cs140 – algorithms
prof. yi chen

september 3, 2013

Big picture

- Solve problems
  - Given a definition
    - put it in context of existing knowledge
    - prove things about it
  - Be able to use common strategies
    - break it down in a structured way
    - decide whether/how to simplify
    - decide if general techniques can be applied and, if so, how
  - Determine if it can be solved efficiently
    - understand the role of heuristics
    - understand the role of implementation
- Collaborate and communicate

Prerequisites/assumptions

- Proficiency with:
  - proof techniques
- Experience with:
  - high-level programming language
  - counting
  - graphs and trees
- Exposure to:
  - big-O notation
  - 1D/2D arrays
  - sorting (one $O(n^2)$ sort)
  - basic matrix/vector operations
  - sequential vs. random memory access

two conjectures

- all primes are odd
- if p is prime, then 2^p-1 is prime

Proof techniques

- by example/counterexample
- by enumeration
- by cases
- by inference (aka direct proof)
- trivially
- by the contrapositive
- by contradiction
- by (weak) induction
- by strong induction

all primes are odd
if p is prime, then 2^p-1 is prime

handouts

- Administrivia
  - synchronization
  - collaboration
  - piazza
- piazza
  - q&a
  - resources

asymptotics

f(n) is O(g(n)) if there exist positive constants c and n_0 such that $f(n) \leq cg(n)$ for all $n>n_0$

- Show that $f(n)=120n^2-11n+250$ is $O(n^2)$
- Show that $f(n)=n^3$ is $O(n^2)$

more asymptotics

- Ω (big-Omega)

f(n) is $\Omega(g(n))$ if there exist positive constants c and n_0 such that $f(n) \geq cg(n)$ for all $n>n_0$

f(n) is $\overset{\infty}{\Omega}(g(n))$ if there exists a positive constant c such that $f(n) \geq cg(n)$ for an infinite number of n

- θ (Theta)
- ω (little-omega)
- o (little-o)

Describe (in English sentences) how you would arrange a set of 10 dates in "ascending sorted order", that is, from earliest to latest. You might consider the following list of dates, but make sure that you describe how to do it with any 10 dates.

December 21, 2010
May 1, 1988
July 21, 1970
August 28, 2001
January 31, 2002
June 6, 2010
May 20, 1988
November 5, 1970
April 2, 1999
September 9, 2002

insertion sort

```
insertion-sort(A)
 1.  for j = 2 to length of A
 2.       key = A[j]
 3.       // insert key into sorted sequence A[1:j-1]
 4.       i = j-1
 5.       while ((i>0) and (A[i]>key))
 6.        A[i+1] = A[i]
 7.                  i = i-1
 8.       endwhile
 9.       A[i+1] = key
10.  endfor
```


correctness – loop invariants

- loop invariant
- initialization
- maintenance
- termination

```
insertion-sort(A)
 1.  for j = 2 to length of A
 2.       key = A[j]
 3.       // insert key into sorted sequence A[1:j-1]
 4.       i = j-1
 5.       while ((i>0) and (A[i]>key))
 6.          A[i+1] = A[i]
 7.          i = i-1
 8.       endwhile
 9.       A[i+1] = key
10.  endfor
```