INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DESIGN AND MANUFACTURING KANCHEEPURAM

LAB ASSIGNMENT 5 - REPORT
ON
MATRIX ADDITION
AND
MATRIX MULTIPLICATION
IN MPI

SUBMITTED BY
AMAR KUMAR
(CED17I029)
TO
DR. NOOR MAHAMMAD

MATRIX ADDITION

Strategy

In my program for matrix addition, the instruction which is running in parallel is in the "for" loop i.e **c[i][j] = a[i][j] + b[i][j];**

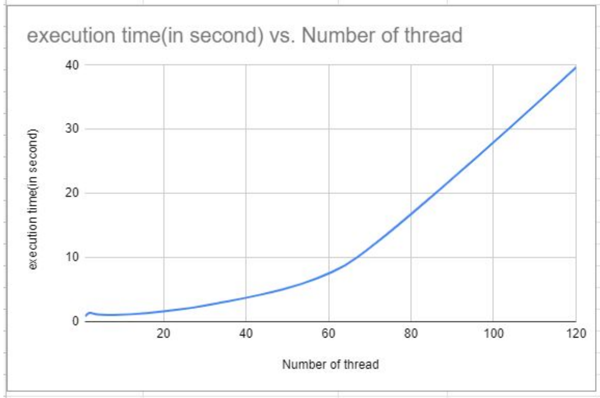
Instead of running the program serially, we can distribute the task of adding matrices between master and worker so that my program could run parallely and in turn save the execution time.
Therefore , I have divided the rows of matrix between master and workers according to the number of processors available so that matrix addition can happen parallely.
I calculated the sum in master and sent the remaining portion of matrices to workers to calculate the sum.

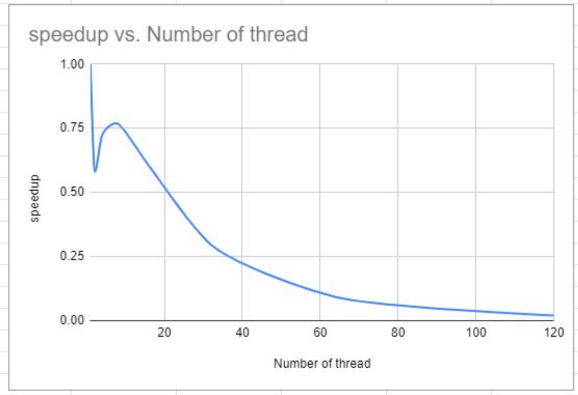
Graph and tables

https://docs.google.com/spreadsheets/d/1LFspvqvPAKYoleIqAJCE7h1Rue09EXuUvWG70YgU-KY/edit#gid=0

Question1

Number of thread	execution time(in second)	speedup	parallelization fraction(f)
1	0.808579	1	0
2	1.362713	0.593359717	-1.370636635
4	1.117148	0.7237886117	-0.5088251529
8	1.052631	0.7681504725	-0.3449465933
16	1.337499	0.6045454987	-0.697744232
32	2.732962	0.2958617793	-2.456729486
64	8.785991	0.0920304835	-10.02256748
120	39.65332	0.0203912055	-48.44445304





Calculation of parallelization fraction

T(1) =0.808579 seconds
Here , for P = 1 the execution time is minimum
T(P) = 0.808579 seconds
Speedup = $\frac{T(1)}{T(P)}$ = $\frac{0.808579}{0.808579}$ = 1
From Amdahl's Law,
Speedup = $\frac{1}{(f/P) + (1-f)}$ Where , f = Parallelization factor P = Thread Number

So, f = $\frac{(1-T(P)/T(1))}{(1-(1/P))}$
Therefore, f = 0 which means that the program is not parallelizable.

MATRIX MULTIPLICATION

Strategy

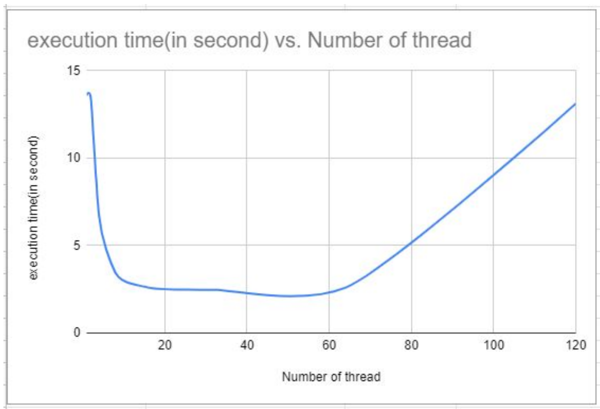
n my program for vector addition, the instruction which is running in parallel is in the "for" loop i.e **c[i][j] += a[i][k] * b[k][j];**

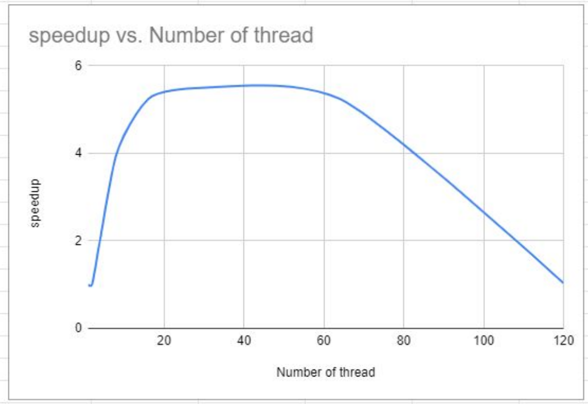
Instead of running the program serially, we can distribute the task of multiplying matrices between master and worker so that my program could run parallely and in turn save the execution time.
Therefore , I have divided the rows of matrix between master and workers according to the number of processors available so that matrix multiplication can happen parallely.
I calculated the multiplication in master and sent the remaining portion of matrices to workers to calculate the matrix multiplication.

Graph and tables

https://docs.google.com/spreadsheets/d/1LFspvqvPAKYoIelqAJCE7h1Rue09EXuUvWG70YgU-KY/edit#gid=0

Number of thread	execution time(in second)	speedup	parallelization fraction(f)
1	13.592836	1	0
2	13.418825	1.012967678	0.02560333988
4	6.698961	2.029096154	0.6762263102
8	3.437566	3.954203643	0.8538338031
16	2.595961	5.236148001	0.862954574
32	2.468261	5.5070497	0.8448150377
64	2.593052	5.242022142	0.8220789058
120	13.110976	1.036752413	0.03574745136





Calculation of parallelization fraction

T(1) =13.592836 seconds
Here , for P = 32 the execution time is minimum
T(P) = 2.468261 seconds
Speedup = $\frac{T(1)}{T(P)}$ = $\frac{13.592836}{2.468261}$ = 5.5070497

From Amdahl's Law,
Speedup = $\frac{1}{(f/P) + (1-f)}$ Where , f = Parallelization factor P = Thread Number

So, f = $\frac{(1-T(P)/T(1))}{(1-(1/P))}$
Therefore, f = 0.8448150377 which means that approx 84% of the program is parallelizable.