Trabajo de Laboratorio - Algoritmos Paralelos

Ximena Sofia Pocco Lozada

April 2018

1 Matrix Vector

Para empzar necesitaremos distribuir la información del array a todos los procesos, luego cada uno deberá compartir esta información con todos, para ello se puede utilizar Gather y posteriormente BCast. Pero existe la opción de hacer todo esto con una sola instrucción: Allgather, la cual se utiliza en el programa, está también forma parte de las instrucciones de MPI de comunicación colectiva.

Run time					
comm_sz	1024	2048	4096	8192	16384
1	4.1	16	64	270	1100
2	2.3	8.5	33	140	560
4	2	5.1	18	70	280
8	1.7	3.3	9.8	36	140
16	1.7	2.6	5.9	19	71

Figure 1: Tiempo de ejecución en milisegundos de multiplicación Matriz vector con 1, 2,4,8,16 procesos

Run time					
comm_sz	1024	2048	4096	8192	16384
1	0.00894	0.0354894	0.1440215	0.5670188	2.2670188
2	0.0069185	0.0239316	0.097413	0.3795881	1.4058375
4	0.006437	0.0273359	0.099546	0.3993084	1.5985645
8	0.0155323	0.0472851	0.1327478	0.5764178	2.4287389
16	0.0268397	0.0835637	0.26501	1.0298161	4.0614511

Figure 2: Tiempo de ejecución en milisegundos de multiplicación Matriz vector con 1, 2,4,8,16 procesos

Comparando los resultados adquiridos se nota que a medida se va aumentando los procesos el tiempo en el que realiza la operación va ir disminuyendo, y cuando se va aumentado el tamaño se de la matriz-vector este también incrementara el tiempo.

2 Odd even Sort

Este algoritmo se divide en dos fases: fase par e impar. En la fase impar, realizamos una clasificación de burbuja en elementos indexados impares y en la fase par, realizamos una ordenación de burbuja en elementos indexados pares.

# de keys					
process	200	400	800	1600	3200
1	88	190	390	830	1800
2	43	91	190	410	860
4	22	46	96	200	430
8	12	24	51	110	220
16	7.5	14	29	60	130

Figure 3: Tiempo de ejecución en milisegundos de Bubble sort con 1, 2,4,8,16 procesos

# de keys					
Process	1024	2048	4096	8192	16384
1	3.69 e-05	4.72 e-05	7.20 e-05	7.27 e-05	9.10 e-04
2	5.91 e-04	4.12 e-04	5.33 e-04	6.31 e-04	7.24 e-04
4	3.64 e-04	3.36 e-04	4.31 e-04	4.96 e-04	5.26 e-04
8	5.45 e-03	5.76 e-03	5.99 e-03	6.62 e-03	6.77 e-03
16	3.23 e-03	4.39 e-03 4	4.44 e-03	5.03 e-03	5.41 e-03

Figure 4: Tiempo de ejecución en milisegundos de Odd even Sort con 1, 2,4,8,16 procesos

Como observamos en odd even sort a medida se va incrementando los procesos va ir disminuyendo el tiempo ya que cuando se hace las particiones cada proceso va ir ordenando para luego comparar entre procesos e intercambiar si es el caso. El tiempo se va ir incrementando cada ves que el tamaño de la lista incremente.

References