CS 4373/6373 High Performance Computing
Spring 2022
Final Project
600 Points

Finding the Determinant of a Matrix Problem

Instructions:

Build a serial and parallel implementation of finding the determinant of a matrix. The determinant must be reported in two formats: the first, the actual determinant; and the second, the $\log_{10}$ (logarithm base-10) of the absolute value of the determinant: $\log_{10}$(absolute_value(determinant)). The speedup and efficiency for each test case must be found. The objectives are to develop a parallel algorithm to solve this problem and then implement this algorithm to achieve (1) the greatest speedup possible for the largest input size and (2) the greatest efficiency for all input sizes and test cases.

Use Hammer for this problem.

Use the serial algorithm to find a benchmark time for each input size and test case. Then use this to determine the speedup and efficiency of the parallel runs for the input size and test case.

Print at least the following items to the standard output: (1) the input matrix file used, (2) the input matrix size, (3) determinant, and (4) the $\log_{10}$(absolute_value(determinant)). Use the following format to print these items (include the "(1)", "(2)", "(3)", and "(4)" in the output):

(1) <input matrix file filename>
(2) <input matrix size>
(3) <value of the actual determinant>
(4) <value of the $\log_{10}$ determinant>

Note: Take the $\log_{10}$(absolute_value(<matrix element>)) for **each** multiplication step in the calculation of the determinant value, additions can be done without taking the $\log_{10}$(absolute_value(<matrix element>)). When performing row or column manipulations on the matrix work with the original floating point values - do not take the $\log_{10}$(absolute_value(<matrix element>)).

Sample code for reading the matrices:

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


#define ARRAYSIZE 16

int main(int argc, char *argv[])
{
   char f_name[50];
   double a[ARRAYSIZE][ARRAYSIZE];
   int i,j;
   double det;
   //Create filename
   sprintf(f_name,"m0016x0016.bin");
   printf("Reading array file %s of size %dx%d\n",f_name,ARRAYSIZE,ARRAYSIZE);
   //Open file
   FILE *datafile=fopen(f_name,"rb");
   //Read elelements
   for (i=0; i< ARRAYSIZE; i++)
      for (j=0; j< ARRAYSIZE; j++)
      {
         fread(&a[i][j],sizeof(double),1,datafile);
         printf("a[%d][%d]=%f\n",i,j,a[i][j]);
      }
         printf("Matrix has been read.\n");
}
```

Matrix files for determinant calculation:

m0016x0016.bin
    det=3.346535e-04
    log(abs(det))=-3.475405e+00
m0032x0032.bin
    det=-2.633256e-01
    log(abs(det))=-5.795069e-01
m0064x0064.bin
    det=3.788989e+07
    log(abs(det))=7.578523e+00
m0128x0128.bin
    det=-3.725889e+38
    log(abs(det))=3.857123e+01
m0256x0256.bin
    det=-1.254321e+115
    log(abs(det))=1.150984e+02
m0496x0496.bin
    det=5.948068e+292
    log(abs(det))=2.927744e+02
m0512x0512.bin
    det=-1.908922e+304
    log(abs(det))=3.042808e+02
m1000x1000.bin
    det=+inf
    log(abs(det))=7.431221e+02
m1024x1024.bin
    det=+inf
    log(abs(det))=7.667341e+02
m2000x2000.bin
    det=+inf
    log(abs(det))=1.786385e+03
m2048x2048.bin
    det=+inf
    log(abs(det))=1.838927e+03
m3000x3000.bin
    det=-inf
    log(abs(det))=2.945781e+03
m4000x4000.bin
    det=-inf
    log(abs(det))=4.176335e+03
m4096x4096.bin
    det=-inf
    log(abs(det))=4.297334e+03

Note: Numeric results generated using the FLOAT datatype (single precision) in ANSI C

Traveling Salesperson Problem

Instructions:

Find the shortest path through all cities in the 1000x1000 city world and return to the starting city. The distance between each city is given in the CSV file included with the project. Please note that the CSV file has some text information on the first few rows that needs to be removed before proceeding. The time limit is 60 seconds for a run and includes reading the input file (list of distances between cities). This time limit does not include writing the output file/standard output (stdout).

The result must be displayed as

- the total distance calculated for the circuit;
- the number of cities traversed (must be all 1000); and
- a list of all cities traversed in the order they are visited: the first city in the list is the first city visited.

The result is to be printed to the standard output.

The objective is to find the best solution that you can find within the given time limit of 60 seconds.

You may visit a city multiple times in a circuit.

You can select the starting city but must return to this city. How you selected the starting city must be described in your report (Hint: It should be part of your algorithm.).

The distance from city A to city B may or may not be identical to the distance from city B to city A.

Use Hammer for this problem.

Report:

Produce a professional report documenting your work, approach, and results for both problems. Make sure to include the following items in the report. Upload a single report as a single PDF with the requested information included and also include the source code files for the parallel and serial implementations.

- Description of algorithm and how this will be parallelized.
- Description of the implementation of the algorithm on Hammer. How was it tuned to Hammer? What parallel programming libraries were used?
- Results obtained from testing.
- Output from the test runs.
- Analysis of those results.
- Is the program weakly or strongly scalable? How do you know or justify your response?
- Source code for the implementation of the parallel algorithm and the serial algorithm.
- One paragraph describing the contributions of each team member (1-2 sentences per team member is fine).

Presentation:

For each problem include a description of your algorithm, the approach used to parallelize the algorithm and its implementation, the results you obtained, your analysis of those results, and responses to all questions in the problem and report. The presentation is to be 10-12 minutes in length, 12 minutes is the limit. A Q/A session will follow each presentation. The format for the presentation slides is Microsoft PowerPoint or PDF.

What to Submit:

Please submit the following items for this project. **All team members are to submit a copy of these materials. All team members must be present for the presentation. All team members will receive the same grade for this project.**

- Report as a single PDF file
- Presentation slides as a single Microsoft PowerPoint or PDF file
- Source code for each problem