Knowledge Representation

CSE 415: Introduction to Artificial Intelligence
University of Washington
Spring, 2017

Outline

- Introduction
- The Directed Acyclic Graph example
- Binary relations
- Partial orders
- Inferences from partial orders
- Redundancy detection
- Valid and invalid inheritance
- Hasse diagrams and transitive reductions
- HAS links
- Preparing for inference: machine representation
- The **Linneus.py** program
- Maintaining ISA hierarchies
- Other kinds of semantic networks

Knowledge Representation

Expert (intelligent) behavior requires knowledge + inference.

Are there differences among "data," "information," and "knowledge?"

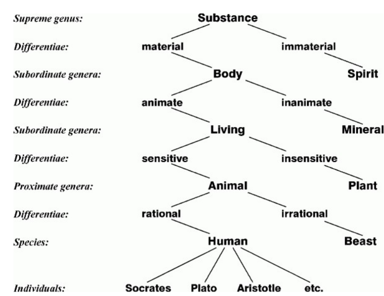
Knowledge

Information

`Data`

Knowledge representation is representation of information with support for inference.

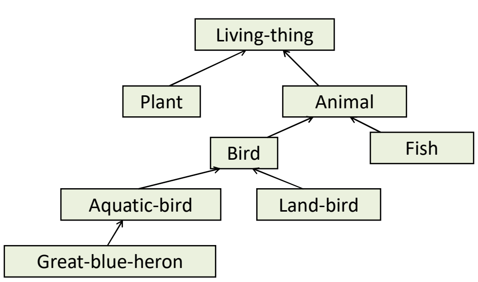
The First Semantic Network Diagram: *Tree of Porphyry*



3rd century AD, by the Greek philosopher Porphyry, illustrating Aristotle's categories. (Taken from an article by John Sowa – this version based on a drawing by Peter of Spain in 1329.)

http://www.jfsowa.com/pubs/semnet.htm

Example of Declarative Representation: An "Isa" Hierarchy



Inference with Isa Hierarchies

"A great-blue-heron is an aquatic-bird."
Isa(great-blue-heron, aquatic-bird).

"An aquatic-bird is a bird."
Isa(aquatic-bird, bird).

"Is a great-blue-heron a bird?"
Isa(great-blue-heron, bird)?

"Isa" represents a relation which has certain properties that support types of inference.

Binary Relations

A *binary relation* R over a domain D is a set of ordered pairs (x,y) where x and y are in D.

For example, we could have,
D = {a,b,c,d} R = {(a, b), (c, a), (d, a)}

If (x, y) is in R, then we write R(x, y) or x R y.

Partial Orders

If for each x in D we have x R x, then R is *reflexive*.

If for each x in D and y in D we have
x R y and y R x imply x = y,
then R is *antisymmetric*.

If for each x in D, y in D, and z in D we have
x R y and y R z imply x R z,
then R is *transitive*.

If R has all 3 properties, R is a *partial order*.

Example of a Partial Order

Let D = the real numbers
Let R_0 = { (x, y) | x ≤ y } Actually, R_0 *is* ≤.

R_0 is reflexive, because
for all x, x ≤ x.

R_0 is antisymmetric, because
for all x, y: if (x ≤ y) and (y ≤ x) then x = y.

R_0 is transitive, because
for all x, y, z: if (x ≤ y) and (y ≤ z) then (x ≤ z).

Examples of the Partial Order Properties

ISA is *reflexive*:
A bear is a bear.
-> ISA(bear, bear)

ISA is *antisymmetric*:
A bear is an ursid, and an ursid is a bear.
Therefore, bear and ursid represent the same things.
ISA(bear, ursid) ∧ ISA (ursid, bear) → *bear = ursid*

ISA is *transitive*:
A grizzly is a bear, and a bear is a mammal.
Therefore, a grizzly is a mammal.
ISA(grizzly, bear) ∧ ISA (bear, mammal) → ISA (grizzly, mammal)

Redundant Facts

Any fact implied by others via the reflexive, antisymmetric, or transitive properties of a partial order can be considered redundant.

Suppose
a ≤ b
b ≤ c
c ≤ b
Then
a ≤ c is redundant.
a ≤ a is redundant.
b = c is redundant.
b and c could be represented by one name.

Inheritance of Properties via Isa

A fox is a mammal.

A mammal bears live young

Therefore a fox bears live young.

Non-inheritance of Certain Properties

A neutron is an atomic particle.

There are three types of atomic particles.

Therefore there are three types of neutrons (??)

Hasse Diagrams

A *Hasse Diagram* is a drawing of a graph that represents the transitive reduction of a partial order.

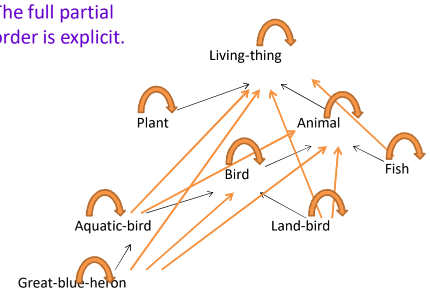
Transitive reduction:
Let R_0 be the original relation and R_1 be its transitive reduction.

That means all shortcuts have been removed;
i.e., if x R_0 y, y R_0 z, and x R_0 z,
then x R_1 y and y R_1 z, but NOT x R_1 z.

In addition, all reflexive pairs are removed:
x R_0 x, but NOT x R_1 x.

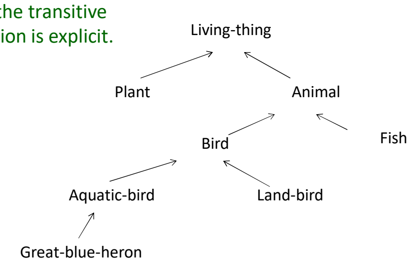
ISA Hierarchy with Redundancy

The full partial order is explicit.



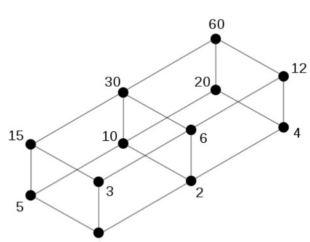
ISA Hierarchy as a Hasse Diagram

Only the transitive reduction is explicit.



Hasse Diagram Example 2

Example for the set {1,2,3,4,5,6,10,12,15,20,30,60} and the relation *divides*.



Arrowheads are not necessary here. All edges point "up."

The "HAS" Relation

"A car has a wheel."
x HAS y = An x has a y as a part.

"A person has a head."
"A head has a face."
"Therefore a person has a face."

"An organism has some tissue."
"Some tissue has a cell."
"Therefore an organism has a cell."

What are the properties of HAS?

Is it transitive?
Is it reflexive?
Is it symmetric?
Is it antisymmetric?

Is it a partial order?
Is it an equivalence relation?

What are the properties of HAS?

Is it transitive? Yes
Is it reflexive? No
Is it symmetric? No
Is it antisymmetric? Yes, vacuously

Is it a partial order? No
Is it an equivalence relation? No

Interoperation of HAS and ISA

An *x* is a *y*.
A *y* has a *z*.
Therefore an *x* has a *z*.

An *x* has a *y*.
A *y* is a *z*.
Therefore an *x* has a *z*.

Chains of HAS and ISA links

A chain is a sequence of the form
$x_1 R_1 x_2, x_2 R_2 x_3, \ldots, x_k R_k x_{k+1}$

x_1 has an x_{k+1} provided…

???

Chains of HAS and ISA links

A chain is a sequence of the form
$x_1 R_1 x_2, x_2 R_2 x_3, \ldots, x_k R_k x_{k+1}$

x_1 has an x_{k+1} provided…

Each R_i is either HAS or ISA,
and at least one of the R_i is HAS.

Preparation for Inference: Python represent. of ISA facts

```
ISA = {}
INCLUDES = {}
ARTICLES = {}

def store_isa_fact(category1, category2):
    'Stores one fact of the form A BIRD IS AN ANIMAL'
    # That is, a member of CATEGORY1 is a member of CATEGORY2
    try :
        c1list = ISA[category1]
        c1list.append(category2)
    except KeyError :
        ISA[category1] = [category2]
    try :
        c2list = INCLUDES[category2]
        c2list.append(category1)
    except KeyError :
        INCLUDES[category2] = [category1]
```

Linneus.py

- Demonstrates representation of ISA facts.
- Permits inference via transitivity.
- Illustrates a variety of methods for answering queries.
- Extensible (as in Assignment 2).

Other Kinds of Semantic Networks

According to John Sowa, there are 6 general kinds of semantic networks:

1. **Definitional** -- including ISA hierarchies.
2. **Assertional** – to represent situations, claims that are not necessarily true, etc. (e.g., logic).
3. **Implicational** – links represent causality or logical implication.
4. **Executable** – processing mechanisms are associated with them; e.g., Petri nets.
5. **Learning** – e.g., neural nets.
6. **Hybrid** – combinations of the above