Laboratorio 3

Autores:

Mario Antonio Guerra Morales | Carné 21008
Daniel Armando Valdez Reyes | Carné 21240

Curso:

Computación Paralela y Distribuida

Catedrático:

Sebastian Galindo

Sección:

10

Universidad del Valle de Guatemala
11 calle 15-79 Zona 15 Vista Hermosa lll
Guatemala, C. A.
Facultad de Ingeniería

Laboratorio 3 - Computación Paralela y Distribuida

Repositorio de Github: https://github.com/MarioGuerra21008/Lab3Paralela

Para compilar el código realizado en el laboratorio:

- gcc -g -Wall -o vector_add2 vector_add2.c
- mpicc -g -Wall -o mpi_vector_add2 mpi_vector_add2.c
- mpicc -g -Wall -o mpi_vector_add3 mpi_vector_add3.c

Para ejecutar el código realizado en el laboratorio:

- ./vector_add2
- mpiexec -n 2 ./mpi_vector_add2
- mpiexec -n 2 ./mpi_vector_add3

1. **(10 pts) Explique por qué y cómo usamos comunicación grupal en las siguientes funciones de mpi_vector_add.c:**
    a. **Check_for_error():**

    Se utiliza para verificar errores en alguno o algunos de los procesos. La comunicación grupal se da coordinando todos los procesos, con el objetivo de que, si un proceso cuenta con un error, todos los demás lo sepan y el programa finalice de forma segura.

    b. **Read_n():**

    Se utiliza en el programa para leer el tamaño de los vectores. La comunicación grupal está presente para que todos los procesos cuenten con el mismo valor de n, con tal de que todos trabajen con los mismos datos.

    c. **Read_data():**

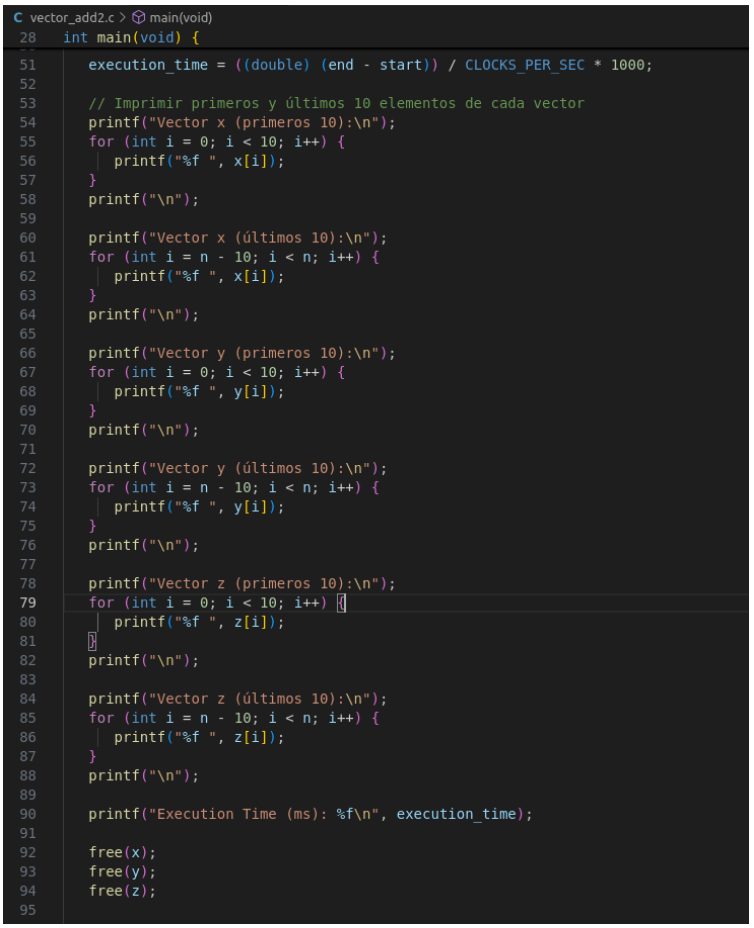
    Distribuye las partes del vector entre los procesos designados por medio de scatter y por partes desde el proceso 0 hacia los demás.

    d. **Print_vector():**

    Utiliza la comunicación grupal para unir los resultados parciales para generar el resultado final de la suma.

2. **(15 pts) Descargue y modifique el programa vector_add.c para crear dos vectores de al menos 100,000 elementos generados de forma aleatoria. Haga lo mismo con mpi_vector_add.c . Imprima únicamente los primeros y últimos 10 elementos de cada vector (y el resultado) para validar. Incluya captura de pantalla.**

vector_add.c:



```
C vector_add2.c > main(void)
28  int main(void) {
51      execution_time = ((double) (end - start)) / CLOCKS_PER_SEC * 1000;
52
53      // Imprimir primeros y últimos 10 elementos de cada vector
54      printf("Vector x (primeros 10):\n");
55      for (int i = 0; i < 10; i++) {
56          printf("%f ", x[i]);
57      }
58      printf("\n");
59
60      printf("Vector x (últimos 10):\n");
61      for (int i = n - 10; i < n; i++) {
62          printf("%f ", x[i]);
63      }
64      printf("\n");
65
66      printf("Vector y (primeros 10):\n");
67      for (int i = 0; i < 10; i++) {
68          printf("%f ", y[i]);
69      }
70      printf("\n");
71
72      printf("Vector y (últimos 10):\n");
73      for (int i = n - 10; i < n; i++) {
74          printf("%f ", y[i]);
75      }
76      printf("\n");
77
78      printf("Vector z (primeros 10):\n");
79      for (int i = 0; i < 10; i++) {
80          printf("%f ", z[i]);
81      }
82      printf("\n");
83
84      printf("Vector z (últimos 10):\n");
85      for (int i = n - 10; i < n; i++) {
86          printf("%f ", z[i]);
87      }
88      printf("\n");
89
90      printf("Execution Time (ms): %f\n", execution_time);
91
92      free(x);
93      free(y);
94      free(z);
95
```

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ gcc -g -Wall -o vector_add2
vector_add2.c
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
23.000000 47.000000 62.000000 14.000000 17.000000 43.000000 47.000000 18.000000
67.000000 26.000000
Vector x (últimos 10):
3.000000 32.000000 87.000000 91.000000 75.000000 72.000000 49.000000 64.000000 4
5.000000 3.000000
Vector y (primeros 10):
48.000000 56.000000 68.000000 77.000000 30.000000 97.000000 84.000000 56.000000
13.000000 79.000000
Vector y (últimos 10):
89.000000 74.000000 79.000000 54.000000 31.000000 19.000000 54.000000 93.000000
70.000000 2.000000
Vector z (primeros 10):
71.000000 103.000000 130.000000 91.000000 47.000000 140.000000 131.000000 74.000
000 80.000000 105.000000
Vector z (últimos 10):
92.000000 106.000000 166.000000 145.000000 106.000000 91.000000 103.000000 157.0
00000 115.000000 5.000000
Execution Time (ms): 4.721000
```

mpi_vector_add2.c:

```c
int main(void) {
    int local_n;
    int comm_sz, my_rank;
    double *local_x, *local_y, *local_z;
    MPI_Comm comm;
    double tstart, tend;

    MPI_Init(NULL, NULL);
    comm = MPI_COMM_WORLD;
    MPI_Comm_size(comm, &comm_sz);
    MPI_Comm_rank(comm, &my_rank);

    local_n = n / comm_sz;

    //Read_n(&n, &local_n, my_rank, comm_sz, comm);
    tstart = MPI_Wtime();
    Allocate_vectors(&local_x, &local_y, &local_z, local_n, comm);

    srand(time(NULL) + my_rank); // Para generar números aleatorios diferentes en cada proceso
    for (int i = 0; i < local_n; i++) {
        local_x[i] = (double)(rand() % 100);
        local_y[i] = (double)(rand() % 100);
    }

    MPI_Barrier(comm); // Sincronización
    tstart = MPI_Wtime();

    Parallel_vector_sum(local_x, local_y, local_z, local_n);

    tend = MPI_Wtime();

    if (my_rank == 0) {
        printf("Tiempo total: %f milisegundos\n", (tend - tstart) * 1000);
    }

    Print_vector(local_x, local_n, n, "Vector x", my_rank, comm);
    Print_vector(local_y, local_n, n, "Vector y", my_rank, comm);
    Print_vector(local_z, local_n, n, "Vector z", my_rank, comm);

    free(local_x);
    free(local_y);
    free(local_z);

    MPI_Finalize();
    return 0;
}  /* main */
```

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ gcc -g -Wall -o vector_add2 vector_add2.c
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
23.000000 47.000000 62.000000 14.000000 17.000000 43.000000 47.000000 18.000000 67.000000 26.000000
Vector x (últimos 10):
3.000000 32.000000 87.000000 91.000000 75.000000 72.000000 49.000000 64.000000 45.000000 3.000000
Vector y (primeros 10):
48.000000 56.000000 68.000000 77.000000 30.000000 97.000000 84.000000 56.000000 13.000000 79.000000
Vector y (últimos 10):
89.000000 74.000000 79.000000 54.000000 31.000000 19.000000 54.000000 93.000000 70.000000 2.000000
Vector z (primeros 10):
71.000000 103.000000 130.000000 91.000000 47.000000 140.000000 131.000000 74.000000 80.000000 105.000000
Vector z (últimos 10):
92.000000 106.000000 166.000000 145.000000 106.000000 91.000000 103.000000 157.000000 115.000000 5.000000
Execution Time (ms): 4.721000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpicc -g -Wall -o mpi_vector_add2 mpi_vector_add2.c
mpi_vector_add2.c: In function ‘main’:
mpi_vector_add2.c:63:10: warning: implicit declaration of function ‘time’ [-Wimplicit-function-declaration]
   63 |     srand(time(NULL) + my_rank); // Para generar números aleatorios diferentes en cada proceso
      |           ^~~~
mpi_vector_add2.c: In function ‘Print_vector’:
mpi_vector_add2.c:274:10: warning: unused variable ‘fname’ [-Wunused-variable]
  274 |     char* fname = "Print_vector";
      |           ^~~~~
mpi_vector_add2.c:273:8: warning: unused variable ‘local_ok’ [-Wunused-variable]
  273 |     int local_ok = 1;
      |         ^~~~~~~~
mpi_vector_add2.c:272:8: warning: unused variable ‘i’ [-Wunused-variable]
  272 |     int i;
      |         ^
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.264818 milisegundos
Vector x (primeros 10):
72.000000 87.000000 77.000000 92.000000 15.000000 41.000000 64.000000 29.000000 29.000000 76.000000
Vector x (últimos 10):
18.000000 24.000000 39.000000 25.000000 41.000000 8.000000 3.000000 72.000000 48.000000 43.000000
Vector y (primeros 10):
50.000000 64.000000 6.000000 21.000000 81.000000 76.000000 36.000000 74.000000 7.000000 49.000000
Vector y (últimos 10):
6.000000 31.000000 56.000000 36.000000 25.000000 27.000000 87.000000 90.000000 49.000000 99.000000
Vector z (primeros 10):
122.000000 151.000000 83.000000 113.000000 96.000000 117.000000 100.000000 103.000000 36.000000 125.000000
Vector z (últimos 10):
24.000000 55.000000 95.000000 61.000000 66.000000 35.000000 90.000000 162.000000 97.000000 142.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$
```

3. **(5 pts) Mida los tiempos de ambos programas y calcule el speedup logrado con la versión paralela. Realice al menos 10 mediciones de tiempo para cada programa y obtenga el promedio del tiempo de cada uno. Cada medición debe estar en el orden de los ~5 segundos para asegurar valores estables (utilice una cantidad de elementos adecuada para que a su máquina le tome por lo menos ~5 cada corrida). Utilice esos promedios para el cálculo del speedup. Incluya capturas de pantalla.**

   Mediciones de tiempo para la versión secuencial:

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
15.000000 59.000000 5.000000 76.000000 18.000000 87.000000 86.000000 13.000000 35.000000 31.000000
Vector x (últimos 10):
71.000000 97.000000 38.000000 17.000000 55.000000 63.000000 55.000000 66.000000 92.000000 64.000000
Vector y (primeros 10):
57.000000 33.000000 19.000000 19.000000 66.000000 16.000000 87.000000 84.000000 60.000000 3.000000
Vector y (últimos 10):
75.000000 10.000000 20.000000 51.000000 10.000000 60.000000 19.000000 85.000000 4.000000 90.000000
Vector z (primeros 10):
72.000000 92.000000 24.000000 95.000000 84.000000 103.000000 173.000000 97.000000 95.000000 34.000000
Vector z (últimos 10):
146.000000 107.000000 58.000000 68.000000 65.000000 123.000000 74.000000 151.000000 96.000000 154.000000
Execution Time (ms): 5.507000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
57.000000 41.000000 8.000000 88.000000 44.000000 14.000000 97.000000 49.000000 51.000000 57.000000
Vector x (últimos 10):
29.000000 80.000000 3.000000 77.000000 1.000000 6.000000 41.000000 11.000000 97.000000 75.000000
Vector y (primeros 10):
28.000000 72.000000 98.000000 34.000000 13.000000 79.000000 21.000000 31.000000 40.000000 34.000000
Vector y (últimos 10):
69.000000 93.000000 31.000000 7.000000 11.000000 61.000000 97.000000 14.000000 79.000000 60.000000
Vector z (primeros 10):
85.000000 113.000000 106.000000 122.000000 57.000000 93.000000 118.000000 80.000000 91.000000 91.000000
Vector z (últimos 10):
98.000000 173.000000 34.000000 84.000000 12.000000 67.000000 138.000000 25.000000 176.000000 135.000000
Execution Time (ms): 6.437000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
59.000000 62.000000 28.000000 49.000000 47.000000 40.000000 27.000000 14.000000 67.000000 6.000000
Vector x (últimos 10):
52.000000 45.000000 16.000000 96.000000 32.000000 29.000000 29.000000 94.000000 93.000000 82.000000
Vector y (primeros 10):
83.000000 33.000000 2.000000 18.000000 58.000000 66.000000 32.000000 75.000000 76.000000 24.000000
Vector y (últimos 10):
64.000000 65.000000 17.000000 89.000000 55.000000 66.000000 54.000000 84.000000 44.000000 15.000000
Vector z (primeros 10):
142.000000 95.000000 30.000000 67.000000 105.000000 106.000000 59.000000 89.000000 143.000000 30.000000
Vector z (últimos 10):
116.000000 110.000000 33.000000 185.000000 87.000000 95.000000 83.000000 178.000000 137.000000 97.000000
Execution Time (ms): 5.881000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
34.000000 93.000000 91.000000 14.000000 23.000000 92.000000 53.000000 39.000000 97.000000 74.000000
Vector x (últimos 10):
32.000000 88.000000 49.000000 47.000000 28.000000 60.000000 74.000000 70.000000 33.000000 11.000000
Vector y (primeros 10):
9.000000 50.000000 93.000000 47.000000 73.000000 53.000000 45.000000 83.000000 36.000000 40.000000
Vector y (últimos 10):
8.000000 62.000000 8.000000 78.000000 73.000000 25.000000 80.000000 89.000000 48.000000 88.000000
Vector z (primeros 10):
43.000000 143.000000 184.000000 61.000000 96.000000 145.000000 98.000000 122.000000 133.000000 114.000000
Vector z (últimos 10):
40.000000 150.000000 57.000000 125.000000 101.000000 85.000000 154.000000 159.000000 81.000000 99.000000
Execution Time (ms): 6.057000
```

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
30.000000 42.000000 31.000000 22.000000 47.000000 87.000000 20.000000 2.000000 42.000000 27.000000
Vector x (últimos 10):
65.000000 86.000000 24.000000 3.000000 6.000000 76.000000 90.000000 83.000000 0.000000 30.000000
Vector y (primeros 10):
19.000000 8.000000 53.000000 16.000000 75.000000 33.000000 68.000000 42.000000 26.000000 13.000000
Vector y (últimos 10):
89.000000 29.000000 42.000000 99.000000 7.000000 18.000000 56.000000 50.000000 14.000000 29.000000
Vector z (primeros 10):
49.000000 50.000000 84.000000 38.000000 122.000000 120.000000 88.000000 44.000000 68.000000 40.000000
Vector z (últimos 10):
154.000000 115.000000 66.000000 102.000000 13.000000 94.000000 146.000000 133.000000 14.000000 59.000000
Execution Time (ms): 6.738000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
51.000000 61.000000 11.000000 88.000000 54.000000 51.000000 61.000000 90.000000 73.000000 25.000000
Vector x (últimos 10):
94.000000 68.000000 45.000000 14.000000 33.000000 66.000000 44.000000 65.000000 65.000000 51.000000
Vector y (primeros 10):
45.000000 54.000000 76.000000 93.000000 15.000000 73.000000 75.000000 94.000000 45.000000 12.000000
Vector y (últimos 10):
36.000000 9.000000 45.000000 12.000000 42.000000 24.000000 86.000000 25.000000 49.000000 9.000000
Vector z (primeros 10):
96.000000 115.000000 87.000000 181.000000 69.000000 124.000000 136.000000 184.000000 118.000000 37.000000
Vector z (últimos 10):
130.000000 77.000000 90.000000 26.000000 75.000000 90.000000 130.000000 90.000000 114.000000 60.000000
Execution Time (ms): 10.029000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
41.000000 27.000000 42.000000 36.000000 35.000000 38.000000 19.000000 89.000000 32.000000 65.000000
Vector x (últimos 10):
48.000000 27.000000 50.000000 52.000000 89.000000 71.000000 86.000000 3.000000 99.000000 77.000000
Vector y (primeros 10):
20.000000 53.000000 7.000000 46.000000 39.000000 99.000000 82.000000 92.000000 50.000000 1.000000
Vector y (últimos 10):
1.000000 73.000000 86.000000 58.000000 75.000000 50.000000 90.000000 8.000000 89.000000 25.000000
Vector z (primeros 10):
61.000000 80.000000 49.000000 82.000000 74.000000 137.000000 101.000000 181.000000 82.000000 66.000000
Vector z (últimos 10):
49.000000 100.000000 136.000000 110.000000 164.000000 121.000000 176.000000 11.000000 188.000000 102.000000
Execution Time (ms): 4.699000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
3.000000 98.000000 14.000000 96.000000 37.000000 67.000000 34.000000 10.000000 16.000000 68.000000
Vector x (últimos 10):
3.000000 86.000000 30.000000 1.000000 12.000000 75.000000 6.000000 32.000000 6.000000 81.000000
Vector y (primeros 10):
49.000000 87.000000 28.000000 36.000000 78.000000 60.000000 30.000000 75.000000 33.000000 85.000000
Vector y (últimos 10):
0.000000 93.000000 23.000000 67.000000 99.000000 13.000000 93.000000 29.000000 39.000000 43.000000
Vector z (primeros 10):
52.000000 185.000000 42.000000 132.000000 115.000000 127.000000 64.000000 85.000000 49.000000 153.000000
Vector z (últimos 10):
3.000000 179.000000 53.000000 68.000000 111.000000 88.000000 99.000000 61.000000 45.000000 124.000000
Execution Time (ms): 3.845000
```

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
23.000000 54.000000 30.000000 43.000000 87.000000 95.000000 41.000000 59.000000 89.000000 64.000000
Vector x (últimos 10):
29.000000 84.000000 22.000000 37.000000 73.000000 5.000000 85.000000 34.000000 68.000000 34.000000
Vector y (primeros 10):
22.000000 25.000000 58.000000 24.000000 58.000000 83.000000 92.000000 42.000000 9.000000 13.000000
Vector y (últimos 10):
64.000000 50.000000 26.000000 2.000000 6.000000 1.000000 80.000000 54.000000 98.000000 23.000000
Vector z (primeros 10):
45.000000 79.000000 88.000000 67.000000 145.000000 178.000000 133.000000 101.000000 98.000000 77.000000
Vector z (últimos 10):
93.000000 134.000000 48.000000 39.000000 79.000000 6.000000 165.000000 88.000000 166.000000 57.000000
Execution Time (ms): 4.141000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ ./vector_add2
Vector x (primeros 10):
66.000000 63.000000 55.000000 98.000000 6.000000 54.000000 46.000000 89.000000 61.000000 52.000000
Vector x (últimos 10):
97.000000 57.000000 73.000000 79.000000 60.000000 69.000000 23.000000 99.000000 56.000000 90.000000
Vector y (primeros 10):
65.000000 59.000000 50.000000 22.000000 50.000000 61.000000 99.000000 91.000000 59.000000 69.000000
Vector y (últimos 10):
43.000000 59.000000 60.000000 90.000000 44.000000 75.000000 45.000000 30.000000 73.000000 16.000000
Vector z (primeros 10):
131.000000 122.000000 105.000000 120.000000 56.000000 115.000000 145.000000 180.000000 120.000000 121.000000
Vector z (últimos 10):
140.000000 116.000000 133.000000 169.000000 104.000000 144.000000 68.000000 129.000000 129.000000 106.000000
Execution Time (ms): 6.110000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$
```

Mediciones de tiempo para la versión paralela:

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.350194 milisegundos
Vector x (primeros 10):
36.000000 28.000000 2.000000 86.000000 45.000000 51.000000 62.000000 9.000000 57.000000 73.000000
Vector x (últimos 10):
50.000000 0.000000 48.000000 19.000000 94.000000 34.000000 52.000000 33.000000 62.000000 17.000000
Vector y (primeros 10):
18.000000 92.000000 8.000000 30.000000 55.000000 89.000000 93.000000 1.000000 57.000000 79.000000
Vector y (últimos 10):
32.000000 30.000000 55.000000 33.000000 94.000000 11.000000 48.000000 90.000000 82.000000 65.000000
Vector z (primeros 10):
54.000000 120.000000 10.000000 116.000000 100.000000 140.000000 155.000000 10.000000 114.000000 152.000000
Vector z (últimos 10):
82.000000 30.000000 103.000000 52.000000 188.000000 45.000000 100.000000 123.000000 144.000000 82.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.337076 milisegundos
Vector x (primeros 10):
78.000000 48.000000 97.000000 87.000000 65.000000 72.000000 20.000000 41.000000 78.000000 58.000000
Vector x (últimos 10):
79.000000 0.000000 68.000000 4.000000 11.000000 91.000000 42.000000 33.000000 84.000000 29.000000
Vector y (primeros 10):
96.000000 88.000000 42.000000 29.000000 45.000000 37.000000 2.000000 34.000000 59.000000 96.000000
Vector y (últimos 10):
5.000000 97.000000 53.000000 35.000000 50.000000 36.000000 56.000000 80.000000 62.000000 60.000000
Vector z (primeros 10):
174.000000 136.000000 139.000000 116.000000 110.000000 109.000000 22.000000 75.000000 137.000000 154.000000
Vector z (últimos 10):
84.000000 97.000000 121.000000 39.000000 61.000000 127.000000 98.000000 113.000000 146.000000 89.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.200652 milisegundos
Vector x (primeros 10):
16.000000 5.000000 58.000000 32.000000 6.000000 21.000000 6.000000 18.000000 66.000000 71.000000
Vector x (últimos 10):
19.000000 22.000000 87.000000 21.000000 58.000000 63.000000 48.000000 88.000000 66.000000 16.000000
Vector y (primeros 10):
71.000000 49.000000 68.000000 7.000000 90.000000 0.000000 53.000000 55.000000 89.000000 29.000000
Vector y (últimos 10):
42.000000 68.000000 3.000000 0.000000 31.000000 59.000000 37.000000 65.000000 1.000000 34.000000
Vector z (primeros 10):
87.000000 54.000000 126.000000 39.000000 96.000000 21.000000 59.000000 73.000000 155.000000 100.000000
Vector z (últimos 10):
61.000000 90.000000 90.000000 21.000000 89.000000 122.000000 85.000000 153.000000 67.000000 50.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.244314 milisegundos
Vector x (primeros 10):
83.000000 65.000000 36.000000 44.000000 87.000000 39.000000 8.000000 40.000000 3.000000 47.000000
Vector x (últimos 10):
9.000000 96.000000 21.000000 90.000000 34.000000 88.000000 22.000000 79.000000 52.000000 76.000000
Vector y (primeros 10):
68.000000 64.000000 10.000000 11.000000 92.000000 58.000000 73.000000 20.000000 53.000000 23.000000
Vector y (últimos 10):
48.000000 68.000000 53.000000 71.000000 4.000000 8.000000 80.000000 61.000000 38.000000 28.000000
Vector z (primeros 10):
151.000000 129.000000 46.000000 55.000000 179.000000 97.000000 81.000000 60.000000 56.000000 70.000000
Vector z (últimos 10):
57.000000 164.000000 74.000000 161.000000 38.000000 96.000000 102.000000 140.000000 90.000000 104.000000
```

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.309858 milisegundos
Vector x (primeros 10):
6.000000 84.000000 63.000000 28.000000 22.000000 18.000000 30.000000 8.000000 54.000000 19.000000
Vector x (últimos 10):
11.000000 39.000000 79.000000 89.000000 19.000000 82.000000 58.000000 14.000000 40.000000 85.000000
Vector y (primeros 10):
44.000000 13.000000 90.000000 45.000000 70.000000 35.000000 41.000000 30.000000 59.000000 8.000000
Vector y (últimos 10):
18.000000 30.000000 63.000000 73.000000 75.000000 41.000000 45.000000 42.000000 90.000000 24.000000
Vector z (primeros 10):
50.000000 97.000000 153.000000 73.000000 92.000000 53.000000 71.000000 38.000000 113.000000 27.000000
Vector z (últimos 10):
29.000000 69.000000 142.000000 162.000000 94.000000 123.000000 103.000000 56.000000 130.000000 109.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.316538 milisegundos
Vector x (primeros 10):
71.000000 40.000000 67.000000 22.000000 35.000000 89.000000 69.000000 84.000000 1.000000 64.000000
Vector x (últimos 10):
41.000000 48.000000 33.000000 39.000000 28.000000 7.000000 23.000000 61.000000 30.000000 53.000000
Vector y (primeros 10):
52.000000 9.000000 3.000000 37.000000 18.000000 87.000000 88.000000 46.000000 58.000000 58.000000
Vector y (últimos 10):
24.000000 92.000000 6.000000 28.000000 64.000000 57.000000 83.000000 69.000000 81.000000 91.000000
Vector z (primeros 10):
123.000000 49.000000 70.000000 59.000000 53.000000 176.000000 157.000000 130.000000 59.000000 122.000000
Vector z (últimos 10):
65.000000 140.000000 39.000000 67.000000 92.000000 64.000000 106.000000 130.000000 111.000000 144.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.192988 milisegundos
Vector x (primeros 10):
93.000000 90.000000 58.000000 34.000000 73.000000 61.000000 27.000000 85.000000 30.000000 69.000000
Vector x (últimos 10):
39.000000 45.000000 92.000000 78.000000 0.000000 73.000000 23.000000 81.000000 61.000000 14.000000
Vector y (primeros 10):
48.000000 26.000000 47.000000 55.000000 38.000000 84.000000 53.000000 68.000000 11.000000 46.000000
Vector y (últimos 10):
61.000000 17.000000 31.000000 6.000000 98.000000 3.000000 8.000000 54.000000 13.000000 75.000000
Vector z (primeros 10):
141.000000 116.000000 105.000000 89.000000 111.000000 145.000000 80.000000 153.000000 41.000000 115.000000
Vector z (últimos 10):
100.000000 62.000000 123.000000 84.000000 98.000000 76.000000 31.000000 135.000000 74.000000 89.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.335225 milisegundos
Vector x (primeros 10):
72.000000 92.000000 37.000000 31.000000 81.000000 25.000000 48.000000 79.000000 65.000000 6.000000
Vector x (últimos 10):
23.000000 75.000000 51.000000 91.000000 6.000000 28.000000 26.000000 9.000000 18.000000 11.000000
Vector y (primeros 10):
13.000000 2.000000 58.000000 63.000000 31.000000 78.000000 51.000000 76.000000 28.000000 44.000000
Vector y (últimos 10):
91.000000 8.000000 87.000000 6.000000 61.000000 65.000000 60.000000 79.000000 39.000000 59.000000
Vector z (primeros 10):
85.000000 94.000000 95.000000 94.000000 112.000000 103.000000 99.000000 155.000000 93.000000 50.000000
Vector z (últimos 10):
114.000000 83.000000 138.000000 97.000000 67.000000 93.000000 86.000000 88.000000 57.000000 70.000000
```

```
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.273938 milisegundos
Vector x (primeros 10):
29.000000 83.000000 34.000000 12.000000 40.000000 26.000000 11.000000 18.000000 66.000000 60.000000
Vector x (últimos 10):
39.000000 81.000000 24.000000 3.000000 4.000000 15.000000 48.000000 20.000000 15.000000 1.000000
Vector y (primeros 10):
47.000000 15.000000 1.000000 40.000000 42.000000 65.000000 89.000000 97.000000 27.000000 87.000000
Vector y (últimos 10):
14.000000 42.000000 6.000000 34.000000 69.000000 58.000000 41.000000 31.000000 8.000000 54.000000
Vector z (primeros 10):
76.000000 98.000000 35.000000 52.000000 82.000000 91.000000 100.000000 115.000000 93.000000 147.000000
Vector z (últimos 10):
53.000000 123.000000 30.000000 37.000000 73.000000 73.000000 89.000000 51.000000 23.000000 55.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$ mpiexec -n 2 ./mpi_vector_add2
Tiempo total: 0.186565 milisegundos
Vector x (primeros 10):
26.000000 30.000000 55.000000 30.000000 78.000000 71.000000 16.000000 18.000000 72.000000 54.000000
Vector x (últimos 10):
55.000000 81.000000 53.000000 4.000000 69.000000 50.000000 59.000000 52.000000 9.000000 53.000000
Vector y (primeros 10):
40.000000 38.000000 47.000000 20.000000 61.000000 18.000000 27.000000 75.000000 35.000000 99.000000
Vector y (últimos 10):
89.000000 61.000000 29.000000 64.000000 29.000000 95.000000 69.000000 14.000000 94.000000 57.000000
Vector z (primeros 10):
66.000000 68.000000 102.000000 50.000000 139.000000 89.000000 43.000000 93.000000 107.000000 153.000000
Vector z (últimos 10):
144.000000 142.000000 82.000000 68.000000 98.000000 145.000000 128.000000 66.000000 103.000000 110.000000
mariogm@mariogm-VirtualBox:~/Paralela/Lab3Paralela$
```

Aquí se encuentra la tabla con el speedup correspondiente:

# / Valores	Versión Secuencial (ms)	Versión Paralela (ms)	Speedup
1	5,51	0,350194	15,72556926
2	6,44	0,337076	19,09658356
3	5,88	0,200652	29,30945119
4	6,06	0,244314	24,7918662
5	6,74	0,309858	21,74544469
6	10,03	0,316538	31,68339978
7	4,70	0,192988	24,34866417
8	3,85	0,335225	11,46990827
9	4,14	0,273938	15,11655922
10	6,11	0,186565	32,7499799
Promedio Total	**5,9444**	**0,2747348**	**21,63686581**

Se puede observar que el speedup promedio es de 21.64 aproximadamente. Indicando así que la versión paralela del programa es bastante superior a la versión secuencial.

4. **(55 pts) Modifique el programa mpi_vector_add.c para que calcule de dos vectores 1) el producto punto 2) el producto de un escalar por cada vector (el mismo escalar para ambos). Verifique el correcto funcionamiento de su programa (para ello puede probar con pocos elementos para validar). Incluya captura de pantalla.**

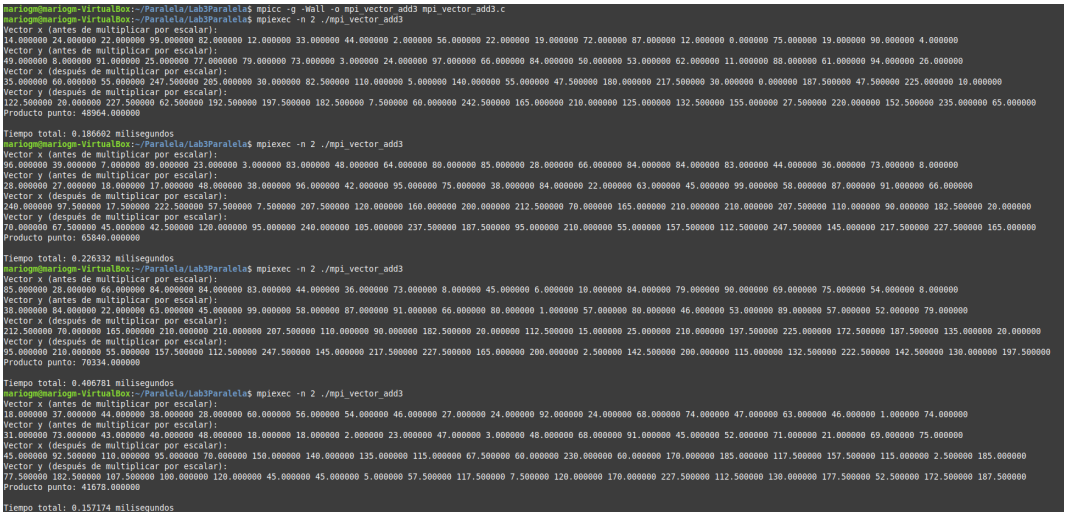
Se utiliza un valor de 20 en n y un escalar de 2.5.

```
int n = 20;
int local_n;
int comm_sz, my_rank;
double *local_x, *local_y, *local_z;
double dot_product;
double scalar = 2.5;
MPI_Comm comm;
double tstart, tend;

MPI_Init(NULL, NULL);
comm = MPI_COMM_WORLD;
MPI_Comm_size(comm, &comm_sz);
MPI_Comm_rank(comm, &my_rank);
```

```c
double Parallel_dot_product(double local_x[], double local_y[], int local_n) {
    double local_dot_product = 0.0;
    double global_dot_product = 0.0;
    for (int i = 0; i < local_n; i++) {
        local_dot_product += local_x[i] * local_y[i];
    }
    MPI_Reduce(&local_dot_product, &global_dot_product, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
    return global_dot_product;
}

void Parallel_scalar_multiplication(double local_x[], double local_y[], double scalar, int local_n) {
    for (int i = 0; i < local_n; i++) {
        local_x[i] *= scalar;
        local_y[i] *= scalar;
    }
}
```



5. **(15 pts) Finalmente, escriba una reflexión del laboratorio realizado en donde hable de las técnicas aplicadas, lo que se aprendió y pudo repasar, elementos que le llamaron la atención, ediciones/mejoras que considera que son posibles y cualquier otra cosa relevante que tengan en mente.**

   El laboratorio nos pareció bastante interesante en el sentido de poder aplicar más directivas de OpenMPI y que sea un programa muy llamativo para nosotros al poder aplicar varios vectores y realizar operaciones con ellos de forma distribuida y usando comunicación grupal entre los procesos que lo están llevando a cabo.

   Pudimos repasar también más sobre el uso de la computación distribuida y sobre comunicación al usar directivas y métodos que llamaran para revisar errores, cálculos ya realizados y el uso de enviar un mensaje a la consola. Son prácticas que podrán parecer comunes pero que con el uso de una herramienta diferente a las usadas durante la carrera se descubren nuevas maneras de implementación.

   Por último, quizá una de las mejoras que podríamos proponer serían incisos en los que se puedan aplicar mayor cantidad de divisiones intentado usar operaciones o productos que permitan realizar esto.