Category theory

Pre-order:

$$\begin{tikzcd} a \arrow[loop above, "\leq"] \arrow[r, "\leq"] \arrow[rr, bend right, "\leq"'] & b \arrow[loop above, "\leq"] \arrow[r, "\leq"] & c \arrow[loop above, "\leq"] \end{tikzcd}$$

• Impose reflexivity, composition, associativity

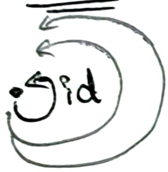
* This is called thin category. (There are either one/more arrows between them.)

Hom-set: represents the set of arrows ~~b/w~~ btw objects.

$C(a,b) \leq 1$. (Partial orders has no loops)

Thick category: (Having multiple maps between categories)

One-object category:



"monoid"

Kleisli category

→ Once we get to categories, the information of elements is lost.

→ We cannot look through the objects but we can see the relationship between two objects.

• We define relation of an object to whole universe and ~~we~~ determine the objects from its relation to universe.

Eg: How can you define a singleton object without elements.

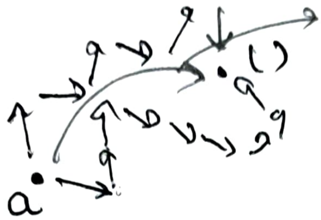
Terminal object $\forall$ objects(a) $\exists$ $f : a \to ()$

$\forall$ objects(a) $f : a \to ()$, $g : a \to ()$

$\Rightarrow f = g$

How you define a empty set? : There is an arrow "to" any objects ⇒ Initial object

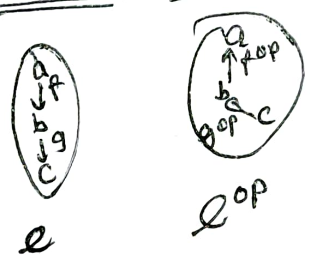
* There is an "unique" path from any object to terminal object.



* How many terminal / initial objects are there?

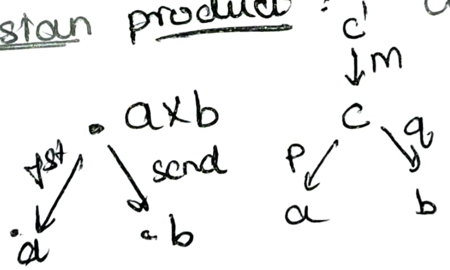
→ If there are two terminal objects, then they are isomorphic

Reversal of arrows :



$(g \circ f)^{op} = f^{op} \circ g^{op}$

Cartesian product : Categorical product



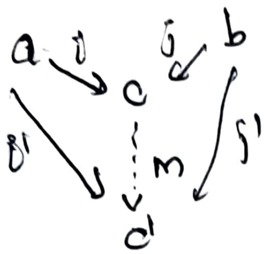
It is between two objects a and b ~~and~~ is equal to c s.t $p : c \to a$ $q : c \to b$

SUCH THAT for any c', $p' : c' \to a$, $q' : c' \to b$ there exist a unique morphism from $c \to c'$

Co-product :

* Any other candidate will be reduced to the product c

<u>Co-product</u>: (Using reversal of maps)

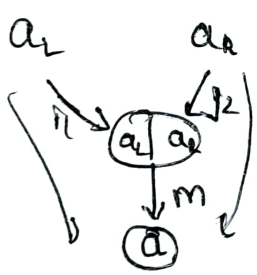


j' factorises to moj
i' factorises to moi

Coproduct is an object and a pair of injections s.t for any other object and a pair of injections there exist a unique morphism.

- Ideal object that contains 2 objects is co-product and morphism 'm' tells which part of c' has c.

→ When the objects are same, we do disjoint union



NOT

Not possible

~~Decorated cospans:~~

~~Category of open graphs~~

* As it is a disjoint union, we can tell about the intersection of these objects through co-equaliser