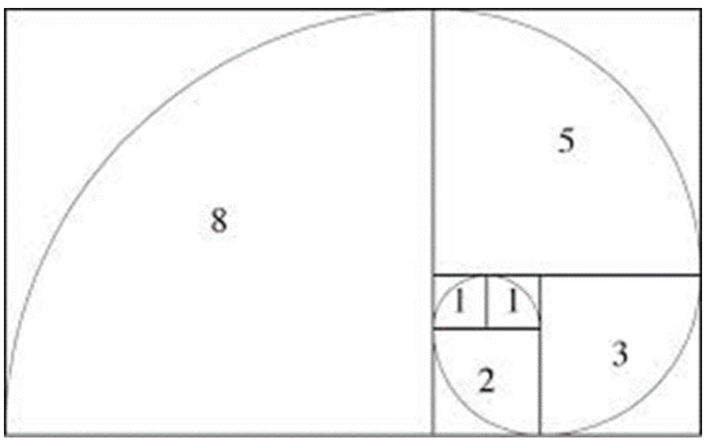
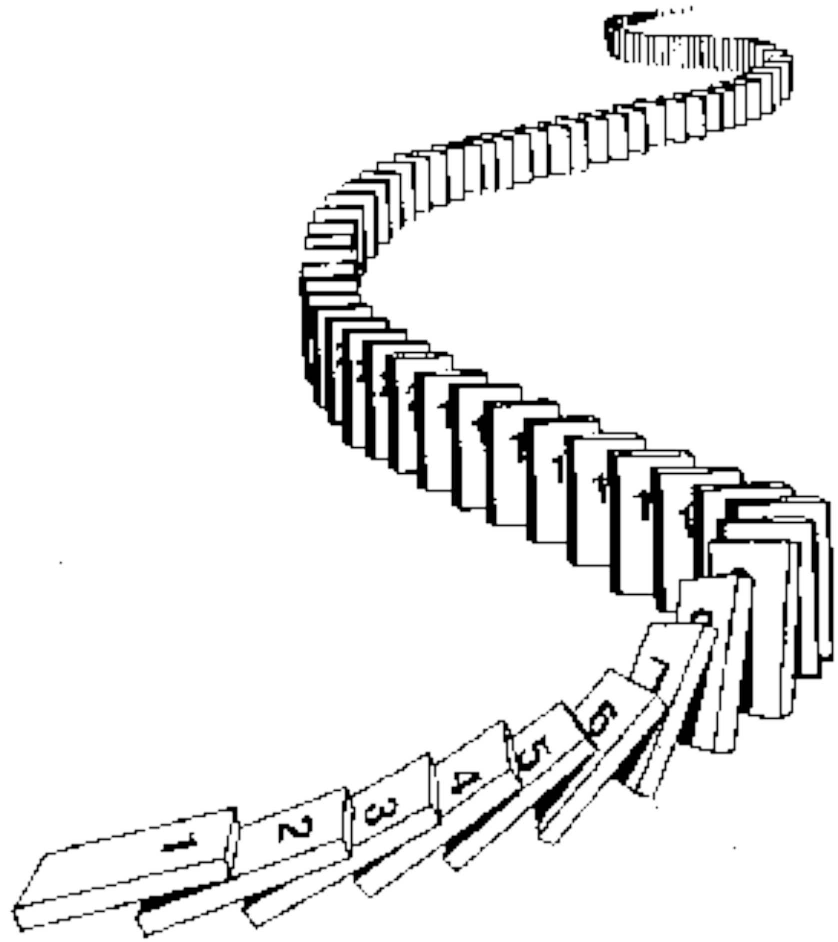
CSE 311: Foundations of Computing

Lecture 16: Recursion & Strong Induction
Applications: Fibonacci & Euclid





Midterm Review

- Review session on Sunday, 3–5pm in Gowen 301
  - TAs will be there
  - come with questions
- Midterm covers material up through (ordinary) induction
- Practice midterm and problems on web site
  - make sure all the concepts we covered are clear
  - more information on exam format coming on Monday

Last time: *Strong* Inductive Proofs In 5 Easy Steps

1. "Let $P(n)$ be... . We will show that $P(n)$ is true for all integers $n \geq b$ by *strong* induction."
2. "Base Case:" Prove $P(b)$
3. "Inductive Hypothesis:

   Assume that for some arbitrary integer $k \geq b$,

   $P(j)$ is true for every integer j from b to k"
4. "Inductive Step:" Prove that $P(k+1)$ is true:

   Use the goal to figure out what you need.

   Make sure you are using I.H. (that $P(b), \ldots, P(k)$ are true) and point out where you are using it.
   (Don't assume $P(k+1)$!!)
5. "Conclusion: $P(n)$ is true for all integers $n \geq b$"

Recall: Fundamental Theorem of Arithmetic

Every integer > 1 has a unique prime factorization

$48 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3$
$591 = 3 \cdot 197$
$45{,}523 = 45{,}523$
$321{,}950 = 2 \cdot 5 \cdot 5 \cdot 47 \cdot 137$
$1{,}234{,}567{,}890 = 2 \cdot 3 \cdot 3 \cdot 5 \cdot 3{,}607 \cdot 3{,}803$

We use strong induction to prove that a factorization into primes exists, but not that it is unique.

Last time: every integer ≥ 2 is a product of primes.

1. **Let** P(n) **be** "n is a product of primes". **We will show that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case** (n=2): 2 **is prime, so it is a product of (one) prime.**
   Therefore P(2) **is true.**
3. **Inductive Hyp: Suppose that for some arbitrary integer** $k \geq 2$,
   P(j) **is true for every integer** j **between** 2 **and** k
4. **Inductive Step:**

   Goal: Show P(k+1); **i.e.** k+1 **is a product of primes**

   Case: k+1 **is prime: Then by definition** k+1 **is a product of primes**

   Case: k+1 **is composite: Then** k+1=ab **for some integers** a **and** b **where** $2 \leq a, b \leq k$. **By our IH,** P(a) **and** P(b) **are true so we have**

   $a = p_1 p_2 \cdots p_r$ **and** $b = q_1 q_2 \cdots q_s$

   for some primes $p_1, p_2, \ldots, p_r, q_1, q_2, \ldots, q_s$.

   Thus, $k+1 = ab = p_1 p_2 \cdots p_r q_1 q_2 \cdots q_s$ **which is a product of primes.**

   Since $k \geq 2$, **one of these cases must happen and so** P(k+1) **is true.**
5. **Thus** P(n) **is true for all integers** $n \geq 2$, **by strong induction.**

Strong Induction is particularly useful when…

…we need to analyze methods that on input k make a recursive call for an input different from $k-1$.

e.g.: Recursive Modular Exponentiation:

- For exponent $k > 0$ it made a recursive call with exponent $\mathrm{j} = k/2$ when k was even or $\mathrm{j} = k - 1$ when k was odd.

Fast Exponentiation

```
public static int FastModExp(int a, int k, int modulus) {

        if (k == 0) {
            return 1;

         } else if ((k % 2) == 0) {
            long temp = FastModExp(a,k/2,modulus);
            return (temp * temp) % modulus;

        } else {
            long temp = FastModExp(a,k-1,modulus);
            return (a * temp) % modulus;
        }

}
```

$$a^{2j} \bmod m = \left(a^j \bmod m\right)^2 \bmod m$$

$$a^{2j+1} \bmod m = \big((a \bmod m) \cdot (a^{2j} \bmod m)\big) \bmod m$$

Strong Induction is particularly useful when…

…we need to analyze methods that on input k make a recursive call for an input different from $k-1$.

e.g.: Recursive Modular Exponentiation:

– For exponent $k > 0$ it made a recursive call with exponent $\mathrm{j} = k/2$ when k was even or $\mathrm{j} = k-1$ when k was odd.

We won't analyze this particular method by strong induction, but we could.

However, we will use strong induction to analyze other functions with recursive definitions.

Recursive definitions of functions

- $F(0) = 0;\ \ F(n+1) = F(n) + 1$ **for all** $n \geq\ 0$**.**

- $G(0) = 1;\ \ G(n+1) = 2 \cdot G(n)$ **for all** $n \geq\ 0$**.**

- $0! = 1;\ \ (n+1)! = (n+1) \cdot n!$ **for all** $n \geq\ 0$**.**

- $H(0) = 1;\ \ H(n+1) = 2^{H(n)}$ **for all** $n \geq\ 0$**.**

Prove $n! \leq n^n$ for all $n \geq 1$

1. **Let** P(n) **be** "$n! \leq n^n$". **We will show that** P(n) **is true for all integers** $n \geq 1$ **by induction.**
2. **Base Case** (n=1): $1!=1\cdot0!=1\cdot1=1=1^1$ **so** P(1) **is true.**
3. **Inductive Hypothesis: Suppose that** P(k) **is true for some arbitrary integer** $k \geq 1$**. I.e., suppose** $k! \leq k^k$.

Prove $n! \leq n^n$ for all $n \geq 1$

1. **Let** P(n) **be** "$n! \leq n^n$". **We will show that** P(n) **is true for all integers** $n \geq 1$ **by induction.**
2. **Base Case** (n=1): $1!=1\cdot 0!=1\cdot 1=1=1^1$ **so** P(1) **is true.**
3. **Inductive Hypothesis: Suppose that** P(k) **is true for some arbitrary integer** $k \geq 1$. **I.e., suppose** $k! \leq k^k$.
4. **Inductive Step:**

   Goal: Show P(k+1), **i.e. show** $(k+1)! \leq (k+1)^{k+1}$

   $$
   \begin{aligned}
   (k+1)! &= (k+1)\cdot k! && \textbf{by definition of !} \\
   &\leq (k+1)\cdot k^k && \textbf{by the IH} \\
   &\leq (k+1)\cdot (k+1)^k && \textbf{since } k \geq 0 \\
   &= (k+1)^{k+1}
   \end{aligned}
   $$

   Therefore P(k+1) **is true.**
5. **Thus** P(n) **is true for all** $n \geq 1$, **by induction.**

More Recursive Definitions

Suppose that $h: \mathbb{N} \to \mathbb{R}$.

Then we have familiar summation notation:

$\sum_{i=0}^{0} h(i) = h(0)$

$\sum_{i=0}^{n+1} h(i) = h(n+1) + \sum_{i=0}^{n} h(i)$ **for** $n \geq 0$

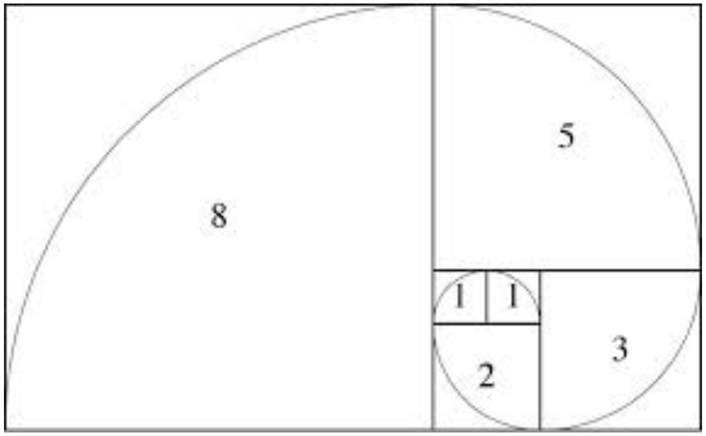
There is also product notation:

$\prod_{i=0}^{0} h(i) = h(0)$

$\prod_{i=0}^{n+1} h(i) = h(n+1) \cdot \prod_{i=0}^{n} h(i)$ **for** $n \geq 0$

Fibonacci Numbers

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



Fibonacci Numbers

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

Tamás Görbe
@TamasGorbe

A Mathematician's Way* of Converting Miles to Kilometers

$$\begin{aligned} 3 \text{ mi} &\approx 5 \text{ km} \\ 5 \text{ mi} &\approx 8 \text{ km} \\ 8 \text{ mi} &\approx 13 \text{ km} \end{aligned}$$

$$f_n \text{ mi} \approx f_{n+1} \text{ km}$$

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be "**$f_n < 2^n$**". We prove that** P(n) **is true for all integers** $n \geq 0$ **by** **strong** **induction.**

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

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be** "$f_n < 2^n$". **We prove that** P(n) **is true for all integers** $n \geq 0$ **by strong induction.**
2. **Base Case:** $f_0 = 0 < 1 = 2^0$ **so** P(0) **is true.**

$\boldsymbol{f_0 = 0} \qquad \boldsymbol{f_1 = 1}$

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

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be** "$f_n < 2^n$". **We prove that** P(n) **is true for all integers** $n \geq 0$ **by strong induction.**
2. **Base Case:** $f_0=0 < 1= 2^0$ **so** P(0) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 0$, **we have** $f_j < 2^j$ **for every integer** j **from** 0 **to** k.

$\boldsymbol{f_0 = 0} \quad \boldsymbol{f_1 = 1}$

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

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be "**$f_n < 2^n$**". We prove that** P(n) **is true for all integers** $n \geq 0$ **by strong induction.**
2. **Base Case:** $f_0 = 0 < 1 = 2^0$ **so** P(0) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 0$**, we have** $f_j < 2^j$ **for every integer** j **from** 0 **to** k**.**
4. **Inductive Step:** **Goal: Show** P(k+1)**; that is,** $f_{k+1} < 2^{k+1}$

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

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be** "$f_n < 2^n$". **We prove that** P(n) **is true for all integers** $n \geq 0$ **by strong induction.**
2. **Base Case:** $f_0=0 < 1= 2^0$ **so** P(0) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 0$, **we have** $f_j < 2^j$ **for every integer** j **from** 0 **to** k.
4. **Inductive Step:** $\boxed{\text{Goal: Show } P(k+1)\text{; that is, } f_{k+1} < 2^{k+1}}$

   Case k+1 = 1:

   Case k+1 ≥ 2:

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

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be "**$f_n < 2^n$**". We prove that** P(n) **is true for all integers** $n \geq 0$ **by strong induction.**
2. **Base Case:** $f_0 = 0 < 1 = 2^0$ **so** P(0) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 0$, **we have** $f_j < 2^j$ **for every integer j from** 0 **to** k.
4. **Inductive Step:** $\boxed{\textbf{Goal: Show } P(k+1)\textbf{; that is, } f_{k+1} < 2^{k+1}}$

   Case $k+1 = 1$: **Then** $f_1 = 1 < 2 = 2^1$ **so** P(k+1) **is true here.**

   Case $k+1 \geq 2$:

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

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be "**$f_n < 2^n$**". We prove that** P(n) **is true for all integers** $n \geq 0$ **by strong induction.**
2. **Base Case:** $f_0 = 0 < 1 = 2^0$ **so** P(0) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 0$**, we have** $f_j < 2^j$ **for every integer** j **from** 0 **to** k**.**
4. **Inductive Step:** $\boxed{\textbf{Goal: Show } P(k+1)\textbf{; that is, } f_{k+1} < 2^{k+1}}$

   <u>Case $k+1 = 1$</u>: **Then** $f_1 = 1 < 2 = 2^1$ **so** P(k+1) **is true here.**

   <u>Case $k+1 \geq 2$</u>: **Then** $f_{k+1} = f_k + f_{k-1}$ **by definition**

   $< 2^k + 2^{k-1}$ **by the IH since** $k-1 \geq 0$

   $< 2^k + 2^k = 2 \cdot 2^k$

   $= 2^{k+1}$

   so P(k+1) **is true in this case.**

These are the only cases so P(k+1) **follows.**

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

Bounding Fibonacci I: $f_n < 2^n$ for all $n \geq 0$

1. **Let** P(n) **be "**$f_n < 2^n$**". We prove that** P(n) **is true for all integers** $n \geq 0$ **by strong induction.**
2. **Base Case:** $f_0=0 < 1= 2^0$ **so** P(0) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 0$, **we have** $f_j < 2^j$ **for every integer** j **from** 0 **to** k**.**
4. **Inductive Step:** $\boxed{\textbf{Goal: Show } P(k+1)\textbf{; that is, } f_{k+1} < 2^{k+1}}$

   <u>Case k+1 = 1</u>: **Then** $f_1 = 1 < 2 = 2^1$ **so** P(k+1) **is true here.**

   <u>Case k+1 ≥ 2</u>: **Then** $f_{k+1} = f_k + f_{k-1}$ **by definition**

   $< 2^k + 2^{k-1}$ **by the IH since** $k-1 \geq 0$

   $< 2^k + 2^k = 2 \cdot 2^k = 2^{k+1}$

   so P(k+1) **is true in this case.**

   These are the only cases so P(k+1) **follows.**
5. **Therefore by strong induction,**

   $f_n < 2^n$ **for all integers** $n \geq 0$.

$\boxed{\begin{array}{l} f_0 = 0 \quad f_1 = 1 \\ f_n = f_{n-1} + f_{n-2} \text{ for all } n \geq 2 \end{array}}$

Bounding Fibonacci II: $f_n \geq 2^{n/2-1}$ for all $n \geq 2$

1. **Let** P(n) **be "**$f_n \geq 2^{n/2 -1}$**". We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**

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

Bounding Fibonacci II: $f_n \geq 2^{n/2-1}$ for all $n \geq 2$

1. **Let** P(n) **be** "$f_n \geq 2^{n/2-1}$". **We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case:** $f_2 = f_1 + f_0 = 1$ and $2^{2/2-1} = 2^0 = 1$ **so** P(2) **is true.**

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

Bounding Fibonacci II: $f_n \geq 2^{n/2 - 1}$ for all $n \geq 2$

1. **Let** P(n) **be "**$f_n \geq 2^{n/2 - 1}$**". We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case:** $f_2 = f_1 + f_0 = 1$ and $2^{2/2 - 1} = 2^0 = 1$ **so** P(2) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 2$, P(j) **is true for every integer** j **from** 2 **to** k.

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

Bounding Fibonacci II: $f_n \geq 2^{n/2-1}$ for all $n \geq 2$

1. **Let** P(n) **be "**$f_n \geq 2^{n/2-1}$**". We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case:** $f_2 = f_1 + f_0 = 1$ and $2^{2/2-1} = 2^0 = 1$ **so** P(2) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 2$, P(j) **is true for every integer** j **from** 2 **to** k**.**
4. **Inductive Step:** **Goal: Show** P(k+1)**; that is,** $f_{k+1} \geq 2^{(k+1)/2-1}$

$\boldsymbol{f_0 = 0} \quad \boldsymbol{f_1 = 1}$

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

Bounding Fibonacci II: $f_n \geq 2^{n/2-1}$ for all $n \geq 2$

1. **Let** P(n) **be "**$f_n \geq 2^{n/2 - 1}$**". We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case:** $f_2 = f_1 + f_0 = 1$ and $2^{2/2-1} = 2^0 = 1$ **so** P(2) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 2$, P(j) **is true for every integer** j **from** 2 **to** k.
4. **Inductive Step:** $\boxed{\textbf{Goal: Show } P(k+1)\textbf{; that is, } f_{k+1} \geq 2^{(k+1)/2-1}}$

No need for cases for the definition here:

$f_{k+1} = f_k + f_{k-1}$ **since** $k+1 \geq 2$

Now just want to apply the IH to get P(k) **and** P(k-1)

Problem: Though we can get P(k) **since** $k \geq 2$,

k-1 **may only be** 1 **so we can't conclude** P(k-1)

Solution: Separate cases for when k-1=1 **(or** k+1=3**).**

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

Bounding Fibonacci II: $f_n \geq 2^{n/2-1}$ for all $n \geq 2$

1. **Let** P(n) **be "**$f_n \geq 2^{n/2 -1}$**".   We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case:** $f_2 = f_1 + f_0 = 1$ and $2^{2/2-1} = 2^0 = 1$ **so** P(2) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 2$, P(j) **is true for every integer** j **from** 2 **to** k**.**
4. **Inductive Step:** **Goal: Show** P(k+1); **that is,** $f_{k+1} \geq 2^{(k+1)/2 -1}$

   Case $k = 2$:

   Case $k \geq 3$:

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

Bounding Fibonacci II: $f_n \geq 2^{n/2-1}$ for all $n \geq 2$

1. **Let** P(n) **be "**$f_n \geq 2^{n/2-1}$**". We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case:** $f_2 = f_1 + f_0 = 1$ and $2^{2/2-1} = 2^0 = 1$ **so** P(2) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 2$, P(j) **is true for every integer** j **from** 2 **to** k**.**
4. **Inductive Step:** $\boxed{\textbf{Goal: Show } P(k+1)\textbf{; that is, } f_{k+1} \geq 2^{(k+1)/2-1}}$

   <u>Case k = 2</u>: **Then** $f_{k+1} = f_3 = f_2 + f_1 = 2 \geq 2^{1/2} = 2^{3/2-1} = 2^{(k+1)/2-1}$

   <u>Case k ≥ 3</u>:

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

Bounding Fibonacci II: $f_n \geq 2^{n/2-1}$ for all $n \geq 2$

1. **Let** P(n) **be "**$f_n \geq 2^{n/2-1}$**".** **We prove that** P(n) **is true for all integers** $n \geq 2$ **by strong induction.**
2. **Base Case:** $f_2 = f_1 + f_0 = 1$ and $2^{2/2-1} = 2^0 = 1$ **so** P(2) **is true.**
3. **Inductive Hypothesis: Assume that for some arbitrary integer** $k \geq 2$, P(j) **is true for every integer** j **from** 2 **to** k**.**
4. **Inductive Step:** $\boxed{\textbf{Goal: Show } P(k+1)\textbf{; that is, } f_{k+1} \geq 2^{(k+1)/2-1}}$

   Case $k = 2$: **Then** $f_{k+1} = f_3 = f_2 + f_1 = 2 \geq 2^{1/2} = 2^{3/2-1} = 2^{(k+1)/2-1}$

   Case $k \geq 3$: $f_{k+1} = f_k + f_{k-1}$ **by definition**

   $\geq 2^{k/2-1} + 2^{(k-1)/2-1}$ **by the IH since** $k-1 \geq 2$

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

   So P(k+1) **is true in both cases.**
5. **Therefore by strong induction,** $f_n \geq 2^{n/2-1}$ **for all integers** $n \geq 0$.

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

Running time of Euclid's algorithm

Theorem: Suppose that Euclid's Algorithm takes n steps for $\gcd(a, b)$ with $a \geq b > 0$. Then, $a \geq f_{n+1}$.

Running time of Euclid's algorithm

Theorem: Suppose that Euclid's Algorithm takes n steps for $\gcd(a, b)$ with $a \geq b > 0$. Then, $a \geq f_{n+1}$.

Why does this help us bound the running time of Euclid's Algorithm?

We already proved that $f_n \geq 2^{n/2 - 1}$ so $f_{n+1} \geq 2^{(n-1)/2}$

Therefore: if Euclid's Algorithm takes n steps
for $\gcd(a, b)$ with $a \geq b > 0$
then $a \geq 2^{(n-1)/2}$

so $(n-1)/2 \leq \log_2 a$ or $n \leq 1 + 2\log_2 a$
i.e., # of steps ≤ 1 + twice the # of bits in a.

Running time of Euclid's algorithm

Theorem: Suppose that Euclid's Algorithm takes n steps for $\gcd(a, b)$ with $a \geq b > 0$. Then, $a \geq f_{n+1}$.

An informal way to get the idea: Consider an n step gcd calculation starting with $r_{n+1}=a$ and $r_n=b$:

$$
\begin{aligned}
r_{n+1} &= q_n r_n + r_{n-1} \\
r_n &= q_{n-1} r_{n-1} + r_{n-2} \\
&\dots \\
r_3 &= q_2 r_2 + r_1 \\
r_2 &= q_1 r_1
\end{aligned}
$$

For all $k \geq 2$, $r_{k-1} = r_{k+1} \bmod r_k$

Now $r_1 \geq 1$ and each q_k must be ≥ 1. If we replace all the q_K's by 1 and replace r_1 by 1, we can only reduce the r_k's. After that reduction, $r_k=f_k$ for every k.

Running time of Euclid's algorithm

Theorem: Suppose that Euclid's Algorithm takes n steps for $\gcd(a, b)$ with $a \geq b > 0$. Then, $a \geq f_{n+1}$.

We go by strong induction on n.

Let P(n) **be** "gcd(a,b) **with** $a \geq b > 0$ **takes** n **steps** $\longrightarrow a \geq f_{n+1}$" **for all** $n \geq 1$.

<u>**Base Case**</u>: n=1 **Suppose Euclid's Algorithm with** $a \geq b > 0$ **takes** 1 **step.**

By assumption, $a \geq b \geq 1 = f_2$ **so** P(1) **holds.**

<u>**Induction Hypothesis**</u>: **Suppose that for some integer** $k \geq 1$, P(j) **is true for all integers** j **s.t.** $1 \leq j \leq k$

Running time of Euclid's algorithm

Theorem: **Suppose that Euclid's Algorithm takes n steps for $\gcd(a, b)$ with $a \geq b > 0$. Then, $a \geq f_{n+1}$.**

We go by strong induction on n.

Let P(n) **be** "gcd(a,b) **with** $a \geq b > 0$ **takes** n **steps** $\rightarrow a \geq f_{n+1}$" **for all** $n \geq 1$.

<u>**Base Case**</u>: n=1 **Suppose Euclid's Algorithm with** $a \geq b > 0$ **takes** 1 **step.**
By assumption, $a \geq b \geq 1 = f_2$ **so** P(1) **holds.**

<u>**Induction Hypothesis**</u>: **Suppose that for some integer** $k \geq 1$, P(j) **is true for all integers** j **s.t.** $1 \leq j \leq k$

<u>**Inductive Step**</u>: **We want to show:** **if** gcd(a,b) **with** $a \geq b > 0$ **takes** k+1 **steps, then** $a \geq f_{k+2}$.

Running time of Euclid's algorithm

Induction Hypothesis: **Suppose that for some integer** $k \geq 1$, $P(j)$ **is true for all integers** j **s.t.** $1 \leq j \leq k$

Inductive Step: **Goal: if** $\gcd(a,b)$ **with** $a \geq b>0$ **takes** $k+1$ **steps, then** $a \geq f_{k+2}.$

Now if $k+1=2$**, then Euclid's algorithm on** a **and** b **can be written as**

$$a = q_2 b + r_1$$
$$b = q_1 r_1$$

and $r_1 > 0$.

Also, since $a \geq b > 0$ **we must have** $q_2 \geq 1$ and $b \geq 1$.

So $a = q_2 b + r_1 \geq b + r_1 \geq 1+1 = 2 = f_3 = f_{k+2}$ as required.

Running time of Euclid's algorithm

Induction Hypothesis: Suppose that for some integer $k \geq 1$, $P(j)$ is true for all integers j s.t. $1 \leq j \leq k$

Inductive Step: Goal: if $\gcd(a,b)$ with $a \geq b>0$ takes $k+1$ steps, then $a \geq f_{k+2}$.

Next suppose that $k+1 \geq 3$ so for the first 3 steps of Euclid's algorithm on a and b we have

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

and there are $k-2$ more steps after this.

Running time of Euclid's algorithm

Induction Hypothesis: Suppose that for some integer $k \geq 1$, $P(j)$ is true for all integers j s.t. $1 \leq j \leq k$

Inductive Step: $\boxed{\text{Goal: if } \gcd(a,b) \text{ with } a \geq b > 0 \text{ takes } k+1 \text{ steps, then } a \geq f_{k+2}.}$

Next suppose that $k+1 \geq 3$ so for the first 3 steps of Euclid's algorithm on a and b we have

$$a = q_{k+1}b + r_k$$
$$b = q_k \; r_k + r_{k-1}$$
$$r_k = q_{k-1}r_{k-1} + r_{k-2}$$

and there are $k-2$ more steps after this. Note that this means that the $\gcd(b, r_k)$ takes k steps and $\gcd(r_k, r_{k-1})$ takes $k-1$ steps.

So since k, $k-1 \geq 1$ by the IH we have $b \geq f_{k+1}$ and $r_k \geq f_k$.

Running time of Euclid's algorithm

Induction Hypothesis: Suppose that for some integer $k \geq 1$, $P(j)$ is true for all integers j s.t. $1 \leq j \leq k$

Inductive Step: Goal: if $\gcd(a,b)$ with $a \geq b > 0$ takes $k+1$ steps, then $a \geq f_{k+2}$.

Next suppose that $k+1 \geq 3$ so for the first 3 steps of Euclid's algorithm on a and b we have

$$a = q_{k+1} b + r_k$$
$$b = q_k \, r_k + r_{k-1}$$
$$r_k = q_{k-1} r_{k-1} + r_{k-2}$$

and there are $k-2$ more steps after this. Note that this means that the $\gcd(b, r_k)$ takes k steps and $\gcd(r_k, r_{k-1})$ takes $k-1$ steps.

So since $k, k-1 \geq 1$ by the IH we have $b \geq f_{k+1}$ and $r_k \geq f_k$.

Also, since $a \geq b$ we must have $q_{k+1} \geq 1$.

So $a = q_{k+1} b + r_k \geq b + r_k \geq f_{k+1} + f_k = f_{k+2}$ as required. ■