Preliminaries

- Introductions
- Why are you taking this course?
- Discuss syllabus: online
- Cluster accounts

A motivating example

- Find the max of n numbers
- How would you do it?
- Assumptions?
- How many comparisons?

Find max of n numbers

- Case 1: sorted (ascending)
  - no comparisons -- just return last number in array
  - O(1) comparisons
  - It's easy when numbers already sorted!
Find max of n numbers

• Case 2: unsorted
  • sort array first, then return max
  • \(O(n \log(n))\) comparisons
  • a lot of work just to return the max!

Find max of n numbers

• Case 3: unsorted
  • Compare unsorted numbers from first to last, keep track of max as you go
  • \(O(n)\) comparisons
  • sure beats sorting them all first!

Find max of n numbers

• Let’s pause for a moment...
  • the unsorted cases are the interesting ones...
  • it takes \(n-1\) comparisons to find the max of \(n\) unsorted numbers \(\sim O(n)\)
  • can we do better than \(O(n)\)?
Find max of n numbers

array A: \[8 \ 6 \ 4 \ 2 \ 1 \ 3 \ 5 \ 7\]

Max = ?

All numbers must be compared at least once to find the max, but the pairs we choose to compare, and the order of those comparisons, depends on the algorithm.

Order. Traditional (imperative) programming causes us to become (necessarily) obsessed with order...

It’s time to break out of the sequential box.

Find max of n numbers

• Why were we counting comparisons?
  • a measure independent of machine speed
• Is this still what we want if not sequential? (i.e., can do >1 comparisons concurrently)
• We need new measures!
• Like what? (time, speedup, cost, work, efficiency)

Find max of n numbers

• Let \( T_1(N) \) be the Best Sequential Algorithm
• Let \( T_P(N) \) be the Time for Parallel Algorithm (P processors)
• The Speedup \( S_P(N) \) is \( T_1(N)/T_P(N) \)
• The Cost \( C_P(N) \) is \( PT_P(N) \), assuming P processors
• The Work \( W_P(N) \) is the summation of the number of steps taken by each of the processors. It is often, but not always, the same as Cost.
• The Cost Efficiency \( CE_P(N) \) (often called efficiency \( Ep(N) \)) is \( S_P(N)/P = C_1(N)/C_P(N) = T_1(N)/(PT_P(N)) \)
• The Work Efficiency \( WE_P(N) \) is \( W_1(N)/W_P(N) = T_1(N)/W_P(N) \)

Find max of n numbers

A: \[8 \ 6 \ 4 \ 2 \ 1 \ 3 \ 5 \ 7\]

A binary tree!

Comparisons in each row can be performed in parallel

Time = # rows = \log(n)

How many processors needed?

Tradeoffs of time, cost, efficiency...
Find max of n numbers

A: 8 6 4 2 1 3 5 7

This problem can be solved in
Time = O(1)

How?
Next time...

Reading Assignment 1

• Read Sutter and Larus article from September 2005 issue of ACM Queue
• Write one-page summary (no more)
  • to discuss during next class
  • points in article most striking to you

Programming Assignment 1

• Write a C program that finds the max of n numbers
  • Goals
    • access the Rocketcalc cluster (ssh)
    • edit files (vim, emacs, your choice...)
    • compile / execute C program
    • implement sequential solution to max

Programming Assignment 1

• Write a C program that finds the max of n numbers
  • main function
    • initializes array (read values from stdin)
    • prints array
    • calls max and prints result
Programming Assignment 1

• Write a C program that finds the max of n numbers
  • max function
    • takes array of integers as parameter
    • iterates through array to find max
    • returns value of max element in given array

Field Trip (right now!)

• Where’s the cluster?
• In the closet!
• Let’s go take a look at it...