Proof by induction (recap)
COMS20010 (Algorithms II)

John Lapinskas, University of Bristol

What is induction?

Induction is like **proof by programming**.

Example: Let's prove by induction that for all n and all $x \neq 1$,

$$\sum_{i=0}^{n} x^i = \frac{x^{n+1} - 1}{x - 1}.$$

Write $S(n)$ for the statement that $\sum_{i=0}^{n} x^i = \frac{x^{n+1}-1}{x-1}$ for all $x \neq 1$.
So for example, $S(0)$ says that $x^0 = \frac{x^{0+1}-1}{x-1} = 1$ for all x, which is true.

Rather than proving $S(n)$ for all n, we write an algorithm which, given n as an input, outputs a proof of $S(n)$. So since we can prove $S(n)$ for all n, it must hold for all n!

Let $S(n)$ be the statement that $\sum_{i=0}^{n} x^i = \frac{x^{n+1}-1}{x-1}$ for all $x \neq 1$.
We want to prove by induction that $S(n)$ holds for all n.

The "program" has two subroutines:

- `BaseCase()` outputs a proof of $S(0)$;
- `InductiveStep`(k) outputs a proof of $S(k+1)$ from $S(0), \ldots, S(k)$.

And the pseudocode reads:

Input: An integer $n \geq 0$.
Output: A proof of $S(n)$.

begin
  Output `BaseCase()`.
  foreach k *in* $\{0, \ldots, n-1\}$ **do**
    Output `InductiveStep`(k).
  Halt.

But this program is the same for every induction proof, so we only have to specify `BaseCase` and `InductiveStep`.

Let $S(n)$ be the statement that $\sum_{i=0}^{n} x^i = \frac{x^{n+1}-1}{x-1}$ for all $x \neq 1$.
We want to prove by induction that $S(n)$ holds for all n.

Or to put it another way, we need to do two things.

Base case: Prove $S(0)$.
We already did this: $x^0 = \frac{x^{0+1}-1}{x-1} = 1$. ✓

Inductive step: Assuming that $S(0), \ldots, S(k)$ are all true (a.k.a. the **induction hypothesis**), prove $S(k+1)$.

We have

$$\sum_{i=0}^{k+1} x^i = \sum_{i=0}^{k} x^i + x^{k+1} = \frac{x^{k+1}-1}{x-1} + x^{k+1} \qquad \text{[Induction hypothesis]}$$

$$= \frac{x^{k+1}-1+x^{k+1}(x-1)}{x-1} = \frac{x^{k+2}-1}{x-1}.$$ ✓

And so we're done! □

Strong induction

Let $S(n)$ be the statement that $\sum_{i=0}^{n} x^i = \frac{x^{n+1}-1}{x-1}$ for all $x \neq 1$.
We want to prove by induction that $S(n)$ holds for all n.

There we used $S(k)$ to prove $S(k+1)$ in the inductive step.
But we could have used **any** of $S(0), \ldots, S(k)$.

For example, this proof of the inductive step is also valid:

$$\sum_{i=0}^{k+1} x^i = \sum_{i=0}^{\lfloor k/2 \rfloor} x^i + \sum_{i=\lfloor k/2 \rfloor+1}^{k+1} x^i = \sum_{i=0}^{\lfloor k/2 \rfloor} x^i + x^{\lfloor k/2 \rfloor+1} \cdot \sum_{i=0}^{k-\lfloor k/2 \rfloor} x^i$$

$$= \frac{x^{\lfloor k/2 \rfloor+1}-1}{x-1} + x^{\lfloor k/2 \rfloor+1} \cdot \frac{x^{k-\lfloor k/2 \rfloor+1}-1}{x-1} \quad \text{[Inductive hypothesis]}$$

$$= \frac{x^{k+2}-1}{x-1}. \qquad \checkmark$$

This technique is sometimes called **strong induction**.

Another example

Consider the following (awful) pseudocode for a function $\texttt{Increment}(y)$:

Input: An integer $y > 0$.
Output: $y + 1$.

```
begin
    if y = 0 then
        Return 1.
    else if y (mod 2) = 0 then
        Return y + 1.
    else
        Return 2 · Increment(⌊y/2⌋).
```

Does it work? Technically, yes! We can prove this by induction on y.

Base case: If $y = 0$, we return $1 = y + 1$. ✓

Another example

Consider the following (awful) pseudocode for a function $\texttt{Increment}(y)$:

Input: An integer $y > 0$.
Output: $y + 1$.

begin
  if $y = 0$ **then**
    Return 1.
  else if $y \pmod 2 = 0$ **then**
    Return $y + 1$.
  else
    Return $2 \cdot \texttt{Increment}(\lfloor y/2 \rfloor)$.

Inductive step: If y is even, we return $y + 1$. ✓

Otherwise, by induction, we return $2(\lfloor y/2 \rfloor + 1)$.

Writing $y = 2z + 1$, we have $\lfloor y/2 \rfloor = z$, so this is $2(z + 1) = y + 1$. $\square$

Other induction schemes

Say you're proving a statement $S(m, n)$ for all m **and** n. Then you could think of an "induction proof program" `Proof`(m, n) using subroutines:

- `BaseCase`() outputs a proof of $S(m, 0)$ for all m and $S(0, n)$ for all n;
- `InductiveStep`(m, n) outputs a proof of $S(m, n)$ from $S(m-1, n-1)$ for all $m, n \geq 1$.

Input: Integers $m, n \geq 0$.
Output: A proof of $S(m, n)$.

begin
    if $m = 0$ *or* $n = 0$ **then**
        Output `BaseCase`().
    else
        Output `Proof`$(m-1, n-1)$.
        Output `InductiveStep`(m, n).
    Halt.

Other induction schemes

This corresponds to an induction proof of:

- **Base case:** Prove $S(m, 0)$ for all m and $S(0, n)$ for all n;
- **Inductive step:** Prove $S(m, n)$ from $S(m-1, n-1)$ for all $m, n \geq 1$.

This is valid! There are lots of other induction schemes, e.g.:

- The base case is $S(0, 0)$, the inductive step proves $S(m, n)$ from:
  - $S(m-1, n)$ and $S(m, n-1)$ when $m, n \geq 1$;
  - $S(m, n-1)$ when $m = 0$ and $n \geq 1$;
  - $S(m-1, n)$ when $m \geq 1$ and $n = 0$.
- For a one-variable statement $S(n)$: The base cases are $S(0)$ and $S(1)$, and the inductive step proves $S(n)$ from $S(n-2)$ for all $n \geq 2$.

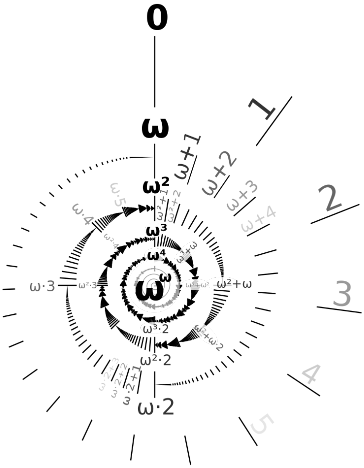
Other induction schemes

Without getting into foundational stuff: if you can put your induction proof in the form of an "induction program" that will output a (finite!) proof for any parameter choice, then you have a valid proof by induction.

This is useful in dealing with functions on multiple variables, or complicated structures that can be broken down into simpler parts.

But beware subtle errors! (See the quiz...)

Non-examinable: transfinite induction



Just say no.