CS4961: Parallel Programming Midterm Exam
October 8, 2009

Instructions:
This is an in-class, open-book, open-note exam. Please use the paper provided to submit your responses. You can include additional paper if needed. The goal of the exam is to reinforce your understanding of issues we have studied in class.

CS4961: Parallel Programming
Midterm Quiz
October 8, 2009

I. Definitions (30 points)

Provide a very brief definition of the following terms:

a. Amdahl's Law

b. Parallelism granularity

c. λ in the CTA model

d. Contention

e. MIMD

f. Data parallelism

g. Concurrent_hash_map in TBB

h. +/ in Peril-L

i. race condition

j. load imbalance

II. Problem Solving (60 points)

In this set of three questions, you will be asked to provide code solutions to solve particular problems. This portion of the exam may take too much time if you write out the solution in detail. I will accept responses that sketch the solution, without necessarily writing out the code or worrying about correct syntax. Just be sure you have conveyed the intent and issues you are addressing in your solution.

a. A multiprocessor consists of 100 processors, each capable of a peak execution rate of 2 Gflops (i.e., 2 billion floating point operations per second). What is the peak performance of the system as measured in Gflops for an application where 10% of the code is sequential and 90% is parallelizable?

b. Given the following code, which is representative of a Fast Fourier Transform:

```
procedure FFT_like_pattern(A,n) {
float *A;
int n, m;

m = $\log_2 n$;
for (j=0; j<m; j++) {
  k = $2^j$;
  for (i=0; i<n; i++)
    A[i] = A[i] + A[i XOR $2^j$];
}
}
```

(i) What are the data dependences on loops i and j?

(ii) Assume n = 16. Provide OpenMP or Peril-L code for the mapping to a shared-memory parallel architecture.

(c) Construct a task-parallel (similar to producer-consumer) pipelined code to identify the set of prime numbers in the sequence of integers from 1 to n. A common sequential solution to this problem is the sieve of Erasthones. In this method, a series of all integers is generated starting from 2. The first number, 2, is prime and kept. All multiples of 2 are deleted because they cannot be prime. This process is repeated with each remaining number, up until but not beyond sqrt(n). A possible sequential implementation of this solution is as follows:

```
for (i=2; i<=n; i++) {
   prime[i] = true;
for (i=2; i<= sqrt(n); i++) {
   if (prime[i]) {
     for (j=i+i; j<=n; j = j+i) { // multiples of i are set to non-prime
         prime[j] = false;
  }
}
```

The parallel code can operate on different values of i. First, a series of consecutive numbers is generated that feeds into the first pipeline stage. This stage eliminates all multiples of 2 and passes the remaining numbers onto the second stage, which eliminates all multiples of 3, etc. Although this is not the most efficient solution, assume that each pipeline stage sends a single number to the next stage. The parallel code terminates when the "terminator" element arrives at each pipeline stage.

III. Essay Question (10 points)
Write a very brief essay about one of the following four topics, no more than 5 sentences.

a. Describe the technology drivers that have led to the multi-core paradigm shift.
b. Why is control flow a performance concern in SIMD architectures?
c. Why is data locality so important to the performance of parallel code?
d. What are the reasons why a parallel version running on 2 or more processors of a sequential code might run slower than the sequential version?