CS 170 NOTES

LUKE YANG
NOTES FROM A COURSE BY AARON COTE

ABSTRACT. These notes were taken during CS 170 (Discrete Methods in Computer Science) taught by Aaron COTE in Spring 2014 at University of Southern California. They were live-LaTeXed during lectures in `TeXShop` and compiled using XeLaTeX. Each lecture gets its own section. The notes are not edited afterward, so there may be typos; please email corrections to `yifeiyan@usc.edu`.

1. LECTURE 1, MONDAY, JANUARY 13

Course goals: Discrete math and problem solving skills with a wide range of topics.
Review: Sets, functions, sequences

Definition 1.1. A set is an unordered collection of objects.

Definition 1.2. Two sets are equal if and only if they have the same elements (aka objects, or members).

Venn graph
Universal set U contains all objects under consideration.

Definition 1.3. The intersection of sets S_1 and S_2, denoted by $S_1 \cap S_2$, is the set that contains those elements in both S_1 and S_2. $S_1 \cap S_2 = \{x | x \in S_1 \wedge x \in S_2\}$

Definition 1.4. The union of sets S_1 and S_2, denoted by $S_1 \cup S_2$, is the set that contains those elements that are either in S_1 or S_2, or both. $S_1 \cup S_2 = \{x | x \in S_1 \vee x \in S_2\}$

Definition 1.5. The complement of set S, denoted by $\overline{S}$, is the set that contains those elements that are in the universal set U but not in S. $\overline{S} = \{x | x \notin S\}$

Empty set $= \overline{U}$

Definition 1.6. Two sets S_1 and S_2 are disjoint if $S_1 \cap S_2 = $.

Generalized intersection

$$\bigcap_{i=1}^{n} S_i = S_1 + S_2 + \cdots + S_n$$

Generalized union

$$\bigcup_{i=1}^{n} S_i$$

Definition 1.7. Set A is a subset of set B, denoted by $A \subseteq B$, if and only if everything in A is in B.

Theorem 1.8. *Any set is a subset of itself. is a subset of any set.*

If two sets are subsets of each other, two sets are equal.

Definition 1.9. Set A is a strict subset of set B, denoted by $A \subset B$, if $A \subseteq B$ and $A \neq B$.

$A \subset B \Rightarrow A \subseteq B$, and its opposite is not true.

Example 1.10. 7 stamps: 2 red, 2 yellow, 3 green. A, B, C, are three perfect logicians. A removes blindfold and can't tell any conclusions about the colors about who's wearing what. B can't either.

From what A said, B and C can't wear red or yellow together. C wears green.

2. Lecture 2, Thursday, January 15

Example 2.1. 3 Truth machines are in stock. A machine corresponds true/false to red/green, but patterns for different machines can be different. 1 machine is broken (it answers arbitrarily) and 2 are working. Ask one single question (with a single yes/no answer) to one machine and determine which two are working.

Solution. Ask M1: is it the case that exactly one of these is true: Red means yes; 2nd machine is broken.

If Red, choose M2; if Green, choose M3.

Assume M1 works, and red means yes. Then it answers red if the 2nd machine works.

Assume M1 works, and green means yes. Then it answers green if the 2nd machine is broken.

Assume M1 works, both M2 and M3 work. Choosing either would be good.

Another method: meta-question: If I'd asked you, "Is machine 2 broken?", would you answer green?

Definition 2.2. The cardinality of a set A, denoted by $|A|$, is the number of distinct elements in A.

Example 2.3. $|\{\text{cake, pie, cake}\}| = 2$

Example 2.4. $|| = |\{\}| = 0$

Example 2.5. $|\{\}| = 1$

Example 2.6. $|\{\mathbf{Z}, \mathbf{N}\}| = 2$

Example 2.7. $|\mathbf{Z}| = \infty$

Definition 2.8. The power set of S, denoted by $P(S)$, is the set of all subsets of S.

Example 2.9. $S = \{\text{cake, pie}\}$, $P(S) = \{, \{\text{cake}\}, \{\text{pie}\}, \{\text{cake, pie}\}\}$

Example 2.10. $P() = \{\}$

Example 2.11. $P(\{\}) = \{, \{\}\}$

Example 2.12. $P(P(\{a\})) = P(\{, \{a\}\}) = \{, \{\}, \{\{a\}\}, \{, \{a\}\}\}$

$P(A) = P(B) \to A = B$
$|A| = n \to |P(A)| = 2^n$

Definition 2.13. An ordered n-tuple is a collection of elements where order matters, i.e., an ordered set.

Example 2.14. $(a, b) \neq (b, a)$

Example 2.15. $(a, a, b) \neq (a, b)$

Definition 2.16. The Cartesian product of two sets A and B, denoted by $A \times B$, is an unordered set that consists all ordered pairs of (a, b) such that a is in A and b is in B.

$$A \times B = \{(a, b) | a \in A \wedge b \in B\}$$

Example 2.17. $\{1, 2\} \times \{1, 3, 4\} = \{(1, 1), (1, 3), (1, 4), (2, 1), (2, 3), (2, 4)\}$

Example 2.18. $\{(1, 1), (1, 3), (1, 4), (2, 1), (2, 3), (2, 4)\} \times \{1, 2\} = \{((1, 1), 1), ((1, 3), 1), \cdots\}$

Definition 2.19. A function for A to B, denoted by $f : A \to B$, takes as input an element from set A and outputs an element from set B.

Definition 2.20. A function $f : A \to B$ is injective or one-to-one if every input maps to a distinct output, i.e., for an injective function f, $f(a) = f(b) \to a = b$

Example 2.21. $f(x) : \mathbf{R} \to \mathbf{Z}, f(x) = \lfloor x \rfloor$ is not injective.

Remark. Floor function $\lfloor x \rfloor$, ceiling function $\lceil x \rceil$.

Definition 2.22. A function $f : A \to B$ is surjective or onto if every element in B can be produced.

Example 2.23. $f(x) : \mathbf{R} \to \mathbf{Z}, f(x) = \lfloor x \rfloor$ is surjective.

Definition 2.24. A function f is bijective or one-to-one correspondence if it's both injective and surjective.

Example 2.25. $f(x) : \mathbf{Z} \to$ even integers, $f(x) = 2x$ is both injective and surjective, thus is bijective.

Definition 2.26. A sequence is a function from $\mathbf{N}$ to some set S.

Example 2.27. $f_0 = 0, f_1 = 1, f_2 = 2, f_3 = 3, f_4 = 5, f_5 = 8, \cdots$

Example 2.28. $1, 2, 3, 4, \cdots$

Example 2.29. $1, 4, 9, 16, \cdots$

Example 2.30. $f_n = f_{n-1} + f_{n-2},\ f_0 = 0,\ f_1 = 1$

Remark. Recurrence relations: recursive definitions of sequences.

Example 2.31. 3, 3, 3, 3, $\cdots$: $f_0 = 3,\ f_n = f_{n-1}$

Example 2.32. $f_n = 2n$: $f_n = 2 + f_{n-1}$

Example 2.33. $f_n = n^2$: $f_n = f_{n-1} + 2n - 1$

Example 2.34. $f_n = n + (-1)^n$: $f_n = f_{n-2} + 2$

Example 2.35. The polulation of world in 2010 is 6.9 billion, assume it grows at an annual rate of 11%. $f_n = 1.011 f_{n-1},\ f_0 = 6.9\text{billion}$

```
1 Function MergeSort(array A [1 : n])
2 if n == 1 then
3 |  return A
4 B = MergeSort (A [1 : n/2])
5 C = MergeSort (B [n/2 + 1 : n])
6 return Merge (B, C)
```

3. Lecture 3, Wednesday, January 22

Running time analysis: analyzing sorting algorithms. Look into details of merge sort and selection sort.

Merge sort is better. Why? It has less total operations than selection sort. Count up the number of operations, but some operations take longer.

"if I double the size of input, how much does the runtime increase?"

$n \to$ double

$20n \to$ double

$10n + 37 \to$ double

large runtime $Cn+$ smaller terms $\to$ double

$n^2 \to 4\times$

$n^4 \to 16\times$

$n^5 + 10n^3 + 21 \to 32\times$

Polynomial times Cn^d+ smaller terms, increase by a constant factor 2^d

Selection fort, number of operations $\simeq n^2$

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \simeq n^2$$

Optimize this algorithm: Check if the list is sorted (n), it not, run Selection sort (n^2). We say n^2 because we analyze worst-case scenario.

In the average case, it'll be the same anyway (in this case it is true). $\frac{n^2+n}{2} \simeq n^2$. Must make guarantees. It's easier.

Analyzing growth rate of functions: look at worst case for large input sizes.

Definition 3.1. $f(n) = O(g(n))$ "big-oh": $\forall n \geq n_0, f(n) \leq cg(n) \leftrightarrow \frac{f(n)}{g(n)} \leq c$, for some constants c, n_0.

Example 3.2. $10n^3 = O(n^3)$

Example 3.3. $20n^2 + 13n + 5 = O(n^2)$

Example 3.4. $10n^3 = O(n^4) = O(n^\infty)$

Constant factors are important, but they don't affect the growth rate.

O is imperfect knowledge of a function's running time. "Algorithm A is at least $O(n^4)$" doesn't actually mean anything.

Definition 3.5. $f(n) = \Omega(g(n))$ "big-omega": $\forall n \geq n_0, f(n) \geq cg(n) \leftrightarrow \dfrac{f(n)}{g(n)} \geq c$, for some constants c, n_0.

Example 3.6. $10n^3 = \Omega(1) = \Omega(n^3)$

Example 3.7. $10n^2 + 10n + 2 = \Omega(n^2)$

Ω is imperfect knowledge as well. Neither is the best case nor worst case.
The optimized selection sort mentioned hereof is $O(n^2)$, and is $\Omega(n^2)$ as well.

Definition 3.8. $f(n) = \Theta(g(n))$ "big-theta" means $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

Θ means perfect knowledge. Not every function has a Θ.

4. Discussion 1, Wednesday, January 22

Set

- common set $\mathbf{Z}$, $\mathbf{Z}^+$, $\mathbf{N}$, $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{R}^+$, $\mathbf{C}$
- Operations $A \cap B$, $A \cup B$, $\overline{A}$, $A - B$

Cardinality
$|A|$ number of distance elements

Example 4.1. $|\{a, a, b, c\}| = 3$

Remark. set order or duplication do not matter
$\{a, b, c\} = \{b, c, a\} = \{a, a, b, c\}$

Ordered n-tuple

Example 4.2. $(a, b, c) \neq (b, c, a)$
$(a, b, c) \neq (a, a, b, c)$

Problem 4.3. Venn diagram

Remark. Operators within the same pair of parentheses have the same level of priority; union and intersection have the same level of priority. Calculate from the innermost pair of parentheses.

Power set $P(S)$
The set of all subsets of S

Problem 4.4. $|| = 0$
$|\{\}| = |\{\{\}\}| = 1$
$|\{\mathbf{Z}^+, 0, \mathbf{Z}^-\}| = 3$
$|\mathbf{Z}| = \infty$

Problem 4.5. Show that if A, B, and C are sets then $\overline{A} \cap B \cap C = \overline{A} \cup \overline{B} \cup \overline{C} \cup$

Proof
Pick arbitrary element from left set, prove it is in the right set (subset). Do the same to the other side, proves they are equal.
Arbitrary X: could be anywhere $\cdots$ test every case
Functions

Must map to ONE element
Injective (one to one)
Each one mapped to unique element. Can have extra elements on right.
Surjective (onto)
Each element is mapped. No extra on the right.
Bijective (perfect match)
Injective and surjective

Problem 4.6. $f : \mathbf{Z} \to \mathbf{O}$ where $f(x) = 2x + 5$ is bijective

Problem 4.7. $f : \mathbf{O} \to \mathbf{Z}^+$ where $f(x) = (x + 5)^2$ is neither

Sequence
$\mathbf{N} \to S$
Fibonacci Sequence
$f_0 = 0$, $f_1 = 1$, $f_n = f_{n-1} + f_{n-2}$

Problem 4.8. $a_0 = 1$, $a_n = 2a_{n-1}$, $a_n = 2^n$

Problem 4.9. $a_0 = -7$, $a_{n+1} = 10 - a_n$, $a_n = 5 - 12(-1)^n$

5. Lecture 4, Monday, January 27

Linear search $\Theta(n)$
Binary search $\Theta(\log n)$

Remark. $\log n$ means $\log_2 n$ in computer science, unless otherwise specified.

Selection sort $\Theta(n^2)$
Insertion sort $\Theta(n^2)$
Quick sort $\Theta(n^2)$ (worst case) $\Theta(n \log n)$ (Average case)
Merge sort $\Theta(n \log n)$
Radix sort $\Theta(kn)$ where k stands for the number of digits in the sorted numbers

Example 5.1. Two functions $f(n) = \log_{10} n$ and $g(n) = \log_2 n = 2 \log n$. Apparently $f(n) = O(g(n))$. True or false: $f(n) = \Theta(g(n))$.

True, since $\log_{10} n = \dfrac{\log_2 n}{\log_2 10}$.

Example 5.2. $\log^2 n = (\log_n)^2$. Apparently $\log_n = O(\log^2 n)$. True or false: $\log_n = \Omega(\log^2 n)$.

False, since $\log^2 n = \log n \cdot \log n$.

Example 5.3. It is true that $\log n = \Theta(\log n^2)$.

Example 5.4. Although for any given number n, the value of $10 \log^{100} n$ is larger than the value of $\dfrac{1}{100}\sqrt{n}$, the rate of growth of $10 \log^{100} n$ is smaller than that of $\dfrac{1}{100}\sqrt{n}$. The former is polylogarithmic ($\log^c n$) and the latter is polynomial (n^d). $\log^c n = O(n^d)$.

Example 5.5. Although for any given number n, the value of $100n^100$ is larger than the value of $1.01^{\frac{n}{100}}$, the rate of growth of $100n^100$ is smaller than that of $1.01^{\frac{n}{100}}$. The former is polynomial (n^d) and the latter is exponential ($c^{\frac{n}{e}}$). $n^d = O(c^{\frac{n}{e}})$, for $c > 1$.

Hierarchy of classifications of the rate of growth of a function:
Constant $\Theta(1)$
Polylogarithmic $O(\log^c n)$
Polynomial $O(n^c)$
Exponential is not polynomial
$c^{\log n}$ undefined

Example 5.6. $f_1 = n^n$ is exponential
$f_2 = \log^2 n$ is polylog
$f_3 = n^{1.0001}$ is polynomial
$f_4 = 1.0001^n$ is exponential
$f_5 = 2^{\sqrt{\log n}}$ is undefined
$f_6 = n \log^{1.0001} n$ is polynomial
Since $f_3 = n \cdot n^{0.0001}$ and $f_6 = n \cdot \log^{1.0001} n$, $n^{0.0001}$ is polynomial and $\log^{1.0001} n$ is polylog, $f_3 > f_6$ in terms of rate of growth.
Since $2^{\sqrt{\log n}} < 2^{\log_2 n} = n$, $f_5 < f_6$ in terms of rate of growth.
Comparing $f_2 = \log^2 n$ and $f_5 = 2^{\sqrt{\log n}}$: $\log f_2 = 2 \log(\log n)$ is polylogarithmic of $\log n$ and $\log f_5 = \sqrt{\log n}$ is polynomial of $\log n$, $\log^2 n = O(2^{\sqrt{\log n}})$, $f_2 < f_5$ in terms of rate of growth.
Comparison of the rates of growth: $f_2 < f_5 < f_6 < f_3 < f_4 < f_1$

Example 5.7. $f_1 = 2^{100n}$ is exponential
$f_2 = 2^{n^2}$ is exponential
$f_3 = 2^{n!}$ is exponential
$f_4 = 2^{2^n}$ is exponential
$f_5 = n^{\log n}$ is undefined
$f_6 = n \log n \log(\log n)$ is polynomial
$f_7 = n^{\frac{3}{2}}$ is polynomial
$f_8 = n \log^{\frac{3}{2}} n$ is polynomial
$f_9 = n^{\frac{4}{3}} \log^2 n$ is polynomial
For the polynomials, take out common factor n:
$f_6 = \log n \log(\log n)$ is polylog
$f_7 = n^{\frac{1}{2}}$ is polynomial
$f_8 = \log^{\frac{3}{2}} n$ is polylog
$f_9 = n^{\frac{1}{3}} \log^2 n$ is polynomial
For f_6 and f_8, take out common factor $\log n$:
$f_6 = \log(\log n)$ is polylog of $\log n$
$f_8 = \log^{\frac{1}{2}} n$ is polynomial of $\log n$
For f_7 and f_9, take out common factor $n^{\frac{1}{3}}$
$f_7 = n^{\frac{1}{6}}$ is polynomial
$f_9 = \log^2 n$ is polylog
For the polynomials, $f_6 < f_8 < f_9 < f_7$.
For the exponentials, take logarithms:

$f_1 = 100n$ is polynomial (linear)
$f_2 = n^2$ is polynomial (quadratic)
$f_3 = n! = O(n^n)$ is exponential
$f_4 = 2^n$ is exponential
For the exponentials, $f_1 < f_2 < f_4 < f_3$
For f_5 and f_1, take logarithms:
$f_5 = \log(n^{\log n}) = \log^2 n$ is polylog
$f_1 = 100n$ is polynomial
$f_6 < f_8 < f_9 < f_7 < f_5 < f_1 < f_2 < f_4 < f_3$

Running time (seconds, unless otherwise specified):

	$n = 10$	10^2	10^3	10^6
$\log n$	3×10^{-11}	7×10^{-11}	10^{-10}	2×10^{-10}
n^2	10^{-9}	10^{-7}	10^{-5}	circa 10^2
2^n	10^8	4×10^{11} years		

$f(n) = O(h(n))$, $g(n) = O(h(n))$. It is intuitively true that $f(n) + g(n) = O(h(n))$

Proof. $f(n) = O(h(n)) \Leftrightarrow f(n) \leq c_0 h(n)$, $\forall n > n_0$, $\exists c_0, n_0$, $g(n) = O(h(n)) \Leftrightarrow g(n) \leq c_1 h(n)$, $\forall n > n_1$, $\exists c_1, n_1$, then $f(n) + g(n) \leq (c_0 + c_1)h(n)$, $\forall n > \max(n_0, n_1)$. Set $c = c_0 + c_1$ and $n_2 = \max(n_0, n_1)$, then $f(n) + g(n) \leq ch(n)$, $\forall n > n_2$. □

6. Discussion 2, Tuesday, January 28

Problem 6.1. For the following sequences,
1) determine the next term in the sequence, and
2) find a recursive formula for S_n (the n^{th} term in the sequence)
a) $S = \{1, -1, 2, -2, 3, -3, \cdots\}$
$S_7 = 4$, $S_{n+2} = (-1)^{n+1}(|S_{n-2}| + 1)$, $S_1 = 1$, $S_2 = -1$
or $S_n = S_{n-1} + (-1)^{n+1} n$, $S_1 = 1$
or $S_n = S_{n-2} + (-1)^{n+1}$, $S_1 = 1$, $S_2 = -1$
b) $S = \{2, 3, 5, 9, 17, 33, \cdots\}$
$S_7 = 65$, $S_{n+1} = S_n + 2^{n-1}$, $S_1 = 2$
or $S_n = 2S_{n-1} - 1$, $S_1 = 2$

Problem 6.2. Show that if A, B, C are sets, then $\overline{A \cap B \cap C} = \overline{A} \cup \overline{B} \cup \overline{C}$
De Morgan's Law.
$x \notin A \wedge x \notin B \wedge x \notin C \Leftrightarrow x \in \overline{A} \wedge x \in \overline{B} \wedge x \in \overline{C} \Leftrightarrow x \in (\overline{A} \cup \overline{B} \cup \overline{C})$

Algorithm, Sorting, Merge Sort, Selection Sort, Runtime, Asymptotic Runtime, Growth Rate

Operations, O: $f(n) = O(g(n))$ where $|f(n)| \leq c|g(n)|$, Ω: $f(n) = \Omega(g(n))$ where $|f(n)| \geq c|g(n)|$, Θ: $f(n) = \Theta(n)$ where $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$, Time Complexity: amount of time required for an algorithm to solve a problem, Brute Force Algorithm

Example 6.3. $n^3 = \Omega(n^2)$

Example 6.4. $n! = \Omega(1)$

Example 6.5. $n^3 + 2n^2 - 1 = O(n^3) = \Omega(n^3) = \Theta(n^3)$

Example 6.6. $n^2 = O(2^n) \neq \Omega(2^n) \neq \Theta(2^n)$

Problem 6.7. Suppose you have 2 sets A and B, each with n elements consisting of real numbers in no particular order. Let C be the Cartesian Product of Sets A and B. An algorithm which takes as input A, B, and C, and outputs the furthest pair of real numbers (from origin $(0,0)$) from the set C.

```
1 Procedure FurthestPair(A [1...n], B [1...n], C [1...n^2][1...2])
2 max = 0
3 for i ← 1 to n^2 do
4 |   dist = sqrt(C[i][1]^2 + C[i][2]^2)
5 |   if dist > max then
6 |   |   max ← dist
7 |   |   FurthestPair = (C[i]_1, C[i]_2)
8 end
9 return FurthestPair
```

Runtime: $O(n^2)$

Problem 6.8. n^2 slots, fill in with numbers 1 to n^2, $\Theta(n^2!)$

Problem 6.9. Suppose you have functions f and g such that $f(n) = O(g(n))$. Are these statements True or False?

a) $f(n)^2 = O(g(n)^2)$ True
b) $2^{f(n)} = O(2^{g(n)})$ False
c) $x^2 + 4x + 5 = \Theta(x^2)$ False
d) $n \log n = \Theta(\log(n!))$ True

Problem 6.10. Suppose $f(n) = O(s(n))$, $g(n) = O(r(n))$, prove or disprove (by counter example):

a) $f(n) - g(n) = O(f(n) - r(n))$ False
e.g.: $f(n) = n^2 + 1$, $g(n) = n$, $s(n) = n^2$. $r(n) = n^2$
b) If $f(n) = O(g(n))$ then $f(n) + g(n) = O(g(n))$ False
e.g.: $f(n) = n + 2$, $g(n) = n^2$, $s(n) = n$, $r(n) = n^2$

7. Lecture 5, Wednesday, January 29

Problem 7.1. $f(n) = O(h(n))$, $g(n) = O(h(n))$, then $f(n)+g(n) = O(h(n))$. True or false: $\sum_{i=1}^{n} f(n) = O(h(n))$.

False. Set $f(n) = n$, $h(n) = n$, then $n = O(n)$, but $\sum_{i=1}^{n} n = n^2 \neq O(n)$.

Remark. The summation of $f(n)$ for any constant number c times $\sum_{i=1}^{c} = O(f(n))$, but in this case n is a variable, which makes the statement false.

Problem 7.2. $f(n) = O(s(n))$, $g(n) = O(r(n))$, True or false: $f(n)-g(n) = O(s(n)-r(n))$.

False. Set $f(n) = n^2$, $g(n) = 1$, $s(n) = n^2$, $r(n) = n^2 - 1$, then $n^2 = O(n^2)$, and $1 = O(n^2 - 1)$, but $n^2 - 1 \neq O(1)$

Remark. $f(n) \leq c_0 s(n)$ for $n > n_0$ and $g(n) \leq c_1 r(n)$ for $n > n_1$ can't lead to $f(n)-g(n) \leq c[s(n) - r(n)]$.

$f(n) - g(n) \leq c_0 s(n) - g(n) \geq c_0 s(n) - c_1 r(n)$. No way to yield a relation between the left side and the right side.

Problem 7.3. $f(n) = \Omega(s(n))$, $g(n) = \Omega(r(n))$, True or false: $f(n) - g(n) = \Omega(s(n) - r(n))$.

False. Set $f(n) = n + 1$, $g(n) = n$, $s(n) = n$, $r(n) = 1$, then $n + 1 = \Omega(n)$, and $n = \Omega(1)$, but $1 \neq \Omega(n - 1)$

Problem 7.4. $f(n) = \Theta(s(n))$, $g(n) = \Theta(r(n))$, True or false: $f(n) - g(n) = \Omega(s(n) - r(n))$.

False. Set $f(n) = n^2 + n$, $g(n) = n^2$, $s(n) = n^2 + 1$, $r(n) = n^2$, then $n^2 + n = \Theta(n^2 + 1)$, and $n^2 = \Theta(n^2)$, but $n \neq \Theta(1)$

Propositional Logic: The building block of proper logic is the proposition.

Definition 7.5. Proposition: a statement of fact which is either true or false.

Example 7.6. *36 is a prime number* is a proposition.
Spaghetti grows on trees is a proposition.
Go play Hearthstone is not a proposition.
I am bored by this class is a proposition.
The cake is a lie is a proposition.
$1 + 1 = 2$ is a proposition.
$x + 1 = 2$ is not a proposition. It is a *predicate*.
This statement is false is not a proposition. It is neither true nor false.

Example 7.7. p = *there is life on Mars.*
q = *there is life on Earth.*
$\neg p$ = *no life on Mars.*
$p \wedge q$ = *life on both*
$p \vee q$

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \oplus q$
T	T	F	T	T	F
T	F	F	F	T	T
F	T	T	F	T	T
F	F	T	F	F	F

$\neg$ is an unary operator. It only takes one argument.
Theoretically there are 16 binary operators, including $\wedge$, $\vee$, $\oplus$.

Definition 7.8. $p \to q$: $\neg p \vee q$, if p then q, p implies q, p only if q, q if p, q is necessary for p, p is sufficient for q

Example 7.9. `If Obama is president then` $2 + 2 = 4$ is true, true.
`If Obama is president then` $2 + 2 = 4$ is true, false.
`If Spaghetti grows on trees then` $2 + 2 = 4$ is false, true.
`If Spaghetti grows on trees then` $2 + 2 = 5$ is false, false.

Definition 7.10. Implication: $p \to q$

Definition 7.11. Converse: $q \to p$

Definition 7.12. Contraposition: $\neg q \to \neg p$

Theorem 7.13. *$p \to q$ and $\neg q \to \neg p$ are equivalent.*

Definition 7.14. $p \Leftrightarrow q$: p implies q and q implies p, p if and only if q, p iff q, $\neg(p \oplus q)$

p	q	$\neg p$	$\neg q$	$\neg q \to \neg p$	$p \to q$	$p \Leftrightarrow q$
T	T	F	F	T	T	T
T	F	F	T	F	F	F
F	T	T	F	T	T	F
F	F	T	T	T	T	T

Remark. In common English, people use *if* a lot but actually they mean *iff*.

Order of operation: first negation, then everything else.

Definition 7.15. A set of compound propositions is consistent if it is possible to assign True or False values to all atomic propositions such that all compound propositions are true.

Example 7.16. *You will get an A in this class* or *you will die trying*,
If *you die trying*, *you will (posthumously) get an A in this class*,
You will get an A in this class and not *die trying*
are consistent.

Example 7.17. *You will get an A in this class* or *you will die trying.*
If *you die trying*, *you will (posthumously) get an A in this class.*
You will get an A in this class.
are inconsistent.

Problem 7.18. Two doors, each has a sign. One prize behind one of the two doors. One sign is true, the other one is false. Sign 1: *There is a prize behind this door and no prize behind other.* Sign 2: *There is a prize behind one door and nothing behind the other.*

The prize is behind door 2. When sign 1 is true, sign 2 must be true, but there's only one true sign, then sign 1 is false and sign 2 is true. Since sign 1 is false, the prize is behind door 2.

$r =$ *There is a prize behind door 1.*
$s =$ *There is a prize behind door 2.*
$p =$ *Sign 1 is true.*
$q =$ *Sign 2 is true.*
Premise: $p \oplus q$
Sign 1: $p \Leftrightarrow (r \wedge \neg s)$
Sign 2: $q \Leftrightarrow (r \oplus s)$

8. Lecture 6, Monday, February 3

Example 8.1. *You cannot ride the roller coaster if you are under 4 feet tall unless you are older than 16 years.*

X unless Y: $(\neg Y) \to X$

If you are not older than 16, then (if you are under 4 ftm then you cannot rider), unless you are older than 16. $(\neg r) \to [q \to (\neg p)]$

Truth table:

p	q	r	$(\neg r) \to [q \to (\neg p)]$	$(\neg p) \to q$	$q \to (\neg p)$	$(\neg r) \to q$
T	T	T	T	T	F	T
T	T	F	F	T	F	T
T	F	T	T	T	T	T
T	F	F	T	T	T	F
F	T	T	T	T	T	T
F	T	F	T	T	T	T
F	F	T	T	F	T	T
F	F	F	T	F	T	F

Some equivalent expressions:
$(p \wedge q) \to r$
$(q \wedge \neg r) \to \neg p$
$q \to (\neg p \vee r)$

Propositional logic are digital logic are one-on-one correspondent. Every propositional logic expression has its unique digital logic expression, vice versa.

First order logic: propositional logic + predicates.

Definition 8.2. A predicate is a sentence with one or more variables with becomes a proposition when specific values substituted for variables.

We write predicate as a function.

Example 8.3. $P(n) = n$ *is an odd integer.*
$P(5) = 5$ *is an odd integer.*
$P(6) = 6$ *is an odd integer.*

You may need to specify *universe*.

Example 8.4. $P(x) = x$ *is larger than cake.* Universe: food items.
P(pie): *pie is larger than cake.*

Example 8.5. $P(x) = x$ *is mortal.* Universe: humans.
All humans are mortal: $\forall x P(x)$
There is a human who is mortal: $\exists x P(x)$
No humans are mortal: $\forall \neg P(x)$
There is a human who is not mortal: $\exists x \neg P(x)$

Example 8.6. $E(x) = x$ *is an elephant*
$P(x) = x$ *is pink.* Universe: everything.
$\forall x[E(x) \wedge P(x)]$: *everything is pink elephant.*
$\forall x[E(x) \to P(x)]$: *all elephants are pink.*
$\exists x[E(x) \wedge P(x)]$: *There is a pink elephant.*
$\exists x[E(x) \to P(x)]$: *There is something which is either pink or not an elephant.*

Remark. $\exists x[E(x) \to P(x)] = \exists x[\neg E(x) \vee P(x)]$

Example 8.7. *All humans are mortal* and *There is a human who is not mortal* are logically opposite.

There is a human who is mortal and *No humans are mortal* are logically opposite.

$\neg\forall x P(x) = \exists x \neg P(x)$
$\neg\exists x P(x) = \forall x \neg P(x)$

Remark. This is actually De Morgan's Law, since $\forall$ is the combination of a lot of $\wedge$, and $\exists$ is equivalent to $\vee$.

Example 8.8. $\neg\forall x[E(x) \to P(x)] = \exists x \neg[E(x) \to P(x)]$

Example 8.9. *All that glitters is not gold* literally translates to $\forall x[Gl(x) \to \neg Go(x)]$, but it actually means $\exists x[\neg Go(x) \wedge Gl(x)]$. Ambiguity of English.

Example 8.10. $L(x, y) = x$ loves y. Universe: people.
Everybody loves somebody: $\forall x \exists y L(x, y)$
Somebody loves everybody: $\exists x \forall y L(x, y)$
There is somebody whom everybody loves: $\exists y \forall x L(x, y)$
Everybody is loved by somebody: $\forall y \exists x L(x, y)$

9. Lecture 7, Wednesday, February 5

Example 9.1. $L(x, y) = x$ loves y. Universe: people.
Not *Everybody loves somebody = Somebody loves Nobody*:

$$\neg\forall x \exists y L(x, y) = \exists x \neg\exists y L(x, y) = \exists x \forall y \neg L(x, y)$$

Not *Somebody loves everybody = Everybody doesn't love somebody*:

$$\neg\exists x \forall y L(x, y) = \forall x \neg\forall y L(x, y) = \forall x \exists y \neg L(x, y)$$

Example 9.2. $has(x, y)$, $Universe(x) =$*dogs*, $Universe(y) =$*tails*
$\forall x \exists y has(x, y)$: *Every dog has a tail*
$\forall y \exists x has(x, y)$: *Every tail is on a dog*
$\exists y \forall x has(x, y)$: *Every dog shares a tail*
$\exists x \forall y has(x, y)$: *There is a dog which has every tail*

Example 9.3. $P(p, h) = p$ *lives in* h
$F(s, p) = s$ *is friends with* p
$Universe(p) =$*people*, $Universe(h) =$*residence halls*, $Universe(s) =$*students in 170*
There is a student in this class who is friends with at least one person from every residence hall: $\exists s \forall h \exists p[F(s, p) \wedge P(p, h)]$

Remark. *There exists a person s in this class such that for every residence hall there exists a person p who is friends with s.*

For every residence hall you can find a person who is friends with someone in this class= $\forall h \exists p \exists s[F(s, p) \wedge P(p, h)]$ is different. In this statement s can change, while in the previous (correct) statement s is fixed.

There exists a person s and *there exists a person p such that for every residence hall s is friends with p and p is from this residence hall* is a stronger statement. $\exists s \exists p \forall h[F(s, p) \wedge P(p, h)] \to \exists s \forall h \exists p[F(s, p) \wedge P(p, h)]$

Generally, the farther right the $\forall$ quantifier, the stronger the statement.

Example 9.4. $Q(x, y, z) : x + y = z$, Universe $= \mathbf{Z}$
$\forall x \forall y \exists z Q(x, y, z)$ True
$\exists x \exists y \forall z Q(x, y, z)$ False

$\exists z \forall x \forall y Q(x, y, z)$ False
$\forall z \forall x \forall y Q(x, y, z)$ True

Remark. Swapping two adjacent $\forall$ or $\exists$ does not change the statement

Example 9.5. $F(x, y) = x$ *is friends with* y
$\exists x \forall y \forall z[(F(x, y) \wedge F(x, z) \wedge (y \neq z)) \rightarrow \neg F(y, z)]$: *There is someone for whom all of his friends dislike each other.*

Definition 9.6. Argument: A sequence of statements starting with assumption (premises) and ending with a conclusion.

Definition 9.7. Valid: It is impossible for the premise to be true and the conclusion to be false.

Example 9.8. If *all roads lead to Rome* and *the 405 does not lead to Rome*, then *the 405 is not a road* is valid.

Example 9.9. If *all professors are absent-minded* and *Aaron is absent-minded*, then *Aaron is a professor* is invalid.

Remark. Any argument that has one contradiction in its premises is valid.

Rules of Inference:
Modus ponens: if p and $p \rightarrow q$ then q
Modus tollens: if $p \rightarrow q$ and $\neg q$ then $\neg p$
Hypothetical syllogism: if $p \rightarrow q$ and $q \rightarrow r$ then $p \rightarrow r$
Simplificaiton: if $p \wedge q$ then p
Resolution: if $p \vee q$ and $\neg p \vee r$ then $q \vee r$
Use rules of inference to construct more complicated arguments.
More on Table 1 on page 72.

Example 9.10. If $\neg$ *Sunny* $\wedge$ *cold, swim* $\rightarrow$ *sunny,* $\neg$ *swin* $\rightarrow$ *canoe,* and*canoe* $\rightarrow$ *home by sunset.*
Prove *home by sunset.*

Proof. 1. $\neg$ *Sunny* $\wedge$ *cold*: premise
2. *swim* $\rightarrow$ *sunny*: premise
3. $\neg$ *swin* $\rightarrow$ *canoe*: premise
4. *canoe* $\rightarrow$ *home by sunset*: premise
5. $\neg$ *sunny*: simplification 1
6. $\neg$ *swin*: modus tollens on 2, 5
7. *canoe*: modus ponens on 6, 3
8. *home by sunset*: modus ponens on 7, 4

□

10. Discussion 3, Wednesday, February 5

Homework problems

Problem 10.1. If $f(n) = O(g(n))$ then $\log f(n) = O(\log g(n))$. True.
$f(n) \leq cg(n)$ for some constant c, then $\log f(n) \leq \log cg(n) = \log c + \log g(n) \leq c' \log g(n)$

Problem 10.2. $f_a = \dfrac{n^3}{\log n} + n^2$, $f_b = 2n^3$, $f_c = \log^2 n$, $f_d = \sum_{i=1}^{n} \sum_{j=1}^{i} j$, $f_e = \log_3 n^2$, $f_f = 2^{\log n}$, $f_g = n^{\frac{8}{3}}$, $f_h = 1.01^n$

Simplify:

$$\sum_{i=1}^{n} \sum_{j=1}^{i} j = \sum_{i=1}^{n} \frac{i(i+1)}{2} = \sum_{i=1}^{n} \left(\frac{i^2}{2} + \frac{i}{2} \right) \leq \sum_{i=1}^{n} \frac{i^2}{2} = \frac{n(n+1)(2n+1)}{12} = \Theta(n^3)$$

$f_a \to \dfrac{n^3}{\log n} = o(n^3) = n^{\frac{8}{3}} \dfrac{n^{\frac{1}{3}}}{\log n} = \omega(n^{\frac{8}{3}})$, $f_b = 2n^3 = \Theta(n^3)$, $f_c = \log^2 n = \Theta(\log^2 n)$, $f_d = \sum_{i=1}^{n} \sum_{j=1}^{i} j = \Theta(n^3)$, $f_e = 2 \log_3 n = \Theta(\log n)$, $f_f = n = \Theta(n)$, $f_g = n^{\frac{8}{3}} = o\left(\dfrac{n^3}{\log n} \right)$, $f_h = 1.01^n$

$f_e < f_c < f_f < f_g < f_a < f_d = f_b < f_h$

Problem 10.3. If $f(n) = O(g(n))$ and $h(n) = O(f(n))$, then $g(n)^2 = \Omega\left(\sqrt{h(n)}\right)$ True.

$f(n) \leq c_1 g(n) \leq c_1 g(n)^2 \Rightarrow \sqrt{f(n)} \leq c_2 g(n)$

$h(n) \leq c_3 f(n) \Rightarrow \sqrt{h(n)} \leq c_4 \sqrt{f(n)} \leq c_5 g(n) \leq c_6 g(n)^2 \Rightarrow g(n)^2 = \Omega\left(\sqrt{h(n)}\right)$

11. Lecture 8, Monday, February 10

Definition 11.1. Propositional Equivalences $\equiv$

Example 11.2. $(p \to q) \equiv (\neg q \to \neg p)$ contrapositive

Example 11.3. $(p \to q) \not\equiv (q \to p)$ converse

Tautology: $p \vee \neg p \equiv$ True

Contradiction: $p \wedge \neg p \equiv$ False

Implication: $(p \to q) \equiv (\neg p \vee q)$

De Morgan's: $\neg(p \wedge q) \equiv \neg p \vee \neg q$, $\neg(p \vee q) \equiv \neg p \wedge \neg q$

More on Tables 6, 7, 8 on pages 27 - 28

Problem 11.4. If *you send me an email*(e), then *I will write my code*(c). If *you don't send me an email*, then *I will go to bed*(b). If *I go to bed*, then *I will wake up refreshed*(r).

Prove: if *I don't write my code*, then *I will wake up refreshed*$(\neg c \to r)$.

Proof. 1: $e \to c$ premise

2: $\neg e \to b$ premise

3: $b \to r$ premise

4: $\neg c \to \neg e$ contrapositive of 1

5: $\neg e \to b$ hypothetical syllogism on 2, 4

6: $\neg c \to r$ hypothetical syllogism on 5, 3

□

Problem 11.5. Prove: $\neg[p \vee (\neg p \wedge q)] \equiv (\neg p \wedge q)$

Proof. 1: $\neg[p \vee (\neg p \wedge q)]$ premise
2: $\neg p \wedge \neg(\neg p \wedge q)$ De Morgan's on 1
3: $\neg p \wedge (\neg\neg p \vee \neg q)$ De Morgan's on 2
4: $\neg p \wedge (p \vee \neg q)$ double negation on 3
5: $(\neg p \wedge p) \vee (\neg p \wedge \neg q)$ distributive on 4
6. False $\vee (\neg p \wedge \neg q)$ contradiction on 5
7. $(\neg p \wedge \neg q)$ identity on 6

□

Distributive: $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$, $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
Identity: $p\vee$ False $\equiv p$, $p\wedge$ True $\equiv p$
Domination: $p\wedge$ False $\equiv F$, $p\vee$ True $\equiv T$
Implication: $p \to q \equiv \neg p \vee q$
Double negation: $\neg\neg p \equiv p$
Tautology: $p\neg \not{p} \equiv$ True
Associative: $(p \vee q) \vee r \equiv p \vee (q \vee r)$, $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

Problem 11.6. Prove that $\neg p \to (p \to q) \equiv$ True is tautology.

Proof. 1. $\neg p \to (p \to q)$ premise
2. $\neg\neg p \vee (p \to q)$ implication on 1
3. $p \vee (p \to q)$ double negation on 2
4. $p \vee (\neg p \vee q)$ implication on 3
5. $(p \vee \neg p) \vee q$ associative on 4
6. True $\vee q$ tautology on 5
7. True domination on 6

□

Definition 11.7. Universal instantiation: If $\forall x P(x)$, then $P(c)$ for any c

Definition 11.8. Existential generalization: If $P(c)$, then $\exists x P(x)$

The two above are easy to use. The following two are dangerous.

Definition 11.9. Universal generalization: If $P(c)$ for any c, then $\forall x P(x)$

Definition 11.10. Existential instantiation: If $\exists x P(x)$, then $P(c)$, specific c. *There's no restriction on c.*

Example 11.11. Prove that $\forall x(P(x) \wedge Q(x)) \equiv (\forall x P(x)) \wedge (\forall x Q(x))$

Proof. 1. $P(c) \wedge Q(c)$ instantiation on the left side
2. $(\forall x P(x)) \wedge Q(c)$ generalization on 1
3. $(\forall x P(x)) \wedge (\forall x Q(x))$ generalization on 2

□

Problem 11.12. Prove that $\exists x(P(x) \vee Q(x)) \equiv (\exists x P(x)) \vee (\exists x Q(x))$

Proof. 1. $P(c) \vee Q(c)$ instantiation on the left side
2. $(\exists x P(x)) \vee Q(c)$
3. $(\exists x P(x)) \vee (\exists x Q(x))$
4. $P(c) \vee (\exists x Q(x))$

5. $P(c) \vee Q(d)$
6. If $P(c)$ is true, then $\exists x(P(x) \vee Q(x))$
7. If $Q(d)$ is true, then $\exists x(P(x) \vee Q(x))$

□

Problem 11.13. Prove that $\exists x(P(x) \wedge Q(x)) \not\equiv (\exists x P(x)) \wedge (\exists x Q(x))$

Proof. 1. $P(c) \wedge Q(c)$
2. $(\exists x P(x)) \wedge Q(c)$
3. $(\exists x P(x)) \wedge (\exists x Q(x))$
4. $P(c) \wedge (\exists x Q(x))$
5. $P(c) \wedge Q(d) \not\equiv \exists x(P(x) \wedge Q(x))$

□

Common fallacies:

Affirming the conclusion: $p \to q$ and q, therefore p

Denying the hypothesis: $p \to q$ and $\neg p$, therefore $\neg q$

Circular reasoning/Begging the question: trying to prove the conclusion q while inside the proof q is assumed to be true.

General proof tips:

1. Convince yourself by running through examples;
2. Keep deriving stuff, do not give up;
3. Check your proof very carefully.

E-mail address: yifeiyan@usc.edu