goals:

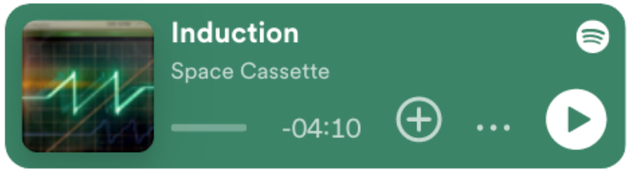
- practice!
- connect induction to recursive algorithms

Middlebury

CSCI 200: Math Foundations of Computing

Spring 2024

Lecture 3F: More Induction



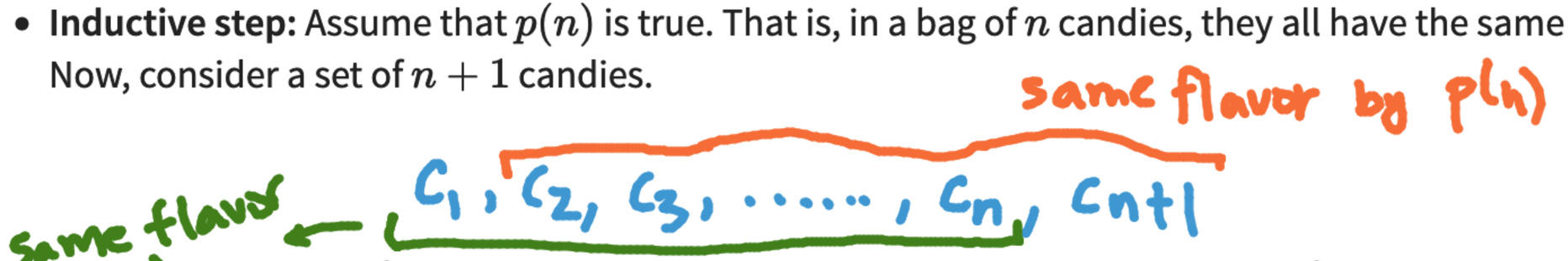
Is this proof okay?

In a bag of n candies ($n \geq 1$), all candies have the same flavor.

Proof: We use a proof by induction on the induction variable n, the number of candies in a bag. Let the induction hypothesis be: $p(n) =$ *in a bag of* n *candies, all candies have the same flavor*.

- **Base case:** Our base case is for $n = 1$, i.e. for a single candy, it has the same flavor as itself.
- **Inductive step:** Assume that $p(n)$ is true. That is, in a bag of n candies, they all have the same flavor.

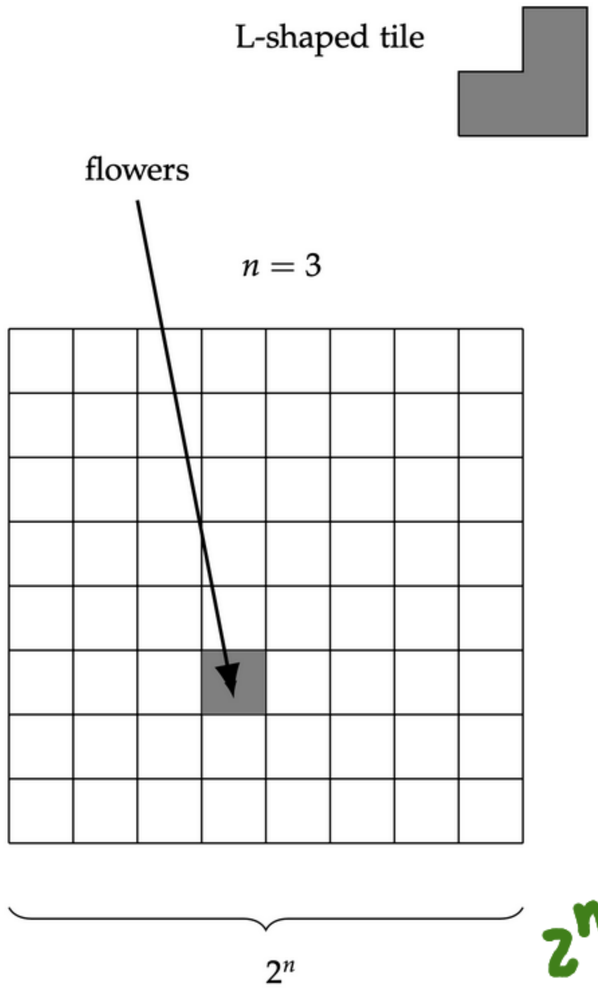
  Now, consider a set of $n + 1$ candies.



  Since c_1 is the same flavor as the candies: $c_2, \ldots, c_n$ and c_{n+1} has the same flavor as the candies: $c_2, \ldots, c_n$, then c_1 and c_{n+1} have the same flavor.

By induction, this means that a bag of n candies all have the same flavor. □

Prove that you can tile any $2^n \times 2^n$ $(n \geq 0)$ courtyard with L-shaped tiles, leaving one space for a pot of flowers.



IH: "a $2^n \times 2^n$ courtyard can be tiled with L-shaped tiles leaving one empty spot (for flowers)."

base case: For $n=0$ $2^0 \times 2^0 = 1$ (no L-shaped tiles)

inductive step: Assume p(n) true.
Show tile $2^{n+1} \times 2^{n+1}$ yard with L leaving 1 spot for flowers.

$= 2(2^n)$

Proving algorithm correctness (or a property).

```
reverseString(s)

    input: s (string)
    output: reversed string
1   if length(s) == 1 :  # base case
2       return s
3   else   # recursive case
4       return reverseString( s[1:] ) + s[0]
```

proof: induction

IH = " reverseString reverses a string of length n"

base: For n=1, Line 2 returns the string itself (which is the reverse of a single char string).

inductive: Assume p(n) true. → reverseString works for strings of length n. Show true for strings of length n+1.

Line 4 returns reverseString(s[1:n+1]) + s[0]

reverseString(s[1:n+1]) — by p(n) this reverses string of len n

s[0] — which is the reverse.

Proving algorithm correctness (or a property).

$\frac{L(1-\alpha^n)}{1-\alpha}$

Prove that the algorithm below draws a total length of $L(1-\alpha^n)/(1-\alpha)$.

```
spiral(n, L, α)

    input: n: number of generations, L: length to draw,
    α: factor to decrease length in next generation
    output: none
1   if n == 0     # base case
2       return
3   else   # recursive case
4       turn_left(30 degrees)     # turns left by 30 degrees
5       draw_line(L)     # draws a straight line of length L
6       spiral(n − 1, αL, α)
```

base case: n=0 Line 2 returns (draw nothing)

$\frac{L(1-\alpha^0)}{1-\alpha} = \frac{L(1-1)}{1-\alpha} = 0$ ✓

inductive step: Assume p(n) → draws $\frac{L(1-\alpha^n)}{1-\alpha}$

Show draws $\frac{L(1-\alpha^{n+1})}{1-\alpha}$

input n+1, L, α → Line 5 draws L

Line 6 draws $\alpha L \frac{(1-\alpha^n)}{1-\alpha}$

n+1−1=n

by p(n)

$L + \frac{\alpha L(1-\alpha^n)}{1-\alpha}$

$\frac{L(1-\alpha) + \alpha L(1-\alpha^n)}{1-\alpha}$

$= L(1-\alpha^{n+1})$ / $(1-\alpha)$