Review

COMPSCI 3331

Outline

Mathematical Necessities:

- Set Theory.
- Induction.
- Logic.

Set Theory

- A set is a collection of objects.
- We can specify sets by listing the elements or describing them all:
  - Finite sets: $S = \{a, 1, y\}$.
  - Infinite sets: $S = \{x \in \mathbb{N} : x \geq 2\}$.
- Descriptions of sets:

$$\{x \in S : x \texttt{ <satisfies some condition> } \}.$$

- "All x in the set S such that `<some condition>` holds".

Set Theory

- $\emptyset$ is the set consisting of no elements.
- Membership: $x \in S$ means x is an element of the set S.
- Inclusion: $S_1 \subseteq S_2$ means every element of S_1 is an element of S_2.
  - Note $\emptyset \subseteq S$ for all sets S.
- Equality $S_1 = S_2$: Two sets S_1, S_2 are equal if everything in S_1 is in S_2 and vice versa.
  - i.e., $S_1 \subseteq S_2$ and $S_2 \subseteq S_1$.

Set Theory

Example:

$$\begin{aligned} S_1 &= \{1,2,3\}, \\ S_2 &= \{1,3\}, \\ S_3 &= \{1,3,2\}. \end{aligned}$$

Then

Set Theory

Operations on sets:

- Union: $S_1 \cup S_2 = \{x \; : \; x \in S_1 \text{ or } x \in S_2\}$.
- Intersection: $S_1 \cap S_2 = \{x \; : \; x \in S_1 \text{ and } x \in S_2\}$.
- Difference: $S_1 - S_2 = \{x \; : \; x \in S_1 \text{ and } x \notin S_2\}$.
- Cross product: $S_1 \times S_2 = \{(x, y) \; : \; x \in S_1 \text{ and } y \in S_2\}$ (aka Cartesian product).
- Complement: every set S has a universe $S \subseteq U$. The complement of S (relative to U) is $\overline{S} = U - S$.

Set Theory

Complement example:

- Let $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$.
- $S = \{x \in \mathbb{N} \ : \ x \text{ is a multiple of } 4\} = \{0, 4, 8, 12, \ldots\}$, ($S$ has universe $\mathbb{N}$)
- then $\overline{S} = \{1, 2, 3, 5, 6, 7, 9, 10, 11, 13, \ldots\} = \{x \in \mathbb{N} \ : \ x \text{ is not a multiple of } 4\}$.
- In this course, the universe will always be either explicitly stated or clear from the context.

Set Theory

- if I is a set (finite or infinite) and S_i are sets for all $i \in I$, then

$$\bigcup_{i \in I} S_i = \{x \ : \ \exists i \in I \text{ such that } x \in S_i\}.$$

- the same applies for intersection.

Power sets:

- if S is a set, then $2^S = \{S' \ : \ S' \subseteq S\}$ is the set of all subsets of S.
- For example, if $S = \{a, b\}$, then

$$2^S = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}.$$

- If S has n elements, 2^S has 2^n elements.

Set Theory

De Morgan's Laws:

$$\begin{aligned} \overline{A \cup B} &= \overline{A} \cap \overline{B} \\ \overline{A \cap B} &= \overline{A} \cup \overline{B} \end{aligned}$$

Other laws:

$$\begin{aligned} \overline{\overline{A}} &= A \\ A \cap \emptyset &= \emptyset \end{aligned}$$

Functions

- Given a function f which takes elements from S and converts them to elements from T, we denote this by $f : S \to T$.
- e.g., g
$$g : \mathbb{N} \to 2^{\mathbb{N}}$$
defined by $g(n) = \{1, 2, 3, \ldots, n\}$.
- Functions which take two or more arguments can be denoted using cross product.
- e.g.,
$$f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
defined by $f(a, b) = ab$. (multiplication).

Induction

- Induction is a method of proving a certain proposition (involving n) holds for all values of n:

$$1+2+\cdots+n=n(n+1)/2$$

- Induction works on more complicated structures:
  - **Binary Trees**: Prove that every binary tree of height n has at most 2^n-1 nodes.
  - **Graphs**: Euler's Formula: $V-E+F=2$.

Induction

Formally: let $P(n)$ be a statement involving the natural number n, Then $P(n)$ holds for all $n \geq 0$ if the following hold:

- base case: $P(0)$ holds. That is, the statement holds for $n = 0$.
- inductive step: For any $k \geq 0$, if $P(k)$ holds, $P(k+1)$ holds also.

Examples of statements involving n:

1. $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$.
2. $(x+y)^n = \sum_{i=0}^{n} \binom{n}{i} x^i y^{n-i}$.

Induction

Example: Prove that for all n, $\sum_{i=0}^{n} i^2 = n(n+1)(2n+1)/6$.

- Also have "strong" induction: Assume $P(i)$ is true for all $i \leq n$, prove $P(n+1)$ is true.
- Example: prove that every integer $n \geq 2$ is either a prime number of a product of two or more primes.

Recursive Definitions

Recursive definitions:

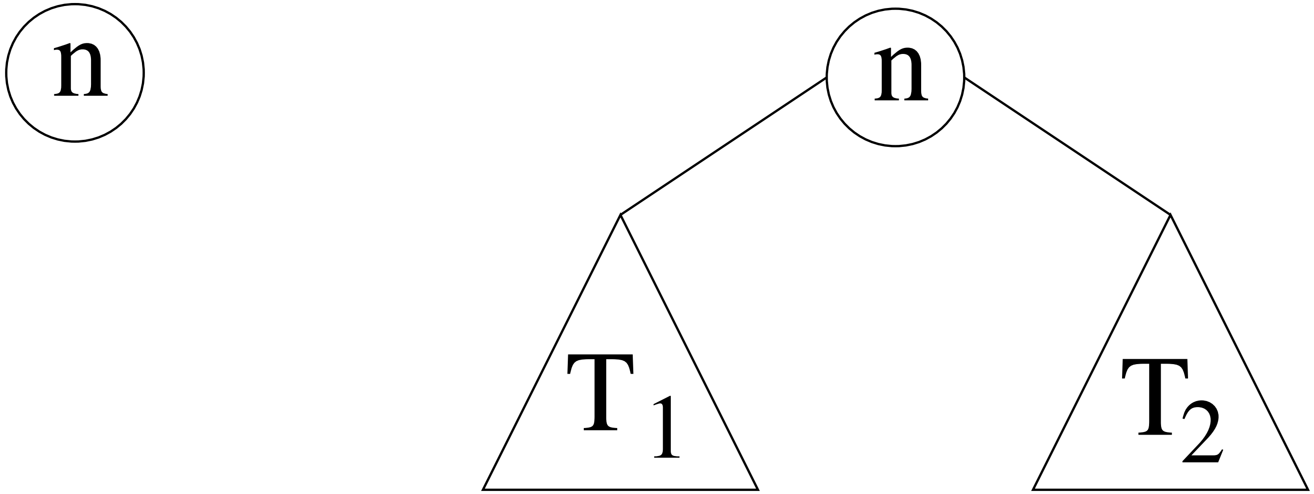
$$n! = \begin{cases} n(n-1)! & \text{if } n \geq 2 \\ 1 & \text{if } n = 1. \end{cases}$$

Binary Trees:

- a node is a binary tree.
- if T_1, T_2 are binary trees (possibly empty), and n is a node, then the structure with root n, left subtree T_1 and right subtree T_2, is also a binary tree.

Recursive Definitions

Binary Trees:



Some structures with recursive definitions are suited to proof by induction: e.g., prove $n! > 2^n$ for all $n \geq 4$.

However, we usually need to use **structural induction**.

Structural Induction

NOT REVIEW

Let S be a set (finite or infinite) of structures defined recursively in the following way:

1. For some finite (easy) set I, $I \subseteq S$ (i.e., each element of I is an element of S, I is the **base set**).
2. For some set of operations op_i $(1 \leq i \leq n)$, if $x_1, \ldots, x_n \in S$ then $op_i(x_1, \ldots, x_n) \in S$ for all $1 \leq i \leq n$. (the ops represent how we **build up** structures in S) .

Implicitly, we agree that anything formed in any way not defined by 1 or 2 is not an element of S.

Example: S is the set of binary trees.

Structural Induction

Example: S is the set of arithmetic expressions:

- (**base set**) n is an arithmetic expression for all $n \in \mathbb{N}$;
- (**building rules**) if x, y are arithmetic expressions, so are

$$(x+y), (xy), (x-y), (x^y), \text{ and } (x/y).$$

Structural Induction

Structural induction on a set S defined by I and O works as follows: Let P be a statement involving members of S.

- e.g., $P =$ "every binary tree with height n has at most 2^n nodes."

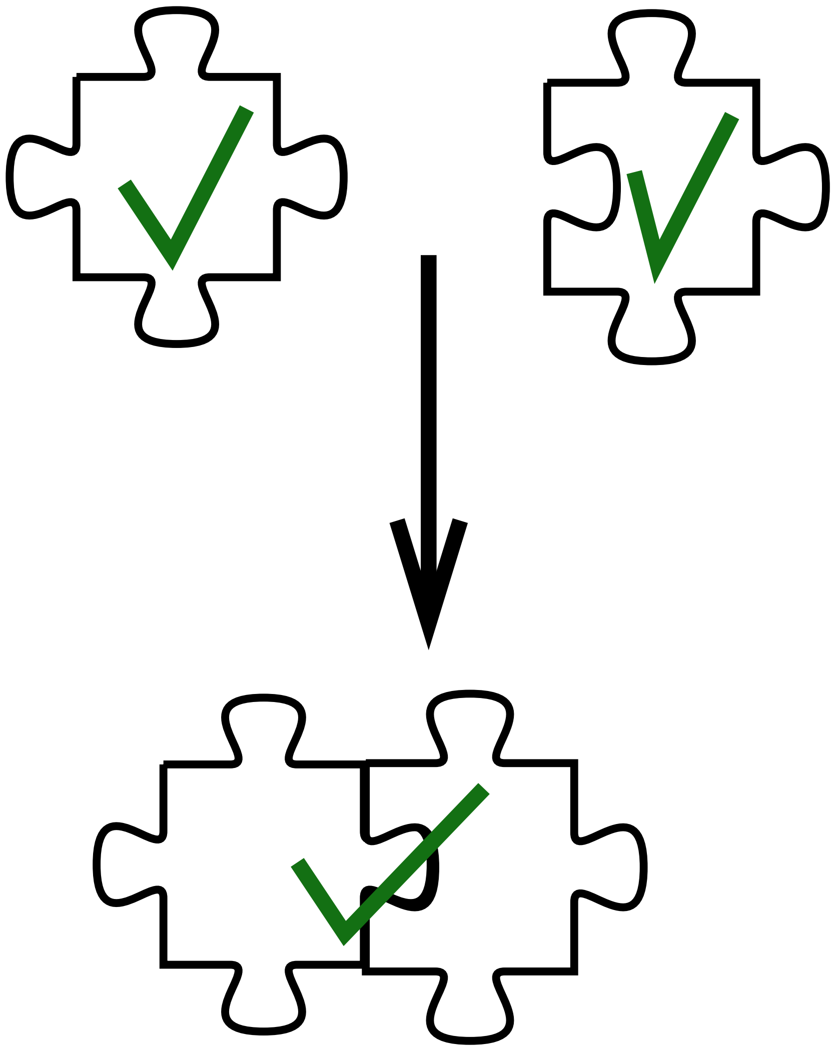
IF

- $P(x)$ holds for all $x \in I$ (**prove for base set**) **and**
- If whenever $x_1, \ldots, x_n \in S$ and $P(x_i)$ holds for all $1 \leq i \leq n$, then $P(op(x_1, \ldots, x_n))$ holds for all $op \in O$. (**prove for building rules**)

THEN

- Every element of $x \in S$ satisfies $P(x)$.

Structural Induction



Structural Induction

Example: **every non-empty binary tree has one more node than edges**.

Proof:

- (**prove for base set**) if T is a node, then it has one node and zero edges.
- (**prove for building rules**) if T is a tree with root n and subtrees T_1, T_2 with $edges(T_i) = nodes(T_i) - 1$, then the tree T has $edges(T_1) + edges(T_2) + 2$ and $nodes(T_1) + nodes(T_2) + 1$.

$$\Rightarrow edges(T) = nodes(T) - 1.$$

Thus, by structural induction, every binary tree has one more node than edges.

Why/When Structural Induction?

- When there is no easy relationship between a recursively defined structure and natural numbers.
- In this course, **regular expressions** and **grammars** will be natural targets for structural induction proofs.

Proofs and Logic

- In this course, we use proofs to establish statements rigorously.
- We will see proofs in class and you will write proofs on assignments.
- Let's review some proof techniques, tricks and common pitfalls.

Proofs and Logic

Given a statement `if A then B`, what can we show using this statement?

- If `A` is true, then we can conclude `B`.
- **contrapositive**: If `B` is not true, then `A` is not true. (`not B` implies `not A`)

Contrapositive Example:

- Statement: `If a student cheats, then they fail the assignment.`
- Contrapositive: `If a student didn't fail, that means` **they didn't cheat.**

Proofs and Logic

De Morgan's law is used to negate `and` and `or`:

- `not ( A and B)` ≡ (`not A`) `or` (`not B`).
- `not ( A or B )` ≡ (`not A`) `and` (`not B`).

Example:

$$
\begin{aligned}
&\texttt{not (cloudy and chance of rain)} \\
\equiv\ &\texttt{(not cloudy) or (not chance of rain).}
\end{aligned}
$$

Proofs and Logic

There are two quantifiers:

- $\forall$: For all.
- $\exists$: There exists.

Use a quantifier in relation with some variable:

$$\forall x \in \mathbb{N}, x \geq 0$$

General form:

$$\forall x . P(x)$$

where $P(x)$ is an expression (using and,not,or,exists) involving x ($P(x) = x \geq 0$)

Proofs and Logic

Negating quantifiers (in stating contrapositives):

- $\texttt{not}\ \forall x.P(x) = \exists x.\texttt{not}\, P(x)$.
- $\texttt{not}\ \exists x.P(x) = \forall x.\texttt{not}\, P(x)$.

Example: **if** a course is hard **then** all students get a bad grade.

- “all students get a bad grade” – $\forall$ student, student.grade = bad.
- negation: $\exists$ student, `not` (student.grade = bad).

Contrapositive: **If** there is a student who got a good grade **then** the course is not hard.

Types of proofs

In this course, remember the following proof techniques:

- Induction and Structural Induction.
- **Using the contrapositive**: To prove **if A then B** we instead prove **if not B then not A**.
- **Proof by contradiction**: To prove **if A then B** we assume A holds then show if **not** B holds as well, a contradiction arises.

Proofs

"Iff" (if and only if):

- make sure you prove both directions.
- `A iff B` = `if A then B` **AND** `if B then A`

"Disprove":

- to disprove $\forall x.P(x)$, need to find one x such that $P(x)$ does not hold – a **counter-example**.
- Example: Every prime number is odd.