Lec 08: Induction II

Prof. Adam J. Aviv

GW

CSCI 1311 Discrete Structures I
Spring 2020

Weak Induction

To prove a proposition $P(n)$ holds for $\forall n \geq a$ by induction, we must:

Base Case: $P(a)$ is true

Inductive Step: $(\forall k \geq a)(P(k) \implies P(k+1))$ is true

The assumption that $P(k)$ is true is our Inductive Hypothesis, and generally, to prove $P(k+1)$ we only need to assume that the prior result holds, namely $P(k)$.

Assuming only the prior result holds, $P(k)$, in proving $P(k+1)$ is called weak induction.

Strong Induction

In strong induction, we allow for a stronger induction hypothesis. To prove that $P(n)$ is true for all $n \geq a$

Base Case: $P(a)$ is true.

Inductive Step:
$(\forall k \geq a)[P(a) \wedge P(a+1) \wedge \ldots \wedge P(k-1) \wedge P(k) \implies P(k+1)]$ is true.

We get to assume the induction hypothesis is not only true for $P(k)$, but also true for $P(a)$ *and* $P(a+1)$ *and* ... $P(k)$, all values between a and k.

What makes this so strong?

The inductive step requires prove the following implication.

$$P(a) \wedge P(a+1) \wedge \ldots P(k-1) \wedge P(k) \implies P(k+1)$$

But, recall that $p \rightarrow q \equiv \neg p \vee q$, so if we have

$$\begin{aligned} p_1 \wedge p_2 \rightarrow q &\equiv \neg(p_1 \wedge p_2) \vee q \\ &\equiv \neg p_1 \vee \neg p_2 \vee q \\ &\equiv \neg p_1 \vee \neg p_2 \vee q \vee q \\ &\equiv (\neg p_1 \vee q) \vee (\neg p_2 \vee q) \\ &\equiv (p_1 \rightarrow q) \vee (\neg p_2 \rightarrow q) \end{aligned}$$

Or to put it another way, to prove the larger implication, we only need to show that **any** $P(j)$ where $a \leq j \leq k$ implies $P(k+1)$.

Using strong induction

Let's revisit the following theorem we proved by contradiction, and instead prove it with strong induction.

Theorem (Prime Divisibility)

For all integers $n > 1$, n is divisible by a prime.

What is the base case?

What is the inductive step and inductive hypothesis?

Prime divisibility proof

Proof of prime divisibility

By strong induction on n, we can show:

Base Case: 2 is divisible by a prime, namely $2|2$.

Inductive Step: If we assume that all integers $k \leq n$ are divisible by a prime, we can show that $n+1$ is divisible by a prime. There are two cases to consider: $n+1$ is prime and $n+1$ is composite.

Prime divisibility proof

Proof of prime divisibility (cont.)

Case $(n+1)$ is prime: If $n+1$ is prime, then $n+1$ divides itself, and is thus divisible by a prime.

Case $(n+1)$ is composite: If $n+1$ is composite, then $n+1 = ab$ for some integers a and b where $1 < a < n+1$ and $1 < b < n+1$. By the inductive hypothesis a and b must be divisible by a prime because $a \leq n$ and $b \leq n$.

Consider a (but the same is true for b). By the IH, there exists a prime p such that $p|a$ and the case assumes that $a|(n+1)$. By transitivity of divisibility, if $p|a$ and $a|(n+1)$ then $p|(n+1)$, proving this case.

Thus, every integer $n > 1$ is divisible by a prime. □

Existence of a prime factorization

Recall that the *fundamental theorem of arithmetic* says that all numbers can be factored into a unique set of primes. There are two parts of the proof, existence and uniqueness. Existence can be proven using strong induction.

Theorem (Existence of a prime factorization)

For all integers $n > 1$, there exists a k and primes $p_1 < p_2 < \ldots < p_k$ such that $n = p_1 p_2 \ldots p_{k-1} p_k$.

What is the base case?

What do we need to show in the inductive step, and what is the inductive hypothesis?

Existence of prime factorization proof

Proof of existence of prime factorization.

Proof by strong induction on n.

Base Case: $n = 2$, the prime factorization is simply 2 as 2 is prime.

Inductive Step: Assume that for all $m \leq n$ there exists a prime factorization, we must show that $n+1$ has a prime factorization. There are two cases, $n+1$ is prime or $n+1$ is composite.

- If $n+1$ is prime, than the prime factorization is simply $n+1$
- If $n+1$ is composite, than there exists integers r and s such that $1 < r < (n+1)$ and $1 < s < (n+1)$ and $rs = n+1$. Both r and s are less than n, so by the IH, we know that there exists prime factorization for both, namely that $r = p_1 \dots p_k$ and $s = q_1 \dots q'_k$. The prime factorization for $n+1$ is then $p_1 \dots p_k q_1 \dots q_{k'}$.

□

Exercise

Proof the following using (strong) induction

For all integers $n > 0$, there exists a $k \geq 0$ and odd integer ℓ, such that $n = \ell \cdot 2^k$

Hint: Start by applying induction on n, and consider even and odd cases for $n+1$. You may not need the inductive hypothesis in both cases.

Thinking differently about induction

Induction is not just about numbers

Induction can be applied to any proposition for which you can work from a base case in some well-ordered sequence.

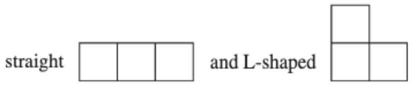
$$P(0) \to P(1) \to \dots \to P(k-1) \to P(k) \to \dots P(n) \to P(n+1) \dots$$

The numbers really mean that we have an obvious path through set of prepositions base case ($P(0)$) through the n'th case ($P(n)$) and beyond.

We can perform induction on other kinds of sequences of objects that have this property.

Trominos

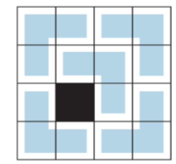
Trominos are objects that can be drawn with 3 squares. There are exacttly two types of trominos, straight and L-shaped.



Trominos is an example of a polyomino, a generalization of domino, introduce by Solomn Golomb in 1954. His work (and others) led to things like Tetris.

Tiling with trominos

You can't fully tile a square grid with a tromino.



But you can tile a square grid of trominos if the dimensions of the grid is a power of 2, e.g., 2x2, 4x4,. . . , and you remove exactly one grid-square.

And you can prove it using induction!

Induction on Tiling Trominos

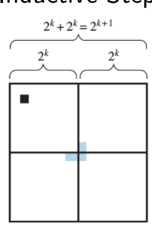
Theorem

For any integer $n \geq 1$, if one square is removed from a $2^n \times 2^n$ grid of squares, then it can be tiled by L-shaped trominos.

Base Case

$n = 1$ is a 2x2 grid.

Inductive Step



IH: 2^k can be tiled
Show: 2^{k+1} can be tiled

Induction on the Structure of Lists

Consider a list L of elements h, we can define a list recursively. It can take exactly two forms: A List is . . .

is empty:

$L = \emptyset$

is a (head) element h appended to a (tail) list T :

$L = h * T$

Example

The list containing the numbers 1-4, can be described as

$$1 * 2 * 3 * 4 * \emptyset$$

The size of the list

Let's define the $|L|$ as the number of elements of the list, where the $|\emptyset|$=0.

Theorem

For all lists L, if $L = \emptyset$, then $|L| = 0$, but if $L = h * T$, then $|h * T| = 1 + |T|$.

This can be proven on the inductive structure of lists. Namely that lists are either empty, $\emptyset$, or they are a head element appended to a tail list, $L = h * T$

Induction on Structure

Proof.

If $L = \emptyset$, then $|L| = 0$ because L has no elements. The remainder of the theorem we can proof by induction on the structure of lists.

Base Case: $L = h * \emptyset$, then $|L| = 1 + |\emptyset|$. The size of the $\emptyset$ is 0. So $|L| = 1$, proving the case since there is 1 element in the list.

Inductive Step: Assume that if $L = h * T$ then $|h * T| = 1 + |T|$, can we show that if $M = a * b * S$ then $|a * b * S| = 1 + |b * S|$.

Let $U = b * S$, then $M = a * U$, and we can now apply the IH with to M, providing us with $|b * S| = 1 + |U|$.

Substituting back in for $U = b * S$, we have $|a * b * S| = 1 + |b * S|$, proving our result. □

Proofs and programming

Proving the theorem is also showing that this program functions properly:

```
def size(L):
  if L is null:
     return 0
  else:
    return 1 + tail(L)
```