Math 135 - More on algorithms

Note Title 9/27/2012

Announcements

Last time

- Algorithms
- Pseudo code : if, for, while, procedure

- Searching : 2 versions
  - Binary
  - Linear

Sorting

Fundamental CS problem
(which is useful everywhere!)

Given a list of n things, put them in order.

How?

Strategy: Find ~~smallest~~ largest + put it ~~first~~ last.

Repeat!

SORT($a_1, a_2, \ldots, a_n$):

coded last time

```
for i := n to 2
    location := FINDMAX(a_1, ..., a_i)
    temp := a_i
    a_i := a_location
    a_location := temp
```

Bubble sort

Key idea:

Compare a pair & swap if out of order. (And repeat!)

3 2 5 1 4

2 3 5 1 4

2 3 1 5 4

2 3 1 4 5

5 4 3 2 1

4 3 2 1 5

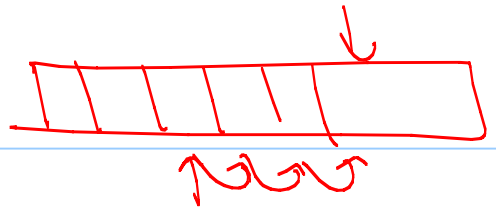
3 2 1 4 5

Pseudo code:

BUBBLESORT $(a_1, \ldots, a_n)$:

```
for i := n down to 2:
    for j := 1 to i-1:
        if a_j > a_{j+1}
            temp := a_j
            a_j := a_{j+1}
            a_{j+1} := temp
```

Insertion Sort



Key Idea:
If first i things are sorted take $(i+1)^{st}$ and move it to the correct spot.

3 4 5 1 2

1 3 4 5 2

Pseudocode:

INSERTIONSORT$(a_1, \ldots, a_n)$:

for $j := 2$ to n:

in book

Another: Making Change

You have a pile of pennies (1¢), nickels (5¢), dimes (10¢), and quarters (25¢).

Goal: Use fewest coins possible to give change.

Ex: 67¢ : 2 quarters, 1 dime, 1 nickel, 2 pennies

39¢ : 1 quarter, 1 dime, 4 pennies

Making Change

This is an example of a greedy algorithm:

At each step, choose largest possible value.

Pseudocode

coin types, with $c_1 > c_2 > \cdots > c_r$

```
MAKINGCHANGE(n, c_1, c_2, ..., c_r):
    for i := 1 to r
        d_i := 0
        while (n ≥ c_i)
            d_i := d_i + 1
            n := n − c_i

    return d_1, ..., d_r
```

$d_1, \ldots, d_r$: how many of each coin type to use.

Thm: The greedy algorithm returns the fewest possible coins.

pf: By contradiction.

Let p, n, d, q be our # of coins,
Let p', n', d', q' be optimal.

and $q' + d' + n' + p' < q + d + n + p$.

Consider quarters:
Opt. can't have more quarters, because my algorithm used maximum possible.

If opt used fewer quarters,
theirs must actually be bigger.
(takes ≥ 3 coins to replace a quarter)

So $q = q'$.

Consider d: can $d' > d$?

impossible — we used max #

If $d' < d$, they're worse, since
≥ 2 coins to replace each dime.

$\Rightarrow d = d'$

⋮

$\Rightarrow q = q'$, $d = d'$, $n = n'$, $p = p'$ same! ↯

Question: Are some problems unsolvable?

Halting problem: Is there a procedure that takes as its input:
- a program
- an input to the program

and then decides if the program halts or runs forever on that input.

Output: yes or no

Why useful? detect infinite loops

Note: Our program can't just run the other program on its input.

Why?

Thm: The halting problem is undecidable
(that is, no such program can exist.)

pf: by contradiction