

Exercise 1 :

Ans) Let playing cricket = P

Watching cricket = W

Reading cricket = R

Then

$$\begin{array}{l} \neg P \rightarrow W \\ \neg W \rightarrow R \end{array} \quad \left. \begin{array}{l} \\ \end{array} \right\}$$

and can't do 2 activities at same time.

so there, 4 case W, P, R, W \wedge P \wedge R

case 1 \Rightarrow W \Rightarrow $\neg P, \neg R$

It can be since it satisfies 3 proposition.

case 2 \rightarrow P, $\neg W, \neg R$

It can't be, since it contradicting 2 statement

case 3 \rightarrow R, $\neg P, \neg W$

It can't be, since it contradicting 1 statement

case 4 \rightarrow P \wedge R \wedge W

It doesn't contradicting anything.

So

either W or W \wedge R \wedge P don't

Exercise 2

This statement can be prove by induction. (meta).

so taking base case as length 1, It can only "(" or any other alphabet excluding ')'. Now taking first character "(", For remaining segment "(" can be come atleast ones. due definition a st

If S=2 it means the segment is the entire propositional formula,

the, mainly "i" & "j" in must be equal by definition wif in PL.

$$\check{\vee}(\phi) \leftarrow \check{\vee}(o) = T \downarrow$$

Ex 3 : To show that an arbitrary function $\tilde{\vee}$ from atomic formulas to {T, L} can be extended to valuation \tilde{V} from wif to {T, L}

consider wif P, You can extend arbitrary function $\tilde{\vee}$ as

1) If P is an atomic, then $V(P) = \tilde{V}(P)$, (from question)

2) If P is complex (having logical connective and other alphabet)

$V(P)$ can be define by

a) for negation $(\tilde{\forall} a) \check{\vee}(P) = \neg \check{\vee}(a)$

b) for binary connective $a, b \quad V(P) = V(a) \tilde{J} V(b)$

c) for $\exists n a$, $V(P) = T$ if only if $V(a) = T$ for all

otherwise $V(P) = L$

d) for $\exists x a$, $V(P) = T$ if exists $V(a)$ for all

Ex.4 Polish language and propositional logic

Let construct map from wff in propositional logic
to polish language as

wff(PL) \rightarrow Polish,

- 1) for atomic forms (P), we represent (P) in polish
- 2) for complex wff,
 - 1) for unary operator $\neg(P)$, we define $\neg(P)$ in polish
 - 2) for binary operator by \square , wff $P \square Q$
where P, Q wff, we represent $\square P Q$

base case

length $n=1 \Rightarrow P$ is atomic form

From Polish to wff(PL)
 \Rightarrow start from left most right-associative part
 \Rightarrow it atomic formula, it represent as wff
 \Rightarrow if it alphanum \Rightarrow our operator
 \Rightarrow if it having next & equal left higher
operator.
 \Rightarrow if it using next one is open

Induction, length $n>1$

If $P = \neg Q$ defined by $\beta = \neg(\bar{Q})$

\bar{Q} exist & length $q=n-1$ may symbol value

for $P = \square_1 Q_1 \square_2 Q_2$

If P has 2 interpretations

$P = \square_2 r_1 v_2$ for some $r_1 = Q_2$

$v_1 = Q_1$

then $\square_1 = \square_2$

by this we can see the uniqueness of wff

Value's

atomic form : -1

unary operator \neg : 0

binary operator ($\vee, \wedge, \rightarrow, \leftrightarrow$) : 1

for wft, the sum should be equal to (-1)

there 3 case

i) Atomic form

$$\text{sum} = -1$$

ii) $\neg P$

$$\text{sum} = 0 + (-1) = -1$$

iii) $P_1 P_2$

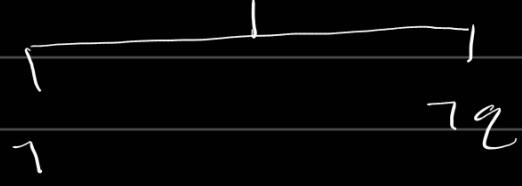
$$\text{sum} = (+(-1)) + (-1) = -1$$

thus proved sum is (-1) wft.

Prefix p has value ≥ 0

proof

case i) $\neg P \Rightarrow$ Prefix P



4) Let Polish language defined by

→ Alphabet :- same in PL, no perfix

→ braces :- wst in pol

→ P

→ $\neg P$ $P \rightarrow Q$ is wst

→ $\Box PQ$ $\neg PQ$ is wst

Constructing a map from polish to PL q
which is unique, we need to show there exist
map b/w

① $P \rightarrow (P)$

② $\neg P \rightarrow \neg P$

③ $\Box PQ \rightarrow P \Box Q$

where P, Q are wst in PL, which need
to define properly.

First

we assign values to Polish variable as follows
atomic form = -1

unary operat = 0

Binary ($\vee, \wedge, \rightarrow, \leftrightarrow$) :-

and state that string is wst in polish if

* atomic form (P)

* If the sum of ϕ (ϕ is the string) is
-1 and the sum(-1) only when string
ends

eg $\wedge \wedge A B C \wedge B$ is not

part of even thou it sum (-1), since its
sum break after 5 al letter.

(This to check whether something Polish
are not with left rule)

* If P, Q are wst

$\neg P, \Box PQ$ are wst

are a wst o & n, \exists a unique wst

length $n = 1$ base m -

P is a factor of

$\Rightarrow \exists$ unique $\beta = P$

Induction

length $n > 1$

\rightarrow first character can't be an atom \in

$\rightarrow P \sqcap q$ or $P = P \sqcap q \sqcap$

If sum is obtained by taking value of an unary character to be $n-1$ and sum of individual's character by assigning

then prefix pos value ≥ 0

Cases

$$\text{Sum}(P) = -1$$

$$\text{Sum}(T) = 1$$

$$\text{Sum}(Tq) = 1 + \text{Sum}(q) \geq 0 \quad ? \text{ prefix}$$

$$\text{Sum}(Tp) = 1 + (-1) \geq 0$$

$$\text{Sum}(Tpq) = 1 + (-1) + \text{Sum}(q) \geq 0$$

$$\text{Sum}(q) \geq 0$$

$$\text{Sum}(Tq) \geq 0$$

convert polish language \rightarrow PL

take all the subset of string which in the form $Tp q$, where the

$T \rightarrow$ binary connect and the prefix of p is 1 and prefix of q is 0

then this block be write some (A_i)

by do this we get new string.

again do this process until we get one block $\rightarrow T(A_i)(A_j)$

After that write it as $(A_i) T(A_j)$

Open it form writing PL

eg : $\neg A \wedge \neg A \quad \neg \rightarrow e \quad \vee y \Leftrightarrow \neg z \ L$

$\neg A \wedge \neg A \quad \neg \rightarrow e \left(\begin{matrix} \neg y & x_1 \\ x_2 \end{matrix} \right)$

$\neg A \wedge \neg A \left(\neg \rightarrow e \ x_1 \right)$

$\neg \left(\begin{matrix} \neg A & x_1 \\ x_2 \end{matrix} \right)$

$\neg \left(\neg A \wedge x_3 \right)$

$\neg \left(\neg A \left(e \rightarrow x_2 \right) \right)$

$\neg \left(\neg A \left(e \rightarrow (\neg y \vee x_1) \right) \right)$

$\neg \left(\neg A \left(e \rightarrow (\neg y \vee (\neg z \Leftrightarrow L)) \right) \right)$

By this we can see that every wff /
in Polish here unique wff in Proposition.

$$\lambda(\rightarrow C_7(\lambda e^{\lambda gh})\ell)$$

() → 10 - 1 → 1 - 1 - 1

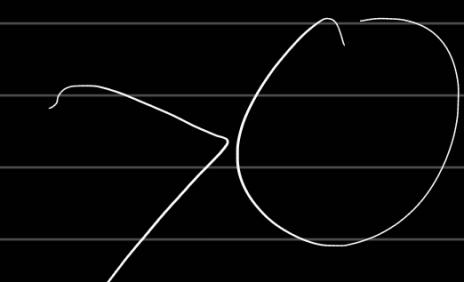
$\lambda \rightarrow C_7 \lambda e^{\lambda gh} \ell$

$$((< \lambda (\lambda e \rightarrow ((g \wedge h) \vee \ell))$$

$$((< \lambda \forall (e \rightarrow (g \wedge h) \vee \ell))$$

o ~ 1 0 ~ 1 0

$\rightarrow \wedge \rightarrow \wedge \vee$

$$\neg \vdash (\wedge AB) \vdash (\vee A \neg B)$$
$$\neg (\wedge P_1 \neg P_2) \vdash (\neg A \neg B)$$
$$\neg P_3$$
$$\vdash (P_1 \wedge \neg P_2)$$
$$\vdash ((A \wedge B) \wedge \vdash (\vee \neg B))$$

$$\vdash \wedge ((\vdash \neg A \neg B) \wedge (\vdash \neg A \neg B))$$
$$\vdash \wedge (\wedge AB) \vdash P_1 P_2$$
$$\text{zeta}$$

$\rightarrow A(A(K)P_1 P_2 P_3)$
 $\rightarrow (A(A(P_4)P_3$

$\rightarrow A(A \rightarrow (B C) D E)$

q

T. pq

)

g ()
- / (D PZ)

P → ah

T a

D

J M

Dg

D P Z

D op Z

D op Z

D P Z

()
sum (x)
z 0

(J P Q)

J P Q

J R S

J T U P Q

(J T U P Q)]

