Lecture 1

Wednesday, January 18, 2023 5:58 PM

Copy req symbols from here math symbols
Or here
https://www.math.utah.edu/~schwede/MichiganClasses/math185/NotationAndTerminology.pdf

Also here : https://wumbo.net/symbols/set-of-natural-numbers/

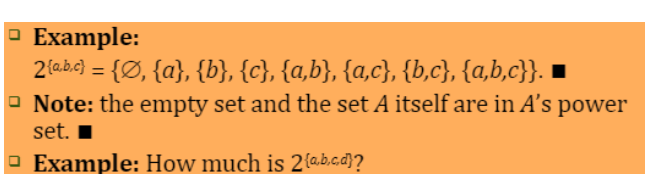
- Logic and Proofs
  - Propositions, Truth tables, true/false
  - Proposition with a free variable (x) can be t/f, dep on val of var
  - Logical connectives
    -

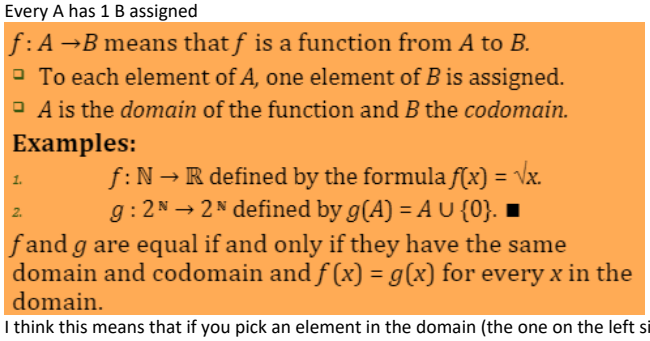
Connective	Symbol	Typical Use	English Translation
conjunction	$\wedge$	$p \wedge q$	p and q
disjunction	$\vee$	$p \vee q$	p or q
negation	$\neg$	$\neg p$	not p
conditional	$\rightarrow$	$p \rightarrow q$	if p then q p only if q
biconditional	$\leftrightarrow$	$p \leftrightarrow q$	p if and only if q

    - Since I never got the conditional, look at where the arrow is pointing. If where it is pointing is true, than no matter what, the result is true. If what is being pointed to is false, check what is pointed from, if that is true, then result is false. If what was pointing was false, (pointing to false) then result is false.
  - Make truth table
    -

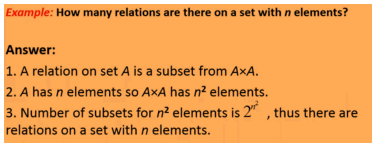
p	q	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

  - Tautology- proposition that's always true
    - Ex: p v¬ p
  - Contradiction- proposition that's always false
    - Ex: p ^¬ p
  - Logically equivalent- if always have same truth values
    - Shown as : P ⇔ Q
  - Logically imply- if whatever one is, the other is also true
    - Shown as : P ⇒ Q
  - P ⇒ Q and P --> Q look similar, but
    - P --> Q is proposition, has truth val
    - P ⇒ Q is meta- statement, relationship btwn P and Q, shows they have tautology
    - Same for P ⇔ Q and P ↔ Q
  - Logical identities, simplify
    - Commutative laws:
      - p v q ⇔ q v p
      - P ^ q ⇔ q ^ p
    - Associative laws:
      - p v (q v r) ⇔ (p v q) v r
      - p ^ (q ^ r) ⇔ (p ^ q) ^ r
    - Distributive Laws:
      - p v (q ^ r) ⇔ (p v q) ^ (p v r)
      - p ^ (q v r) ⇔ (p ^ q) v (p ^ r)
    - DeMorgan Laws
      - ¬ (p v q) ⇔ ¬ p ^ ¬ q
      - ¬ (p ^ q) ⇔ ¬ p v ¬ q
    - Equivalent formulation of conditional
      - (p --> q) ⇔ (¬p v q)
    - Contrapositive of conditional
      - (p --> q) ⇔ (¬q --> ¬p)
    - Equivalent form of biconditional
      - (p ↔ q) ⇔ (p --> q) ^ (q --> p)
  - Can use logic quantifies for statement not about specific elements in domain, but about domain itself
    - ∀ - for every/all
    - ∃ - for some/exists
  - W/ >1 quantifier, order matters
  - Proof is series of statements, derived from initial assumptions, statements derived previously, and generally accepted facts
  - Direct, constructive proof: follow definition, build on that
  - Indirect proof: contrapositive, negate the stuff (or becomes and), implies something
  - Proof by contradiction: like contrapositive, but refer to definition
  - Proof by cases: look at all possible cases, look at truth tables and other stuff
- Sets
  - Finite set, sometimes use ellipses (…), sometimes # after the ellipses to show it's finite set
  - "Such that" is "|", defining properties looks like:

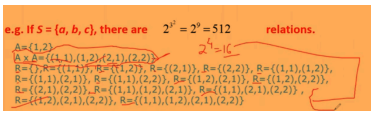
        - B = {x|x is a nonnegative integer multiple of 3}
        - It reads "B is the set of all x such that x is a nonnegative
        - integer multiple of 3"
    - "Element of" look like "∈"
        - X ∈ A means x is an element of A
    - "⊆" is "subset"
    - "Empty set" is "Ø", order not important, repetition not important
    - Frequent sets:
        - ℕ is natural numbers, or nonnegative integers
            - Positive whole numbers, not 0
        - ℤ is set of all integers
            - Whole number, no fractions
        - ℝ is set of all real numbers
            - Infinite decimal extension, cannot be imaginary
        - ℝ+ is nonnegative real numbers
            - The positive real numbers
    - Union – "U", add all types, no repetition
    - Intersection – "∩", what they have in common
    - Difference - "-", everything except what is in common, nothing of the second set
    - Complement of set: if A is subset of (universal) U, complement of A is A', literally U-A, everything that is not in A
    - () doesn't include, [] includes
    - Sets are disjoint if intersection is empty (nothing in common), pairwise disjoint if every 2 distinct sets in collection are disjoint, partition is collection of pairwise disjoint sets
    - U use some, ∩ uses every, power set is 2^A, ex:
        - 

        - Set of all subsets of a set A is called "the power set of A", shown as 2^A
    - Cartesian product, AxB is set of all ordered pairs, AxB={(a,b)|a ∈ A and b ∈ B}
        - This literally means that the cartesian product btw AxB will get you a thing that looks like (a,b) where "a is an element of A" (the set A) and "b is an element of B" (the set B)
    - f:**A**-> **B** means f is function from A to B, A is domain, B is codomain
        - Every A has 1 B assigned
        - 

        - I think this means that if you pick an element in the domain (the one on the left side of the arrow) to plug in, the result is in the codomain (the one on the right side of the arrow)
    - Range of function is set of elements of the codomain that are actually values of the function
        - f is <u>one-to-one</u> if f never assigns the same value to two different elements of its domain
        - f is <u>onto</u> if its range is the entire set B (codomain)
        - If both one-to-one and onto, it is called <u>bijection</u>
    - If f: A->B is a bijection, then define the inverse f^{-1}, from B to A, by formals: for every x ∈ A and y ∈ B:
        - $f^{-1}(f(x)) = x$
        - $f(f^{-1}(y)) = y$
- Functions and Equivalence relations
    - N-ary operation on set A is a function that assigns to every ordered n-tuple of elements of <u>A</u> an element of A
        - N-tuple is an ordered set with n elements, so it's a finite set, with specific values in it, ordered just means that it goes in order
    - Urnary/binary op on set A
        - Binary operations on the integers include addition
        - For every set (S), binary ops on 2^s includes union and intersection
        - Unary operations include negation (on sets of integers (as an example)) and complementation (on the set 2^A)
        - "Subset A_1 of A is closed under the op" if result of applying the op to elements of A1 is an element of A1
    - A = 2^N and A_1 is set of nonempty subsets of A, then A_1 is closed under union but not under intersection
    - Set of even natural numbs closed under addition and multiplication, set of odd nat numbs closed under multiplication but not addition multiplication but not addition



- Relations
  - A relation btwn A and B from A to B is subset of A x B, relation on set A is relation from A to A, or a subset of AxA
  - Express relationships many ways: if R is a relation on a set, write "a is related to b" as a R b , or (a,b) ∈ R



  - Equivalence Relations (must satisfy these 3 properties) https://www.youtube.com/watch?v=FI6j5QZNVx0
    1. R is reflexive: for every x ∈ A, x R x
    2. R is symmetric: for every x and every y in A, if x R y , then y R x
    3. R is transitive: for every x, every y, and every z in A, if x R y & y R z, then x R z
  - For equivalence relation R on a set A, and an element x ∈ A, the equivalence class having x is:
    - $[x]_R = \{y \in A \mid y \, R \, x\}$
    - "Theorem: If R is an equivalence relation on A, the equivalence classes with respect to R form a partition of A, and two elements of A are equivalent if and only if they are elements of the same equivalence class."

- Languages
  - Alphabet is finite set of symbols, denoted by sigma (Σ)
  - String over Σ is finite sequence of symbols
  - |x| stands for length of string x
  - $n_a(x)$ is number of occurrences of a in string x
  - Null string Λ (it's lambda) is string over any alphabet Σ
  - |Λ| = 0
  - Set of all strings over Σ is Σ*, lang over Σ is a subset of Σ*
    - Lots of examples of stuff
  - Definition: xy is concatenation of two strings x & y, this is the basic op on strings
    - Ex: if x= ab and y = bab, then
    - xy = abbab
    - yx = babab
    -
    - For every string x, xΛ = Λx = x
    - |xy| = |x| + |y|
  - Concatenation is associative ((xy)z = x(yz)), so can just write xyz
  - If s = tuv , then t is a prefix of s, v is a suffix, and u is a substring
    - (every string is a prefix & suffix & substring of itself)
  -
  - For languages L_1, and L_2 over Σ
    - L1 U L2, L1∩L2, and L1-L2 all langs over Σ
  - If L ⊆ Σ*, the complement of L is a lang, Σ* – L
    - Remember that subtraction is complement
  - For languagesL1, L2 over Σ: L1L2 is the language {xy | x ∈ L1 and y ∈ L2}
  - Use exponential notation a^k = aaa...a, where there are k occurrences of a, also applies to strings and languages
  - $a^0 = x^0 = \Lambda, \; L^0 = \{\Lambda\} \quad (\text{for every } a \in \Sigma, x \in \Sigma^*, L \subseteq \Sigma^*).$
  -
  - If L is a lang over Σ, the L* denotes lang of all strings that can be gotten from concatenating 0 or more strings in L (Kleene star)
    - Concatenating as in the green highlight
    - $L^* = U\{L^k \mid k \in \mathbb{N}\}$
    - $\Lambda \in L^*$ for every lang L, since $L^0 = \{\Lambda\}$
  - Strings are finite, lang may not be
  - Strings are finite, and languages may not be, but to use a language we need a finite description:
    - $L_1 = \{ab, bab\}^* \cup \{b\}\{ba\}^*\{ab\}^*.$
    - $L_2 = \{x \in \{a, b\}^* \mid n_a(x) > n_b(x)\}.$

- Recursive Definitions
  - It of a set has a basis statement that specifies at least one member of the set, and a recursive part that specifies how additional members of the set can be generated in terms of given members
  - First ex is $\mathbb{N}$,the set of natural numbers
    - Defined as:
    - Basis statements: $0 \in \mathbb{N}$
    - Recursive Part: if $n \in \mathbb{N}$ then $n + 1 \in \mathbb{N}$
    - Every element of can be got from 1st 2 statements (^<those 2)
  - So, 3rd statement in def of is what says is smallest set that has 0 and is closed under the successor op (add by 1), the statement that "the set being defined is the smallest" is usually omitted but always understood
  - Slides 46-48
- Structural Induction(start here for actual notes)
  - When use open circle dot (°), it means +, and when use close dot (•), it means *
  - Also, ◊(x) is (x) (idk)
  - We use these ops for strings, but + and * for numbers
  - Prove stuff using structural induction
    - P(a) is true
    - X is element of Expr, satisfies condition P(x)
    - For every x and every y in Expr, if P(x) and P(y) are ture, then P(x°y) and P(x•y) are true

- In other words, set of elements x satisfying the property P has a and is closed under °, •, and ◊
- Recursive def of EXPR has a basis part (a (curved E) Expr)
  - First case of induction step is show that P(x°y) is true
- Induction hypothesis is that x,y (curved E) EXPR
-
  - Examples: $a + b$ has 3 symbols.
    - How many symbols does $c^*(a + b)$ have? 7 symbols.
  - Let us consider $P(x)$ is "x has odd length, where $x \in Expr$."
  - The *basis step* of the proof is to show that a has odd length, and this is clearly true: $|a| = 1$.
  - The *induction hypothesis* is that $x, y \in Expr$ and that x and y have odd length (i.e., $|x|$ and $|y|$ are odd).
  - The three cases in the induction step are to show that $x \circ y$, $x \bullet y$, *and* $\Diamond(x)$ have odd length.
  - These are all true, because:
    - $|x \circ y| = |x + y| = |x| + |y| + 1$, and odd + odd + 1 = odd (1 because of +).
    - $|x \bullet y| = |x^* y| = |x| + |y| + 1$ (1 because of *).
    - $|\Diamond(x)| = |(x)| = |x| + 2$, and odd + 2 = odd (2 because of '(' and ')')
- For the || pt, the 1 is bc the symbols count, so the total length of it includes the one as well-
- 7 symbols for c*(a+b)
- Circle (view as addition) , full circle (view of multiplication) and diamond (view as parenthesis) ?
- Math induction is structural induction based on recursive def of $\mathbb{N}$ given earlier
  -
    - This is used to prove statements of the form "for every integer $n \geq n_0$, $P(n)$".
    - *Basis step:* prove the statement $P(n)$ for $n = n_0$.
    - *Induction hypothesis:* k is an integer $\geq n_0$ and $P(k)$ is true.
    - *Induction step:* show using the induction hypothesis that $P(k+1)$ is true.
  -
    - **Exercise:** For every $n \in \mathbb{N}$, every set A with n elements, 2^A has exactly 2^n elements. ■
    - Basis: for every set A with 0 elements, 2^A has 2^0 elements; this is true because only $\emptyset$ has zero elements, and $2^\emptyset = \{\emptyset\}$, which has one element.
    - Induction hypothesis: $k \in \mathbb{N}$, and for every set A with k elements, 2^A has 2^k elements.
    - Induction step: to show that for every set A with $k+1$ elements, 2^A has 2^{k+1} elements. ■
  -
    - **Prove:** For every $n \in \mathbb{N}$, and every set A with n elements, 2^A has exactly 2^n elements (cont'd.) ■
    - Proof of induction step:
      - Let A be a set with k + 1 elements, and let a be any element of A (there is one, since $k + 1 \geq 1$).
      - Then $A - \{a\}$ has k elements, and $2^{A-\{a\}}$ has 2^k elements by the induction hypothesis; therefore, A has 2^k subsets that do not contain a and 2^k subsets that do contain a, for a total of 2^{k+1} subsets. ■
  -
- Strong induction- make sure all previous steps are used
  - Similar to normal by only basic step
  -
    - **Exercise:** Prove that for every $n \geq 2$, n is either prime or a product of two or more primes. ■
    - Let us strengthen the statement: for $n \geq 2$, every number m such that $2 \leq m \leq n$ is either prime or a product of two or more primes.
    - The reason for this, as we'll see, is that it gives us a stronger induction hypothesis but does not actually

require that we prove any more than we would have anyway.

- D

Proof:

- *Basis step*: To show that every m satisfying $2 \le m \le 2$ is either prime or a product of primes. This reduces to showing that 2 is, and 2 is prime.
- *Induction hypothesis*: $k \ge 2$, and for every m satisfying $2 \le m \le k$, m is either a prime or a product of primes.
- Statement to prove in *induction step*: For every m satisfying $2 \le m \le k + 1$, m is either prime or a product of primes.

Proof of *induction step*: For every m with $2 \le m \le k$, we already have the conclusion we want, from the induction hypothesis.

- The only additional statement we need to prove is that $k + 1$ is either prime or a product of primes.

If $k + 1$ is prime, we're done; if not, $k + 1$ is the product of two smaller numbers, both bigger than 1.

- By the (stronger) induction hypothesis, both are prime or the product of primes.
- Therefore (in either case), $k + 1$ is the product of primes.

- Balanced language
  - $\Lambda \in Balanced$
  - If $x, y \in Balanced$, then $xy \in Balanced$ and $(x) \in Balanced$ ■
- Making statement stronger makes it easier to prove