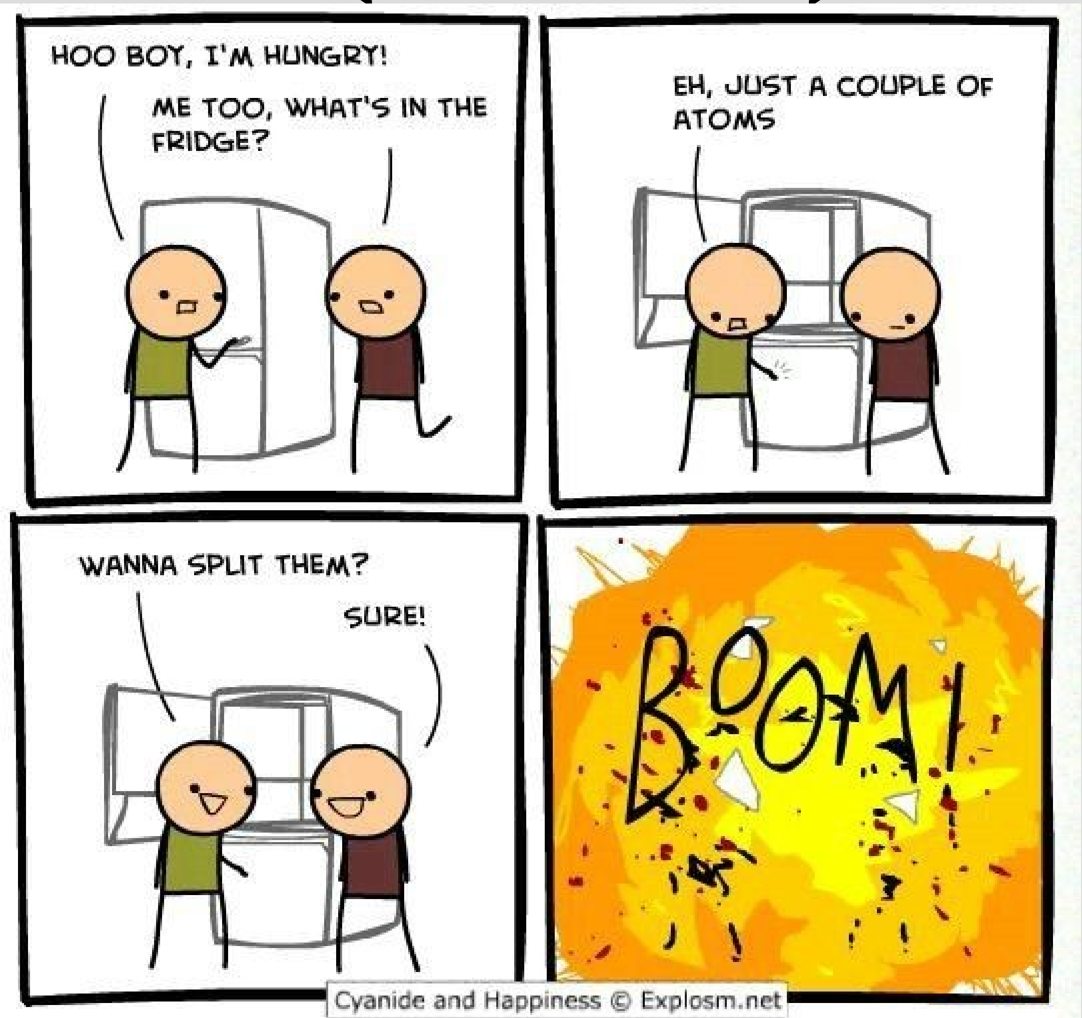
Knowledge Representation (Ch. 12)



Announcements

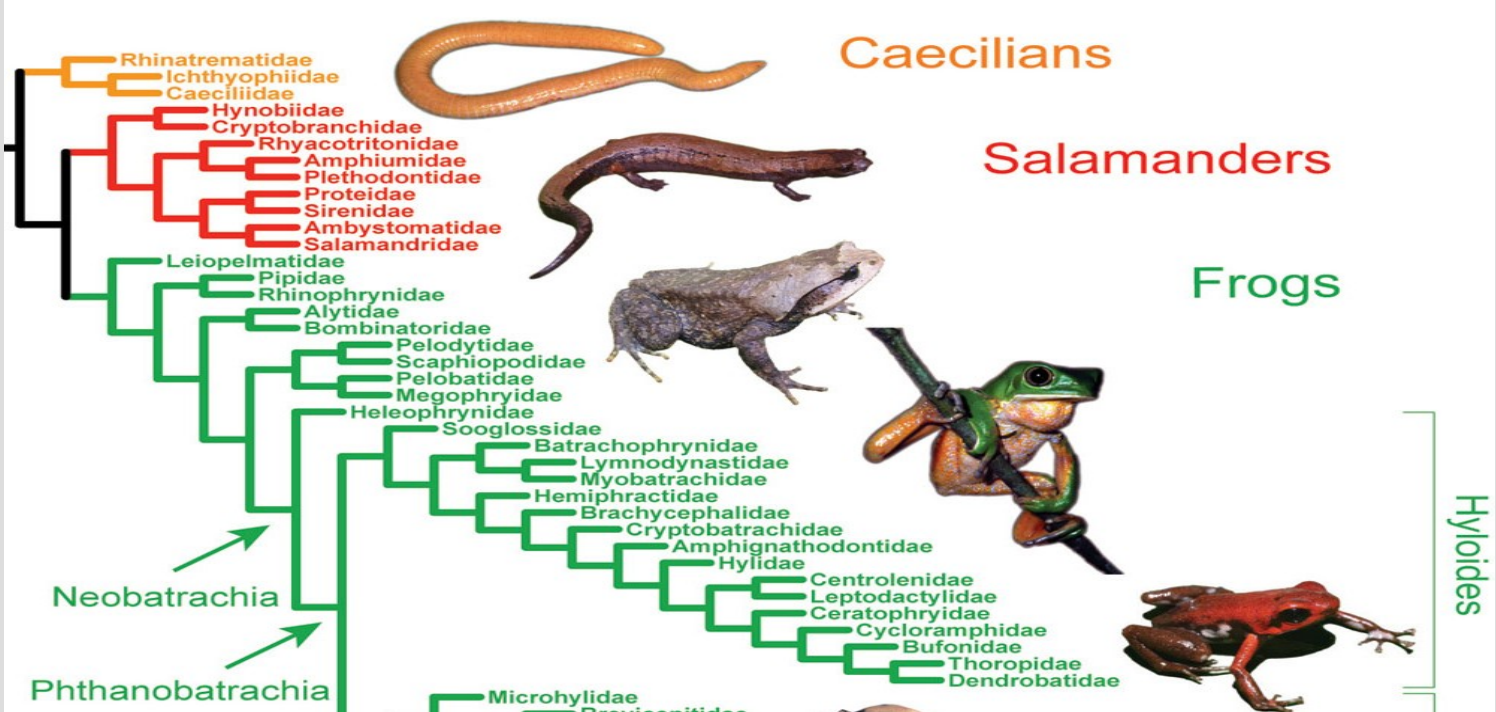
HW 5 correction

Writing 3 up for real now

Ontology

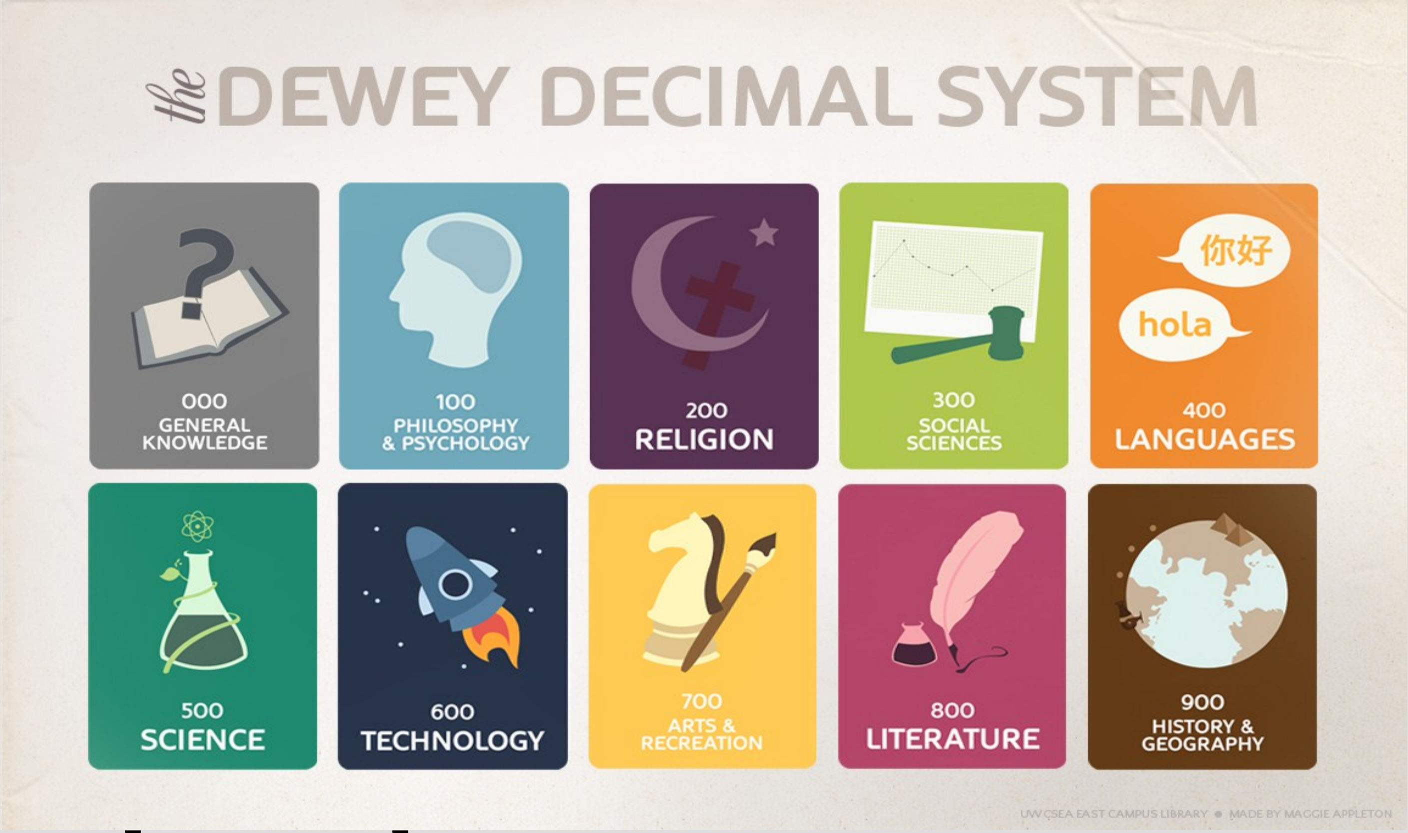
An ontology is a model of data/information and their interactions

Biology has a well known ontology:



Dewey decimal

Most libraries use the Dewey decimal system:



Textbook number: 006.3 20

Chamber of Secrets: 823.914

Dewey decimal

In this system, each digit represents a different classification:

500 = Natural science

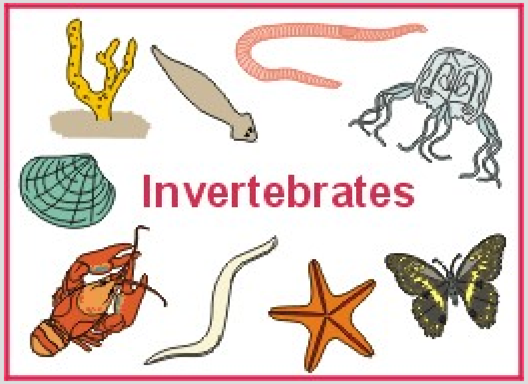
590 = Zoological sciences

595 = Other invertebrates

595.7 = Insects

595.78 = Lepidoptera

595.789 = Butterflies



Ontology

There have been attempts to create an ontology of the real world inside computers

A well known (and still continuing) example is the Cyc project (software proprietary): www.cyc.com

For example, you should be able to ask:
(#$isa #$UofM #$University)
... and it should say "true"

Ontology

Having a general purpose knowledge base
is very time consuming for two main reasons:

1. Inputting all information possible
2. Defining relationships between information correctly and succinctly

Perhaps not too surprising, all-encompassing ontologies have not been too successful, though more limited ones work

Ontology

Thankfully, often you do not need to get too specific and still have a useful ontology

Quite often (but not always), you are interested in a category of something, not a specific one

Ontology

To describe an ontology, we will use first order logic (and add a few new general relations)

The major difference is that we will allow “objects” to be sets in addition to single items

For example, we might make a set Person which both you and I are part of (this is changing it from a relation “Person(x)” to an object “CannotFly(People)”)

Ontology

The main reason for creating a grouping object is to add a Member() and Subset() relations

Member(x, y): “x” (an item) is in “y” (a set)
Subset(x,y): “x” (a set) where very item in “x” is also in “y” (another set)

We will simplify notation by borrowing math's:

$$\left.\begin{array}{l} x \in y \\ Member(x,y) \end{array}\right\} \text{same meaning} \left\{\begin{array}{l} x \subset y \\ Subset(x,y) \end{array}\right.$$

Ontology

This is useful as we can declare general properties and inherit/reuse relations

Suppose we wanted to put everyone in this class into the an ontology

We would have to say:
Person(Alice)^Class(Alice)^Name(Alice)...
Person(Bob) ^ Class(Bob) ^ Name(Bob)...
Person(Catherine) ^ Class(Catherine) ^ ... ...

Ontology

We can define transitivity of both Member() and Subset(), namely:

$$x \in y \wedge y \subset z \Rightarrow x \in z$$

$$x \subset y \wedge y \subset z \Rightarrow x \subset z$$

This allows properties to “transfer” from more general parts of the ontology to specifics

(This is very similar to inheritance in object oriented programming)

Ontology

A more concise way of saying this is then:

$$x \subset People \Rightarrow Name(x) \wedge Person(x)$$
$$InClass \subset People$$
$$Class(x) \Rightarrow x \in InClass$$

Then just:

Class(Alice) ^ Class(Bob) ^ Class(Catherine)...

This simplifies the expression and makes it easier to query the ontology

Ontology

We will borrow more from set theory:

Disjoint(x) - nothing in x shares members (i.e. no overlap between parts of x)

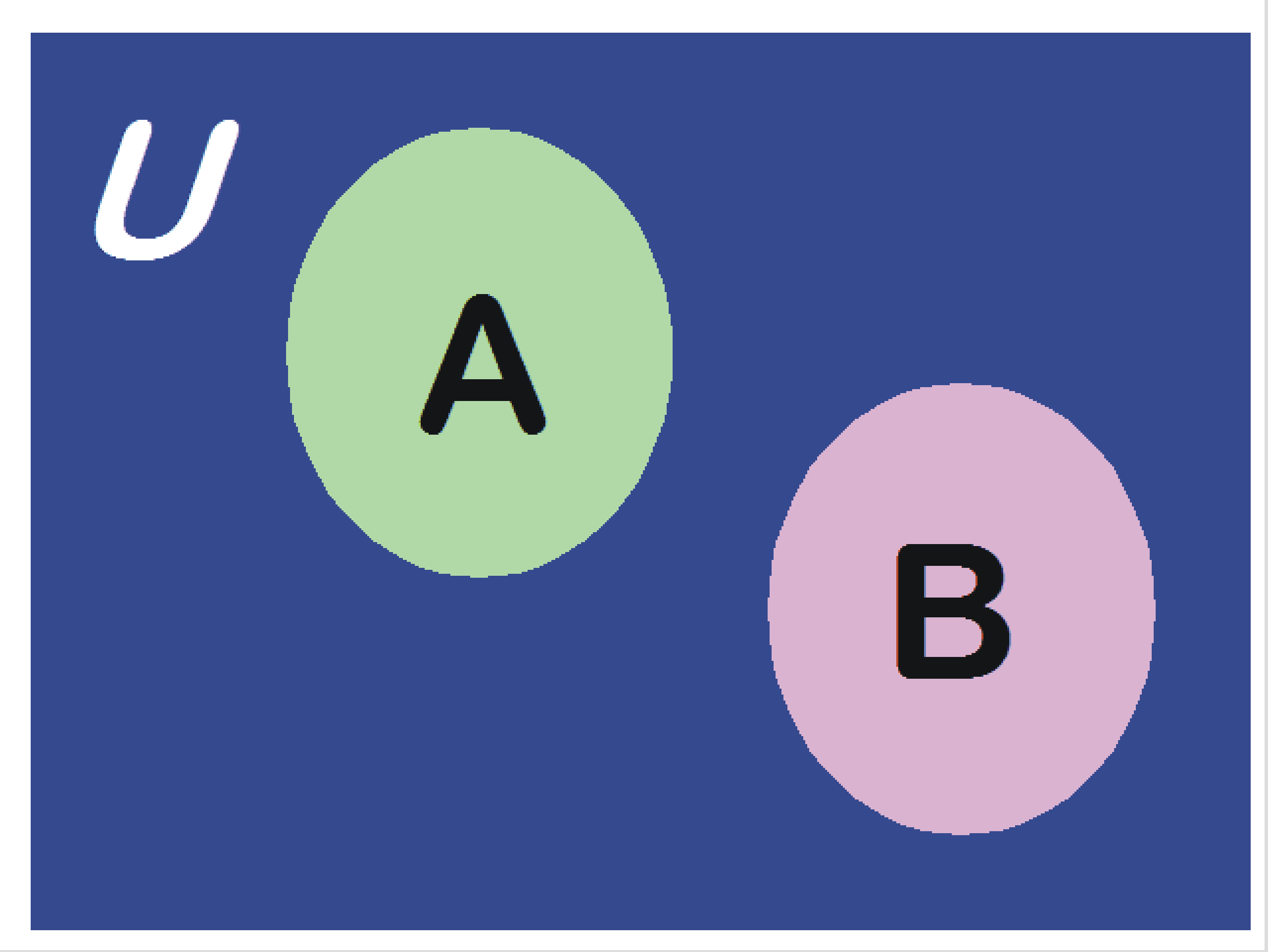
ExhaustiveDecomposition(x,y) - x contains the list of all things in y (i.e. if something is in y, it must also be something in x)

Partition(x,y) - Combination of above two (i.e. if something is in y, it is also in a single x)

Disjunction

$$Disjoint(x) \iff$$
$$(\forall c_1, c_2\ c_1 \in s \wedge c2 \in s \wedge c1 \neq c_2 \Rightarrow Inersection(c_1, c_2) =)$$

x = {A, B}



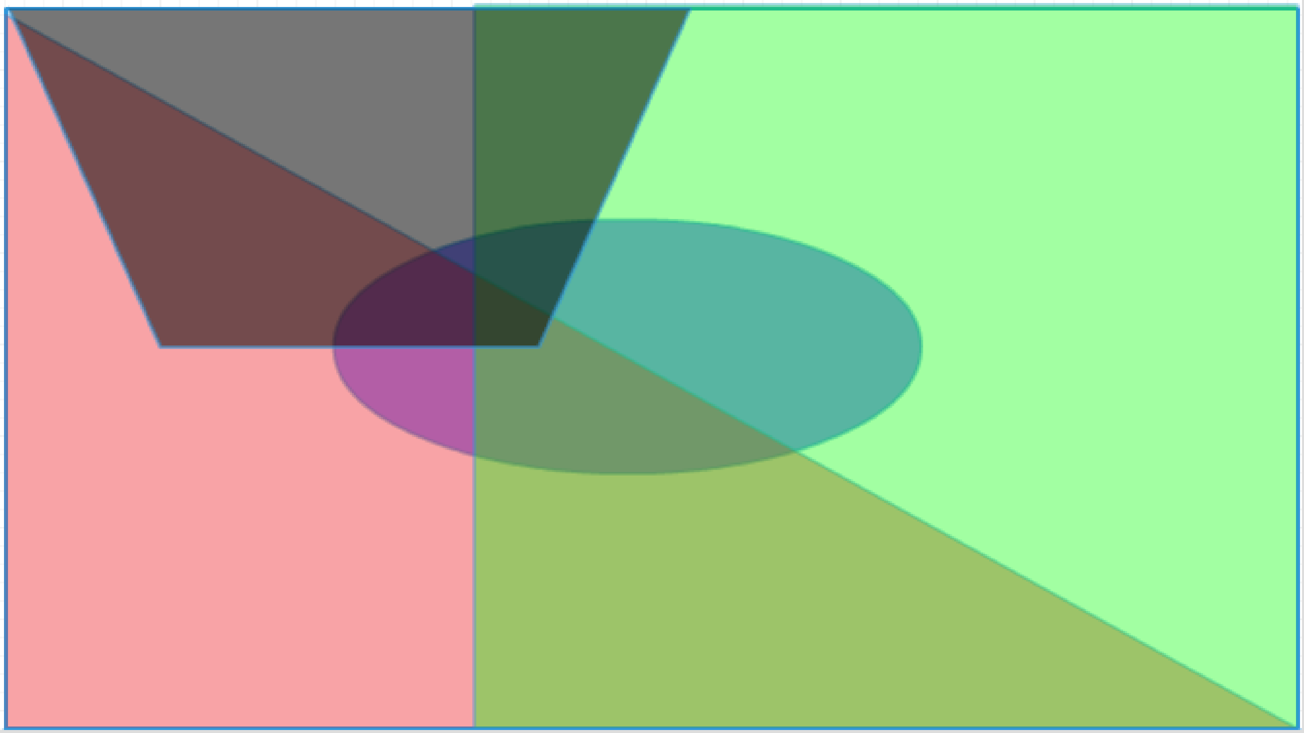
Disjunction

Disjoint({Phones, Dogs}) as there are no objects which are dogs and you can call someone upon (yet...)

Exhaustive decomposition

$$ExhaustiveDecomposition(x, y) \iff (\forall i \; i \in y \iff \exists c_2 \; c_2 \in x \wedge i \in c_2)$$

y = any point in bounding rectangle



x = {red triangle, blue ellipse, green square, black trapezoid}

Exhaustive decomposition

ExhaustiveDecomposition({Ink, Graphite, Petroleum, OtherChemical}, WritingUtensil)

While every writing utensil is one of these types, there can be overlap

For example:

x = that

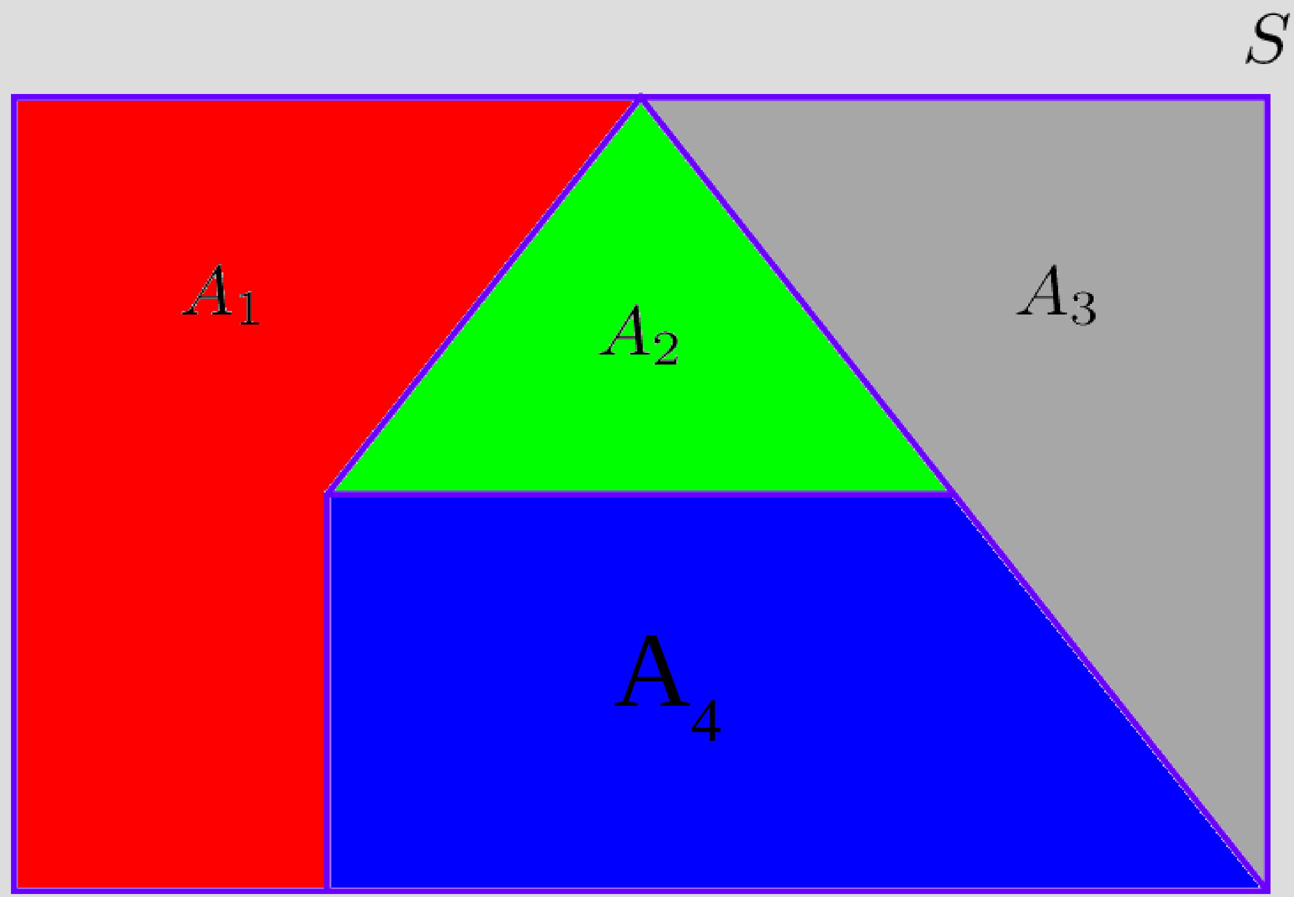
$$x \in WritingUtensil$$

$$x \in Ink \land x \in OtherChemical$$

Partition

$$Partition(x, y) \iff Disjoint(x) \land ExhaustiveDecomposition(x, y)$$

Every point in S is either in {A_1, A_2, A_3, A_4} but never in more than one

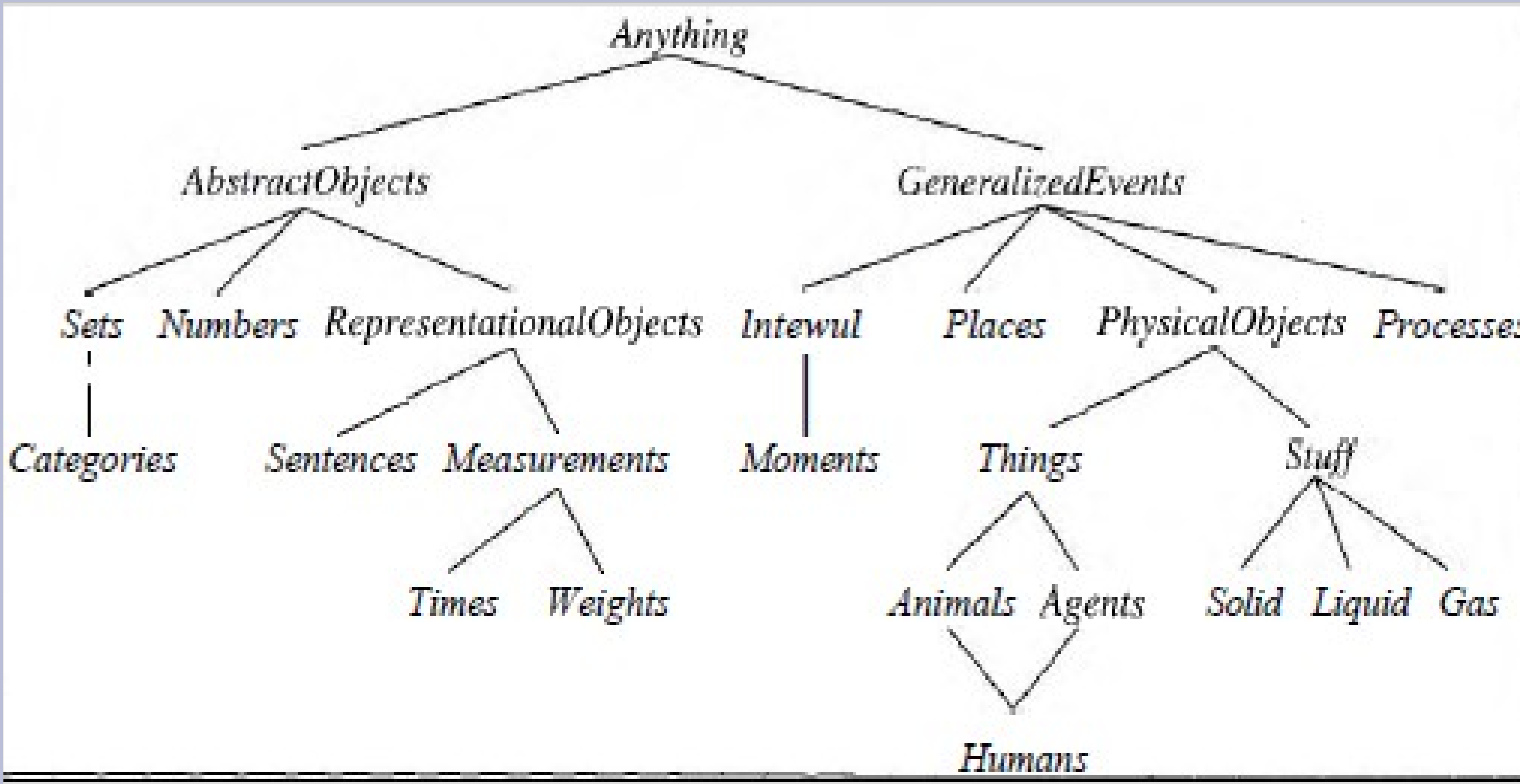


Partition

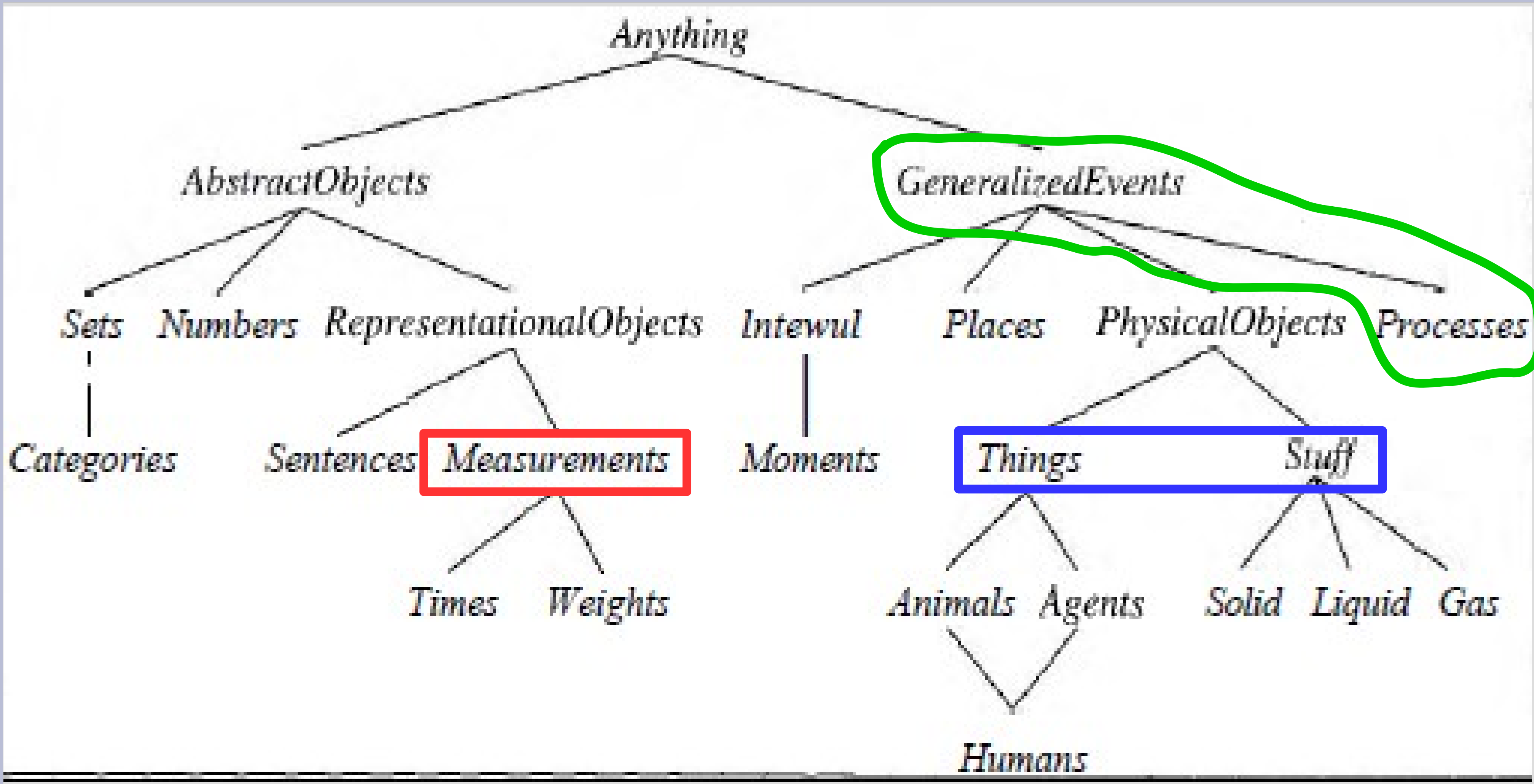
Partition({A, B, C, D, F}, Grade)

Every grade is either an A, B, C, D or F and you can only get one grade (you cannot have both a B and F at the same time)

Book's ontology



Book's ontology



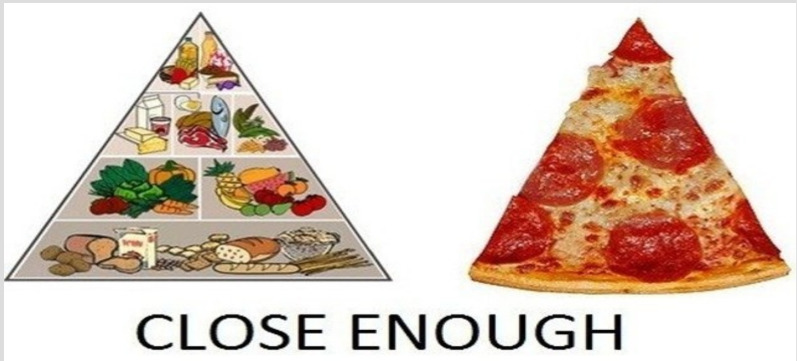
We will look at 3 things: 1, 2 and 3

Measurement

Measurements add another special relation,
a relative compare

This makes sense as Mass(50) > Mass(20)
(i.e. Op>(Mass(50), Mass(20)))

This is also important for qualitative measures:
Tasty(Pizza) > Tasty(Carrot)



Things vs. Stuff

Thing (count nouns) = a single countable item

1 llama  3 llamas 

Stuff (mass nouns) = objects that are only measurable as there is no “whole”

little smoke  lots of smoke 

Things vs. Stuff

The key difference between things and stuff is whether or not it is divisible and keeps the same properties

→ Divide = Different →

→ Divide = Same →

Things vs. Stuff

Intrinsic property = Unchanging properties (i.e. core aspects)

Mostly properties of “stuff”... For example: color, smell, chemical makeup, etc.

Extrinsic property= Properties of the collection

Mostly properties of “things”... For example: mass, shape, length, etc.

Events/Time

Time allows us to have a object that changes value over time (but there is a single object)

4511Teacher(James) would simply say that I am a teacher for this class

If you also say “4511Teacher(Amy)” then it would seem to imply that there are two instructors for this class (not a first!)

Events/Time

To overcome this, we add a True relation, which also takes a time whether or not this is true at that time:

True(4511Teacher(James), Spring2018)

... and also ...

True(4511Teacher(Amy), Fall2016)

This clears up that Amy was teaching a few semesters ago, and me now

Events/Time

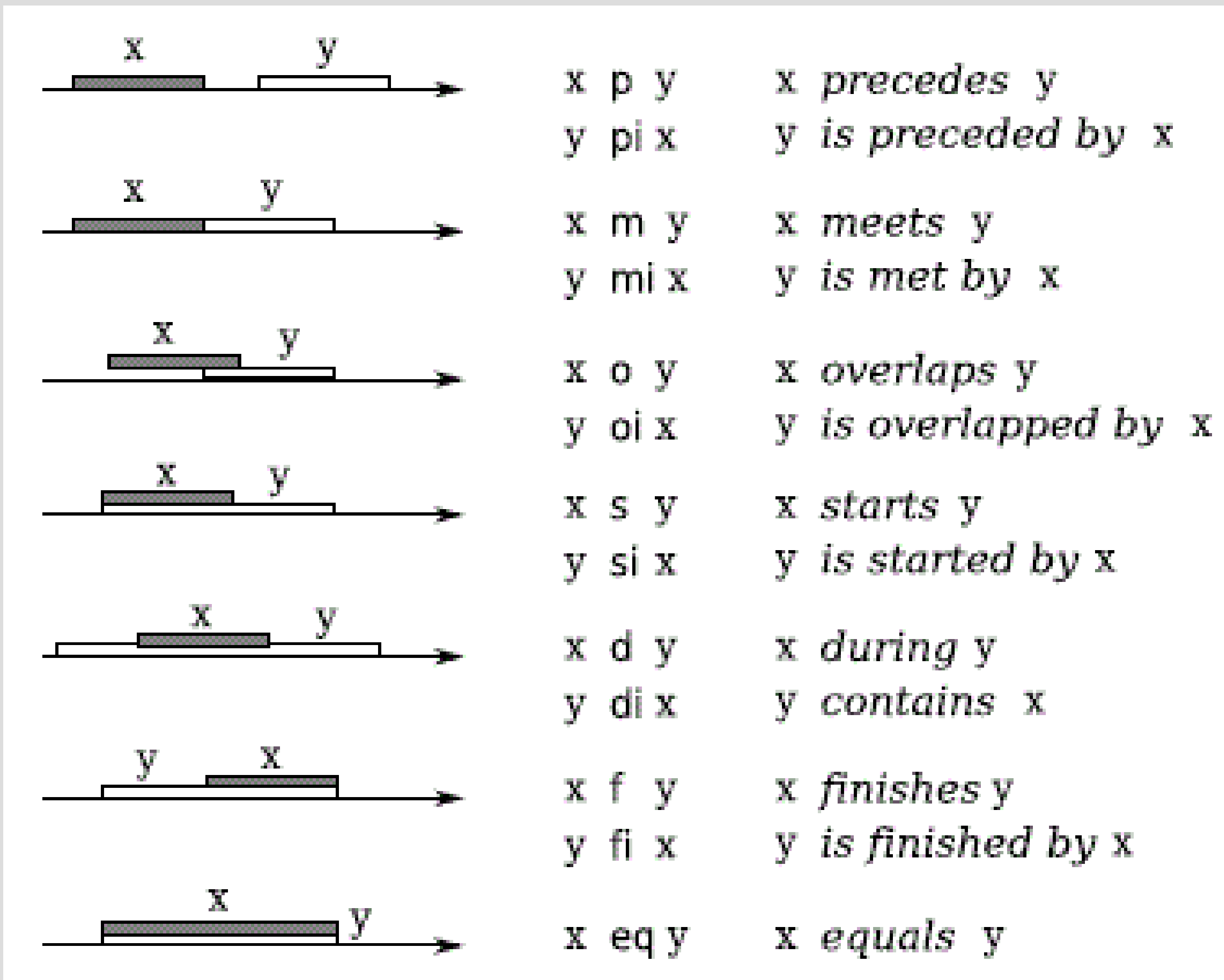
<u>Discrete events</u> have a fixed start and end time, while a <u>process</u> is a fluid transition

This is similar to the difference between things and stuff:

Discrete events = non-divisble = things
  (for example: final exam time)
Process = divisible = stuff
  (for example: global warming)

Events/Time

Additional time relations... Zzz...



Ontology: knowledge relations

So far we focused on defining relationships between different pieces of information

For example, if we know “frogs hop” and “frogs are amphibians”, we can conclude “some amphibians hop”

Deducing new facts are fundamental to having an expressive knowledge base, as it would be too hard to encode every single fact

Mental models

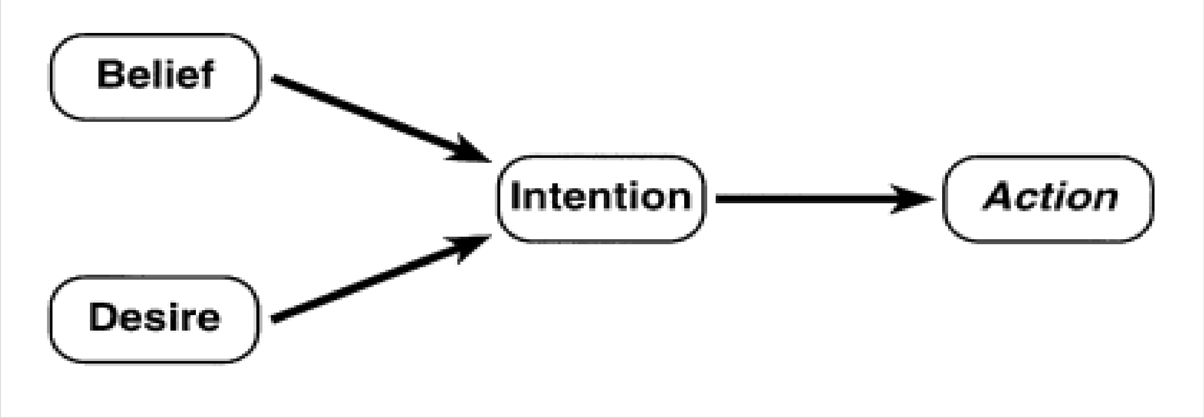
However, not all facts are transferable

Consider this information: “I know someone in Italy” and “That friend knows the weather”, but I **cannot** conclude that “I know the weather in Italy” unless my friend tells me it

Here this is less a physical world fact and more a fact inside my head, which is essentially unknown to anyone else

Mental models

A common way to frame ways of thinking are "Belief, Desire and Intention" (BDI)



https://www.youtube.com/watch?v=96_RSlx2jL0

Each agent has their own local knowledge and goals, which can be communicated

Mental models

Full mental models are an active part of research, so we will focus on just “knows”

$K_a(P)$ will denote agent “a” knows fact “P”

For example, K_{James}(next slide) as I am aware what the next slide is

We will denote this as $K_J(N)$ for short

Mental models

While I know about myself, I do not know if you know what the next slide is

However, I do know that you either “know the next slide” or “don't know...”

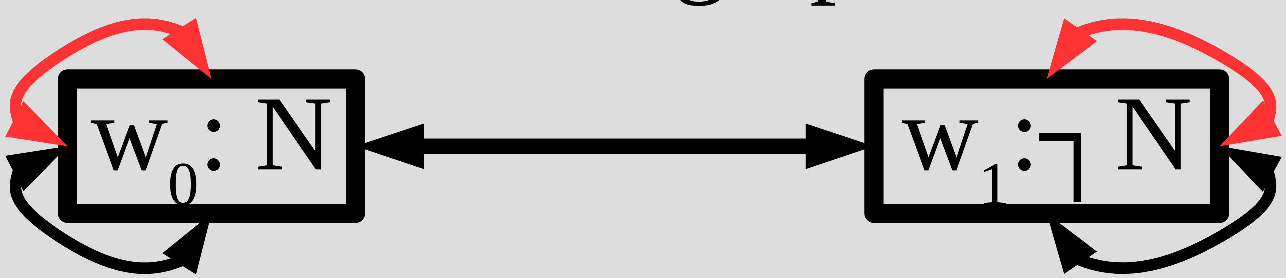
Thus, $K_J(K_{You}(N) \text{ or } K_{You}(\neg N))$

A <u>world</u> is a possible state of the world that I can be in (i.e. possible cases)

Mental models

A world/case is <u>accessible</u> from another world, if the knowledge of a person is consistent

In our example with “N” = you know the next slide, we can make a graph:



 = my accessibility

 = your accessibility

Mental models

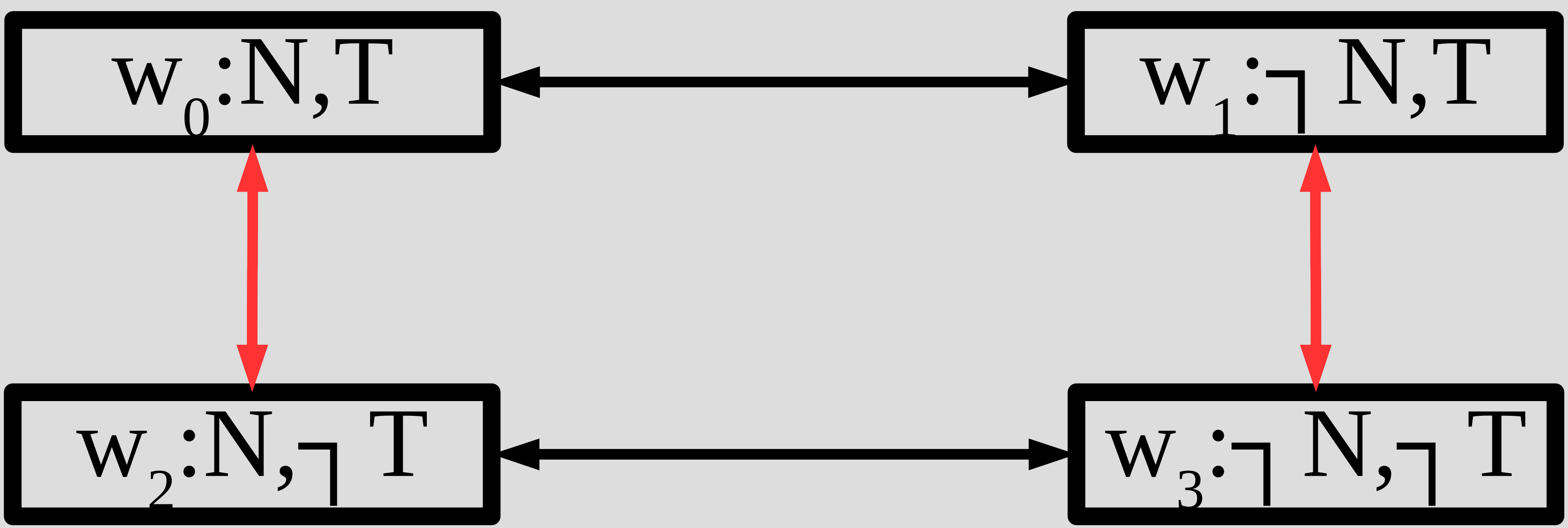
I have a link between w_0 and w_1, as in both worlds $[K_{You}(N)$ or $K_{You}(\neg N)]$ is true

You however know whether or not you know the next slide (e.g. $K_{You}(N)$), so you cannot go between these two worlds

Both possible worlds exist, as in the model I am unsure (despite you knowing)

Mental models

Let's model another fact: next Tuesday's topic (denoted "T"), which only I know... graph is:



= my accessibility (no self arrows)

= your accessibility (no self arrows)

Mental models

You try it! What if I did not know Tuesday's topic either? How would this change?

w_0:N,T

w_1:¬ N,T

w_2:N,¬ T

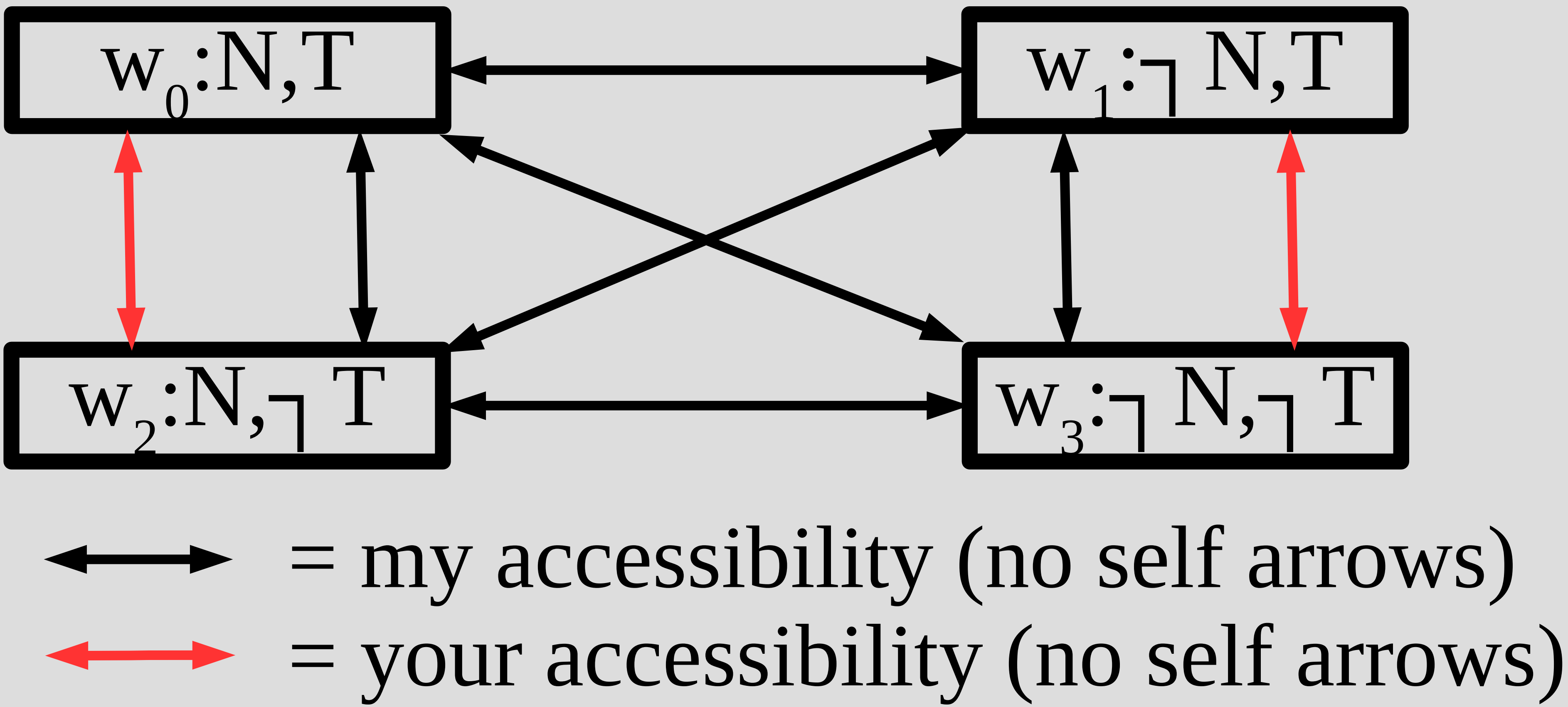
w_3:¬ N,¬ T

= my accessibility (no self arrows)

= your accessibility (no self arrows)

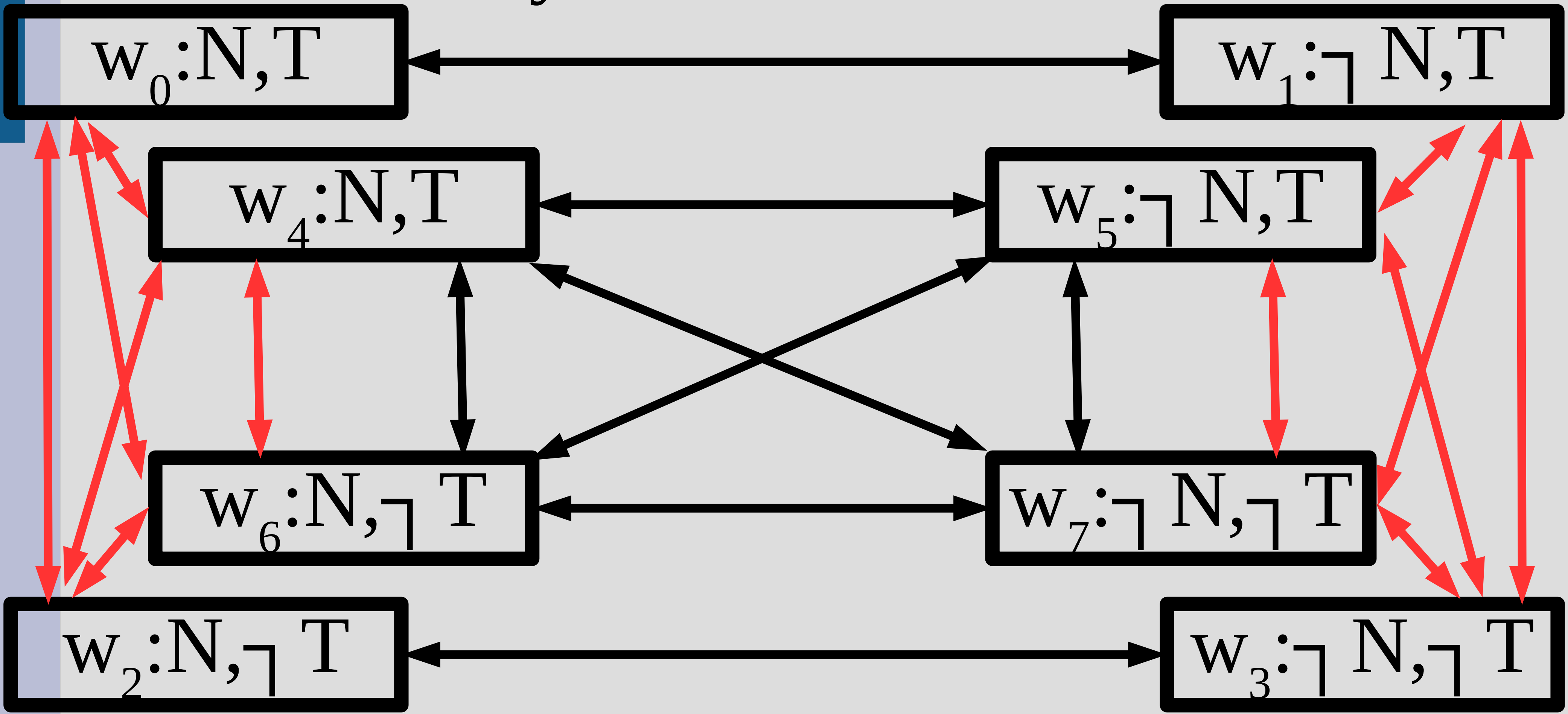
Mental models

You try it! What if I did not know Tuesday's topic either? How would this change?



Mental models

We can actually combine them:



w_0 to w_3 = I do know T, w_4 to w_7 = I don't

Mental models: logic

Logic rules apply to this “knows” as well

For example, if Bird(x) => Fly(x)
Then, K_J(Bird(tweety)) => K_J(Fly(tweety))

This can extend to mental implication as well (for facts that only one entity knows):
$(K_J(P) \wedge K_J(P \Rightarrow Q)) \Rightarrow K_J(Q)$

Mental models: logic

However, you have to be careful with K_a, as the order matters with previous logic ops

For example:

$\exists x \; K_a(Friend(x))$ = in every possible world, one person is always your friend

$K_a(\exists x \; Friend(x))$ = you have a friend in every world, but could be different people

Mental models: logic

You must also put K_A before any knowledge piece that is specific to a person

$K_A(P \text{ or } \neg P) \equiv K_A(\text{True})$, which is a worthless statement ("A knows true things are true")

$[K_A(P) \text{ or } K_A(\neg P)]$ is a useful statement, which indicates that "A knows the state of P"

Mental models: infe

With the additional rules for K_a, you can use ordinary first-order logic to resolve statements

However, this might not be enough... Consider this picture:

It is just some trees, right?
... Right?!