Semantic Networks in Prolog

Tim Finin
University of Maryland
Baltimore County

Version 1.0

- **class/1** is true for nodes
- **isa/2** captures the subclass relation
- **FOO/2** where FOO is the name of an arc asserts an act between two nodes

Version 1.1

- **class/1** is true for nodes
- **isa/2** captures the subclass relation
- **arc/3** where the first argument is the name of an arc asserts an act between two nodes

Version 2

- **class/1** is true for nodes
- **isa/2** captures the subclass relation
- **hasa/4** where the arguments are
  - Frame name
  - Slot name
  - Facet name
  - Datum
- **Slot facets**: type, cardinality, inverse, value, etc.

`class(thing).`  
`class(person).`  
`class(man).`  
`class(woman).`  
`class(integer).`  
`isa(integer,thing).`  
`isa(person,thing).`  
`isa(man,person).`  
`isa(woman,person).`  
`isa(john,man).`  
`age(john,25).`  
`parent(john,mary).`
**Syntactic Sugar**

A person is a thing with
1 age with type integer,
1 sex with type oneof(male,female),
2 parent with type person and inverse child,
any child with type person.

John is a man with
age = 25,
parent = Mary.

**Inheritance**

- A logical model of inheritance is easy to implement
- Characteristics: everything that is true for a class is true for all its subclasses and individual members. (i.e., no defaults, shadowing, overriding)

```
is(C, C) :- class(C).
is(C1, C2) :- isa(C1, C2).
is(C1, C2) :- isa(C1, X), is(X, C2).
hasa(Class, Slot, Facet, Value) :-
is(Class, C2),
hasa(C2, Slot, Facet, Value).
```

**Lots of issues**

- Detecting inconsistencies
- Own slots vs. inherited slots
- Instances vs. classes
- Subslots (e.g., father is a subslot of parent, or father(X,Y) => parent(X,Y).
- Defaults
- Attached procedures (if-added, if-removed, if-needed)
- Attached axioms
- When to do inferencing, caching stuff