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

```
class(thing).
class(person).
class(man).
class(woman).
class(integer).
isa(integer,thing).
isa(person,thing).
isa(man,person).
isa(woman,person).
age(person,integer).
parent(person,person).
inverse(person,child).
child(person,person).
inverse(child,parent).
sex(man,male).
isa(john,man)
age(john,25).
parent(john,mary).
```


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

```
class(thing).
class(person).
class(man).
class(woman).
class(integer).
isa(integer,thing).
isa(person,thing).
isa(man,person).
isa(woman,person).
arc(age,person,integer).
arc(parent,person,person).
arc(inverse,person,child).
arc(child,person,person).
arc(inverse,child,parent).
arc(sex,man,male).
arc(isa,john,man)
arc(age,john,25).
arc(parent,john,mary).
```


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,think).
isa(man,person).
isa(woman,person).
hasa(person,age,type,integer).
hasa(person,age,cardinality,1).
hasa(person,sex,type,oneof(male,female)).
hasa(person,sex,cardinality,1).
hasa(person,parent,type,person).
hasa(person,parent,cardinaliry,2).
hasa(person,parent,inverse,child).
hasa(person,father,type,man).
hasa(person,father,cardinality,1).
hasa(person,father,inverse,child).
hasa(person,father,value,X) :-
  hasa(person,parent,value,X),
  is(X,male).
hasa(person,child,type,person).
hasa(person,child,cardinality,(0,infinity)).
hasa(man,sex,value,male).
hasa(woman,sex,value,male).
```

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.

Inhertance

- A logical model of inheritance is easy to implement

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

- Characteristics: everthing that is true for a class is true for all it's subclasses and indifidual members. (i.e., no defaults, shadowing, overriding)

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