Lecture 5: Multithreaded Programming

CS178: Programming Parallel and Distributed Systems

February 7, 2001
Steven P. Reiss

I. Overview

A. Topics covered last time

1. Threads and their operations

2. Synchronizationn primitives

a) Critical sections

b) Producer-consumer problem

c) Readers-writers problem

d) Waiting in a critical region

3. Multithreaded Java Programming

a) Concepts

(1) Determine what can be done in parallel

(2) Determine what are the shared data structures

(3) Determine how to create/stop the threads

b) Problems

(1) Deadlock -- dining philosophers

(2) Performance

II. Prime Number Computations

A. Problem

1. Consider the problem of finding prime numbers

a) Generate list of prime numbers

b) For each number, check it against each number in the list up to its square root

2. What can be done in parallel

3. What are the shared data structures

4. How to create/stop threads

B. Basic loop

```
add(2)
add(3)
M = 3
for ( ; ; ) {
   for (N = 5; N < (M+1)**2; N += 2) {
      th = create thread(N)
      start th
   next
   waitfor(th(M+2))
   M = M+2
next
```

1. **Need to modify this to limit the number of threads running at once**

C. Thread code

D. List data structure

1. **Lots of readers -- don't want to block these**
2. **Multiple writers**
3. **Note that writers are adding to list beyond where readers are reading -- they need not block readers**
4. **How to maintain this**
   a) Simple -- list with synchronized add end
   b) Alternative -- list with synchronized adds

III. Collocations

A. The problem

1. **Web base (20G+ of web pages)**
2. **Want to find what strings constitute phrases**
   a) f(A B) compared to f(A) and f(B) statistically
   b) Need to get counts for all words and sequences
   c) Up to length 7.

B. Simple approach

1. **Keep hash table of words, phrases in memory**
2. **Go through source, increment counts for each word and phrase as a word is read**
3. **Access counts at the end to determine which phrases are valid ones.**
4. **Problems**

a) Can't fit any of the hash tables in memory
b) Lots of file I/O processing needs to be done

C. How might we make this run faster

1. What can be done in parallel (what are the threads)
2. What are the common data structures
3. How to start/stop threads

IV. Sorting (Next Assignment)

A. What have you learned about sorting

1. What is the costly part of the process

a) I/O -- what helps here if anything
b) Allocation in memory

2. Other things

B. What should you consider in multithreading

1. What can be done in parallel
2. What are the common data structures
3. How to start/stop threads