16/7/22 FOL:-

→ Object:-

1. John
2. Richard
3. crown
4. left leg (John)
5. " (Richard).

→ Relationship

1. Binary :-

Two object

⇒ Brother.

↳ {< Richard, John>, < John, Richard>}

⇒ ONHead

↳ {< the crown, John>}

→ unary Relation or properties:-

1. Person
   - John
   - Richard
2. king → John.

→ Function :- [Try to map, single relation real set has [2n]]

① Left leg
→ < King Richard → left leg of Richard >
→ < King John → left leg of John >.

→ Syntax & Semantics of FOL :-

① Syntax :-

~~i) Terms~~
↓ It is used to represent object.

(1) Symbols :- (three)
1. constant symbol
2. Predicate "
3. Function "

with 'CAP' letter

→ It is used to represent the given domain [John, Richard]

→ It is used to Relation in the given domain [Brother, OnHead, Person, King]

→ It is used to represent the functions [left leg].

② Term :- Logical rep | Single Term or Complex

↳ It is used to rep the object

↳ It is used to rep the constant symbol

↳ It is called as term.

* We cannot use the separation constant symbol.

i) → John, Richard → simple

ii) → Left leg (John) → complex.

③ ~~Predicate~~ ~~Symbol~~ : Atomic Sentence :- | $A \& B \rightarrow ST$ | $A \& B \rightarrow CT$

i) Brother (John, Richard)

simple

simple term

simple term

It will be formed by predicate and terms ~~simple~~ ~~sentence~~

ii) Atomic sentence → complex term (function)

Married (Mother (John), Father (Richard))

→ How to Evaluate :- T or F

will take from the environment

(4) Complex Sentence:-

* Complex sentences will be framed by the combining the atomic by logic connectors.

[And, Or, Not, →, ⇔]

Ex.

¬ King(Richard) ⇒ King(John)

(5) Quantifiers:-

Used to rep ~~Atoms~~ Atoms of all by sum

There are two types:-

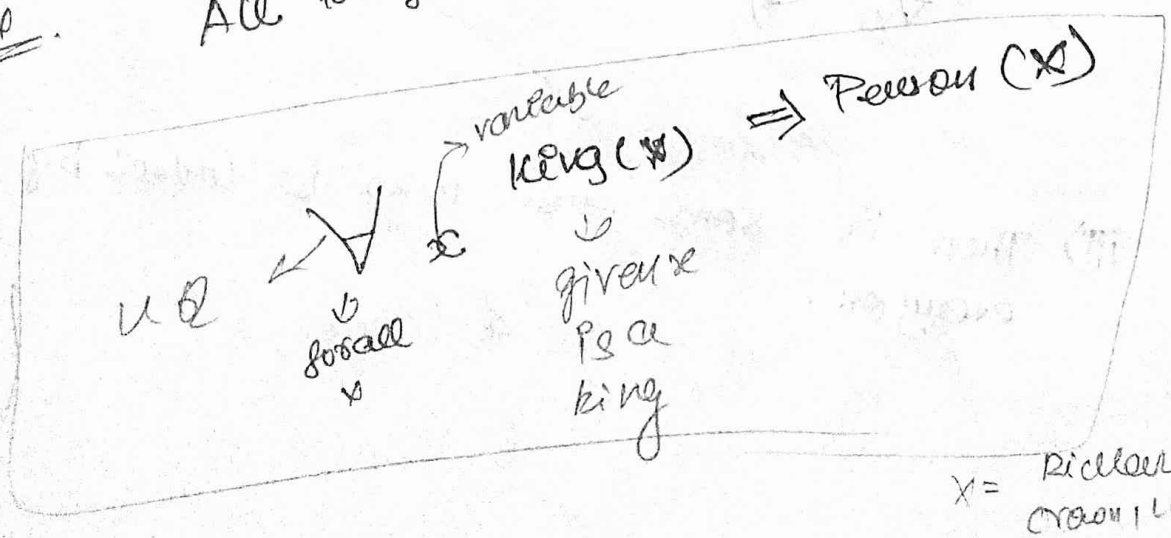
i) Universal (All) ∀

ii) Existential (some) ∃

i) Universal quantifier:-

Instead of using one by one, we can use the universal quantifier to rep all the obj with some property.

Ex. All kings are person.



x = Richard, John
crown, Left leg J, Left leg R.

ii) Existential Quantifier :- ∃

at least one object identified

If there exists, one of the crown and should be on king of John (crown)

$\exists x$ Crown (x) ∧ On Head (x, John)

→ Nested Quantifiers :- => Two are "same one"

combined of uni or Exi

Same type

Ex i) $\forall x \forall y$ Brother (x, y) ⇒ sibling (x, y)

variable

value. $\forall x, y$ Brother (x, y) => sibling (x, y)

ii) diff type :-

Every body loves some body.

$\forall x \exists y$ loves (x, y)

↓ Ever ↓ some body

iii) There is some one who is loved by every one.

$\exists x \forall y$ loves (x, y)

→ Conn

Ex i)

How

ii) E

→ Connection with ∀, ∃ :-

Ex i) Everyone dislike the king

$\neg \forall_x$ like (x, king)

or : ↑ NOT

$\forall_x$ (¬) like (x, king)

↓

How to replace with Existen Quan

$\neg \exists_x$ like (x, king)

$\forall_x \neg$ ⇒ $\neg \exists x$

ii) Everyone likes Programmes :-

$\forall_x$ likes (x, Programmes)

⇓

13/4/22. Unit - 3. AI

→ Situation Calculus :-

↓

* Replace Time by situation

Time ⟷ Situation

* There will be something the Initial state → S_0 to the DOS

* I am applying action, Result (S_0, a)

Result (S_0, a) ⇒ Situation.

* From the outcome situation is equal, it is possible if and if

$S_0 = S_1 \wedge a = a_1$ ✓

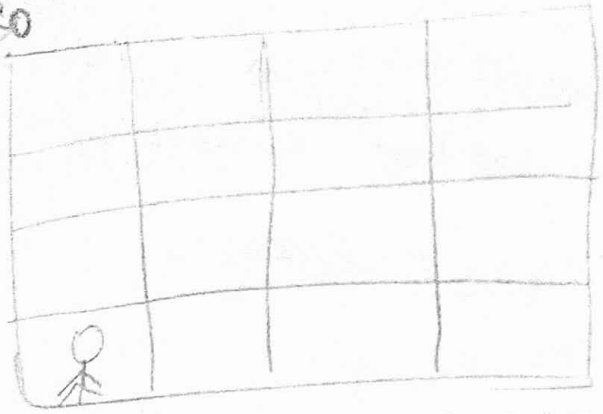
Result (S_0, a) = Result (S_1, a)

iff

$S_0 = S_1 \wedge a = a_1$

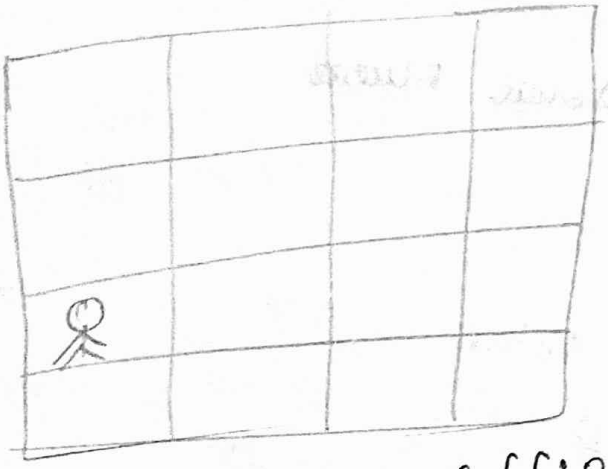
Eg.

S0

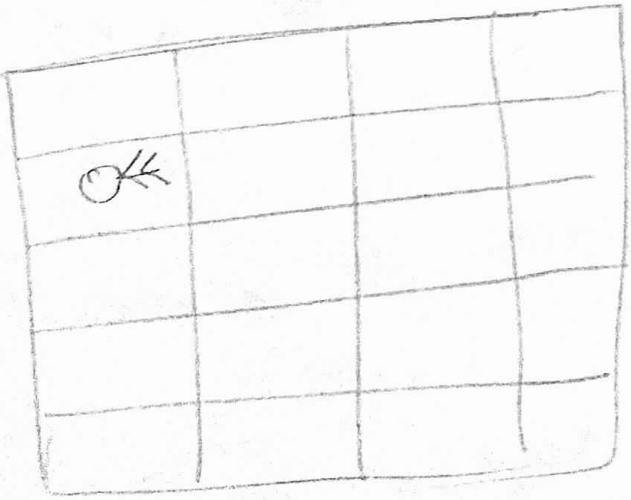


action

↓ Result (S0, forward)



↓ Result ((S0, forward), Turn right).



→ **Fluent** :-

* Given relation or function, if it changes from one situation to another or next situation is known Fluent.

→ **Two types** :-

① Relational Fluent

② Functional Fluent.

i) **Relational Fluent** :-

* Given relation :- Eg

At (x, s, l) location.

x ↓ object → location (l)

s → sit.

at = function(s) next situation

if x is in l ✓

s ⇒ next sit

② **Functional** :-

Location (x, s) = l.

→ Axiom :-

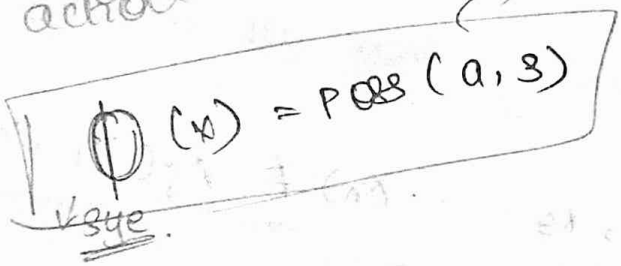
↓

Telling, what will be happening to the action when fluent is Process.

⇒ Types :-

① Possibility Axiom.
② Successor state "
③ Unique Action "

① What are the possible actions to take



action a will be taken for the (s).

Eg :- With

Have (Agent, s) ∧ Have (Agent, Arrow, s)
⇒ Agent (shoot, s).

② Status of the Fluent.

Eg :-

Holding (Agent, g, Result (s, a))

↓ selection ↓ given ↓ gold ↓ after holding

⇕

a = Grab(g) @ V

(or)

Holding (Agent g, s) ∧ a ≠ Release (g))

Overall:-

Holding (Result, g, Result (s, a))

⇕

a = Grab(g) ∨ (Holding (Agent, g, s) ∧ a ≠ Release (g))

(3) * It is describing that all actions are unique.

$A_i(x_1, x_2, x_3 \ldots x_n) \neq A_j(y_1, y_2 \ldots y_n)$

↓ diff actions. ↓ arguments

$A_i(x_1, x_2, x_3 \ldots x_n) \neq A_i(y_1, y_2 \ldots y_n)$

same actions.