# Next-Next-Gen Notes

Object-Oriented Maths

Dark JP

March 28, 2018

Model Theory: semantics; Proof Theory: syntax

# 1 Kleene

## 1.1 Linguistic considerations: formulas

undefined terms: *lolm2k*

(1)

paradox: logic in terms on logic; solution: compartmentalize logic within "languages" (2)

object language/logic: the particular logic to be studied (3)

observer's language/logic: the logic used in studying the object language/logic (4)

sentences - declarative: a proposition; interrogative: a question; imperative: a command (5)

assume that object languages have a class of declarative sentences which serves as the building blocks, - (6)

and other sentences can be built from them by certain operations which are called "formulas" (A, ..., O) (7)

a language has "prime formulas"/"atoms" (P, ..., Z) which are distinct sentences that don't change meanings (8)

a language has 5 operations for building "composite formulas"/"molecules", - (9)

and these are - $\sim$: equivalence; $\supset$: implication; $\&$: conjunction; $\vee$: disjunction; $\neg$: negation (10)

(P, ..., Z) represent distinct prime formulas; (A, ..., O) represent formulas (11)

operator precedence: $\sim, \supset, \&, \vee, \neg, ..., (\_)$, where the higher ranks are evaluated first, same ranks right first (12)

the "scope" of an operator is the parts of the formula where it acts upon (13)

## 1.2 Model theory: truth tables, validity

undefined terms: *lolm2k*

$$ \tag{14} $$

this chapter discusses the system of logic called classical logic (15)

different systems of logic are conceptually equally possible, but classical logic is the simplest (16)

classical logic: assumes that atom/declarative sentence/proposition can either be true or false, but not both (17)

do truth table for: $\sim, \supset, \&, \vee, \neg$ (18)

"valid"/"identically true"/"tautology" formulas evaluate to true independent of its prime formula values (19)

### 1.3 Model theory: the substitution rule, a collection of valid formulas

undefined terms: $lolm2k$

$$ \tag{20} $$

start thm 1, whatever logic done with atoms, you can swap with formulas (21)

Note: Operators (op)s preserve type; Relations (rel)s return truths; include setOps; fix

## 2 Logic and Set Theory

### 2.1 D: Logical Truths and Operators

undefined terms: $:=, =, (\_), , , `, .,$

$$ \tag{22} $$

$$ truth[t][] :=_{or} \left( \begin{cases} t = T \\ t = F \end{cases} \right) \tag{23} $$

$$ operatorLogic[\odot][x,y] :=_{and} \left\{ \begin{aligned} (truth[x][]) \\ (truth[y][]) \\ (truth[x \odot y][]) \end{aligned} \right) \tag{24} $$

$$ operatorOR[\vee][x,y] :=_{1} \big(truth[x][]\big),_{1} \big(truth[y][]\big),_{1} \left( truth[x \vee y][] = \begin{cases} F & x = F, y = F \\ T & x = F, y = T \\ T & x = T, y = F \\ T & x = T, y = T \end{cases} \right) ._{1} \tag{25} $$

$$ operatorAND[\wedge][x,y] :=_{1} \big(truth[x][]\big),_{1} \big(truth[y][]\big),_{1} \left( truth[x \wedge y][] = \begin{cases} F & x = F, y = F \\ F & x = F, y = T \\ F & x = T, y = F \\ T & x = T, y = T \end{cases} \right) ._{1} \tag{26} $$

$$operatorNOT[\neg][x] :=_1 \big(truth[x][]\big),_1 \left(truth[\neg x][] = \begin{cases} T & x=F \\ F & x=T \end{cases}\right)._1 \tag{27}$$

---

$$operatorXOR[\veebar][x,y] :=_1 \big(truth[x][]\big),_1 \big(truth[y][]\big),_1 \left(truth[x \veebar y][] = \begin{cases} F & x=F, y=F \\ T & x=F, y=T \\ T & x=T, y=F \\ F & x=T, y=T \end{cases}\right)._1 \tag{28}$$

---

$$operatorIF[\implies][x,y] :=_1 \big(truth[x][]\big),_1 \big(truth[y][]\big),_1 \left(truth[x \implies y][] = (\neg x) \vee y = \begin{cases} T & x=F, y=F \\ T & x=F, y=T \\ F & x=T, y=F \\ T & x=T, y=T \end{cases}\right)._1$$

$$\text{\# a counterexample cannot follow from a false precedence, thus the conditional cannot be false} \tag{29}$$

---

$$operatorOIF[\impliedby][x,y] :=_1 \big(truth[x][]\big),_1 \big(truth[y][]\big),_1 \left(truth[x \impliedby y][] = (\neg y) \vee x = \begin{cases} T & x=F, y=F \\ F & x=F, y=T \\ T & x=T, y=F \\ T & x=T, y=T \end{cases}\right)._1 \tag{30}$$

---

$$\begin{aligned} &operatorIIF[\iff][x,y] :=_1 \big(truth[x][]\big),_1 \big(truth[y][]\big),_1 \\ &\left(truth[x \iff y][] = (x \implies y) \wedge (y \implies x) = \begin{cases} T & x=F, y=F \\ F & x=F, y=T \\ F & x=T, y=F \\ T & x=T, y=T \end{cases}\right)._1 \end{aligned} \tag{31}$$

P

## 2.2 P: Boolean Algebra

## 2.3 Predicates, Sets, Tuples

$arg\_(\_), set, \in, \{\_\},$

$$predicate[P][] := truth[P(v_{free})][] \tag{43}$$

---

$$\begin{aligned} universalQuantifier[\forall][P] &:=_1 (predicate[P][]),_1 \\ &(\forall_{x_{free}}(P(x_{free})) = P(y_{free}))._1 \end{aligned} \tag{44}$$

---

$$existentialQuantifier[\exists][Q,P] := (\exists_{arg_x(Q(x))}(P(x)) = \neg\forall_{arg_x(Q(x))}(\neg P(x))) \tag{45}$$

---

$$uniquenessQuantifier[\exists!][Q,P] := (\exists!_{arg_x(Q(x))}(P(x)) = \exists_{arg_x(Q(x))}(P(x) \wedge \neg\exists_{arg_y(Q(y))}(P(y) \wedge \neg(y=x)))) \tag{46}$$

---

$$relationSetEq[=][X,Y] := (\forall_{arg_z(z \in X \vee z \in Y)}(z \in X \wedge z \in Y)) \tag{47}$$

$$operatorIntersection[\bigcap][X] := (z \in \bigcap(X) \iff \forall_{x \in X}(z \in x)) \tag{48}$$

$$operatorUnion[\bigcup][X] := (z \in \bigcup(X) \iff \exists_{x \in X}(z \in x)) \tag{49}$$

$$orderedPair[<x,y>][] == <x,y> = <a,b> iff x = a and y = b == \{\{x\},\{x,y\}\} \tag{50}$$