Lecture Overview

Induction, Strong Induction, and Well-Ordering

Structural induction

Induction and Recursive Algorithms

Sets

You may state the predicate of the Induction such that it includes all i's

$Q(n)\ \forall i, 2 \leq i \leq n\ i$ can be factored, and prove $Q(n)$ using ordinary induction.

The induction and strong induction have "identical power"

Well-ordering:

Axiom: Every nonempty subset $S \subseteq N$ has a smallest element

Theorem: Any tournament that contains a cycle contains a 3-cycle.

Let n be the smallest cycle length in a tournament.
n must be at least 3.
Suppose $n \geq 4$.
Consider the arrow between p_1 and p_3.
Either close a 3-cycle
or yields a shorter cycle contradicting the assumption that n is smallest...

Not only induction over N,
Proofs for data types that are defined recursively

Fully parenthesized Boolean algebra

Definition: the set of fully parenthesized boolean expressions F

1. 0 and 1
2. if $e, e' \in F$ then $(e \wedge e') \in F$.
3. if $e, e' \in F$ then $(e \vee e') \in F$.
4. if $e \in F$ then $(\neg e) \in F$.
5. F contains nothing else.

Theorem Every Boolean expression has the same number of left and right parentheses.

Base: 0 and 1

Inductive step:
steps 1-4 preserve the assertion (if e and e' satisfy the assertion so does the result).

Definition: the set of binary trees T

1. A single node is in T
2. if t is in T, then an addition of a root node r and an arrow connecting t to be the left child (subtree) of the root is also in T
3. if t is in T, then an addition of a root node r and an arrow connecting t to be the right child (subtree) of the root is also in T
4. if t and t' are in T, then an addition of a root node r and two arrows, where t is the left child and t' is the right child is also in T

Theorem: The number of edges in any binary tree is exactly one fewer than the number of nodes.

Base: a single-node tree

Inductive step:
Steps 2 and 3 of the construction preserves the assertion:
—- t has $n-1$ edges and a node and an edge are added.
Steps 4 of the construction preserves the assertion:
—- t and t' has $n-1$ and $n'-1$ edges, respectively. 1 node is added (total $n+n'+1$) and 2 edges are added (total $(n-1)+(n'-1)+2 = n+n'$ edges).

Definition: the set S of (finite length) strings over alphabet A:

1. λ, the empty string, is in S.
2. If s is in the set S and $a \in A$, then sa is in S.
3. S contains nothing else.

To prove a property $P(s)$ for strings over alphabet $A = \{0, 1\}$:

Prove: $\forall s \in S(P(s))$

1. (Base) $P(\lambda)$
2. (Inductive step) $\forall s \in S \ \ (P(s) \Rightarrow P(s0))$
3. (Inductive step) $\forall s \in S \ \ (P(s) \Rightarrow P(s1))$
4. QED Structural induction on strings.

Theorem: In a string of 0s and 1s, the number of occurrence of the pattern 01 $num(01, s)$ is less than or equal to the number of occurrence of 10 $num(10, s)$, plus one.

It turned out that it is better to have a stronger induction predicate.
(Adding 1 at the end may increase $num(01, s)$)

$num(01, s) \leq num(10, s) + 1$, and
If s ends in 0 then $num(01, s) \leq num(10, s)$
(Thus, addition of 1 at the end will not violate our main predicate).

Base: λ
Step:
1. (*(Inductive step) $\forall s \in S \ (P(s) \Rightarrow P(s0))$*)

1a. if s ends in 0 then $num(01, s0) = num(01, s)$ and $num(10, s0) = num(10, s)$
o.k.

1b. if s ends in 1 then $num(01, s0) = num(01, s)$ and $num(10, s0) = num(10, s) + 1$
We should show that $num(01, s0) \leq num(10, s0)$ given
$num(01, s) \leq num(10, s) + 1$, replacing terms we get
$num(01, s0) \leq num(10, s0)$ as needed.

2. (*(Inductive step) $\forall s \in S \ (P(s) \Rightarrow P(s1))$*)

2a. if s ends in 0 then $num(01, s1) = num(01, s) + 1$ and $num(10, s1) = num(10, s)$
We should show that $num(01, s1) \leq num(10, s1) + 1$ given
$num(01, s) \leq num(10, s)$, replacing terms we get
$num(01, s1) - 1 \leq num(10, s1)$ as required

2b. if s ends in 1 then $num(01, s1) = num(01, s)$ and $num(10, s1) = num(10, s)$
o.k.

Why use induction?
Induction and computation is one step at a time.

Can any 2^n by 2^n board be tiled with Ls ?
the inductive proof is actually a *procedure*: break the board to four, tile each piece, and merge the tiling.

This is a *recursive* algorithm — it calls itself on smaller subproblems.

The ranking B, L and x is similar.

RSA is used in secure communication protocols SSH, SSL, etc.

Computes remainder r of x^a on division by b, where a and b are specially constructed *large* numbers.

If you know the right secret about the origin of a and b then you get x.

We will try to calculate the Exponentiation:

$x^0 = 1$
$x^n = x \cdot x^{n-1}$

Procedure $Exp(x, n)$

if $n = 0$
return 1
else
return $x * Exp(x, n-1)$

Theorem: For all $x, n \in N$, a call to $Exp(x, n)$ returns x^n

We should prove that the algorithm does not run forever

The algorithm returns the right value

By induction on n
Base: $Exp(x, 0) = x^0 = 1$
Step: Assume $Exp(x, n)$ returns x^n and prove for $Exp(x, n + 1)$
the **else** is chosen calling $Exp(x, n)$, which returns x^n,
thus we return x^{n+1}.

n has 128 bits, we should call the procedure 2^{128} times........

if $x^a = x^{2c}$ (a is even)
then compute $y = x^c$ and then y^2.

Otherwise, (a is odd)
then compute $y = x^{a-1}$ and then $y \cdot x$.

Procedure $FastExp(x, n)$

if $n = 0$
return 1
else
if n is even
    let $y = FastExp(x, n/2)$
    return y^2
else
    let $y = FastExp(x, (n-1)/2)$
    return $x * y^2$

Theorem: For all $x, n \in N$, a call to $FastExp(x, n)$ returns x^n

Base: $n = 0$
Step: (strong induction) true for every $k < n$ and prove for n
n is even, say $n = 2m$ then we call $FastExp(x, m)$, by strong induction $y = x^m$, thus we return $(x^m)^2 = x^n$ as required.
n is odd, say $n = 2m+1$ then we call $FastExp(x, m)$, by strong induction $y = x^m$, thus we return $x \cdot (x^m)^2 = x^{2m+1}$
Proof works for all cases.

no more than 256 multiplications ...

Theorem: If $n > 1$ has b bits, then $FastExp(x, n)$ performs at most $2b$ multiplications.

Base: $b = 2$ then $n = 0$, $n = 1$, $n = 2$, $n = 3$
if $n = 0$ then we do not multiply
if $n = 1$ we multiply $3 < 4$ times $(x \cdot 1^2)$
if $n = 2$ we multiply $2 < 4$ times (y^2)
if $n = 3$ we multiply $3 < 4$ times $(x \cdot x^2)$

Step: we assume b and prove for $b + 1$
We execute recursive call with $n/2$ or $(n-1)/2$ both have b bits, by induction $2b$ multiplications.
Then we do either 1 (even case) or 2 more (odd case) multiplications.
Thus overall at most $2b + 2$

Given 2 positive natural numbers, $a, b \in N^+$, their *greatest common divisor* (GCD) is the largest number that divides both without remainder.

Euclid's algorithm

Lemma: Given $x > y$, let $r = x - y$. Then any common divisor of x and y is also a common divisor of y and r, and vice versa.

Proof: Suppose $d|x$ and $d|y$. So $x = zd$ and $y = wd$. Thus, $r = x - y = d(z - w)$.
Suppose $d|y$ and $d|r$ then $x = y + r = wd + ud = (w + u)d$.

Corollary: GCD(x, y) =GCD(y, r)

Proof: Since (x, y) and (y, r) have exactly the same set of divisors, the biggest number in each set is identical.

So?

Procedure $Euclid(x, y)$

if $x = 0$ return y
else if $y = 0$ return x
else if $x > y$ return $Euclid(y, x - y)$
else return $Euclid(x, y - x))$

Example:
Find $GCD(112, 84) = ?$
Step 1: $112 - 84 = 28$, reduce to $(28, 84)$.
Step 2: $84 - 28 = 56$, reduce to $(28, 56)$.
Step 3: $56 - 28 = 28$, reduce to $(28, 28)$.
Step 4: $28 - 28 = 0$, reduce to $28, 0$
Now done, answer is 28.

First prove that the algorithm terminates

Lemma For any natural number inputs x and y, $Euclid(x, y)$ returns a value.

Induction on $x + y$

if $x + y = 0$ then we return immediately
Assume for $x + y < n$ and prove for $x + y = n$
We either return (when x or y equal 0) or call with smaller $x + y$.

Lemma For any natural number inputs x and y, $Euclid(x, y)$ returns GCD(x, y).

Induction on $x + y$

Base: GCD(0,0)=0 (*in fact we need GCD(x,0)=x*)
Step: $x+y = n > 0$, assume without loss of generality $x \geq y$. Return the result on $Euclid(y, x-y)$ (strong induction) and lemma complete the proof.

A *set* is a collection of objects.

Order is not important $\{1, 2\}$ and $\{2, 1\}$ are the same set.

Element is included once $\{a, a, b\}$ and $\{a, b\}$ are the same set.

Examples of sets:
$\{1,2,3\}$
$\emptyset$ the *empty set.*
$\{ \emptyset \}$ a set with one element, which is a set.
$\{1,\{1,2\}\}$ an element is included only once
$\{n \in N | n \text{ is even}\}$, in general domain D and predicate P over D, $\{x \in D | P(x)\}$ the set of all x in D that satisfy P

$a \in A$ in words a *is an element of* A,
$(\forall x) x \notin \emptyset$.

$A \subseteq B$ ("A is a subset of B" or "A is contained in B") if every element of A is also an element of B
$(\forall x)(x \in A \to x \in B))$.
A is a *proper* subset of B $(A \subset B)$ if $A \subseteq B$ and $A \neq B$.

Note $(\forall X)\emptyset \subseteq X$, i.e. the empty set is contained in any other set.

$A = B$ if and only if $A \subseteq B$ and $B \subseteq A$.

To prove that the sets $\{n \in N | 2 \ divides\ n\}$ and $\{2n | n \in N\}$ are equal we show containment in both ways:

$\subseteq$:
Assume $x \in N$ and 2 divides x and show that $\exists y \in N$ s.t. $x = 2y$.
$\supseteq$:
Assume $x = 2y$ then x is divisible by 2.

The *size* of a set A — the number of elements in A is $|A|$.
$|\{a, b, c\}| = 3$, $|\emptyset| = 0$, the size of an infinite set is ∞.

$|A \cup B| \leq |A| + |B|$ (equal when $A \cap B = \emptyset$) holds for infinite sets when $x + \infty = \infty$.

Union — $A \cup B = \{x | x \in A \text{ or } x \in B\}$.

$\cup_{i=1}^{n} A_i$
$i = 0$ results in $\emptyset$

Intersection — $A \cap B = \{x | x \in A \text{ and } x \in B\}$.

$\cap_{i=1}^{n} A_i$
$i = 0$ results in U the universal set.

Difference — $A - B = \{x | x \in A \text{ and } x \notin B\}$.
(or $A \setminus B$)

Complement — $A^c = \{x | x \notin A\}$.
(or $\bar{A}$).
$A^c = U - A$.

Symmetric difference — $A \oplus B = A \cup B - A \cap B$.

Venn diagrams

Given predicates p and q,
$P = \{x \mid p(x)\}$,
$Q = \{x \mid q(x)\}$.
Then $P \cup Q = \{x \mid p(x) \vee q(x)\}$.
$P \cap Q = \{x \mid p(x) \wedge q(x)\}$.
Etc.

$\mathcal{P}(A)$ the set of all subsets of A.
In other words, $\mathcal{P}(A) = \{x | x \subseteq A\}$.

$\emptyset \in \mathcal{P}(A)$ and $A \in \mathcal{P}(A)$.

If $A = \{1, 2\}$ then $\mathcal{P}(A) = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$.

If A has n elements, then $\mathcal{P}(A)$ has 2^n elements.

Induction on n
Base: $n = 0$ $\mathcal{P}(A) = \{\emptyset\}$.
Step: let a be the "new" element
$\mathcal{P}(A)$ contains all subsets without a which are 2^{n-1} subsets
for each such subset a subset that includes a.
$2^{n-1} \cdot 2$.
(the number of binary words with n bits).

$\mathcal{P}(A)$ can be denoted 2^A.

(Distributivity: Union distributes over intersection)
$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

We show that each is contained in the other

1. $A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C)$.

Suppose $x \in A \cup (B \cap C)$.
Then, $x \in A$ or $x \in (B \cap C)$.
We argue by cases.

If $x \in A$, then $x \in A \cup B$ and similarly $x \in A \cup C$.
By definition of intersection x is in the intersection of these two sets.

On the other hand, if $x \in (B \cap C)$, then $x \in B$ and $x \in C$.
By definition of union, $x \in A \cup B$ and $x \in A \cup C$.
It again follows that x is in the intersection.

2. $A \cup (B \cap C) \supseteq (A \cup B) \cap (A \cup C)$.

Suppose $x \in (A \cup B) \cap (A \cup C)$.
It follows that $x \in A \cup B$ and $x \in A \cup C$.
We argue by cases.

If $x \in A$, then certainly $x \in A \cup (B \cap C)$.

If $x \notin A$, then since $x \in A \cup B$, we know $x \in B$.
Similarly, $x \in C$.
Therefore, $x \in B \cap C$.
It follows that $x \in A \cup (B \cap C)$.

$(A \cup B)^c = A^c \cap B^c$.

We have to show that $(A \cup B)^c \subseteq A^c \cap B^c$ and that $A^c \cap B^c \subseteq (A \cup B)^c$.

1. $(A \cup B)^c \subseteq A^c \cap B^c$.

Let $x \in (A \cup B)^c$.
By definition of complement, $x \notin A \cup B$.
(proof by contradiction) $x \notin A$ and $x \notin B$.
By definition of complement, $x \in A^c$ and $x \in B^c$.
Then by definition of intersection, $x \in A^c \cap B^c$.

2. $A^c \cap B^c \subseteq (A \cup B)^c$.

Let $x \in A^c \cap B^c$.
Then x is in A^c and in B^c.
Therefore, x is in neither A nor B and therefore not in $A \cup B$.
We conclude that $x \in (A \cup B)^c$.

Dual holds, swapping $\cup$ with $\cap$ and $\emptyset$ with U.

$A \times B = \{(a, b) | a \in A \; and \; b \in B\}$.
(a, b) is an ordered pair $\neq (b, a)$.

$A = \{0, 1\}$ and $B = \{x, y\}$ then $\{A \times B\} = \{(0, x), (0, y), (1, x), (1, y)\}$.

Generalize Cartesian products
$A_1 \times \cdots \times A_n = \prod_{i=1}^{n} A_i = \{(a_1, \cdots, a_n) | a_i \in A_i$ for $1 \leq i \leq n\}$.
Also, A^n is $\prod_{i=1}^{n} A$.

Examples:
$\{0, 1\}^n$ is the set of all n-tuples of 0's and 1's.
Or, length n strings of 0's and 1's.

$\Re^2 = \Re \times \Re$ is the real plane.