CIENCIA DE LA COMPUTACIÓN

INFORME

ALGORITMOS PARALELOS

Angel Sucapuca

SEMESTRE VII

AÑO 2016

"El alumno declara haber realizado el presente trabajo de acuerdo a las normas de la Universidad Católica San Pablo"

Firma

Multiplicación de Matriz por Vector

Algoritmo utilizando MPI_Allgather:

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

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

	int m, n, n_local, i,j,local_m,local_n;
	double local_start, local_finish, local_elapsed, elapsed;
	MPI_Comm comm;

	m=atoi(argv[1]);
	n=atoi(argv[1]);
	srand(time(NULL));

	int v[n], result[n];
	int *A = (int *)malloc(sizeof(int) * m*n);
	int rank;
	int size;
	printf("Vivo\n");
	  MPI_Init(&argc, &argv);
	comm = MPI_COMM_WORLD;
	MPI_Comm_rank(MPI_COMM_WORLD, &rank);
	MPI_Comm_size(MPI_COMM_WORLD, &size);

	local_m=m/size;
	local_n=n/size;

	if(rank==0){

		for(i=0;i<m*n;++i)
			A[i]=(rand() %100000);

		for(i=0; i<n; ++i){
			v[i]=(rand() %100000);
			//printf("%d, ",v[i]);
		}
```

```
	}
	MPI_Barrier(comm);
	local_start=MPI_Wtime();
	int *local_A = (int *)malloc(sizeof(int) * local_m*n);
	int *local_x = (int *)malloc(sizeof(int) * local_n);
	int *local_y = (int *)malloc(sizeof(int) * local_m);
	int *res = (int *)malloc(sizeof(int) * m);
	MPI_Scatter(A, local_m*n, MPI_INT, local_A,
		local_m*n, MPI_INT, 0, comm);
	MPI_Scatter(v, local_n, MPI_INT, local_x,
		local_n, MPI_INT, 0, comm);

	int *x=(int*)malloc(n*sizeof(int));
	MPI_Allgather(local_x, local_n, MPI_INT, x, local_n, MPI_INT, comm);

	for(i=0; i<local_m; ++i){

		local_y[i]=0;
		for(j=0; j<n; ++j){
			local_y[i]+=local_A[i*n+j]*x[j];
			//printf("Proceso %d, opero %d x %d\n",
rank,local_A[i*n+j],x[j]);
		}
	}
	free(x);
	MPI_Gather(local_y, local_m, MPI_INT, res, local_m, MPI_INT, 0,
	     MPI_COMM_WORLD);
	local_finish=MPI_Wtime();
	local_elapsed=local_finish-local_start;
	MPI_Reduce(&local_elapsed, &elapsed, 1, MPI_DOUBLE, MPI_MAX, 0,
comm);
	if(rank==0){

		printf("Elapsed time = %f seconds\n", elapsed);
	}

	MPI_Finalize();

	return 0;
}
```

Ejecuciones:

Se ejecutó el algoritmo 10 veces para cada caso y luego se obtuvo un valor promedio para cada uno:

Ejecuciones					
	1024	2048	4096	8192	16384
1	0,008955	0,035377	0,141282	0,563947	2,36124
	0,008885	0,035484	0,141108	0,565297	2,319567
	0,008948	0,035437	0,144954	0,562056	2,252569
	0,008977	0,03561	0,142285	0,567061	2,272706
	0,008983	0,035384	0,145787	0,563295	2,25695
	0,00895	0,0353	0,145613	0,563944	2,250536
	0,008938	0,035314	0,143529	0,585373	2,251094
	0,008973	0,035323	0,151014	0,569035	2,239631
	0,008988	0,035241	0,14083	0,566512	2,242265
	0,008903	0,036424	0,143813	0,563668	2,235656
2	0,009546	0,022077	0,088664	0,375716	1,415271
	0,005925	0,022177	0,103724	0,35027	1,421098
	0,005627	0,023988	0,148086	0,350129	1,399014
	0,009522	0,02228	0,087631	0,349345	1,400372
	0,00566	0,022457	0,087709	0,596208	1,420216
	0,005728	0,022149	0,106581	0,349666	1,397593
	0,00569	0,022114	0,08805	0,370833	1,396216
	0,005859	0,022314	0,0884	0,352816	1,400252
	0,006123	0,022235	0,087729	0,350892	1,398934
	0,009505	0,037525	0,087556	0,350006	1,409409
4	0,006383	0,025198	0,098629	0,392985	1,590762
	0,006392	0,025034	0,099834	0,400996	1,57913
	0,006511	0,03265	0,099469	0,398353	1,589608
	0,006297	0,025801	0,099654	0,398828	1,629073
	0,006527	0,036533	0,09887	0,400721	1,589553
	0,006534	0,025268	0,101967	0,396055	1,615821
	0,006372	0,025031	0,101167	0,417661	1,598418
	0,00625	0,024988	0,098272	0,392075	1,58998
	0,0063	0,025315	0,099087	0,395886	1,581738

	0,006802	0,02754	0,098511	0,399524	1,621562
8	0,019056	0,034776	0,163812	0,587064	2,394955
	0,013223	0,04726	0,128963	0,571657	2,501915
	0,012691	0,052775	0,126607	0,569569	2,411406
	0,017294	0,041721	0,121736	0,574774	2,438641
	0,013098	0,055694	0,132495	0,572658	2,420457
	0,017539	0,061334	0,148944	0,57485	2,415124
	0,012763	0,042391	0,133264	0,563532	2,458999
	0,012982	0,0335	0,121932	0,578227	2,439173
	0,020272	0,063107	0,119596	0,580608	2,387418
	0,016415	0,040293	0,130129	0,591239	2,419301
16	0,027406	0,084753	0,296774	1,050799	4,004641
	0,034026	0,075889	0,257011	1,012635	3,991913
	0,022118	0,074058	0,260654	1,030657	4,102444
	0,033397	0,070854	0,261732	1,040919	4,089053
	0,030389	0,082299	0,26547	1,007034	4,001436
	0,027518	0,117671	0,265122	1,007019	4,118658
	0,022669	0,080746	0,257433	1,010845	4,10194
	0,021738	0,076353	0,254667	1,009879	4,060337
	0,021299	0,094779	0,265738	1,026529	4,065278
	0,027836	0,078235	0,265499	1,101845	4,078811

Run - Time:

Run-time					
	1024	2048	4096	8192	16384
1	0,00895	0,0354894	0,1440215	0,5670188	2,2682214
2	0,0069185	0,0239316	0,097413	0,3795881	1,4058375
4	0,0064368	0,0273358	0,099546	0,3993084	1,5985645
8	0,0155333	0,0472851	0,1327478	0,5764178	2,4287389
16	0,0268396	0,0835637	0,26501	1,0298161	4,0614511

Speedup:

Speedup					
	1024	2048	4096	8192	16384
1	1	1	1	1	1
2	1,293633013	1,482951412	1,478462834	1,493773909	1,613430713
4	1,390442456	1,298275521	1,446783397	1,420002184	1,418911405
8	0,576181494	0,7505408681	1,084925701	0,9836941191	0,9339091164
16	0,3334624957	0,4246987627	0,5434568507	0,5506019958	0,5584756148

Efficiency:

Efficiency					
	1024	2048	4096	8192	16384
1	1	1	1	1	1
2	0,6468165065	0,7414757058	0,7392314168	0,7468869546	0,8067153565
4	0,347610614	0,3245688804	0,3616958492	0,3550005459	0,3547278511
8	0,07202268674	0,09381760851	0,1356157127	0,1229617649	0,1167386395
16	0,02084140598	0,02654367267	0,03396605317	0,03441262474	0,03490472592

Conclusiones:

Observando los datos de la tabla de eficiencia, podemos decir que la eficiencia disminuye siempre que se aumenta el número de procesos, así como también disminuye cuando se incrementa el tamaño del problema, por lo tanto podemos concluir que el programa no es fuertemente escalable, ni débilmente escalable pues no cumple con ninguna de las dos definiciones. No mantiene una eficiencia constante independientemente del número de procesos. No mantiene una eficiencia constante cuando se incrementa el número de procesos y el tamaño del problema.

Observamos que el mejor desempeño se logra cuando se utilizan 2 procesos, esto puede deberse a que está ocurriendo un paralelismo real ya que las pruebas se realizaron en un procesador con dos núcleos reales y dos virtuales