Propositional Logic

- Test for Consistency vs. Validity:

Consistency:

S1
S2
S3

NOT consistent

X X

* inconsistent & valid look the same.

Validity

P_1 P_2 / Q

$\neg Q$

Valid

X X

- Tautologies – sentences that are always true.

ex. $\neg(A \vee B) \leftrightarrow (\neg A \wedge \neg B)$

A	B	¬A	¬B	¬(A∨B)	↔	(¬A∧¬B)
T	T	F	F	F	T	F
T	F	F	T	F	T	F
F	T	T	F	F	T	F
F	F	T	T	T	T	T

- Logical Equivalence – 2 sentences are logically equivalent if they have the same truth table.

ex (above)

$\neg(A \vee B)$ & $(\neg A \wedge \neg B)$ are logically equivalent

- Translations:

Its the weekend iff its Saturday or Sunday. Since it's Saturday, it's the weekend.

$W \leftrightarrow (a \vee b)$

$a \rightarrow W$

W = weekend
a = Saturday
b = Sunday

- Check validity:

$W \leftrightarrow (a \vee b)$

✓ $\neg(a \rightarrow W)$

a

$\neg W$

W / $\neg W$

X / $\neg(a \vee b)$

$\neg a$

X

Valid

Test consistency:

✓ $A \rightarrow B$

✓ $A \rightarrow \neg B$

$\neg A$ / B

$\neg A$ $\neg B$ / $\neg A$ $\neg B$

X

A = F

B = T, F

Quantifiers

- Exists:

$\exists x$ Inference Rule

✓ $\exists x \ldots x \ldots$

$\ldots a \ldots$

* only done once

- For All:

$\forall x$ Inference Rule

$\forall x \ldots x \ldots$

$\ldots a \ldots$

* Repeat for all names in tree

- Syntax vs Semantics
  - Syntax – grammar, is it correctly formed? (use Formation tree)
  - Semantics – what does the sentence mean? (T/F in interpretation)

- Formation Tree

ex: P_a / P_b M_b

$P_b \wedge M_b$

$\exists x (P_x \wedge M_x)$

$P_a \rightarrow \exists x (P_x \wedge M_x)$

- Interpretations:

You need: Domain, Extensions of all names, Extensions of all predicates

Robinson Arithmetic

Q1. $\forall x \forall y \ (x \neq y \rightarrow Sx \neq Sy)$
Q2. $\forall x \ Sx \neq 0$
Q3. $\forall x \ (x \neq 0 \rightarrow \exists y \ Sy = x)$
Q4. $\forall x \ (x + 0 = x)$
Q5. $\forall x \forall y \ (x + Sy = S(x+y))$
Q6. $\forall x \ x \cdot 0 = 0$
Q7. $\forall x \forall y \ (x \cdot Sy = (x \cdot y) + x)$

Induction Rule of Inference:

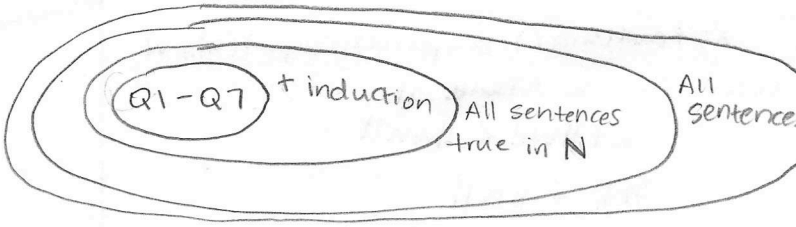
$\ldots 0 \ldots$

$\neg \ldots a \ldots$

$\ldots b \ldots$

$\neg \ldots Sb \ldots$

IF we can prove something from Q1–Q7, then that is true in every interpretation in which Q1–Q7 are true



Induction & Q1-Q7 exs

$\forall x\,(0 \cdot x) = 0$

1. $\neg \forall x\,(0 \cdot x) = 0$ — Negate conclusion
2. $\neg (0 \cdot a) = 0$ — line 1
3. $0 \cdot 0 = 0$ — Q6
4. $0 \cdot b = 0$ — [induction]
5. $\neg 0 \cdot Sb = 0$
6. $0 \cdot Sb = (0 \cdot b) + 0$ — Q7
7. $0 \cdot Sb = 0$ — lines 4, 6, Q4

Paradoxes & Complexity

- Richard-Berry Paradox: smallest integer not describable in less than 100 characters.
  - * it was just described in 64 characters ☺
  - * Need to define describable
- Complexity of n is length of the shortest program which prints n.
  $C(w) = K + \|n\|$ where K is length of program.
- Compressibility - if $C(n) = K + \|n\| \geq \|w\|$
  - * n = input, w = word
- * Prove incompressible words exist:
  words of length n
  95^n words
  $1 + 95 + 95^2 + \ldots + 95^{n-1}$ descriptions

$$\frac{95^n - 1}{95 - 1} < \frac{95^n}{94} << 95^n$$

[* ≤ 1.06% words are compressible]

Ex: estimate complexity of $\|ww\|$

$C(ww) \leq K + \|w\|$

X = readData(); output(x); output(x); w

ww is compressible if

$K + \|w\| < \|ww\|$

↳ $K < \|w\|$

- Church-Turing Thesis: if you think something is an algorithm, then it must be one.

Gödel's Incompleteness Theorem:

- the question "Is x compressible?" is undecidable.

Computably Enumerable - there is a program which prints out the elements of the subset.

- Computably enumeral sets are NOT always computable

S	N-S

* if S AND N-S are computably enumerable, then they are computable

N: Standard Model for language of arithmetic. connected to Robinson Arithmetic, induction, and Incompleteness Theorem, is recursively Enumerable

- Show in N a number can't be odd and even

$\varphi(x) = (\neg \exists_y (y+y=x) \vee \neg \exists_y (S(y+y)) = x)$

* Prove $\forall x\,\varphi(x)$ *show 1 branch

1. $\varphi(0)$ — Q2
2. $\neg \forall x\,\varphi(x)$ — Negate conclusion
3. $\neg \varphi(a)$ — 2
4. $\varphi(b)$
5. $\neg \varphi(Sb)$ — [induction, 1, 3]
6. $\neg \exists_y (y+y=b)$ — 4 *rec splits $\neg \exists_y (S(y+y)) = b)$
7. $\forall y\,(y+y \neq b)$ — 6
8. $\exists_y\, y+y = Sb$ — 5
9. $\exists_y\, S(y+y) = Sb$ — 5
10. $S(c+c) = Sb$ — 9
11. $c+c = b$ — 10, Q1
12. X — 7, 11