CS311H: Discrete Mathematics

Structural Induction

Instructor: Işıl Dillig

Structural Induction

- Last time, we talked about recursively defined structures like sets and strings
- Stuctural induction is a technique that allows us to apply induction on recursive definitions even if there is no integer
- Structural induction is also no more powerful than regular induction, but can make proofs much easier

Structural Induction Overview

- Suppose we have:
  - a recursively defined structure S
  - a property P we'd like to prove about S
- Structural induction works as follows:
  1. Base case: Prove P about base case in recursive definition
  2. Inductive step: Assuming P holds for sub-structures used in the recursive step of the definition, show that P holds for the recursively constructed structure.

Example 1

- Consider the following recursively defined set S:
  1. $a \in S$
  2. If $x \in S$, then $(x) \in S$
- Prove by structural induction that every element in S contains an equal number of right and left parantheses.
- Base case: a has 0 left and 0 right parantheses
- Inductive step: By the inductive hypothesis, x has equal number, say n, of right and left parantheses.
- Thus, (x) has $n+1$ left and $n+1$ right parantheses.

Example 2

- Consider the set S defined recursively as follows:
  - Base case: $3 \in S$
  - Recursive step: If $x \in S$ and $y \in S$, then $x + y \in S$
- Prove S is set of all positive integers that are multiples of 3
- Let A be the set of all positive integers divisible by 3
- We want to show that $A = S$
- To do this, we need to prove $S \subseteq A$ and $A \subseteq S$

Proof, Part I

Consider the set S defined recursively as follows: $3 \in S$ and if $x \in S$ and $y \in S$, then $x + y \in S$

- Let's first prove $S \subseteq A$, i.e., any element in S is divisible by 3
- Base case:
- Inductive step:

Proof, Part II

- Next, need to show S includes all positive multiples of 3
- Therefore, need to prove that $3n \in S$ for all $n \geq 1$
- We'll prove this by induction on n:
  - Base case (n=1):
  - Inductive hypothesis:
  - Need to show:

Proving Correctness of Reverse

- Earlier, we defined a $\text{reverse}(w)$ function for length of strings:
  - Base case: $\text{reverse}(\epsilon) = \epsilon$
  - Recursive step: $\text{reverse}(wa) = a \cdot \text{reverse}(w)$ where $w \in \Sigma^*$ and $a \in \Sigma$
- Prove $\forall y, x \in \Sigma^*.\ \text{reverse}(xy) = \text{reverse}(y) \cdot \text{reverse}(x)$
- Let $P(y)$ be the property

$$\forall x \in \Sigma^*.\ \text{reverse}(xy) = \text{reverse}(y) \cdot \text{reverse}(x)$$

- We'll prove by structural induction that $\forall y \in \Sigma^*.\ P(y)$ holds

Proof of Correctness of Reverse, cont.

$$P(y):\ \forall x \in \Sigma^*.\ \text{reverse}(xy) = \text{reverse}(y) \cdot \text{reverse}(x)$$

- Base case: $y = \epsilon$
- Need to show: reverse$(x \cdot \epsilon)$ = reverse(ϵ) · reverse(x)
- What is reverse$(x \cdot \epsilon)$? reverse(x)
- What is reverse(ϵ) · reverse(x)? reverse(x)
- Thus, $P(y)$ holds for base case

Proof of Correctness of Reverse, cont.

$$P(y):\ \forall x \in \Sigma^*.\ \text{reverse}(xy) = \text{reverse}(y) \cdot \text{reverse}(x)$$

- Inductive step: $y = za$ where $z \in \Sigma^*$ and $a \in \Sigma$
- Want to show: reverse(xza) = reverse(za) · reverse(x)
- $\text{reverse}(xza) = a \cdot \text{reverse}(xz)$
- By the inductive hypothesis, $\text{reverse}(xz) = \text{reverse}(z) \cdot \text{reverse}(x)$
- Thus, $a \cdot \text{reverse}(xz) = a \cdot \text{reverse}(z) \cdot \text{reverse}(x)$
- By definition, $a \cdot \text{reverse}(z) = \text{reverse}(za)$
- Hence, $\text{reverse}(xza) = \text{reverse}(za) \cdot \text{reverse}(x)$ $\square$

One More Reverse Example

- Prove that $\text{reverse}(\text{reverse}(s)) = s$
- We'll prove this by structural induction
- But need previous lemma for the proof to go through!

One More Reverse Example, cont.

Properties of Length

- Prove the following property about the length function:

$$\forall y, x \in \Sigma^*.\ len(xy) = len(x) + len(y)$$

Structural vs. Strong Induction

- Structural induction may look different from other forms of induction, but it is an implicit form of strong induction
- Intuition: We can define an integer k that represents how many times we need to use the recursive step in the definition
- For base case, $k = 0$; if we use recursive step once, $k = 1$ etc.
- In inductive step, assume $P(i)$ for $0 \leq i \leq k$ and prove $P(k+1)$
- Hence, structural induction is just strong induction, but you don't have to make this argument in every proof!

General Induction and Well-Ordered Sets

- Inductive proofs can be used for any well-ordered set
- A set S is well-ordered iff:
  1. Can define a total order $\preceq$ between elements of S ($a \preceq b$ or $b \preceq a$, and $\preceq$ is reflexive, symmetric, and transitive)
  2. Every subset of S has a least element according to this total order
- Example: $(\mathbb{Z}^+, \leq)$ is well-ordered set with least element 1
- What is a total order $\preceq$ such that $(\mathbb{Z}^-, \preceq)$ is a well-ordered set with least element -1?

Generalized Induction

- Can use induction to prove properties of any well-ordered set:
  - Base case: Prove property about least element in set
  - Inductive step: To prove $P(e)$, assume $P(e')$ for all $e' \prec e$
- Mathematical induction is just a special case of this

Ordered Pairs of Natural Numbers

- Consider the set $\mathbb{N} \times \mathbb{N}$, pairs of non-negative integers
- Let's define the following order $\preceq$ on this set:

$$(x_1, y_1) \preceq (x_2, y_2) \ \text{ if } \ \begin{cases} x_1 < x_2 \\ \text{or } x_1 = x_2 \wedge y_1 \leq y_2 \end{cases}$$

- This is an example of lexicographic order, which is a kind of total order
- Therefore, $(\mathbb{N} \times \mathbb{N}, \preceq)$ is a well-ordered set
- Question: What is the least element of this set?

Generalized Induction Example

- Suppose that $a_{m,n}$ is defined recursively for $(m, n) \in \mathbb{N} \times \mathbb{N}$:

$$\begin{array}{lll} a_{0,0} & = & 0 \\ a_{m,n} & = & \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0 \text{ and } m > 0 \\ a_{m,n-1} + n & \text{if } n > 0 \end{cases} \end{array}$$

- Show that $a_{m,n} = m + n(n+1)/2$
- Proof is by induction on (m, n) where $(m, n) \in (\mathbb{N} \times \mathbb{N}, \preceq)$
- Base case:
- By recursive definition, $a_{0,0} = 0$
- $0 + 0 \cdot 1/2 = 0$; thus, base case holds.

Inductive Step

Show $a_{m,n} = m + n(n+1)/2$ for:

$$\begin{aligned} a_{0,0} &= 0 \\ a_{m,n} &= \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0 \text{ and } m > 0 \\ a_{m,n-1} + n & \text{if } n > 0 \end{cases} \end{aligned}$$

- Inductive hypothesis: For all $(0,0) \leq (i,j) < (k_1,k_2)$:

$$a_{i,j} = i + \frac{j(j+1)}{2}$$

- Want to show:

Example, cont.

Show $a_{m,n} = m + n(n+1)/2$ for:

$$\begin{aligned} a_{0,0} &= 0 \\ a_{m,n} &= \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0 \text{ and } m > 0 \\ a_{m,n-1} + n & \text{if } n > 0 \end{cases} \end{aligned}$$

- Since recursive step of definition has two cases, we need to do proof by cases:
  - Case 1: $k_2 = 0,\ k_1 > 0$
  - Case 2: $k_2 > 0$

Example, cont.

Show $a_{m,n} = m + n(n+1)/2$ for:

$$\begin{aligned} a_{0,0} &= 0 \\ a_{m,n} &= \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0 \text{ and } m > 0 \\ a_{m,n-1} + n & \text{if } n > 0 \end{cases} \end{aligned}$$

- Case 1: $k_2 = 0, k_1 > 0$. Then, $a_{k_1,k_2} = a_{k_1-1,k_2} + 1$
- Since $(k_1 - 1, k_2) < (k_1, k_2)$, inductive hypothesis applies.
- By the IH, we know:

$$a_{k_1-1,k_2} = k_1 - 1 + \frac{k_2(k_2+1)}{2}$$

- But then $a_{k_1,k_2} = a_{k_1-1,k_2} + 1 = k_1 + \frac{k_2(k_2+1)}{2}$

Example, cont.

Show $a_{m,n} = m + n(n+1)/2$ for:

$$\begin{aligned} a_{0,0} &= 0 \\ a_{m,n} &= \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0 \text{ and } m > 0 \\ a_{m,n-1} + n & \text{if } n > 0 \end{cases} \end{aligned}$$

- Case 2: $k_2 > 0$. Then, $a_{k_1,k_2} = a_{k_1,k_2-1} + k_2$
- Since $(k_1, k_2 - 1) < (k_1, k_2)$, inductive hypothesis applies.
- By the IH, we know: $a_{k_1,k_2-1} =$
- But then $a_{k_1,k_2} = k_1 + \frac{k_2(k_2-1)}{2} + k_2$
- $a_{k_1,k_2} = k_1 + \frac{k_2^2 - k_2 + 2k_2}{2} = k_1 + \frac{k_2(k_2+1)}{2}$

□

Another Example

- Consider the function $\mathbb{Z}^- \to \mathbb{Z}^-$ defined recursively as follows:

$$\begin{aligned} f(-1) &= -1 \\ f(n) &= f(n+1) + n \quad \text{for } n < -1 \end{aligned}$$

- Prove that:

$$f(n) = -\frac{|n| \cdot (|n|+1)}{2}$$