CSE 311

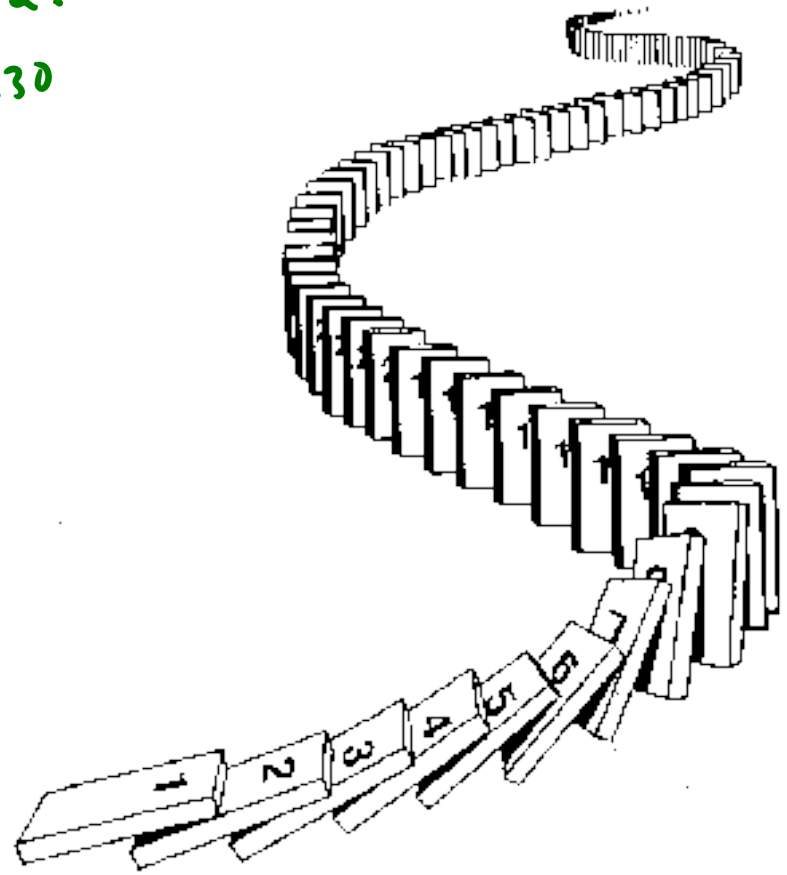
Foundations of Computing I

* All slides are a combined effort between previous instructors of the course

CSE 311: Foundations of Computing

Lecture 15 (6): Strong Induction

Review Session #2:
Tomorrow 2:30 – 4:30
in THO 101



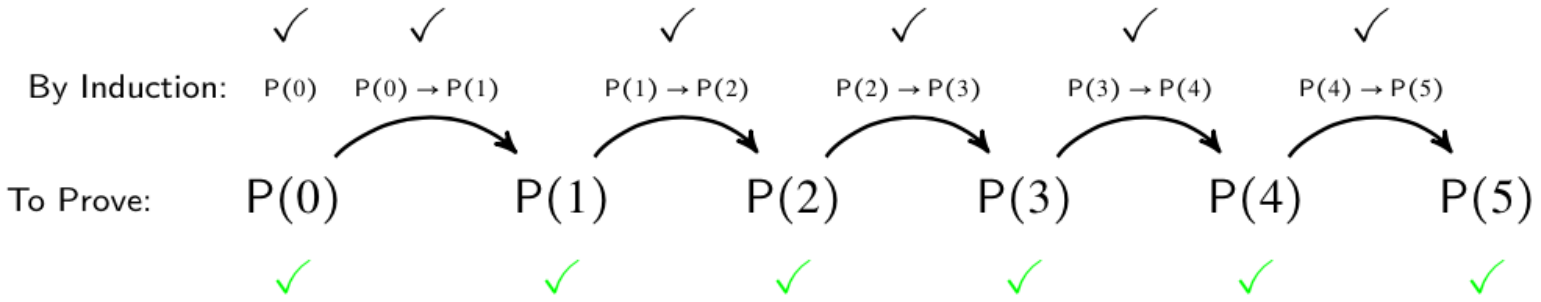
Induction Is A Rule of Inference

Domain: Natural Numbers

$$P(0)$$
$$\forall\, k\, (P(k) \rightarrow P(k+1))$$

$$\therefore\ \forall\, n\, P(n)$$

How does this technique prove P(5)?



First, we prove P(0).
Since P(n) → P(n+1) for all n, we have P(0) → P(1).
Since P(0) is true and P(0) → P(1), by Modus Ponens, P(1) is true.
Since P(n) → P(n+1) for all n, we have P(1) → P(2).
Since P(1) is true and P(1) → P(2), by Modus Ponens, P(2) is true.

Induction Is A Rule of Inference…Again

1.	P(0)	("Given")
2.	$\forall n\,(\mathrm{P}(n) \rightarrow \mathrm{P}(n+1))$	("Given")
3.	P(1)	(MP: 2, 1)
4.	P(2)	(MP: 2, 3)
5.	P(3)	(MP: 2, 4)
6.	P(4)	(MP: 2, 5)

Induction Is A Rule of Inference

"Induction"

1.	P(0)	("Given")
2.	$\forall n\ (\mathrm{P}(n) \rightarrow \mathrm{P}(n+1))$	("Given")
3.	P(1)	(MP: 2, 1)
4.	P(2)	(MP: 2, 3)
5.	P(3)	(MP: 2, 4)
6.	P(4)	(MP: 2, 5)

Notice how when we use regular induction, we're already proving the things necessary to use strong induction.

This is no extra work with a benefit!

"Strong Induction"

1.	P(0)	("Given")
2.	$\forall n\ ((\mathrm{P}(0) \wedge \mathrm{P}(1) \wedge \cdots \wedge \mathrm{P}(n) \rightarrow \mathrm{P}(n+1))$	("Given")
3.	P(1)	(MP: 2, 1)
4.	P(2)	(MP: 2, 1, 3)
5.	P(3)	(MP: 2, 1, 3, 4)
6.	P(4)	(MP: 2, 1, 3, 4, 5)

Strong Induction

$$P(0)$$

$$\forall k \left(\left(P(0) \wedge P(1) \wedge P(2) \wedge \cdots \wedge P(k) \right) \rightarrow P(k+1) \right)$$

$$\therefore \forall n \, P(n)$$

Strong Induction English Proof

1. By induction we will show that $P(n)$ is true for every $n \geq 0$
2. Base Case: Prove $P(0)$
3. Inductive Hypothesis:
   Assume that for some arbitrary integer $k \geq 0$, $P(j)$ is true for every j from 0 to k
4. Inductive Step:
   Prove that $P(k+1)$ is true using the Inductive Hypothesis (that $P(j)$ is true for all values $\leq k$)
5. Conclusion: Result follows by induction

Every $n \geq 2$ can be expressed as a product of primes.

Let $P(n)$ be "$n = p_0 p_1 \cdots p_j$, where $p_0, p_1, \ldots, p_j$ are prime."

We go by strong induction on n.

Base Case (n=2): 2 is prime which is a product of primes.

Induction Hypothesis: Suppose $P(2) \wedge P(3) \wedge \cdots \wedge P(k)$ for some $k \in \mathbb{N}$ for $k \geq 3$

Induction Step: We go by cases.

$k+1 = p_1 p_2 \cdots p_n$

$k = q_1 q_2 \cdots q_m$

Choose a, b, where $a \neq k+1, b \neq k+1, a \neq 1, b \neq 1$ s.t.

$k+1 = a \cdot b$

We know (by IH)...

All numbers smaller than k can be expressed as a product of primes.

We're trying to get...

k can be expressed as a product of primes.

Every $n \geq 2$ can be expressed as a product of primes.

Let $P(n)$ be "$n = p_0 p_1 \cdots p_j$, where $p_0, p_1, \ldots, p_j$ are prime."

We go by induction on n.

Base Case (n=2): Note that 2 is prime (which means it's a product of primes).

Induction Hypothesis: Suppose that P(2), P(3), …, P(k – 1) are true for some $k \geq 2$.

Induction Step: We go by cases.

Case 1 (k is prime):

Then, since k is prime, k is a product of primes.

Case 2 (k is composite):

Then, by definition of composite, we have non-trivial $1 < a, b < k$ such that k = ab. Since a and b are between 2 and k – 1, we know P(2) and P(k – 1) are true. So, we have:

$$a = p_0 p_1 \cdots p_j \text{ and } b = p_{j+1} p_{j+2} \cdots p_{j+\ell}$$

Then, $k = ab = p_0 p_1 \cdots p_j p_{j+1} p_{j+2} \cdots p_{j+\ell}$

So, k can be expressed as a product of primes.

So, P(n) is true for all $n \geq 2$ is true by induction.

We know (by IH)…

All numbers smaller than k can be expressed as a product of primes.

We're trying to get…

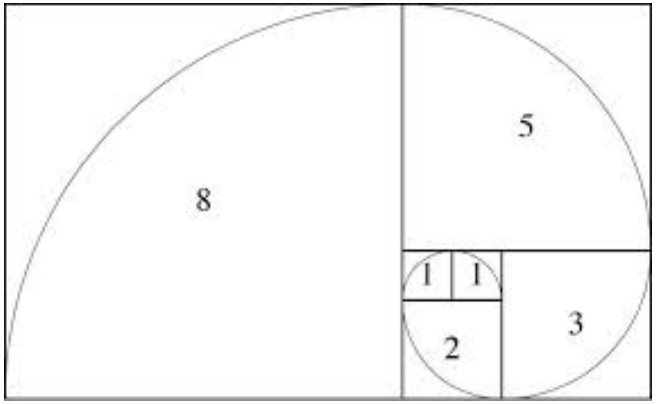
k can be expressed as a product of primes.

Fibonacci Numbers

$f_0 = 0$

$f_1 = 1$

$f_n = f_{n-1} + f_{n-2}$ **for all** $n \geq 2$



Bounding the Fibonacci Numbers

Theorem:
$2^{n/2-1} \leq f_n$ for all $n \geq 2$

$f_0 = 0$
$f_1 = 1$
$f_n = f_{n-1} + f_{n-2}$ for all $n \geq 2$

Let P(n) be "$2^{n/2-1} \leq f_n$" for all $n \geq 2$.
We go by strong induction on n.

Base Case:

note $2^{\frac{2}{2}-1} = 2^0 = 1 = 1 = f_0 + f_1 = f_2$

note

Induction Hypothesis:

Suppose $P(2) \wedge P(3) \wedge \cdots \wedge P(k)$ for some $k \geq 3$, $k \in \mathbb{N}$

Bounding the Fibonacci Numbers

Theorem:
$2^{n/2-1} \le f_n$ for all $n \ge 2$

$f_0 = 0$
$f_1 = 1$
$f_n = f_{n-1} + f_{n-2}$ for all $n \ge 2$

Let P(n) be "$2^{n/2-1} \le f_n$" for all $n \ge 2$.
We go by strong induction on n.

Induction Step:

pick $2^{(k+1)/2 - 1} \le$

$= 2^{k/2-1} + 2^{(k-1)/2-1}$ $\quad k+1 \ge 2$

$\le f_k + f_{k-1}$

$= f_{k+1}$

we know
$2^{\ell/2-1} \le f_\ell$
$2 \le \ell \le k$

WTP
$2^{(k+1)/2-1} \le f_{k+1}$

Bounding the Fibonacci Numbers

Theorem:
$f_n < 2^n$ for all $n \geq 2$

$f_0 = 0$
$f_1 = 1$
$f_n = f_{n-1} + f_{n-2}$ for all $n \geq 2$

Let $P(n)$ be "$f_n < 2^n$" for all $n \geq 2$. We go by induction on n.

Base case ($n = 2$):

Note $f_2 = f_0 + f_1 = 0 + 1 = 1 < 4 = 2^2$. So, $P(2)$ is true.

Note $f_3 = \quad = 2^3$. So, $P(3)$ is true.

IH: Suppose $P(2) \wedge P(3) \wedge \cdots \wedge P(k)$ for some $k \in \mathbb{N}$, where $k \geq 3$.

IS: $f_{k+1} = f_k + f_{k-1}$ by f_n def.

$< 2^k + 2^{k-1}$ by IH.

$= \frac{3}{2} \cdot 2^k$

$< 2 \cdot 2^k$

$= 2^{k+1}$

Bounding the Fibonacci Numbers

Define f_n as: $f_0 = 0$
$f_1 = 1$
$f_n = f_{n-1} + f_{n-2}$ for all $n \geq 2$

Theorem:
$2^{n/2-1} \leq f_n$ and $f_n < 2^n$ for all $n \geq 2$

Proof:
Let P(n) be "$2^{n/2-1} \leq f_n$ and $f_n < 2^n$" for all $n \geq 2$.
We go by strong induction on n.

Base Case: $2^{2/2-1} = 2^0 = 1 \leq 0 + 1 = f_2$, and
$f_2 = 0 + 1 = 1 < 4 = 2^2$. So, P(2) is true.

Induction Hypothesis:
Suppose P(j) for all integers j s.t. $2 \leq j \leq k$ for some $k \geq 2$.

Induction Step: We want to show $2^{(k+1)/2-1} \leq f_{k+1}$ and $f_{k+1} < 2^n$

Bounding the Fibonacci Numbers

Define f_n as: $f_0 = 0$
$f_1 = 1$
$f_n = f_{n-1} + f_{n-2}$ for all $n \geq 2$

Theorem:
$2^{n/2-1} \leq f_n$ and $f_n < 2^n$ for all $n \geq 2$

Induction Step: We want to show $2^{(k+1)/2-1} \leq f_{k+1}$ and $f_{k+1} < 2^n$

If k+1=3, $2^{3/2-1} = 2^{1/2} \leq 2 = 1 + 1 = f_3$, and
$f_3 = 1 + 1 = 2 < 8 = 2^3$. So, P(3) is true.

Otherwise, note that $f_{k+1} = f_k + f_{k-1}$ by definition.

Taking each inequality separately:

$$f_{k+1} = f_k + f_{k-1} < 2^k + 2^{k-1} \quad \text{(by IH)}$$
$$< 2^k + 2^k \quad (2^{k-1} < 2^k)$$
$$= 2^{k+1}$$

$$f_{k+1} = f_k + f_{k-1} \geq 2^{k/2-1} + 2^{(k-1)/2-1} \quad \text{(by IH)}$$
$$\geq 2^{(k-1)/2-1} + 2^{(k-1)/2-1} \quad \text{(Because } 2^{k/2-1} > 2^{(k-1)/2-1}\text{)}$$
$$= 2(2^{(k-1)/2-1}) \quad \text{(Combining terms)}$$
$$= 2^{2/2+(k-1)/2-1} \quad \text{(Multiplying)}$$
$$= 2^{(k+1)/2-1}$$

So, the claim is true by strong induction.

Running time of Euclid's algorithm

Theorem: **Suppose that Euclid's Algorithm takes n steps for gcd(a,b) with a > b. Then, $a \geq f_{n+1}$.**

We go by strong induction on n.
Let P(n) be "gcd(a,b) with a > b takes n steps $\rightarrow$ $a \geq f_{n+1}$" for all n ≥ 1.

Base Case:
If Euclid's Algorithm on a, b, with a > b, takes 1 step, then it must be the case that b | a.
Note that $f_2 = 1$.
Note that if a were 0, then gcd(0, b), which takes zero steps. So, the smallest possible value for a is 1, which is f_2.
Induction Hypothesis: Suppose P(j) for all integers j s.t. $1 \leq j \leq k$ for some $k \geq 1$.

Induction Step: We want to show if gcd(a,b) takes k+1 steps, then $a \geq f_{k+2}$.
If k = 2, note that a > 1, because gcd(1, b) takes one step. Also, $f_3 = 2$.

Running time of Euclid's algorithm

Theorem: **Suppose that Euclid's Algorithm takes n steps for gcd(a,b) with a > b. Then, $a \geq f_{n+1}$.**

Since the algorithm took k+1 steps, let's give them names:

Say $r_{k+1} = a$ and $r_k = b$, and $r_i = r_{i-1} \bmod r_{i-2}$.

So, $\gcd(a, b) = \gcd(r_{k+1}, r_k)$

$= \gcd(r_k, r_k \bmod r_{k+1}) = \gcd(r_k, r_{k-1})$

$= \gcd(r_{k-1}, r_{k-1} \bmod r_k) = \gcd(r_{k-1}, r_{k-2})$

$= \ldots$

$2^{n-1} \leq f_n \leq 2^n$

Writing these as equations, we have:

$r_{k+1} = q_k r_k + r_{k-1}$

$r_k = q_{k-1} r_{k-1} + r_{k-2}$

$\ldots$

$r_3 = q_2 r_2 + r_1$

$r_2 = q_1 r_1$

Note that $q_i \geq 1$, $r_i \geq 1$.

Note that after one iteration of the algorithm, we're left with $\gcd(r_k, r_{k-1})$ which takes k steps. By the IH, $r_k \geq f_{k+1}$. So,

$r_{k+1} = q_k r_k + r_{k-1}$ (by gcd algorithm)

$\geq q_k f_{k+1} + f_k$ (by IH)

$\geq f_{k+1} + f_k$ ($q_k \geq 1$)

$\geq f_{k+2}$ (definition of f)