Next-Next-Gen Notes

Object-Oriented Maths

Dark JP

March 28, 2018

Algebra: structure; Calculus: formal manipulations

1 Tourlakis

1.1 Basic Logic

(1)

proof theory (it studies the structure, properties, and limitations of proofs) (2)

model theory (it studies the interplay between syntax semantics – (3)

by looking at the algebraic structures where formal languages are interpreted) (4)

recursion theory (or computability, which studies the properties and limitations of algorithmic processes) (5)

set theory; 4 areas that consist modern mathematical logic (6)

this book is good, but we need to bootstrap propositional calculus and naive set theory before p.5 (7)

1.2 Bootstrap

$$0 \tag{8}$$

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: $:=, =, (\_), , , ‘, .,$

(9)

$$truth[t][] :=_{or} \left(\begin{cases} t = T \\ t = F \end{cases} \right) \tag{10}$$

$$operatorLogic[\odot][x,y] :=_{and} \begin{Bmatrix} (truth[x][]) \\ (truth[y][]) \\ (truth[x \odot y][]) \end{Bmatrix} \tag{11}$$

$$operatorOR[\vee][x,y] :=_1 (truth[x][]) ,_1 (truth[y][]) ,_1 \left(truth[x \vee y][] = \begin{cases} F & x=F, y=F \\ T & x=F, y=T \\ T & x=T, y=F \\ T & x=T, y=T \end{cases} \right) ._1 \tag{12}$$

$$operatorAND[\wedge][x,y] :=_1 (truth[x][]) ,_1 (truth[y][]) ,_1 \left(truth[x \wedge y][] = \begin{cases} F & x=F, y=F \\ F & x=F, y=T \\ F & x=T, y=F \\ T & x=T, y=T \end{cases} \right) ._1 \tag{13}$$

$$operatorNOT[\neg][x] :=_1 (truth[x][]) ,_1 \left(truth[\neg x][] = \begin{cases} T & x=F \\ F & x=T \end{cases} \right) ._1 \tag{14}$$

$$operatorXOR[\veebar][x,y] :=_1 (truth[x][]) ,_1 (truth[y][]) ,_1 \left(truth[x \veebar y][] = \begin{cases} F & x=F, y=F \\ T & x=F, y=T \\ T & x=T, y=F \\ F & x=T, y=T \end{cases} \right) ._1 \tag{15}$$

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

a counterexample cannot follow from a false precedence, thus the conditional cannot be false (16)

$$operatorOIF[\Longleftarrow][x,y] :=_1 (truth[x][]) ,_1 (truth[y][]) ,_1 \left(truth[x \Longleftarrow y][] = (\neg y) \vee x = \begin{cases} T & x=F, y=F \\ F & x=F, y=T \\ T & x=T, y=F \\ T & x=T, y=T \end{cases} \right) ._1 \tag{17}$$

$$operatorIIF[\Longleftrightarrow][x,y] :=_1 (truth[x][]) ,_1 (truth[y][]) ,_1 \left(truth[x \Longleftrightarrow y][] = (x \Longrightarrow y) \wedge (y \Longrightarrow x) = \begin{cases} T & x=F, y=F \\ F & x=F, y=T \\ F & x=T, y=F \\ T & x=T, y=T \end{cases} \right) ._1 \tag{18}$$

P

2.2 P: Boolean Algebra

2.3 Predicates, Sets, Tuples

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

$$predicate[P][] := truth[P(v_{free})][] \tag{30}$$

$$universalQuantifier[\forall][P] :=_{1} (predicate[P][]),_{1} \\ (\forall_{x_{free}}(P(x_{free})) = P(y_{free})).{}_{1} \tag{31}$$

$$existentialQuantifier[\exists][Q,P] := (\exists_{arg_x(Q(x))}(P(x)) = \neg\forall_{arg_x(Q(x))}(\neg P(x))) \tag{32}$$

$$uniquenessQuantifier[\exists!][Q,P] := (\exists!_{arg_x(Q(x))}(P(x)) = \exists_{arg_x(Q(x))}(P(x) \wedge \neg\exists_{arg_y(Q(y))}(P(y) \wedge \neg(y = x)))) \tag{33}$$

$$relationSetEq[=][X,Y] := (\forall_{arg_z(z \in X \vee z \in Y)}(z \in X \wedge z \in Y)) \tag{34}$$

$$operatorIntersection[\bigcap][X] := (z \in \bigcap(X) \iff \forall_{x \in X}(z \in x)) \tag{35}$$

$$operatorUnion[\bigcup][X] := (z \in \bigcup(X) \iff \exists_{x \in X}(z \in x)) \tag{36}$$

$$orderedPair[<x,y>][] == <x,y> = <a,b> iff x = a and y = b == \{\{x\},\{x,y\}\} \tag{37}$$