suma a[4]=4 suma=4
thread 3 suma a[12]=12 suma=12
thread 3 suma a[13]=13 suma=25
thread 1 suma a[5]=5 suma=9
thread 3 suma a[14]=14 suma=39
thread 0 suma a[7]=7 suma=7
thread 3 suma a[15]=15 suma=54
thread 0 suma a[8]=8 suma=15
thread 1 suma a[6]=6 suma=15
thread 0 suma a[9]=9 suma=24
thread 2 suma a[11]=11 suma=27
Fuera de 'parallel for' suma=54

```

```
JavierRamirezPulido>dom abr 19 ./scheduled-clause 16 4
thread 1 suma a[0]=0 suma=0
thread 1 suma a[1]=1 suma=1
thread 1 suma a[2]=2 suma=3
thread 1 suma a[3]=3 suma=6
thread 1 suma a[4]=4 suma=10
thread 1 suma a[5]=5 suma=15
thread 1 suma a[6]=6 suma=21
thread 1 suma a[7]=7 suma=28
thread 1 suma a[8]=8 suma=36
thread 1 suma a[9]=9 suma=45
thread 1 suma a[10]=10 suma=55
thread 1 suma a[11]=11 suma=66
thread 1 suma a[12]=12 suma=78
thread 1 suma a[13]=13 suma=91
thread 1 suma a[14]=14 suma=105
thread 1 suma a[15]=15 suma=120
Fuera de 'parallel for' suma=120
JavierRamirezPulido>dom abr 19
```

RESPUESTA:

Una diferencia es que el número de iteraciones es tomado del valor de chunk para los casos de static y Dynamic y para guided su valor es equivalente al tamaño mínimo del bloque. Además, en static las iteraciones son ordenadas y se realizan a raíz del identificador de la hebra. Tanto en Dynamic, como en guided, las iteraciones no van ordenadas según el identificador y sí según la hebra que va llegando antes. En caso de static, el valor de chunk expresa el número de iteraciones de cada hebra.

3. Añadir al programa `scheduled-clause.c` lo necesario para que imprima el valor de las variables de control `dyn-var`, `nthreads-var`, `thread-limit-var` y `run-sched-var` dentro (debe imprimir sólo un thread) y fuera de la región paralela. Realizar varias ejecuciones usando variables de entorno para modificar estas variables de control antes de la ejecución. Incorporar en su cuaderno de prácticas volcados de pantalla de estas ejecuciones. ¿Se imprimen valores distintos dentro y fuera de la región paralela?

CAPTURA CÓDIGO FUENTE: `scheduled-clauseModificado.c`

```

#include<stdio.h>
#include<stdlib.h>
#ifndef _OPENMP
#include<omp.h>
#else
#define omp_get_thread_num() 0
#endif

main(int argc,char**argv)
{
    int i,n=200,chunk,a[n],suma=0;

    if(argc!=3){
        fprintf(stderr,"Tienes que poner: ./ejecutables iteraciones chunk\n");
        exit(-1);
    }

    n =atoi(argv[1]); if(n>200) n=200;chunk=atoi(argv[2]);

    omp_sched_t version;
    int run_sched_var;
    int nthreads_var = omp_get_max_threads(), dyn_var = omp_get_dynamic(), thread_limit_var = omp_get_thread_limit();
    omp_get_schedule(&version, &run_sched_var);

    for(i=0;i<n;i++) a[i]=i;

    #pragma omp parallel for firstprivate(suma) lastprivate(suma)schedule(dynamic,chunk)
    for(i=0;i<n;i++){
        suma+=a[i];
        printf(" thread %d suma a[%d]=%d suma=%d \n",omp_get_thread_num(),i,a[i],suma);
    }
    #pragma omp single //Solo realizará lo siguiente la hebra que llegue antes a esta clausula. Otra opcion seria poner un identificador cualquiera
    //A que únicamente esa hebra entre en el if que contenga el código siguiente
    if(version == omp_sched_static) printf("\nVersion: estatica\n");
    if(version == omp_sched_guided) printf("\nVersion: guided\n");
    if(version == omp_sched_dynamic) printf("\nVersion: dinamica\n");
    printf("Y el valor de las variables es: \n");
    printf("\tLa hebra %d muestra dyn-var %d \n", omp_get_thread_num(), dyn_var);
    printf("\tLa hebra %d muestra nthreads_var %d \n", omp_get_thread_num(), nthreads_var);
    printf("\tLa hebra %d muestra thread_limit_var %d \n", omp_get_thread_num(), thread_limit_var);
    printf("\tLa hebra %d muestra run-sched-var mod %d \n", omp_get_thread_num(), run_sched_var );

    printf("\nFuera de 'parallel for' las variables son:");
    if(version == omp_sched_static) printf("\nVersion: estatica\n");
    if(version == omp_sched_guided) printf("\nVersion: guided\n");
    if(version == omp_sched_dynamic) printf("\nVersion: dinamica\n");
    printf("tsuma=%d\ntdyn-var %d\nnthreads-var %d\nthread-limit-var %d\ntrun-sched-var mod %d\n",suma,dyn_var,nthreads_var, thread_limit_var, run_sched_var);

}

```

CAPTURAS DE PANTALLA:

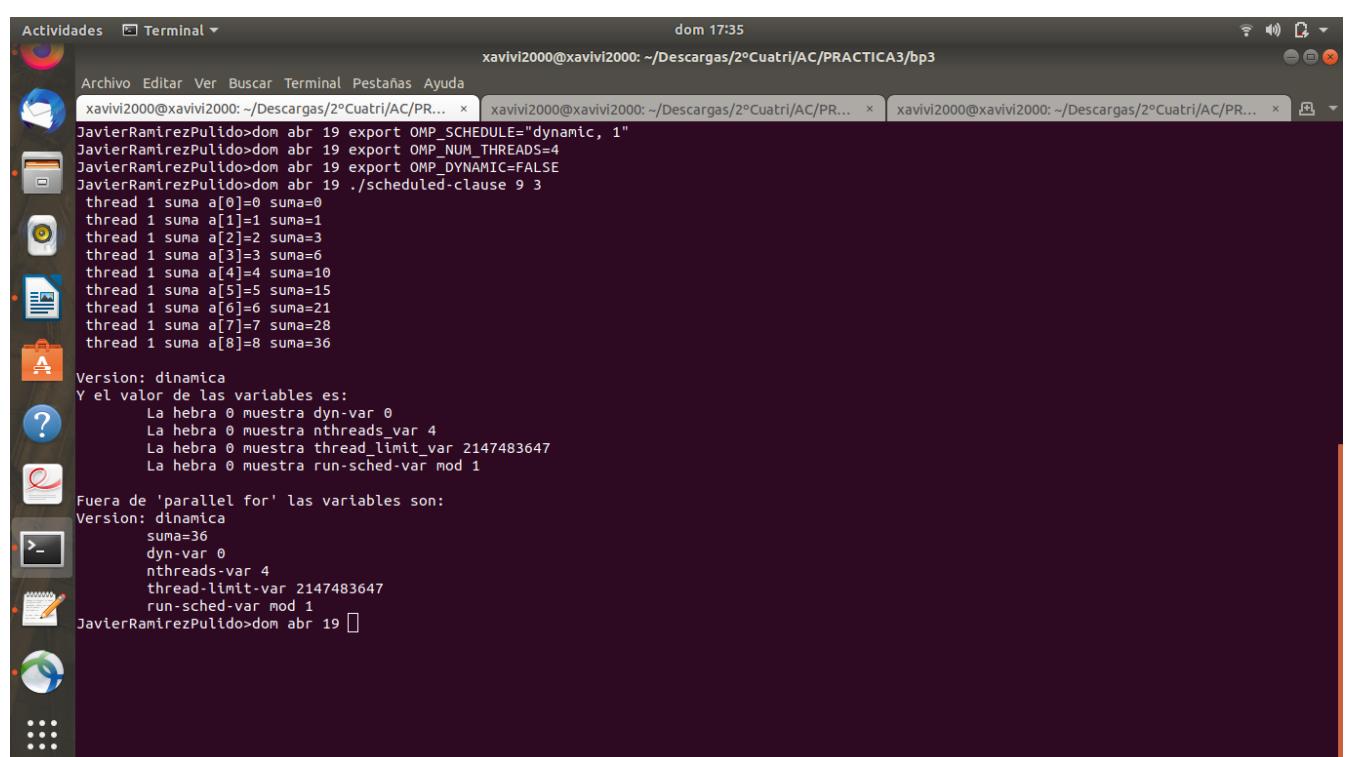
En esta primera captura se han cambiado los siguientes datos:

OMP_SCHEDULE para darle los valores Dynamic y 1

OMP_NUM_THREADS para introducir 4 hebras

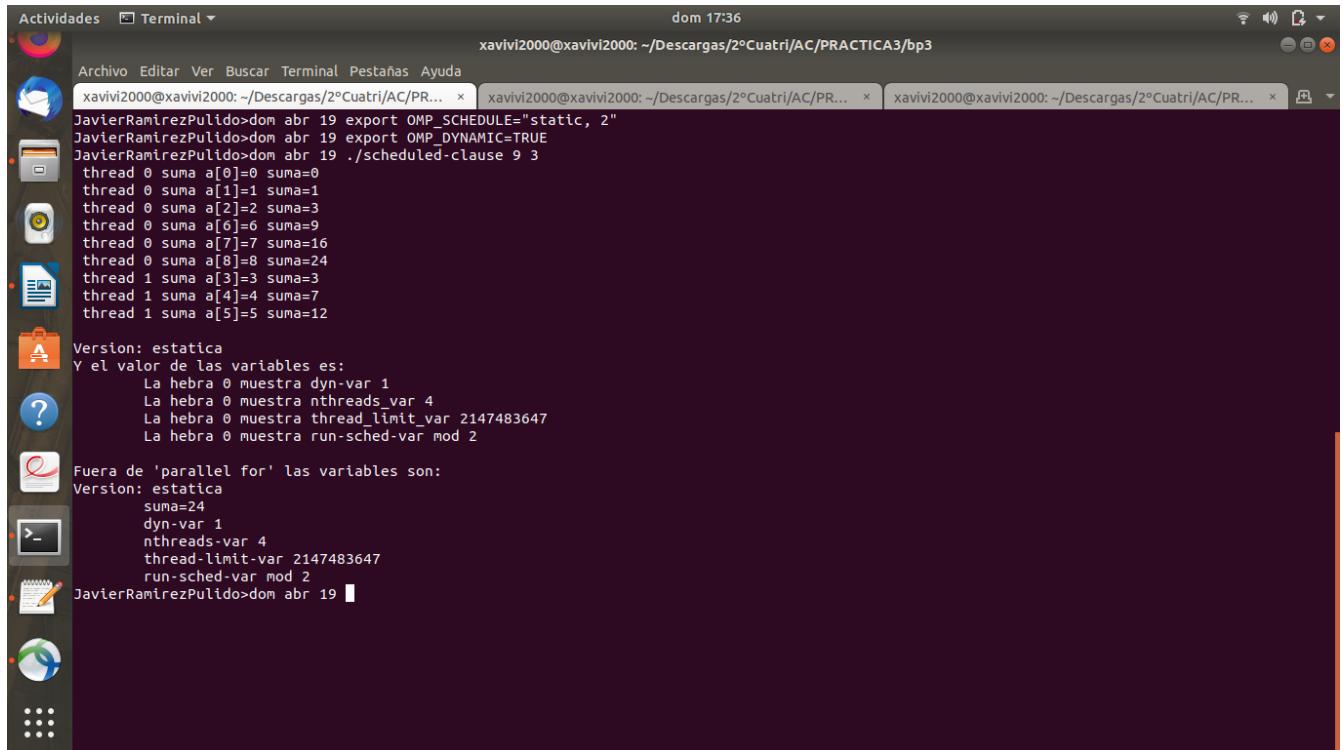
OMP_NOM_THREADS para introducir
OMP_DYNAMIC para ponerlo a false

Y con estas modificaciones, los valores de las variables son:



En esta segunda captura se han cambiado los siguientes datos:
 OMP_SCHEDULE para darle los valores Static y 2
 OMP_DYNAMIC para ponerlo a true.

OMP_NUM_THREADS no se modifica por lo que sigue habiendo 4 hebras
 Y con estas modificaciones, los valores de las variables son:



```

Actividades Terminal dom 17:36
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3
JavierRamirezPulido>dom abr 19 export OMP_SCHEDULE="static, 2"
JavierRamirezPulido>dom abr 19 export OMP_DYNAMIC=true
JavierRamirezPulido>dom abr 19 ./scheduled-clause 9 3
    thread 0 suma a[0]=0 suma=0
    thread 0 suma a[1]=1 suma=1
    thread 0 suma a[2]=2 suma=3
    thread 0 suma a[6]=6 suma=9
    thread 0 suma a[7]=7 suma=16
    thread 0 suma a[8]=8 suma=24
    thread 1 suma a[3]=3 suma=3
    thread 1 suma a[4]=4 suma=7
    thread 1 suma a[5]=5 suma=12

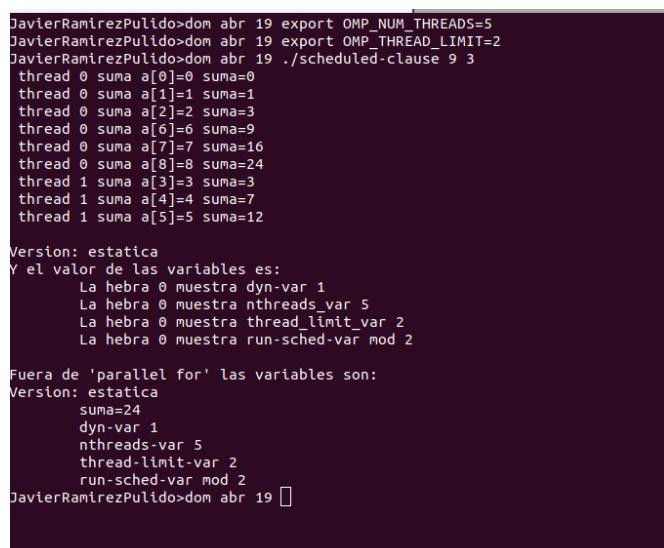
Version: estatica
Y el valor de las variables es:
    La hebra 0 muestra dyn-var 1
    La hebra 0 muestra nthreads_var 4
    La hebra 0 muestra thread_limit_var 2147483647
    La hebra 0 muestra run-sched-var mod 2

Fuera de 'parallel for' las variables son:
Version: estatica
    suma=24
    dyn-var 1
    nthreads-var 4
    thread-limit-var 2147483647
    run-sched-var mod 2
JavierRamirezPulido>dom abr 19

```

En esta tercera captura se han cambiado los siguientes datos:
 OMP_THREAD_LIMIT para ajuste el límite de hebras
 OMP_NUM_THREADS para ponerlo en 5.

OMP_DYNAMIC no se modifica por lo que sigue a true
 OMP_SCHEDULE que sigue en Static y 2
 Y con estas modificaciones, los valores de las variables son:



```

JavierRamirezPulido>dom abr 19 export OMP_NUM_THREADS=5
JavierRamirezPulido>dom abr 19 export OMP_THREAD_LIMIT=2
JavierRamirezPulido>dom abr 19 ./scheduled-clause 9 3
    thread 0 suma a[0]=0 suma=0
    thread 0 suma a[1]=1 suma=1
    thread 0 suma a[2]=2 suma=3
    thread 0 suma a[6]=6 suma=9
    thread 0 suma a[7]=7 suma=16
    thread 0 suma a[8]=8 suma=24
    thread 1 suma a[3]=3 suma=3
    thread 1 suma a[4]=4 suma=7
    thread 1 suma a[5]=5 suma=12

Version: estatica
Y el valor de las variables es:
    La hebra 0 muestra dyn-var 1
    La hebra 0 muestra nthreads_var 5
    La hebra 0 muestra thread_limit_var 2
    La hebra 0 muestra run-sched-var mod 2

Fuera de 'parallel for' las variables son:
Version: estatica
    suma=24
    dyn-var 1
    nthreads-var 5
    thread-limit-var 2
    run-sched-var mod 2
JavierRamirezPulido>dom abr 19

```

En esta cuarta captura se han cambiado los siguientes datos:
 OMP_DYNAMIC que pasa a false
 OMP_NUM_THREADS para ponerlo en 2.
 OMP_SCHEDULE para que sea Guided y 3

OMP_THREAD_LIMIT no se cambia y se mantiene el valor en 2
 Y con estas modificaciones, los valores de las variables son:

```

dom 17:38
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3
JavierRamirezPulido>dom abr 19 export OMP_SCHEDULE="guided, 3"
JavierRamirezPulido>dom abr 19 export OMP_NUM_THREADS=2
JavierRamirezPulido>dom abr 19 export OMP_DYNAMIC=FALSE
JavierRamirezPulido>dom abr 19 ./scheduled-clause 9 3
thread 0 suma a[0]=0 suma=0
thread 0 suma a[1]=1 suma=1
thread 0 suma a[2]=2 suma=3
thread 0 suma a[3]=6 suma=9
thread 0 suma a[7]=7 suma=16
thread 0 suma a[8]=8 suma=24
thread 1 suma a[4]=4 suma=7
thread 1 suma a[5]=5 suma=12

Version: guided
Y el valor de las variables es:
    La hebra 0 muestra dyn-var 0
    La hebra 0 muestra nthreads_var 2
    La hebra 0 muestra thread_limit_var 2
    La hebra 0 muestra run-sched-var mod 3

Fuera de 'parallel for' las variables son:
Version: guided
    suma=24
    dyn-var 0
    nthreads-var 2
    thread-limit-var 2
    run-sched-var mod 3
JavierRamirezPulido>dom abr 19

```

RESPUESTA:

Como conclusión, observamos de qué depende el cambio de valor de las variables:

Tipo -> OMP_SCHEDULE = “ **tipo , num**“

Dyn-var -> OMP_DYNAMIC = [TRUE, FALSE]

Nthreads-var -> OMP_NUM_THREADS = num

Thread-limit-var -> THREAD_LIMIT_VAR = num

Run-sched-var mod -> OMP_SCHEDULE = “ **tipo , num**“

La muestra de los datos es la misma tanto dentro del parallel como fuera.

4. Usar en el ejemplo anterior las funciones `omp_get_num_threads()`, `omp_get_num_procs()` y `omp_in_parallel()` dentro y fuera de la región paralela. Imprimir los valores que obtienen estas funciones dentro (lo debe imprimir sólo uno de los threads) y fuera de la región paralela. Incorporar en su cuaderno de prácticas volcados de pantalla con los resultados de ejecución obtenidos. Indicar en qué funciones se obtienen valores distintos dentro y fuera de la región paralela.

CAPTURA CÓDIGO FUENTE: scheduled-clauseModificado4.c

```

#include<stdio.h>
#include<stdlib.h>
#ifndef _OPENMP
#include<omp.h>
#else
#define omp_get_thread_num() 0
#endif

main(int argc,char**argv)
{
    int i,n=200,chunk,a[n],suma=0;

    if(argc!=3){
        fprintf(stderr,"\nTienes que poner: ./ejecutables iteraciones chunk\n");
        exit(-1);
    }

    n =atoi(argv[1]); if(n>200) n=200;chunk=atoi(argv[2]);

    omp_sched_t version;
    int run_sched_var;
    int nthreads_var = omp_get_max_threads(), dyn_var = omp_get_dynamic(), thread_limit_var = omp_get_thread_limit();
    omp_get_schedule(&version, &run_sched_var);

    for(i=0;i<n;i++) a[i]=i;

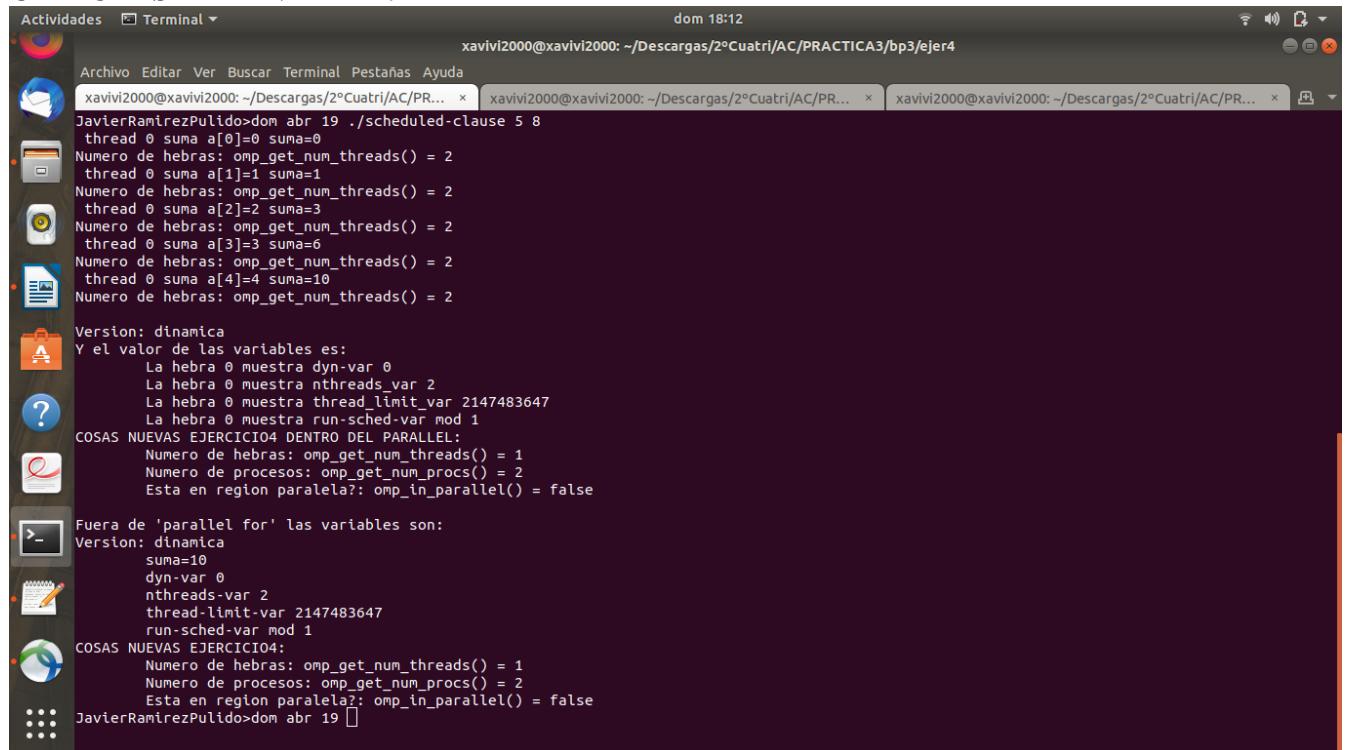
#pragma omp parallel for firstprivate(suma) lastprivate(suma)schedule(dynamic,chunk)
for(i=0;i<n;i++){
    suma =suma +a[i];
    printf(" thread %d suma a[%d]=%d suma=%d \n",omp_get_thread_num(),i,a[i],suma);
    printf("Numero de hebras: omp_get_num_threads() = %d\n", omp_get_num_threads());
}

#pragma omp single //Solo realizará la hebra que llegue antes a esta cláusula.
                //Otra opción sería poner un identificador cualquiera y que únicamente esa hebra entre en el if que contenga el código siguiente
if(version == omp_sched_static) printf("\nVersion: estatica\n");
if(version == omp_sched_guided) printf("\nVersion: guided\n");
if(version == omp_sched_dynamic) printf("\nVersion: dinamica\n");
printf(" el valor de las variables es: \n");
printf("La hebra 0 muestra dyn-var %d \n", omp_get_thread_num(), dyn_var);
printf("La hebra 0 muestra nthreads_var %d \n", omp_get_thread_num(), nthreads_var);
printf("La hebra 0 muestra thread_limit_var %d \n", omp_get_thread_num(), thread_limit_var);
printf("La hebra 0 muestra run-sched-var mod %d \n", omp_get_thread_num(), run_sched_var );
printf("COSAS NUEVAS EJERCICIO4 DENTRO DEL PARALLEL: \n");
printf("\tNúmero de hebras: omp_get_num_threads() = %d\n", omp_get_num_threads());
printf("\tNúmero de procesos: omp_get_num_procs() = %d\n", omp_get_num_procs());
_Bool x =omp_in_parallel();
printf("\tEstá en region paralela?: omp_in_parallel() = "); fputs(x ? "true" : "false", stdout);
printf("\n");

printf("\nFuera de 'parallel for' las variables son:");
if(version == omp_sched_static) printf("\nVersion: estatica\n");
if(version == omp_sched_guided) printf("\nVersion: guided\n");
if(version == omp_sched_dynamic) printf("\nVersion: dinamica\n");
printf("\tsuma=%d\n\tdyn-var %d\n\tthreads-var %d\n\tthread-limit-var %d\n\trun-sched-var mod %d\n",suma,dyn_var,nthreads_var, thread_limit_var, run_sched_var);
printf("COSAS NUEVAS EJERCICIO4: \n");
printf("\tNúmero de hebras: omp_get_num_threads() = %d\n", omp_get_num_threads());
printf("\tNúmero de procesos: omp_get_num_procs() = %d\n", omp_get_num_procs());
x=omp_in_parallel();
printf("\tEstá en region paralela?: omp_in_parallel() = "); fputs(x ? "true" : "false", stdout);
printf("\n");
}

```

CAPTURAS DE PANTALLA:



RESPUESTA: Dentro del bucle el número de hebras es 2 y se ve en el código con la línea:

```
printf("Numero de hebras: omp_get_num_threads() = %d\n", omp_get_num_threads());
```

y en la ejecución con la parte que muestra:

```
Numero de hebras: omp_get_num_threads() = 2
```

Que se repite por cada iteración del bucle.

Una vez nos salimos de este, el número de hebras es 1 tanto dentro como fuera del parallel. El resto de cosas a comprobar, procesos y si está en parallel o no, se mantienen igual.

5. Añadir al programa scheduled-clause.c lo necesario para modificar las variables de control dyn-var, nthreads-var y run-sched-var y para poder imprimir el valor de estas variables antes y después de dicha modificación. Incorporar en su cuaderno de prácticas volcados de pantalla con los resultados de ejecución obtenidos.

CAPTURA CÓDIGO FUENTE: scheduled-clauseModificado5.c

```
#include<stdio.h>
#include<stdlib.h>
#ifndef _OPENMP
#include<omp.h>
#else
#define omp_get_thread_num() 0
#endif

main(int argc,char**argv)
{
    int i,n=200,chunk,a[n],suma=0;

    if(argc!=3){
        fprintf(stderr,"Tienes que poner: ./ejecutables iteraciones chunk\n");
        exit(-1);
    }

    n =atoi(argv[1]); if(n>200) n=200;chunk=atoi(argv[2]);
    omp_sched_t version;
    int run_sched_var;
    int nthreads_var = omp_get_max_threads(), dyn_var = omp_get_dynamic(), thread_limit_var = omp_get_thread_limit();

    for(i=0;i<n;i++) a[i]=i;

    #pragma omp parallel for firstprivate(suma) lastprivate(suma)schedule(dynamic,chunk)
    for(i=0;i<n;i++){
        suma = suma +a[i];
        printf(" thread %d suma a[%d]=%d suma=%d \n",omp_get_thread_num(),i,a[i],suma);
        printf("Numero de hebras: omp_get_num_threads() = %d\n", omp_get_num_threads());
    }

    #pragma omp parallel for firstprivate(suma) lastprivate(suma)schedule(dynamic,chunk)
    for(i=0;i<n;i++){
        suma = suma +a[i];
        printf(" thread %d suma a[%d]=%d suma=%d \n",omp_get_thread_num(),i,a[i],suma);
    }
    #pragma omp single
    omp_get_schedule(&version, &run_sched_var); //Se actualizan las variables

    printf("\nValores actuales: ");
}
```

```

printf("\ndyn-var: %d ", omp_get_dynamic());
printf("\nnthreads-var: %d ", omp_get_dynamic());
printf("\nrun-sched-var: ");
printf("\nttipo: %d", version);
printf("\ntmodificacion: %d", run_sched_var);
printf("\n");

omp_set_dynamic(1);
omp_set_num_threads(5);
omp_set_schedule(omp_sched_static, 2*chunk);

omp_get_schedule(&version, &run_sched_var); //Se actualizan las variables tras los cambios

printf("\nValores modificados: ");
printf("\ndyn-var: %d ", omp_get_dynamic());
printf("\nnthreads-var: %d ", omp_get_dynamic());
printf("\nrun-sched-var: ");
printf("\nttipo: %d", version);
printf("\ntmodificacion: %d", run_sched_var);
printf("\n");

printf("Fuera de 'parallel for' suma=%d\n", suma);
}

```

CAPTURAS DE PANTALLA:

```

Actividades Terminal dom 18:39
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer4
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido@dom abr 19 gcc -fopenmp scheduled-clauseModificado5.c -o scheduled-clause
scheduled-clauseModificado5.c:9::: warning: return type defaults to 'int' [-Wimplicit-int]
^~~~
main(int argc,char**argv)
^~~~

JavierRamirezPulido@dom abr 19 ./scheduled-clause 5 8
thread 0 suma a[0]=0 suma=0
Numero de hebras: omp_get_num_threads() = 2
thread 0 suma a[1]=1 suma=1
Numero de hebras: omp_get_num_threads() = 2
thread 0 suma a[2]=2 suma=3
Numero de hebras: omp_get_num_threads() = 2
thread 0 suma a[3]=3 suma=6
Numero de hebras: omp_get_num_threads() = 2
thread 0 suma a[4]=4 suma=10
Numero de hebras: omp_get_num_threads() = 2
thread 1 suma a[0]=0 suma=10
thread 1 suma a[1]=1 suma=11
thread 1 suma a[2]=2 suma=13
thread 1 suma a[3]=3 suma=16
thread 1 suma a[4]=4 suma=20

Valores actuales:
dyn-var: 0
nthreads-var: 0
run-sched-var:
    tipo: 2
    modificacion: 1

Valores modificados:
dyn-var: 1
nthreads-var: 1
run-sched-var:
    tipo: 1
    modificacion: 16
Fuera de 'parallel for' suma=20
JavierRamirezPulido@dom abr 19

```

RESPUESTA:

La modificaciones realizadas entre una muestra y otra son:

omp_set_dynamic(1); -> Pone a “true” la variable dinámica y te devuelve un 1 tras los cambios
 omp_set_num_threads(5); -> Cambia el numero de hebras a 5
 omp_set_schedule(omp_sched_static, 2*chunk); -> Pone el tipo en estatico que corresponde con 1 y en el valor de la modificación se mete por ejemplo 2*chunk, que en este caso eran 8 (quedando modificación=16)

Tras los cambios se llama de nuevo a esta función:

```
omp_get_schedule(&version, &run_sched_var); //Se actualizan las variables tras los cambios
```

Que actualiza los valores de las variables modificadas.

Resto de ejercicios

6. Implementar un programa secuencial en C que multiplique una matriz triangular por un vector (use variables dinámicas). Compare el orden de complejidad del código que ha implementado con el código que implementó para el producto matriz por vector.

NOTAS: (1) el número de filas/columnas debe ser un argumento de entrada; (2) se debe inicializar las matrices antes del cálculo; (3) se debe imprimir siempre la primera y última componente del resultado antes de que termine el programa.

CAPTURA CÓDIGO FUENTE: pmtv-secuencial.c

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

main(int argc,char**argv)
{
    //Comprueba que la cantidad de argumentos es la correcta
    if(argc!=2){
        printf("\nFalta el numero de elementos del vector\n");
        exit(-1);
    }

    //Declaracion de variables
    unsigned int tamano_total = atoi(argv[1]);
    struct timespec tiempo_anteriores, tiempo_despues;
    double tiempo_ejecucion;

    //Reserva para vectores
    int *vector = (int*) malloc(tamano_total*sizeof(int));
    int *vector_auxiliar = (int*) malloc(tamano_total*sizeof(int));

    //Reserva para matrices
    int **matriz = (int **)malloc(tamano_total*sizeof(int *));
    for(int i = 0; i<tamano_total; i++)
        matriz[i] = (int *) malloc (tamano_total*sizeof(int));

    //Comprueba que la reserva de memoria fue correcta
    if(vector==0 || vector_auxiliar==0 || matriz==0){
        printf("La reserva de espacio para la matriz y el vector ha dado fallo\n");
        exit(-2);
    }

    //Comprueba que la reserva de memoria fue correcta
    for(int i=0; i<tamano_total; i++)
        if(matriz[i] == 0){
            printf("La reserva de espacio ha dado fallo\n");
            exit(-2);
        }
}
```

```

//Rellenar la matriz, vector y vector de resultados
for(int i=0; i<tamano_total; i++){
    for(int j=i; j<tamano_total; j++){
        matriz[i][j] = 1;
    }
    vector[i] = 3;
    vector_auxiliar[i] = 0;
}

//Realiza las cuentas y se mide el tiempo antes y despues de hacerlo para calcular el tiempo de ejecucion
clock_gettime(CLOCK_REALTIME, &tiempo_anterior);

for(int i=0; i<tamano_total; i++){
    for(int j=i; j<tamano_total; j++)
        vector_auxiliar[i] += matriz[i][j] * vector[i];
}

clock_gettime(CLOCK_REALTIME, &tiempo_despues);
tiempo_ejecucion=(double)(tiempo_despues.tv_sec-tiempo_anterior.tv_sec) + (double)((tiempo_despues.tv_nsec-tiempo_anterior.tv_nsec)/(1.e+9));

//Muestra la matriz
printf("\nMatriz: \n");
for(int i=0; i<tamano_total; i++){
    printf("\t");
    for(int j=0; j<tamano_total; j++){
        if(j >= i) printf("%d ", matriz[i][j]);
        else printf("0 ");
    }
    printf("\n");
}

//Muestra el vector por el que multiplica
printf("\nVector: \n");
printf("\t");
for(int i=0; i<tamano_total; i++) printf("%d ", vector[i]);
printf("\n");

//Muestra el vector que contiene los resultados
printf("\nDatos finales: \n");
printf("\t");
for(int i=tamano_total-1; i>=0; i--) printf("%d ", vector_auxiliar[i]);
printf("\n");

//Muestra dimension y tiempo final
printf("\nDimension de vector y matriz: %d\n\nTiempo: %11.9f segundos\n\n", tamano_total, tiempo_ejecucion);

//libera espacio de la matriz
for(int i=0; i<tamano_total; i++) free(matriz[i]);
free(matriz);

//Libera espacio del vector
free(vector);

return 0;
}

```

CAPTURAS DE PANTALLA:

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1  
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1  
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1  
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1  
  
Vector:  
      3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3  
  
Datos finales:  
      3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51  
  
Dimension de vector y matriz: 17  
  
Tiempo: 0.000004462 segundos  
  
JavierRamirezPulido>dom abr 19 ./pmtv-secuencial 6  
  
Matriz:  
      1 1 1 1 1 1  
      0 1 1 1 1 1  
      0 0 1 1 1 1  
      0 0 0 1 1 1  
      0 0 0 0 1 1  
      0 0 0 0 0 1  
  
Vector:  
      3 3 3 3 3 3  
  
Datos finales:  
      3 6 9 12 15 18  
  
Dimension de vector y matriz: 6  
  
Tiempo: 0.000003127 segundos  
  
JavierRamirezPulido>dom abr 19 □
```

```
JavierRamirezPulido>dom abr 19 ./pmtv-secuencial 17

Matriz:
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1
 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1
 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

Vector:
 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3

Datos finales:
 3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51

Dimension de vector y matriz: 17

Tiempo: 0.000004462 segundos

JavierRamirezPulido>dom abr 19
```

7. Implementar en paralelo la multiplicación de una matriz triangular por un vector a partir del código secuencial realizado para el ejercicio anterior utilizando la directiva `for` de OpenMP. El código debe repartir entre los threads las iteraciones del bucle que recorre las filas. Dibujar en el cuaderno de prácticas la descomposición de dominio utilizada (Lección 4/Tema 2) en el código paralelo implementado para asignar tareas a los threads (Lección 5/Tema 2). Añadir lo necesario para que el usuario pueda fijar la planificación de tareas usando la variable de entorno `OMP_SCHEDULE`. Obtener en atcgrid los tiempos de ejecución del código paralelo (usando, como

siempre, -O2 al compilar) que multiplica una matriz triangular por un vector con las alternativas de planificación static, dynamic y guided para chunk de 1, 64 y el chunk por defecto para la alternativa. Use un tamaño de vector N múltiplo del número de cores y de 64 que no sea inferior a 15360. El número de threads en las ejecuciones debe coincidir con el número de cores. Rellenar la Tabla 3 dos veces con los tiempos obtenidos. Representar el tiempo para static, dynamic y guided en función del tamaño del chunk en una gráfica. ¿Qué alternativa ofrece mejores prestaciones? Razone por qué. Incluya los scripts utilizado en el cuaderno de prácticas. NOTA: Nunca ejecute en atcgrid código que imprima todos los componentes del resultado.

Conteste a las siguientes preguntas: (a) ¿Qué valor por defecto usa OpenMP para chunk con static, dynamic y guided? Indique qué ha hecho para obtener este valor por defecto para cada alternativa. (b) ¿Qué número de operaciones de multiplicación y suma realizan cada uno de los threads en la asignación static para cada uno de los chunks? (c) Con la asignación dynamic y guided, ¿qué cree que debe ocurrir con el número de operaciones de multiplicación y suma que realizan cada uno de los threads?

RESPUESTA:

CAPTURA CÓDIGO FUENTE: pmtv-OpenMP.c

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include <omp.h>

main(int argc,char**argv)
{
    //Comprueba que la cantidad de argumentos es la correcta
    if(argc!=2){
        printf("\nFalta el numero de elementos del vector\n");
        exit(-1);
    }

    //Declaracion de variables
    unsigned int tamano_total = atoi(argv[1]);
    double tiempo_antes, tiempo_despues, tiempo_ejecucion;

    //Reserva para vectores
    int *vector = (int*) malloc(tamano_total*sizeof(int));
    int *vector_auxiliar = (int*) malloc(tamano_total*sizeof(int));

    //Reserva para matrices
    int **matriz = (int **)malloc(tamano_total*sizeof(int *));
    for(int i = 0; i<tamano_total; i++)
        matriz[i] = (int *) malloc (tamano_total*sizeof(int));

    //Comprueba que la reserva de memoria fue correcta
    if(vector==0 || vector_auxiliar==0 || matriz==0){
        printf("La reserva de espacio para la matriz y el vector ha dado fallo\n");
        exit(-2);
    }

    //Comprueba que la reserva de memoria fue correcta
    for(int i=0; i<tamano_total; i++)
        if(matriz[i] == 0){
            printf("La reserva de espacio ha dado fallo\n");
            exit(-2);
        }
}
```

```

//Rellenar la matriz, vector y vector de resultados
for(int i=0; i<tamano_total; i++){
    for(int j=i; j<tamano_total; j++)
        matriz[i][j] = j+3;

    vector[i] = 2*i;
    vector_auxiliar[i] = 0;
}

//Realiza las cuentas y se mide el tiempo antes y despues de hacerlo para calcular el tiempo de ejecucion
tiempo_anterior = omp_get_wtime();

for(int i=0; i<tamano_total; i++)
    for(int j=i; j<tamano_total; j++)
        vector_auxiliar[i] += matriz[i][j] * vector[i];

tiempo_despues = omp_get_wtime();
tiempo_ejecucion=tiempo_despues-tiempo_anterior;

if(tamano_total<10){

    //Muestra La matriz
    printf("\nMatriz: \n");
    for(int i=0; i<tamano_total; i++){
        printf("\t");
        for(int j=0; j<tamano_total; j++)
            if(j >= i) printf("%d ", matriz[i][j]);
            else printf("0 ");

        printf("\n");
    }

    //Muestra el vector por el que multiplica
    printf("\nVector: \n");
    printf("\t");
    for(int i=0; i<tamano_total; i++) printf("%d ", vector[i]);
    printf("\n");

    //Muestra el vector que contiene los resultados
    printf("\nDatos finales: \n");
    printf("\t");
    for(int i=tamano_total-1; i>=0; i--) printf("%d ", vector_auxiliar[i]);
    printf("\n");
}

//Muestra dimension y tiempo final
printf("\nDimension de vector y matriz: %d Tiempo: %11.9f segundos\n\n", tamano_total, tiempo_ejecucion);
printf("\nPrimer elemento (matriz[0][0]): %d Ultimo elemento (matriz[%d][%d]): %d \n\n", matriz[0][0], tamano_total-1, tamano_total-1, matriz[tamano_total-1][tamano_total-1]);

//Libera espacio de la matriz
for(int i=0; i<tamano_total; i++) free(matriz[i]);
free(matriz);

//Libera espacio del vector
free(vector);
free(vector_auxiliar);

return 0;
}

```

DESCOMPOSICIÓN DE DOMINIO:

En esta versión, se reparten los valores a calcular, uno por cada hueco del vector resultado, y estos se asignan a cada hebra..

CAPTURAS DE PANTALLA:

```
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10
lun 05:04 ●
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido>lun abr 20 ./pmvt-OpenMP 30
Dimension de vector y matriz: 30    Tiempo: 0.000010688 segundos
Primer elemento (matriz[0][0]): 3    Ultimo elemento (matriz[29][29]): 32
JavierRamirezPulido>lun abr 20 ./pmtv-OpenMP 20
Dimension de vector y matriz: 20    Tiempo: 0.000010500 segundos
Primer elemento (matriz[0][0]): 3    Ultimo elemento (matriz[19][19]): 22
JavierRamirezPulido>lun abr 20 ./pmtv-OpenMP 10
Dimension de vector y matriz: 10    Tiempo: 0.000004492 segundos
Primer elemento (matriz[0][0]): 3    Ultimo elemento (matriz[9][9]): 12
JavierRamirezPulido>lun abr 20 ./pmtv-OpenMP 100
Dimension de vector y matriz: 100    Tiempo: 0.000135674 segundos
Primer elemento (matriz[0][0]): 3    Ultimo elemento (matriz[99][99]): 102
JavierRamirezPulido>lun abr 20 ./pmtv-OpenMP 1000
Dimension de vector y matriz: 1000   Tiempo: 0.010827121 segundos
Primer elemento (matriz[0][0]): 3    Ultimo elemento (matriz[999][999]): 1002
JavierRamirezPulido>lun abr 20 
```

TABLA RESULTADOS, SCRIPT Y GRÁFICA atcgrid

SCRIPT: pmvt-OpenMP_atcgrid.sh

```
#!/bin/bash

#Declaracion de array con los tipos que hay
declare -a versiones=("dynamic" "static" "guided")

#Numero de hebras
export OMP_NUM_THREADS=12

#for que recorra cada tipo de los declarados en el vector
for i in "${versiones[@]}"
do
    echo -e
    echo "*****$i*****"
    echo -e
    let j=0
    #bucle que se hace 3 veces para probar cada tipo de los disponibles
    while [ $j -le 3 ];do
        #Si es 0 coincide con dinamico
        if [ $j -eq 0 ]
        then
            echo "Chunck: 1"
            echo "Tipo: $i"
            export OMP_SCHEDULE="$i,1"
        #Si es 1 coincide con estatico
        elif [ $j -eq 1 ]
        then
            echo "Chunck: Por defecto"
            echo "Tipo: $i"
            export OMP_SCHEDULE="$i"
        #Si es 2 coincide con guided
        elif [ $j -eq 2 ]
        then
            echo "Chunck: 64"
            echo "Tipo: $i"
            export OMP_SCHEDULE="$i,64"
        fi
        #Ejecutamos para 1536
        ./pmtv-OpenMP 15360
        echo "-----"
        #Aumentamos el valor de j para el bucle
        let j=$j+1
    done
done
```

```
JavierRamirezPulido>lun abr 20 ./pmtv-OpenMP_atcgrid.sh
*****
dynamic*****
Chunck: 1
Tipo: dynamic
Dimension de vector y matriz: 15360 Tiempo: 0.288275268 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
Chunck: Por defecto
Tipo: dynamic
Dimension de vector y matriz: 15360 Tiempo: 0.288582314 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
Chunck: 64
Tipo: dynamic
Dimension de vector y matriz: 15360 Tiempo: 0.288999062 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
Dimension de vector y matriz: 15360 Tiempo: 0.288409382 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
```

```
Actividades Terminal lun 11:19 •
c3estudiante18@atcgrid:~/bp3/ejer7
Dimension de vector y matriz: 15360 Tiempo: 0.288999062 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
Dimension de vector y matriz: 15360 Tiempo: 0.288409382 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
*****static*****
? Chunck: 1
Tipo: static
Dimension de vector y matriz: 15360 Tiempo: 0.288375922 segundos
>- Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
? Chunck: Por defecto
Tipo: static
Dimension de vector y matriz: 15360 Tiempo: 0.288968347 segundos
>- Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
? Chunck: 64
Tipo: static
Dimension de vector y matriz: 15360 Tiempo: 0.288681060 segundos
>- Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
? Chunck: 64
Tipo: static
Dimension de vector y matriz: 15360 Tiempo: 0.288899086 segundos
>- Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
*****guided*****
? Chunck: 1
Tipo: guided
```

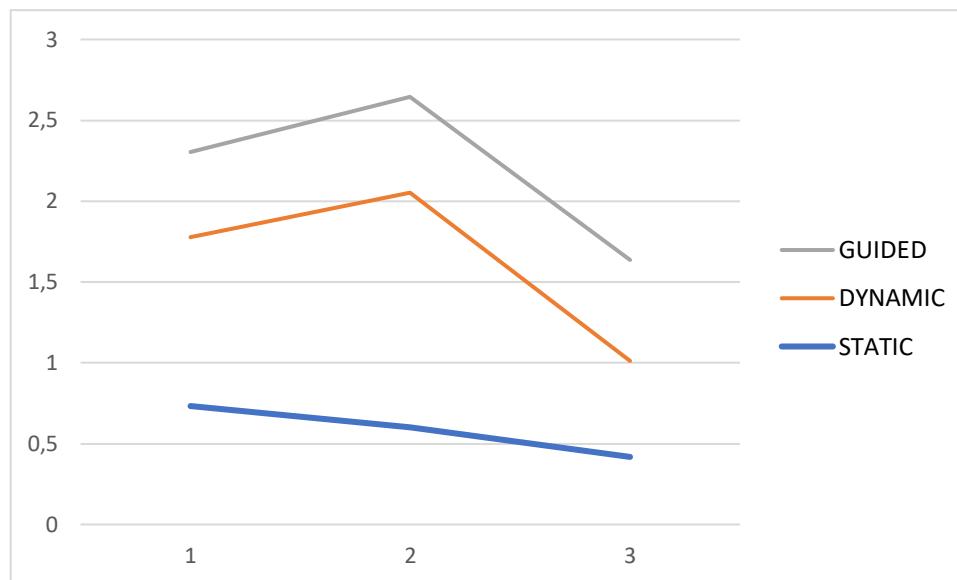
```

Actividades Terminal lun 11:19 *
c3estudiante18@atcgrid:~/bp3/ejer7
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer7 x xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR...
*****guided*****
Chunck: 1
Tipo: guided
Dimension de vector y matriz: 15360 Tiempo: 0.288347457 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
Chunck: Por defecto
Tipo: guided
Dimension de vector y matriz: 15360 Tiempo: 0.288668282 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
Chunck: 64
Tipo: guided
Dimension de vector y matriz: 15360 Tiempo: 0.288576040 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
Dimension de vector y matriz: 15360 Tiempo: 0.287902143 segundos
Primer elemento (matriz[0][0]): 3 Ultimo elemento (matriz[15359][15359]): 15362
-----
JavierRamirezPulido@lun:~$ 

```

Tabla 3 . Tiempos de ejecución de la versión paralela del producto de una matriz triangular por un vector r para vectores de tamaño N= 1360 , 12 threads

Chunk	Static	Dynamic	Guided
por defecto	0,732546324	1,045724893	0,524572834
1	0,600342565	1,451983725	0,592734673
64	0,417813497	0,59387341	0,624598723
Chunk	Static	Dynamic	Guided
por defecto	0,735274643	1,053891743	0,542983743
1	0,610387264	1,223742387	0,701398478
64	0,442394719	0,643873294	0,601938134



8. Implementar un programa secuencial en C que calcule la multiplicación de matrices cuadradas, B y C:

$$A = B \bullet C; A(i, j) = \sum_{k=0}^{N-1} B(i, k) \bullet C(k, j), i, j = 0, \dots, N - 1$$

NOTAS: (1) el número de filas/columnas debe ser un argumento de entrada; (2) se deben inicializar las matrices antes del cálculo; (3) se debe imprimir siempre las componentes (0,0) y (N-1, N-1) del resultado antes de que termine el programa.

CAPTURA CÓDIGO FUENTE: pmm-secuencial.c

```

#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#ifndef _OPENMP
    #include <omp.h>
#else
    #define omp_get_thread_num() 0
    #define omp_get_num_threads() 1
    #define omp_set_num_threads(int)
    #define omp_in_parallel() 0
    #define omp_set_dynamic(int)
#endif

main(int argc,char**argv)
{
    //Comprueba que la cantidad de argumentos es la correcta
    if(argc!=2){
        printf("\nFalta el numero de elementos de la primera_matriz\n");
        exit(-1);
    }

    //Declaracion de variables
    unsigned int tamano_total = atoi(argv[1]);
    double tiempo_antes, tiempo_despues, tiempo_ejecucion;

    //Reserva para matrices
    int **primera_matriz = (int**) malloc(tamano_total*sizeof(int*));
    int **segunda_matriz = (int**) malloc(tamano_total*sizeof(int*));
    int **matriz = (int **)malloc(tamano_total*sizeof(int *));

    for(int i = 0; i<tamano_total; i++){
        matriz[i] = (int *) malloc (tamano_total*sizeof(int));
        primera_matriz[i] = (int *) malloc (tamano_total*sizeof(int));
        segunda_matriz[i] = (int *) malloc (tamano_total*sizeof(int));
    }

    //Comprueba que la reserva de memoria fue correcta
    if(primera_matriz==0 || segunda_matriz==0 || matriz==0){
        printf("La reserva de espacio para las matrices ha dado fallo\n");
        exit(-2);
    }

    //Comprueba que la reserva de memoria fue correcta
    for(int i=0; i<tamano_total; i++){
        if(primera_matriz[i]==0 || segunda_matriz[i]==0 || matriz[i]==0){
            printf("La reserva de espacio ha dado fallo\n");
            exit(-2);
        }
    }

    //Rellenar la matriz, primera_matriz y primera_matriz de resultados
    for(int i=0; i<tamano_total; i++){
        for(int j=0; j<tamano_total; j++){
            matriz[i][j] = 0;
            primera_matriz[i][j] = j*j;
            segunda_matriz[i][j] = j+j;
        }
    }

    //Realiza las cuentas y se mide el tiempo antes y despues de hacerlo para calcular el tiempo de ejecucion
    tiempo_antes = omp_get_wtime();

    for(int i=0; i<tamano_total; i++)
        for(int j=0; j<tamano_total; j++)
            for(int k=0; k<tamano_total; k++)
                matriz[i][j] += primera_matriz[i][k] * segunda_matriz[k][j];

    tiempo_despues = omp_get_wtime();
    tiempo_ejecucion=tiempo_despues-tiempo_antes;
}

```

```

//Muestra dimension y tiempo final
printf("\nDimension de las matrices: %d    Tiempo: %11.9f segundos\n\n", tamano_total, tiempo_ejecucion);
printf("\nPrimer elemento (matriz_resultado[0][0]) = %d,    Ultimo elemento (matriz_resultado[%d][%d]) = %d\n\n",
    matriz[0][0],tamano_total-1,tamano_total-1,matriz[tamano_total-1][tamano_total-1]);
public int __cdecl printf (const char * __restrict__ _Format, ...)

return 0;
}

```

PARTE EXTRA DEL CODIGO PARA MOSTRAR MATRICES SIEMPRE Y CUANDO EL TAMAÑO NO SEA LO SUFFICIENTEMENTE GRANDE PARA MANCHAR LA SALIDA DE DATOS POR PANTALLA

```

if(tamano_total < 10){

    //Muestra la matriz1
    printf("\nMatriz 1: \n");
    for(int i=0; i<tamano_total; i++){
        for(int j=0; j<tamano_total; j++)
            printf("\t%d ", primera_matriz[i][j]);

        printf("\n");
    }

    //Muestra la matriz2
    printf("\nMatriz 2: \n");
    for(int i=0; i<tamano_total; i++){
        for(int j=0; j<tamano_total; j++)
            printf("\t%d ", segunda_matriz[i][j]);

        printf("\n");
    }

    //Muestra la matriz resultado
    printf("\nMatriz Resultado: \n");
    for(int i=0; i<tamano_total; i++){
        for(int j=0; j<tamano_total; j++)
            printf("\t%d ", matriz[i][j]);

        printf("\n");
    }
}

else{
    printf("\nPrimer elemento (matriz_resultado[0][0]) = %d,    Ultimo elemento (matriz_resultado[%d][%d]) = %d\n\n",
        matriz[0][0],tamano_total-1,tamano_total-1,matriz[tamano_total-1][tamano_total-1]);
}

```

CAPTURAS DE PANTALLA:

```
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10
JavierRamirezPulido>lun abr 20 ./pmm-secuencial 15
Primer elemento (matriz_resultado[0][0]) = 420, Ultimo elemento (matriz_resultado[14][14]) = 3360
Dimension de las matrices: 15 Tiempo: 0.000129491 segundos

JavierRamirezPulido>lun abr 20 ./pmm-secuencial 10
Primer elemento (matriz_resultado[0][0]) = 180, Ultimo elemento (matriz_resultado[9][9]) = 990
Dimension de las matrices: 10 Tiempo: 0.000031387 segundos

JavierRamirezPulido>lun abr 20 ./pmm-secuencial 12
Primer elemento (matriz_resultado[0][0]) = 264, Ultimo elemento (matriz_resultado[11][11]) = 1716
Dimension de las matrices: 12 Tiempo: 0.000071487 segundos

JavierRamirezPulido>lun abr 20 ./pmm-secuencia_salida_reducida 8
Dimension de las matrices: 8 Tiempo: 0.000020347 segundos

Primer elemento (matriz_resultado[0][0]) = 112, Ultimo elemento (matriz_resultado[7][7]) = 504
JavierRamirezPulido>lun abr 20 ./pmm-secuencia_salida_reducida 4
Dimension de las matrices: 4 Tiempo: 0.000005768 segundos

Primer elemento (matriz_resultado[0][0]) = 24, Ultimo elemento (matriz_resultado[3][3]) = 60
JavierRamirezPulido>lun abr 20 
```

EXTRA CON LA MUESTRA DE LAS PANTALLAS COMO COMPROBACION DE RESULTADOS

```
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10
JavierRamirezPulido>lun abr 20 ./pmm-secuencial 8
Matriz 1:
 0   2   4   6   8   10  12  14
 0   2   4   6   8   10  12  14
 0   2   4   6   8   10  12  14
 0   2   4   6   8   10  12  14
 0   2   4   6   8   10  12  14
 0   2   4   6   8   10  12  14
 0   2   4   6   8   10  12  14
 0   2   4   6   8   10  12  14

Matriz 2:
 2   3   4   5   6   7   8   9
 2   3   4   5   6   7   8   9
 2   3   4   5   6   7   8   9
 2   3   4   5   6   7   8   9
 2   3   4   5   6   7   8   9
 2   3   4   5   6   7   8   9
 2   3   4   5   6   7   8   9
 2   3   4   5   6   7   8   9

Matriz Resultado:
 112   168   224   280   336   392   448   504
 112   168   224   280   336   392   448   504
 112   168   224   280   336   392   448   504
 112   168   224   280   336   392   448   504
 112   168   224   280   336   392   448   504
 112   168   224   280   336   392   448   504
 112   168   224   280   336   392   448   504
 112   168   224   280   336   392   448   504

Dimension de las matrices: 8 Tiempo: 0.000021014 segundos
JavierRamirezPulido>lun abr 20 
```

9. Implementar en paralelo la multiplicación de matrices cuadradas con OpenMP a partir del código escrito en el ejercicio anterior. Use las directivas, las cláusulas y las funciones de entorno que considere oportunas. Se debe parallelizar también la inicialización de las matrices. Dibuje en su cuaderno de prácticas la descomposición

de dominio que ha utilizado en el código paralelo implementado para asignar tareas a los threads (Lección 4/Tema 2,Lección 5/Tema 2).

10.

DESCOMPOSICIÓN DE DOMINIO: El reparto se realiza únicamente con filas de la primera matriz. Cualquier otro elemento de esta u otra matriz es recorrida por todas las hebras.

CAPTURA CÓDIGO FUENTE: pmm-OpenMP.c

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#ifndef _OPENMP
    #include <omp.h>
#else
    #define omp_get_thread_num() 0
    #define omp_get_num_threads() 1
    #define omp_set_num_threads(int)
    #define omp_in_parallel() 0
    #define omp_set_dynamic(int)
#endif

main(int argc,char**argv)
{
    //Comprueba que la cantidad de argumentos es la correcta
    if(argc!=2){
        printf("\nFalta el numero de elementos del primera_matriz\n");
        exit(-1);
    }

    //Declaracion de variables
    unsigned int tamano_total = atoi(argv[1]);
    double tiempo_antes, tiempo_despues, tiempo_ejecucion;

    //Reserva para matrices
    int **primera_matriz = (int**) malloc(tamano_total*sizeof(int *));
    int **segunda_matriz = (int**) malloc(tamano_total*sizeof(int *));
    int **matriz = (int **)malloc(tamano_total*sizeof(int *));
    for(int i = 0; i<tamano_total; i++){
        matriz[i] = (int *) malloc (tamano_total*sizeof(int));
        primera_matriz[i] = (int *) malloc (tamano_total*sizeof(int));
        segunda_matriz[i] = (int *) malloc (tamano_total*sizeof(int));
    }
}
```

```

//Comprueba que la reserva de memoria fue correcta
if(primer_matriz==0 || segunda_matriz==0 || matriz==0){
    printf("La reserva de espacio para las matrices ha dado fallo\n");
    exit(-2);
}

//Comprueba que la reserva de memoria fue correcta
for(int i=0; i<tamano_total; i++)
    if(primer_matriz[i]==0 || segunda_matriz[i]==0 || matriz[i]==0){
        printf("La reserva de espacio ha dado fallo\n");
        exit(-2);
    }

//Rellenar la matriz, primer_matriz y primera_matriz de resultados
int j,k;
#pragma omp parallel for private(j)
for(int i=0; i<tamano_total; i++){
    for(j=0; j<tamano_total; j++){
        matriz[i][j] = 0;
        primer_matriz[i][j] = i+2;
        segunda_matriz[i][j] = i+9;
    }
}

//Realiza las cuentas y se mide el tiempo antes y despues de hacerlo para calcular el tiempo de ejecucion
tiempo_antes = omp_get_wtime();

#pragma omp parallel for private(j,k)
for(int i=0; i<tamano_total; i++)
    for( j=0; j<tamano_total; j++)
        for( k=0; k<tamano_total; k++)
            matriz[i][j] += primer_matriz[i][k] * segunda_matriz[k][j];

tiempo_despues = omp_get_wtime();
tiempo_ejecucion=tiempo_despues-tiempo_antes;

//Muestra dimension y tiempo final
printf("\nDimension de las matrices: %d    Tiempo: %11.9f segundos\n\n", tamano_total, tiempo_ejecucion);
printf("\nPrimer elemento (matriz_resultado[0][0]) = %d,    Ultimo elemento (matriz_resultado[%d][%d]) = %d\n\n",
    matriz[0][0],tamano_total-1,tamano_total-1,matriz[tamano_total-1][tamano_total-1]);

return 0;

```

PARTE EXTRA DEL CODIGO PARA MOSTRAR MATRICES SIEMPRE Y CUANDO NO SEAN DEMASIADO GRANDES

```

if(tamano_total < 10){

    //Muestra la matriz1
    printf("\nMatriz 1: \n");
    for(int i=0; i<tamano_total; i++){
        for(j=0; j<tamano_total; j++)
            printf("\t%d ", primera_matriz[i][j]);
        printf("\n");
    }

    //Muestra la matriz2
    printf("\nMatriz 2: \n");
    for(int i=0; i<tamano_total; i++){
        for( j=0; j<tamano_total; j++)
            printf("\t%d ", segunda_matriz[i][j]);
        printf("\n");
    }

    //Muestra la matriz resultado
    printf("\nMatriz Resultado: \n");
    for(int i=0; i<tamano_total; i++){
        for( j=0; j<tamano_total; j++)
            printf("\t%d ", matriz[i][j]);
        printf("\n");
    }

}else{
    printf("\nPrimer elemento (matriz_resultado[0][0]) = %d, Ultimo elemento (matriz_resultado[%d][%d]) = %d\n",
    matriz[0][0],tamano_total-1,tamano_total-1,matriz[tamano_total-1][tamano_total-1]);
}

```

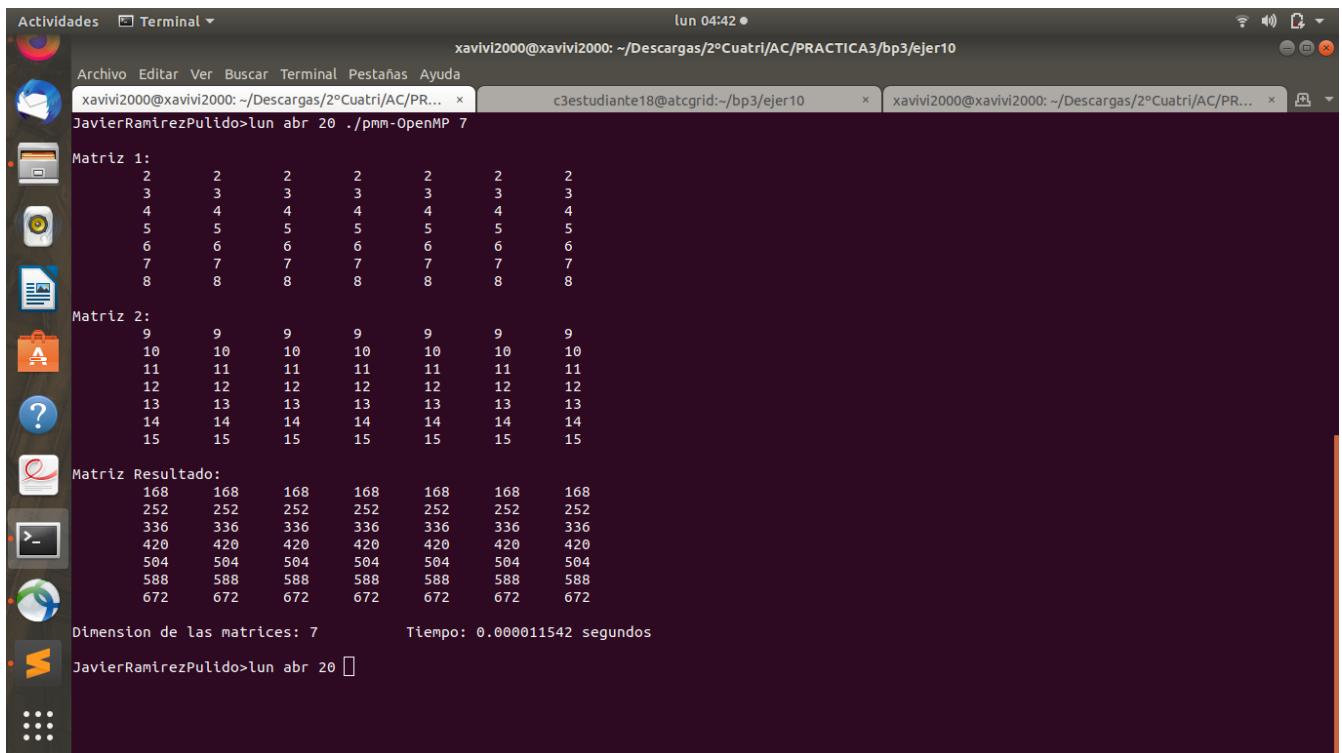
CAPTURAS DE PANTALLA:

```

Actividades Terminal lun 04:41 ●
Terminal xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido>lun abr 20 ./pmm-OpenMP 15
Primer elemento (matriz_resultado[0][0]) = 480, Ultimo elemento (matriz_resultado[14][14]) = 3840
Dimension de las matrices: 15 Tiempo: 0.000076728 segundos
JavierRamirezPulido>lun abr 20 ./pmm-OpenMP 10
Primer elemento (matriz_resultado[0][0]) = 270, Ultimo elemento (matriz_resultado[9][9]) = 1485
Dimension de las matrices: 10 Tiempo: 0.000066259 segundos
JavierRamirezPulido>lun abr 20 ./pmm-OpenMP 12
Primer elemento (matriz_resultado[0][0]) = 348, Ultimo elemento (matriz_resultado[11][11]) = 2262
Dimension de las matrices: 12 Tiempo: 0.000039350 segundos
JavierRamirezPulido>lun abr 20 ./pmm-OpenMP_salida_reducida 8
Dimension de las matrices: 8 Tiempo: 0.000014550 segundos
Primer elemento (matriz_resultado[0][0]) = 200, Ultimo elemento (matriz_resultado[7][7]) = 900
JavierRamirezPulido>lun abr 20 []

```

EXTRA CON LA MUESTRA DE LAS PANTALLAS COMO COMPROBACION DE RESULTADOS



The screenshot shows a terminal window with three tabs. The active tab displays the output of a matrix multiplication script named pmm-OpenMP.sh. The script takes two 7x7 matrices as input and prints the resulting 7x7 matrix. It also outputs the dimension of the matrices (7), the execution time (0.000011542 seconds), and the user's name (JavierRamirezPulido). The terminal window has a dark theme and includes a sidebar with various icons.

```

Actividades Terminal lun 04:42 ●
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
JavierRamirezPulido>lun abr 20 ./pmm-OpenMP 7

Matriz 1:
 2   2   2   2   2   2   2
 3   3   3   3   3   3   3
 4   4   4   4   4   4   4
 5   5   5   5   5   5   5
 6   6   6   6   6   6   6
 7   7   7   7   7   7   7
 8   8   8   8   8   8   8

Matriz 2:
 9   9   9   9   9   9   9
10  10  10  10  10  10  10
11  11  11  11  11  11  11
12  12  12  12  12  12  12
13  13  13  13  13  13  13
14  14  14  14  14  14  14
15  15  15  15  15  15  15

Matriz Resultado:
168   168   168   168   168   168   168
252   252   252   252   252   252   252
336   336   336   336   336   336   336
420   420   420   420   420   420   420
504   504   504   504   504   504   504
588   588   588   588   588   588   588
672   672   672   672   672   672   672

Dimension de las matrices: 7 Tiempo: 0.000011542 segundos
JavierRamirezPulido>lun abr 20 []

```

10. Hacer un estudio de escalabilidad (ganancia en velocidad en función del número de cores) en atcgrid y en su PC del código paralelo implementado para dos tamaños de las matrices. Debe recordar usar -O2 al compilar. El número de núcleos máximo en este estudio debe ser el igual al de núcleos físicos del computador. Presente los resultados del estudio en tablas de valores y en gráficas. Escoger los tamaños de manera que se observe diferentes curvas de escalabilidad en las gráficas que entregue en su cuaderno de prácticas (pruebe con valores de N entre 100 y 1500). Consulte la Lección 6/Tema 2. Incluya los scripts utilizado en el cuaderno de prácticas. NOTA: Nunca ejecute en atcgrid código que imprima todos los componentes del resultado.

ESTUDIO DE ESCALABILIDAD EN atcgrid:

SCRIPT: pmm-OpenMP_atcgrid.sh

```
#!/bin/bash

echo "Secuencial:"
./pmm-secuencia_salida_reducida 105
./pmm-secuencia_salida_reducida 500
./pmm-secuencia_salida_reducida 750
./pmm-secuencia_salida_reducida 900

echo "-----"
echo "Paralelo con 2 threads:"
export OMP_NUM_THREADS=2

./pmm-OpenMP_salida_reducida 105
./pmm-OpenMP_salida_reducida 500
./pmm-OpenMP_salida_reducida 750
./pmm-OpenMP_salida_reducida 900

echo "-----"
echo "Paralelo con 3 threads:"
export OMP_NUM_THREADS=3

./pmm-OpenMP_salida_reducida 105
./pmm-OpenMP_salida_reducida 500
./pmm-OpenMP_salida_reducida 750
./pmm-OpenMP_salida_reducida 900

echo "-----"
echo "Paralelo con 4 threads:"
export OMP_NUM_THREADS=4

./pmm-OpenMP_salida_reducida 105
./pmm-OpenMP_salida_reducida 500
./pmm-OpenMP_salida_reducida 750
./pmm-OpenMP_salida_reducida 900

echo "-----"
echo "Paralelo con 5 threads:"
export OMP_NUM_THREADS=5

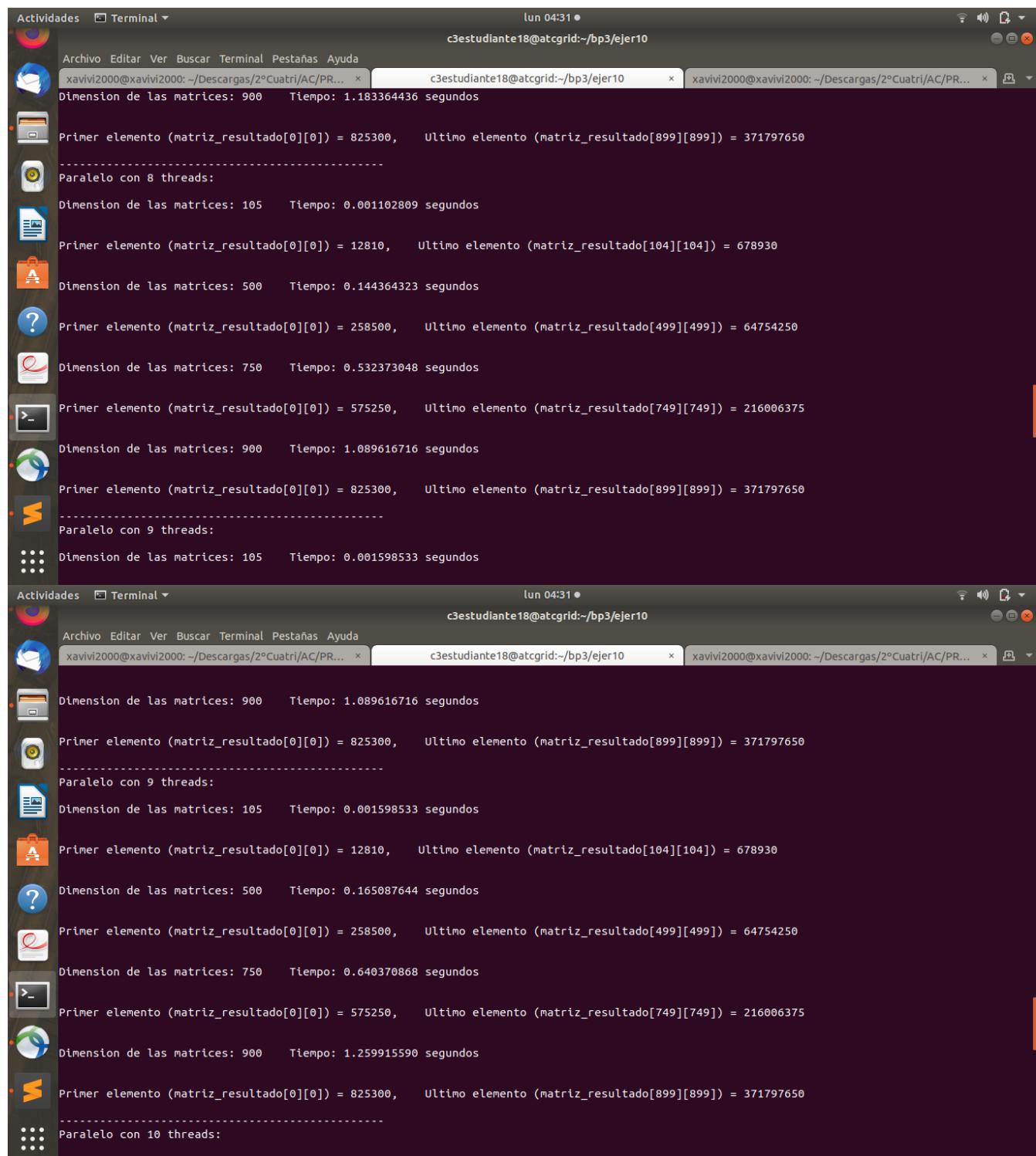
./pmm-OpenMP_salida_reducida 105
./pmm-OpenMP_salida_reducida 500
./pmm-OpenMP_salida_reducida 750
./pmm-OpenMP_salida_reducida 900
```

```
echo "-----"  
echo "Paralelo con 6 threads;"  
export OMP_NUM_THREADS=6  
  
.pmm-OpenMP_salida_reducida 105  
.pmm-OpenMP_salida_reducida 500  
.pmm-OpenMP_salida_reducida 750  
.pmm-OpenMP_salida_reducida 900  
  
echo "-----"  
echo "Paralelo con 7 threads;"  
export OMP_NUM_THREADS=7  
  
.pmm-OpenMP_salida_reducida 105  
.pmm-OpenMP_salida_reducida 500  
.pmm-OpenMP_salida_reducida 750  
.pmm-OpenMP_salida_reducida 900  
  
echo "-----"  
echo "Paralelo con 8 threads;"  
export OMP_NUM_THREADS=8  
  
.pmm-OpenMP_salida_reducida 105  
.pmm-OpenMP_salida_reducida 500  
.pmm-OpenMP_salida_reducida 750  
.pmm-OpenMP_salida_reducida 900  
  
echo "-----"  
echo "Paralelo con 9 threads;"  
export OMP_NUM_THREADS=9  
  
.pmm-OpenMP_salida_reducida 105  
.pmm-OpenMP_salida_reducida 500  
.pmm-OpenMP_salida_reducida 750  
.pmm-OpenMP_salida_reducida 900  
  
echo "-----"  
echo "Paralelo con 10 threads;"  
export OMP_NUM_THREADS=10  
  
.pmm-OpenMP_salida_reducida 105  
.pmm-OpenMP_salida_reducida 500  
.pmm-OpenMP_salida_reducida 750  
.pmm-OpenMP_salida_reducida 900  
  
echo "-----"  
echo "Paralelo con 11 threads;"  
export OMP_NUM_THREADS=11  
  
.pmm-OpenMP_salida_reducida 105  
.pmm-OpenMP_salida_reducida 500  
.pmm-OpenMP_salida_reducida 750  
.pmm-OpenMP_salida_reducida 900  
  
echo "-----"  
echo "Paralelo con 12 threads;"  
export OMP_NUM_THREADS=12  
  
.pmm-OpenMP_salida_reducida 105  
.pmm-OpenMP_salida_reducida 500  
.pmm-OpenMP_salida_reducida 750  
.pmm-OpenMP_salida_reducida 900
```

```
Actividades Terminal lun 04:31 •
c3estudiante18@atcgrid:~/bp3/ejer10
Dimension de las matrices: 900 Tiempo: 3.461814411 segundos
Primer elemento (matriz_resultado[0][0]) = 1618200, Ultimo elemento (matriz_resultado[899][899]) = 728999100
-----
Paralelo con 2 threads:
Dimension de las matrices: 105 Tiempo: 0.002161410 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.298491396 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 1.005357265 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.774483219 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 3 threads:
Dimension de las matrices: 105 Tiempo: 0.001436546 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
Dimension de las matrices: 900 Tiempo: 1.774483219 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 3 threads:
Dimension de las matrices: 105 Tiempo: 0.001436546 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.200297631 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.670570720 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.173172873 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 4 threads:
Dimension de las matrices: 105 Tiempo: 0.001133881 segundos
```

```
Actividades Terminal lun 04:31 • c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x
-----
Paralelo con 4 threads:
Dimension de las matrices: 105 Tiempo: 0.001133081 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.153328378 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 1.023097731 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 0.911827680 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 5 threads:
Dimension de las matrices: 105 Tiempo: 0.001674604 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.220730610 segundos
lun 04:31 • Actividades Terminal c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 5 threads:
Dimension de las matrices: 105 Tiempo: 0.001674604 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.220730640 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.830438547 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.665327512 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 6 threads:
Dimension de las matrices: 105 Tiempo: 0.001436453 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
```

```
Actividades Terminal lun 04:31 • c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x
-----
Paralelo con 6 threads:
Dimension de las matrices: 105 Tiempo: 0.001436453 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.188782107 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.717739888 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.357582793 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Actividades Terminal lun 04:31 • c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x
Dimension de las matrices: 900 Tiempo: 1.357582793 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 7 threads:
Dimension de las matrices: 105 Tiempo: 0.001227006 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.162805911 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.605717104 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.183364436 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 8 threads:
Dimension de las matrices: 105 Tiempo: 0.001102800 segundos
```



```
Actividades Terminal lun 04:31 • c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR...
Dimension de las matrices: 900 Tiempo: 1.183364436 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 8 threads:
Dimension de las matrices: 105 Tiempo: 0.001102809 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.144364323 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.532373048 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.089616716 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 9 threads:
Dimension de las matrices: 105 Tiempo: 0.001598533 segundos
Actividades Terminal lun 04:31 • c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR...
Dimension de las matrices: 900 Tiempo: 1.089616716 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 9 threads:
Dimension de las matrices: 105 Tiempo: 0.001598533 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.165087644 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.640370868 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.259915590 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 10 threads:
Dimension de las matrices: 105 Tiempo: 0.001499431 segundos
```

Actividades Terminal lun 04:31 ● c3estudiante18@atcgrid:~/bp3/ejer10

```
Dimension de las matrices: 900 Tiempo: 1.259915590 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 10 threads:
Dimension de las matrices: 105 Tiempo: 0.001489434 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.148785044 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.581900135 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.146418415 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 11 threads:
Dimension de las matrices: 105 Tiempo: 0.001617875 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.158670682 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.566711653 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.115870520 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 12 threads:
Dimension de las matrices: 105 Tiempo: 0.001579460 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
```

Actividades Terminal lun 04:32 ● c3estudiante18@atcgrid:~/bp3/ejer10

```
Dimension de las matrices: 900 Tiempo: 1.259915590 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 11 threads:
Dimension de las matrices: 105 Tiempo: 0.001617875 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.158670682 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.566711653 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.115870520 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 12 threads:
Dimension de las matrices: 105 Tiempo: 0.001579460 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
```

```

Actividades Terminal lun 04:32 ●
c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.115870520 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 12 threads:
Dimension de las matrices: 105 Tiempo: 0.001579460 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 0.153181072 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 0.570999868 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 1.118343119 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
JavierRamirezPulido>lun abr 20 [ ]
Actividades Terminal lun 04:20 ●
c3estudiante18@atcgrid:~/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000:~/Descargas/2ºCuatri/AC/PR... x
JavierRamirezPulido>lun abr 20 ./pmm-OpenMP_atcgrid.sh
Secuencial:
Dimension de las matrices: 105 Tiempo: 0.006768294 segundos
Primer elemento (matriz_resultado[0][0]) = 21840, Ultimo elemento (matriz_resultado[104][104]) = 1157520
Dimension de las matrices: 500 Tiempo: 0.578423958 segundos
Primer elemento (matriz_resultado[0][0]) = 499000, Ultimo elemento (matriz_resultado[499][499]) = 124999500
Dimension de las matrices: 750 Tiempo: 1.954933774 segundos
Primer elemento (matriz_resultado[0][0]) = 1123500, Ultimo elemento (matriz_resultado[749][749]) = 421874250
Dimension de las matrices: 900 Tiempo: 3.461814411 segundos
Primer elemento (matriz_resultado[0][0]) = 1618200, Ultimo elemento (matriz_resultado[899][899]) = 728999100
-----
Paralelo con 2 threads:
Dimension de las matrices: 105 Tiempo: 0.002161410 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930

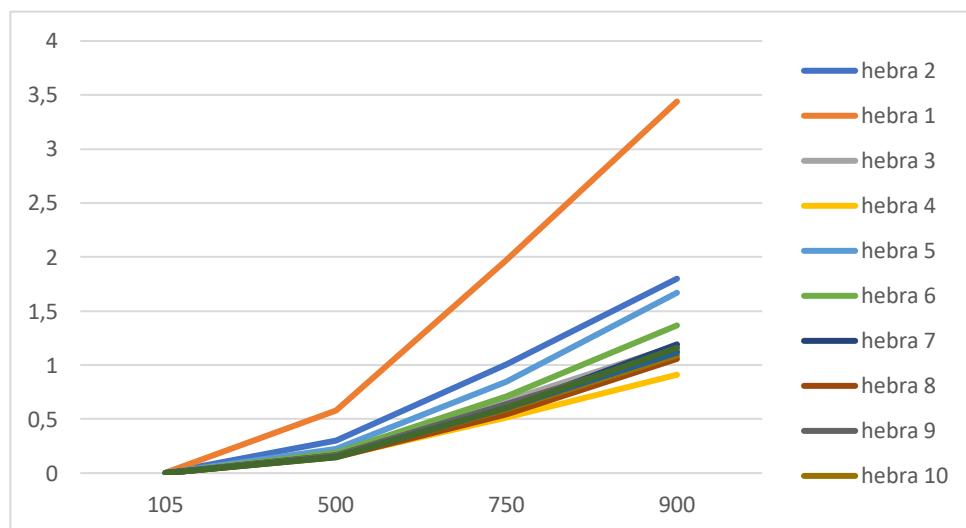
```

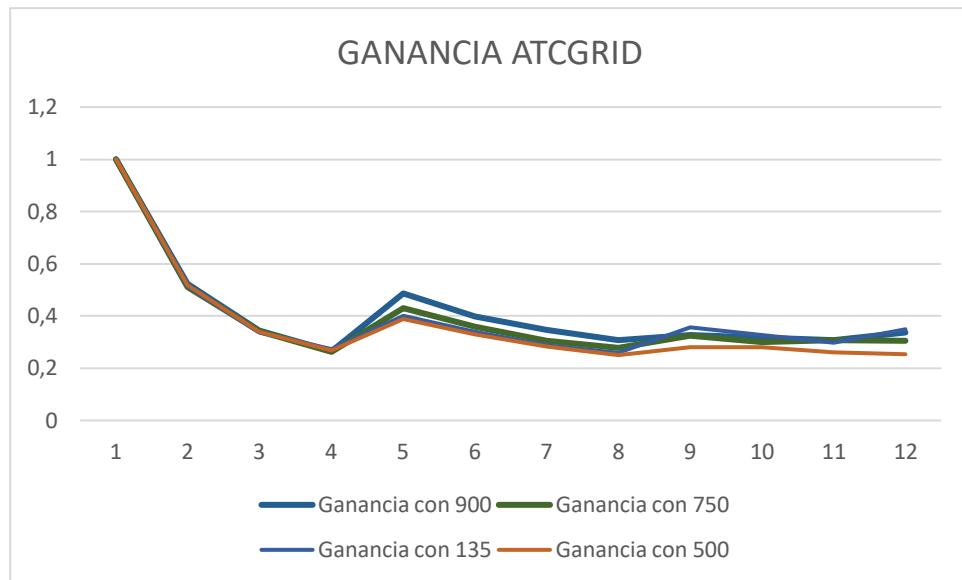
Dimension	Secuencial	Paralelo 1 hebra	Paralelo 2 hebras
105	0,006572787	0,004254565	0,002186157
500	0,579129584	0,578475323	0,299593665
750	1,959748887	1,971152641	1,005249307
900	3,483895693	3,439575668	1,79958947

Paralelo 3 hebras	Paralelo 4 hebras	Paralelo 5 hebras
0,001432333	0,00116213	0,001704585
0,196018003	0,155068472	0,224443357
0,671755824	0,517707344	0,84574689
1,180300914	0,910833247	1,669932719

Paralelo 6 hebras	Paralelo 7 hebras	Paralelo 8 hebras
0,001446947	0,001221202	0,001102563
0,190164711	0,163298547	0,14474842
0,709141809	0,603163116	0,546592735
1,367361467	1,191892359	1,057443921

Paralelo 9 hebras	Paralelo 10 hebras	Paralelo 11 hebras	Paralelo 12 hebras
0,001521457	0,001395173	0,001266472	0,00148616
0,162681762	0,161705241	0,151440822	0,146026116
0,639665835	0,592454314	0,604447957	0,602616191
1,128858164	1,089177176	1,116526771	1,159554273





ESTUDIO DE ESCALABILIDAD EN PCLOCAL:

SCRIPT: pmm-OpenMP_pclocal.sh

```
#!/bin/bash

echo "Secuencial:"
./pmm-secuencia_salida_reducida 105
./pmm-secuencia_salida_reducida 500
./pmm-secuencia_salida_reducida 750
./pmm-secuencia_salida_reducida 900

echo "-----"
echo "Paralelo con 2 threads:"
export OMP_NUM_THREADS=2

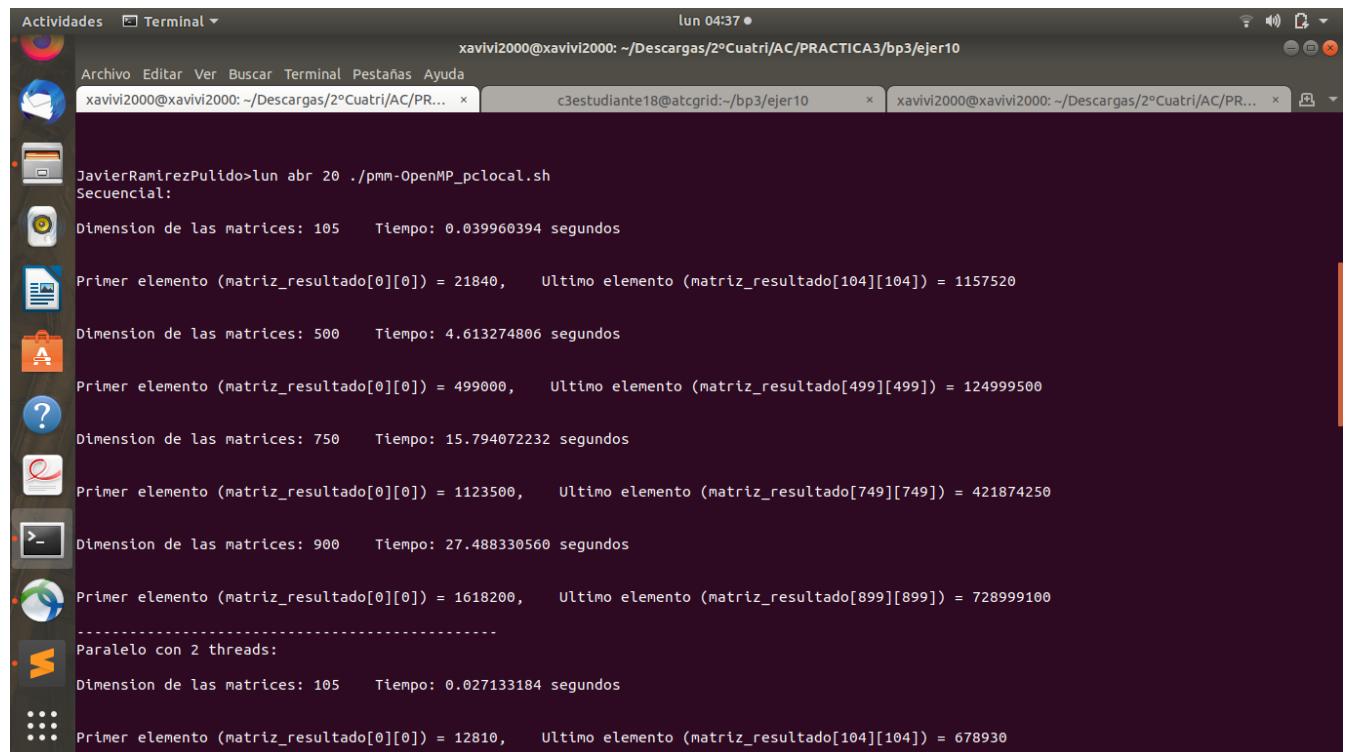
./pmm-OpenMP_salida_reducida 105
./pmm-OpenMP_salida_reducida 500
./pmm-OpenMP_salida_reducida 750
./pmm-OpenMP_salida_reducida 900

echo "-----"
echo "Paralelo con 3 threads:"
export OMP_NUM_THREADS=3

./pmm-OpenMP_salida_reducida 105
./pmm-OpenMP_salida_reducida 500
./pmm-OpenMP_salida_reducida 750
./pmm-OpenMP_salida_reducida 900

echo "-----"
echo "Paralelo con 4 threads:"
export OMP_NUM_THREADS=4

./pmm-OpenMP_salida_reducida 105
./pmm-OpenMP_salida_reducida 500
./pmm-OpenMP_salida_reducida 750
./pmm-OpenMP_salida_reducida 900
```



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "Terminal" and the current directory is "xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10". The terminal displays the output of a shell script named "pmm-OpenMP_pclocal.sh". The script performs matrix multiplication and measures execution time for different matrix sizes. It includes both sequential and parallel execution sections.

```
JavierRamirezPulido>lun abr 20 ./pmm-OpenMP_pclocal.sh
Secuencial:
Dimension de las matrices: 105      Tiempo: 0.039960394 segundos
Primer elemento (matriz_resultado[0][0]) = 21840,      Ultimo elemento (matriz_resultado[104][104]) = 1157520

Dimension de las matrices: 500      Tiempo: 4.613274806 segundos
Primer elemento (matriz_resultado[0][0]) = 499000,      Ultimo elemento (matriz_resultado[499][499]) = 124999500

Dimension de las matrices: 750      Tiempo: 15.794072232 segundos
Primer elemento (matriz_resultado[0][0]) = 1123500,      Ultimo elemento (matriz_resultado[749][749]) = 421874250

Dimension de las matrices: 900      Tiempo: 27.488330560 segundos
Primer elemento (matriz_resultado[0][0]) = 1618200,      Ultimo elemento (matriz_resultado[899][899]) = 728999100
-----
Paralelo con 2 threads:
Dimension de las matrices: 105      Tiempo: 0.027133184 segundos
Primer elemento (matriz_resultado[0][0]) = 12810,      Ultimo elemento (matriz_resultado[104][104]) = 678930
```

```
Actividades Terminal lun 04:37 ●
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR...
Dimension de las matrices: 900 Tiempo: 27.488330560 segundos
Primer elemento (matriz_resultado[0][0]) = 1618200, Ultimo elemento (matriz_resultado[899][899]) = 728999100
-----
Paralelo con 2 threads:
Dimension de las matrices: 105 Tiempo: 0.027133184 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 2.384798150 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 8.098266297 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 14.027206305 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 3 threads:
Dimension de las matrices: 105 Tiempo: 0.028391985 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 2.360774470 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 8.089329159 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 14.146429283 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 4 threads:
Dimension de las matrices: 105 Tiempo: 0.029386235 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
```

```

Actividades Terminal lun 04:37 •
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PRACTICA3/bp3/ejer10
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x c3estudiante18@atcgrid:~/bp3/ejer10 x xavivi2000@xavivi2000: ~/Descargas/2ºCuatri/AC/PR... x
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 14.146429283 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
-----
Paralelo con 4 threads:
Dimension de las matrices: 105 Tiempo: 0.029386235 segundos
Primer elemento (matriz_resultado[0][0]) = 12810, Ultimo elemento (matriz_resultado[104][104]) = 678930
Dimension de las matrices: 500 Tiempo: 2.351503955 segundos
Primer elemento (matriz_resultado[0][0]) = 258500, Ultimo elemento (matriz_resultado[499][499]) = 64754250
Dimension de las matrices: 750 Tiempo: 8.138685784 segundos
Primer elemento (matriz_resultado[0][0]) = 575250, Ultimo elemento (matriz_resultado[749][749]) = 216006375
Dimension de las matrices: 900 Tiempo: 14.140089823 segundos
Primer elemento (matriz_resultado[0][0]) = 825300, Ultimo elemento (matriz_resultado[899][899]) = 371797650
JavierRamirezPulido>lun abr 20 0

```

Dimension	Secuencial	Paralelo 1 hebras	Paralelo 2 hebras
105	0,038690792	0,035769835	0,025026893
500	4,619480149	5,127453475	4,155073936
750	16,01277702	9,012362589	8,963642766
900	27,51280272	13,75849502	14,03799794

Paralelo 3 hebras	Paralelo 4 hebras
0,020172752	0,018368013
4,323619242	2,884703076
8,365346042	8,161604293
14,5250514	15,28439034

