Tuesday

Week 5 - [redacted] lecture highlights

- Distinguish between and use as appropriate each of structural induction, mathematical induction, and strong induction
- Prove correctness of iterative and recursive algorithms using induction

Recall: Proof by Strong Induction (Rosen 5.2 p337, zybooks 8.1)
To prove that a universal quantification over the set of all integers greater than or equal to some base integer b holds, pick a fixed non-negative integer j and then:

Basis Step:	Show the statement holds for b, $b+1$, $\ldots$, $b+j$.
Recursive Step:	Consider an arbitrary integer n greater than or equal to $b+j$, assume (as the **strong induction hypothesis**) that the property holds for **each of** b, $b+1$, $\ldots$, n, and use this and other facts to prove that the property holds for $n+1$.

For which non-negative integers n can we make change for n with coins of value 5 cents and 3 cents?

Restating: We can make change for __________, we cannot make change for _______________, and

______________________________________$\star$

Proof of $\star$ by mathematical induction ($b = 8$)

Basis step: WTS property is true about 8

Inductive step: Consider an arbitrary $n \geq 8$. Assume (as the IH) that there are nonnegative integers x, y such that $n = 5x + 3y$. WTS that there are nonnegative integers x', y' such that $n + 1 = 5x' + 3y'$. We consider two cases, depending on whether any 5 cent coins are used for n.

Case 1: Assume .
Define $x' =$
and $y' =$
(both in $\mathbb{N}$ by case assumption).
Calculating:

$5x' + 3y' \overset{\text{by def}}{=}$

$\overset{\text{rearranging}}{=}$

$\overset{\text{IH}}{=}$

Case 2: Assume .
Therefore $n = 3y$ and $n \geq 8$, by case assumption. Therefore, $y \geq 3$ Define $x' = 2$ and $y' = y - 3$ (both in $\mathbb{N}$ by case assumption). Calculating:

$$5x' + 3y' \overset{\text{by def}}{=} 5(2) + 3(y - 3) = 10 + 3y - 9$$
$$\overset{\text{rearranging}}{=} 3y + 10 - 9$$
$$\overset{\text{IH and case}}{=} n + 10 - 9 = n + 1$$

Proof of $\star$ by strong induction ($b = 8$ and $j = 2$)

Basis step: WTS property is true about $8, 9, 10$

Inductive step: Consider an arbitrary $n \geq 10$. Assume (as the IH) that the property is true about each of $8, 9, 10, \ldots, n$. WTS that there are nonnegative integers x', y' such that $n + 1 = 5x' + 3y'$.

Algorithms for making change

Think about values for which the algorithm does not work → give a witness

Change making (greedy) algorithm in pseudocode

```
procedure change(c_1, c_2, ..., c_r: values of denominations of coins, where c_1 > c_2 > ... > c_r; n: a positive integer)

for i := 1 to r
  d_i := 0 {d_i counts the number of coin of denomination c_i used}
  while n ≥ c_i
    d_i := d_i + 1 {Add a coin of denomination c_i}
    n := n − c_i

return d_1, d_2, ..., d_r {d_i the number of coins of denomination c_i in the change for i=1, 2, ..., r}
```

$c_1 = 5$ & $c_2 = 3$

The greedy approach doesn't work with 5¢ and 3¢ coins even for large values of n. However, we can write two new algorithms inspired by the proofs that we completed using mathematical induction and strong induction.

Recursive algorithms for making change

Write an algorithm based on our proof for coins [induction & strong induction]

using inductive proof

One recursive algo for making change using 5¢ and 3¢ coins

```
procedure change1(n: a positive integer)

    if (n=8)     (d1, d2) = (1,1)

    else {
            (x,y) = change1(n-1)

            if (x=0)     (d1,d2) = (2, y-3)        case (ii)

            else         (d1,d2) = (x-1, y+2)      case (i)
    }
return (d_1, d_2) {d_1, d_2 are the number of 5¢ and 3¢ coins respectively }
```

using strong induction proof

Another recursive algo for making change using 5¢ and 3¢ coins

```
procedure change2(n: a positive integer)

    if (n=8)          (d1, d2) = (1,1)
    else if (n=9)     (d1,d2) = (0,3)
    else if (n=10)    (d1,d2) = (2,0)
    else {
            x,y = change(n-3)
            (d1,d2) = (x, y+1)
    }
return (d_1, d_2) {d_1, d_2 are the number of 5¢ and 3¢ coins respectively }
```

in our strong induction proof the inductive case was proved for $n+1$ & used $n-2 = 3x + y$

$(n+1)-(n-2) = 3$

Note: For ease prove strong inductive case for n assuming claim for $8, \ldots, n-1$ (instead of $n+1$)

Proving correctness of algorithms What does it take to show that some algorithm is correct?

We used induction / strong induction to show that $n = d_1 \cdot 5 + d_2 \cdot 3$

✓ (i) Given a task

✓ (ii) Come up with an algorithm

✓ (iii) To show the correctness of the algorithm

Example 1: Prove that the algorithm findMax is correct

Task: *findMax* takes an input a sequence of numbers and returns the maximum number in the sequence

```
procedure findMax(a_1, a_2, ..., a_n: a sequence of n integers)        recursive
  if (n=1) return a_1
  else{  m = findMax(a_1, ..., a_{n-1})
         if a_n > m
               return a_n
         else
               return m
  }
return
```

How do we show the correctness

→ We'll use induction along with functions

$\text{input} \longrightarrow \boxed{\text{alg}} \longrightarrow \text{output}$

fn

Define

f_{max} : a sequence of integers $\{a_1, \ldots, a_n\}$ $\longrightarrow$ $m \in \{a_1, \ldots, a_n\}$ s.t $m \geq a_i \ \forall i \in \{1, \ldots, n\}$

Claim: The algorithm findMax returns $f_{max}(a_1, \ldots, a_n)$ | $findMax(a_1, \ldots, a_n) = f_{max}(a_1, \ldots, a_n)$

We are going to prove the claim by induction on n.

Prove that the algorithm findMax is correct (contd)

Base Case: $n=1$, the sequence is $\{a_1\}$

$f_{max}(a_1) = a_1$ by definition of f_{max}.

The algorithm returns a_1 & thus returns $f_{max}(a_1)$ proving the claim

Mathematical induction

Inductive Case: We need to show that the algorithm returns $f_{max}(a_1, \dots, a_n)$ for $n>1$.

(i.e) $findMax(a_1, \dots, a_n) = f_{max}(a_1, \dots, a_n)$

By inductive hypothesis we can assume that,

$findMax(a_1, \dots, a_{n-1}) = f_{max}(a_1, \dots, a_{n-1})$

Therefore, let $m = findMax(a_1, \dots, a_{n-1})$

$\forall i \in \{1, \dots, n-1\}, \quad a_i \leq m$ by IH — 1

If $a_n \leq m$ then,

by definition of f_{max} and eqn 1,

$m = f_{max}(a_1, \dots, a_n)$

else if $a_n > m$

then $\forall i \in \{1, \dots, n-1\} \quad a_i \leq m < a_n$

$\therefore a_i < a_n$

by definition of f_{max}, $a_n = f_{max}(a_1, \ldots, a_n)$

Example 2: Prove the correctness of the Division Algorithm

$\therefore$ This completes the proof.

$\therefore$ The algorithm is correct.

Division Algorithm

```
procedure DivisionAlgo(n: positive integer; d: positive integer)











return
```

Example 2: Prove the correctness of the Division Algorithm (contd)