CSI 702 FALL 2008 FINAL Exam

The test is closed book, closed notes, no web access. Answers should be brief and concise. Please use additional paper to answer these questions, and write your name at the top of each page. **Good luck!**

Honor Code Certification
Name :

Time started:

Time completed:

I certify that I have abided by the GMU honor code in taking this examination. The work on this exam is my own. I have received no assistance from other persons in completing this exam. I have not consulted any sources of information other than a calculator.

Signature:

1. (20 pts) Optimization
   (a) (10 pts) List the five most important rules you use in order to optimize your serial code (in C, C++ or Fortran) run efficiently on a typical PC. (This is somewhat subjective, so explain their importance VERY briefly.)
   (b) (10 pts) Briefly define/explain the following terms:
      i. Mutex variable
      ii. Atomic operations
      iii. Race conditions
      iv. Deadlocks
      v. Joining a thread

2. (20 pts) The following code segment calculates forces, potentials, and kinetic energies for particles in a molecular dynamics simulation. Rewrite this code so it will execute using MPI. Assume this code will run both the workstations (wk01-wk09) and also on the gmice machine. Explain when the code will run efficiently. **Assume that you only need to rewrite THIS section of code and that the data you need has been read into each node.**

```
! modified from  http://www.openmp.org/drupal/samples/md.html
do i=1,np

  ! compute potential energy and forces
  f(1:nd,i) = 0.0

  do j=1,np

   if (i .ne. j) then

     ! find the distance and radius vector
     ! between the ith and jth particle
     call dist(nd,pos(1,i),pos(1,j),rij,d)

     ! attribute half of the potential energy to particle 'j'
     pot = pot + 0.5*v(d)

     ! for each fo the dimensions, calc the forces on the
     ! ith particle from the jth particle
     do k=1,nd
       f(k,i) = f(k,i) - rij(k)*dv(d)/d
     enddo

    endif

  enddo

! compute kinetic energy
  kin = kin + dotr8(nd,vel(1,i),vel(1,i))

enddo
```

3. (20 pts) Explain what each of the following CUDA statements does and where it is used in a CUDA program.

   (a) `const int i = blockDim.x*blockIdx.x + threadIdx.x;`

   (b) `cudaMalloc((void**)&a, sizeof(float)*N);`

   (c) `initialize<<<Dg, Db>>>(b, 1.2f);`

   (d) `cudaMemcpy(host_c, c, N*sizeof(float), cudaMemcpyDeviceToHost);`

   (e) `dim3 Dg(N/block_size), Db(block_size);`

4. (15 pts) In about one page, compare the following HPC hardware solutions. Explain the advantages, disadvantages and limitations to each approach. Include a very short overview of how we program each of these types of hardware. In general, what are the advantages and disadvantages of using special purpose hardware for High Performance computing?

   (a) FPGA (Fully Programmable Gate Arrays)

   (b) GPGU (General Purpose Graphics Units)

   (c) The GRAPE project

5. (15 pts) Table 1 and Figure 1 show the user time and real time it takes to run a particular code running on a desktop workation as a function of the problem size. Answer the following questions about these figures and plots.

   (a) Problem sizes of less than 90, how does the cpu time scale with the problem size?

   (b) What is the most likely cause for the difference between the "real time" and "user time" in the graphs?

   (c) What is the most likely reason why the graph suddenly changes at a problem size of about 90?

   (d) Explain two approaches that might be used to improve the performance of this program for large problem sizes?

6. (25 pts) Table 2 and Figure 2 show the real time needed to run a particular code on the gmice cluster using between 1 and 64 nodes. Answer the following questions about these figures and plots.

   (a) What is the speedup of the code when use 32 nodes?

(b) What is the efficiency of the code when we use 32 nodes?

(c) What is the best speed up we could expect with this code assuming we use all 640 cores with no additional optimizations?

(d) What is the main reason a code loses efficiency as we increase the number of nodes it is running on?

(e) At what is the maximum number of nodes you would want to run this code on before you worked on improving its performance? (This is somewhat subjective, so justify your answer.)

Table 1

problem size	user time	real time
15	4.82231	17.6979
20	11.8697	40.527
25	19.0307	87.1334
30	33.3316	137.425
35	68.1827	233.92
40	85.7763	302.707
45	114.493	456.803
50	172.176	568.532
55	209.8	791.61
60	291.746	976.595
65	365.931	1379.35
70	486.894	1589.39
75	577.796	2236.98
80	670.466	2617.25
85	807.306	3720.83
90	1034.54	4445.97
95	1131.57	3813.03
100	11585.7	305393
105	15793	361976
110	18229.3	432683

Table 2

number of nodes	run time
1	813.989
2	397.597
4	215.124
8	124.914
16	73.5243
32	53.2772
64	39.0162

Figure 1

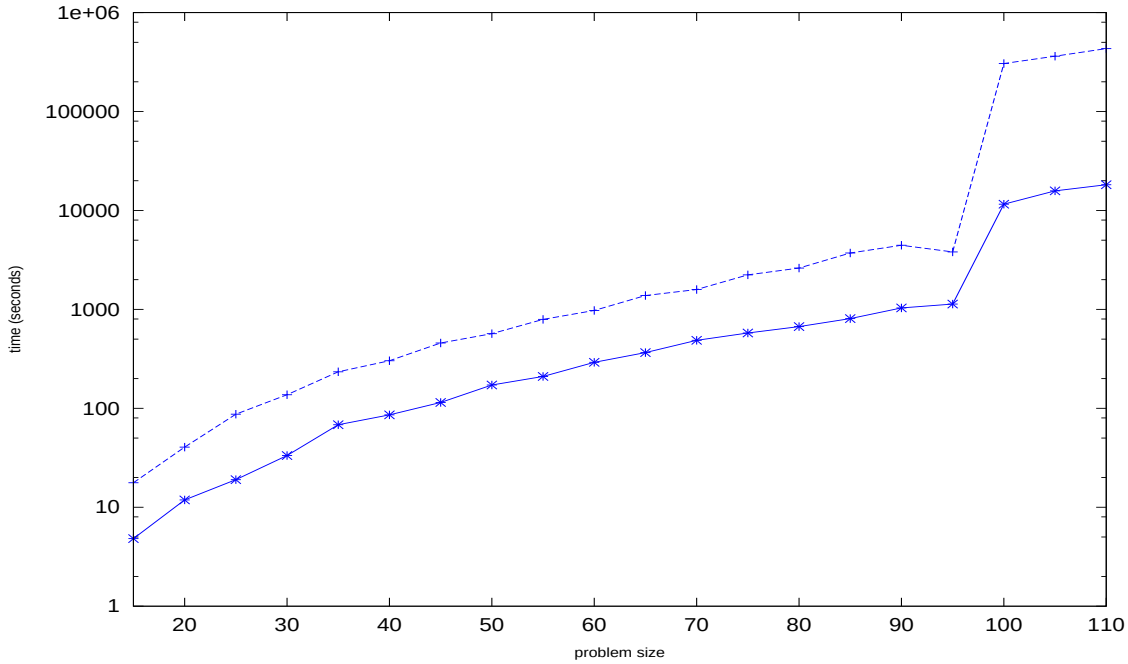

Figure 2