Ontologies and Knowledge-based Systems

- Is there a flexible way to represent relations?
- How can knowledge bases be made to interoperate semantically?

Choosing Individuals and Relations

How to represent: “Pen #7 is red.”

Choosing Individuals and Relations

How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"

Choosing Individuals and Relations

How to represent: “Pen #7 is red.”

- $red(pen_7)$. It’s easy to ask “What’s red?”
  Can’t ask “what is the color of pen_7?”
- $color(pen_7, red)$. It’s easy to ask “What’s red?”
  It’s easy to ask “What is the color of pen_7?”
  Can’t ask “What property of pen_7 has value red?”

Choosing Individuals and Relations

How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"
- $prop(pen_7, color, red)$. It's easy to ask all these questions.

Choosing Individuals and Relations

How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"
- $prop(pen_7, color, red)$. It's easy to ask all these questions.

$prop(Individual, Property, Value)$ is the only relation needed:
called individual-property-value representation
or triple representation

Universality of *prop*

To represent “a is a parcel”

Universality of *prop*

To represent "a is a parcel"

- $prop(a, type, parcel)$, where *type* is a special property
- $prop(a, parcel, true)$, where *parcel* is a Boolean property

Reification

- To represent $\mathit{scheduled}(cs422, 2, 1030, cc208)$. "section 2 of course $cs422$ is scheduled at 10:30 in room $cc208$."

Reification

- To represent $scheduled(cs422, 2, 1030, cc208)$. "section 2 of course $cs422$ is scheduled at 10:30 in room $cc208$."
- Let $b123$ name the booking:

  $$prop(b123, course, cs422).$$

  $$prop(b123, section, 2).$$

  $$prop(b123, time, 1030).$$

  $$prop(b123, room, cc208).$$
- We have reified the booking.
- Reify means: to make into an individual.
- What if we want to add the year?

Semantic Networks / Knowledge Graphs

When you only have one relation, *prop*, it can be omitted without loss of information.

Logic:

$$prop(Individual, Property, Value)$$

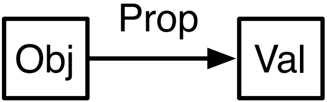
triple:

$$\langle Individual, Property, Value\rangle$$

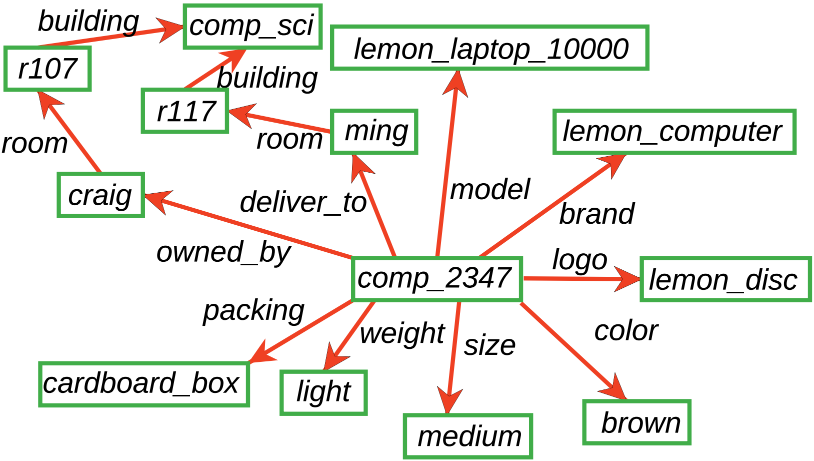
simple sentence:

$$Individual\ \ Property\ \ Value.$$

graphically:



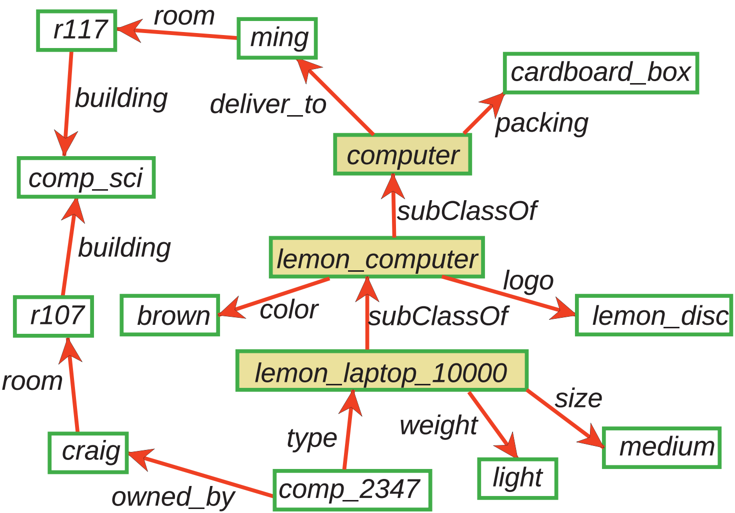
An Example Semantic Network / Knowledge Graph



Equivalent Logic Program

```
prop(comp_2347, owned_by, craig).
prop(comp_2347, deliver_to, ming).
prop(comp_2347, model, lemon_laptop_10000).
prop(comp_2347, brand, lemon_computer).
prop(comp_2347, logo, lemon_disc).
prop(comp_2347, color, brown).
prop(craig, room, r107).
prop(r107, building, comp_sci).
    ⋮
```

A Structured Semantic Network / Knowledge Graph



Logic of Property

An arc $c \xrightarrow{p} v$ from a class c with a property p to value v means every individual in the class has value v on property p:

$$
\begin{array}{l}
prop(Obj, p, v) \leftarrow \\
\qquad prop(Obj, type, c).
\end{array}
$$

Example:

$$
\begin{array}{l}
prop(X, weight, light) \leftarrow \\
\qquad prop(X, type, lemon\_laptop\_10000). \\
prop(X, packing, cardboard\_box) \leftarrow \\
\qquad prop(X, type, computer).
\end{array}
$$

Logic of Property Inheritance

You can do inheritance through the subclass relationship:

$$\begin{aligned} & prop(X, type, T) \leftarrow \\ & \qquad prop(S, subClassOf, T) \wedge \\ & \qquad prop(X, type, S). \end{aligned}$$

Multiple Inheritance

- An individual is usually a member of more than one class. For example, the same person may be a wine expert, a teacher, a football coach,. . . .
- The individual can inherit the properties of all of the classes it is a member of: multiple inheritance.
- With default values,what is an individual inherits conflicting defaults from the different classes? multiple inheritance problem.

Choosing Primitive and Derived Properties

- Associate an property value with the most general class with that property value.
- Don't associate contingent properties of a class with the class. For example, if all of current computers just happen to be brown.