



CS 681 Fall 2008
Designing Expert Systems

Ontology Design and Development: Answers to Questions

Prof. Gheorghe Tecuci
tecuci@gmu.edu
<http://lac.gmu.edu/>

Learning Agents Center
and Computer Science Department
George Mason University

Generalization (cont.)

What are the possible relationships between two concepts A and B, from a generalization point of view?

- A is more general than B
- B is more general than A
- There is no generalization relationship between A and B

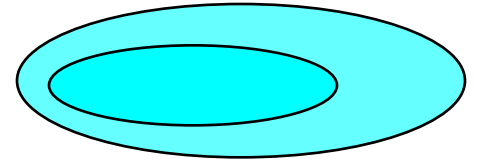
Provide examples of concepts A and B in each of these three situations.

Generalization (cont.)

How could one prove that A is more general than B?

Show that all the instances of B are also instances of A

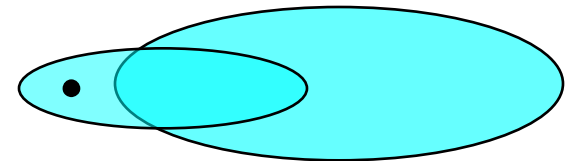
Is this always a practical procedure?



No if A and B have an infinite number of instances.

How can one prove that A is not more general than B?

Show that B contains an instance which is not an instance of A.

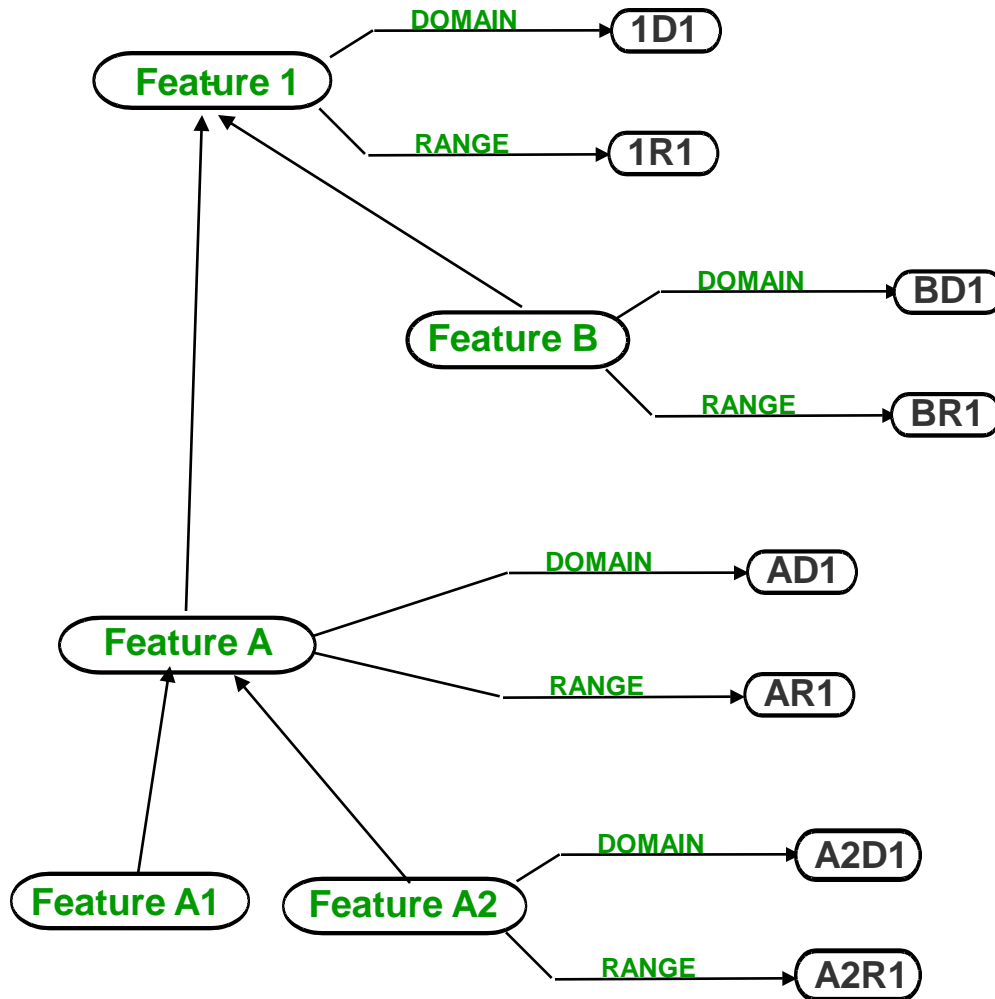


Is this a more practical procedure?

Yes, even for concepts with an infinite number of instances.

Exercise: Hint

Consider the following feature hierarchy:



Is there any necessary relationship between:

BD1 and 1D1?

BR1 and 1R1?

A2D1 and 1D1?

AD1 and BD1?

1D1 and 1R1?

If feature1 is a subfeature of feature2 then the domain of feature1 should be included in the domain of feature2.

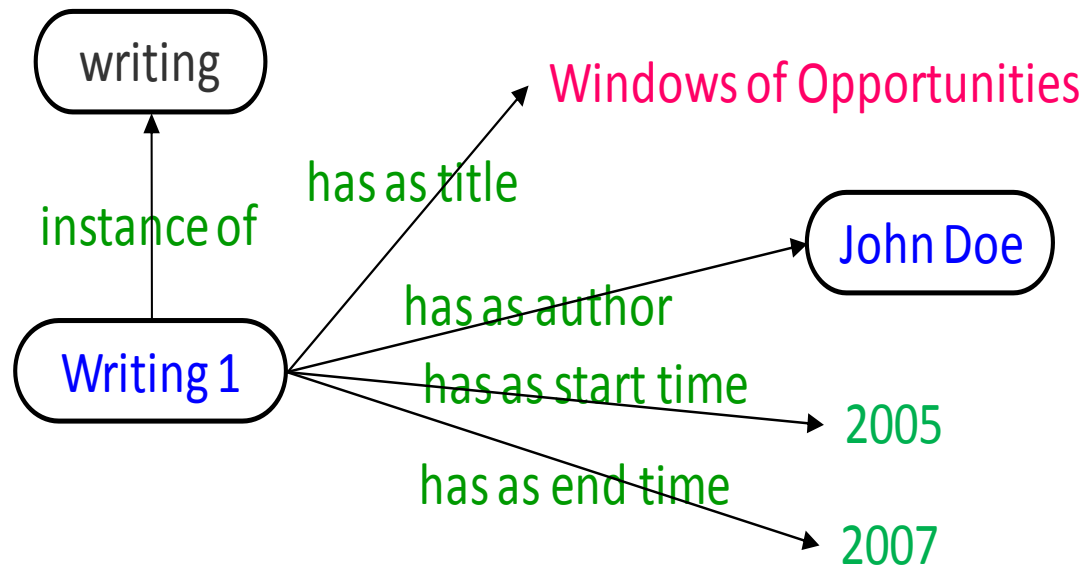
There is a similar rule for the range.

Concept, Instance or Property Value?

John Doe has written “Windows of Opportunities”.

John Doe has as writing “Windows of Opportunities”

he has written it from 2005 until 2007



Practice: Using the Ontology Design Pattern

Extend the object ontology with a new criterion called “support” based on the following information:

Support

Is the director's research work funded?

Do the director's students get academic year support?

Do the director's students get any summer support?

Does the director assist students in obtaining their own funding from such outside sources, such as fellowship programs?

Do the director's students go to conferences?

Using the Ontology Design Pattern

Support

Is the director's research work funded? [advisor funding](#)

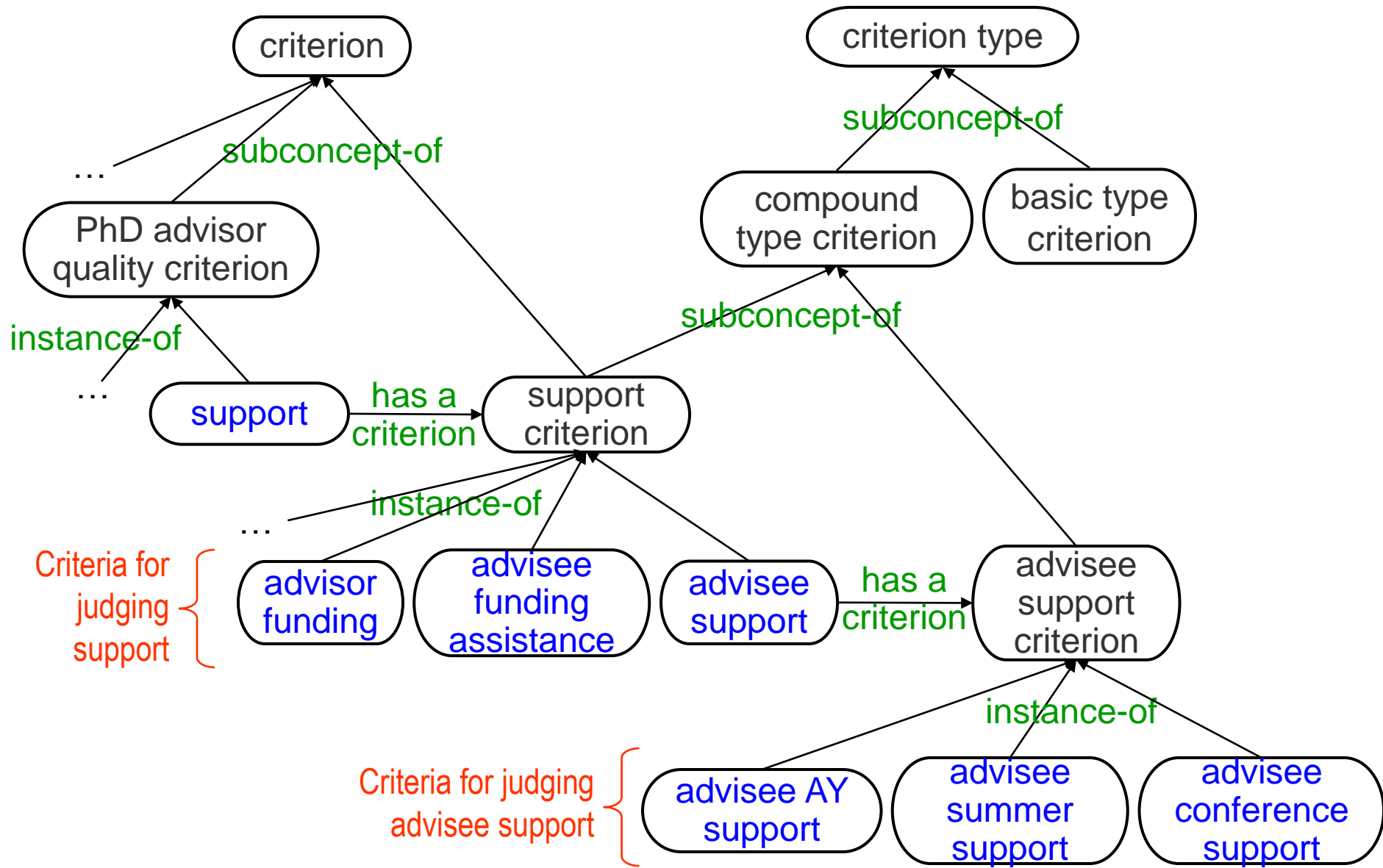
Do the director's students get academic year support? [advisee AY support](#)

Do the director's students get any summer support? [advisee summer support](#)

Does the director assist students in obtaining their own funding from such outside sources, such as fellowship programs? [advisee funding assistance](#)

Do the director's students go to conferences? [advisee conference support](#)

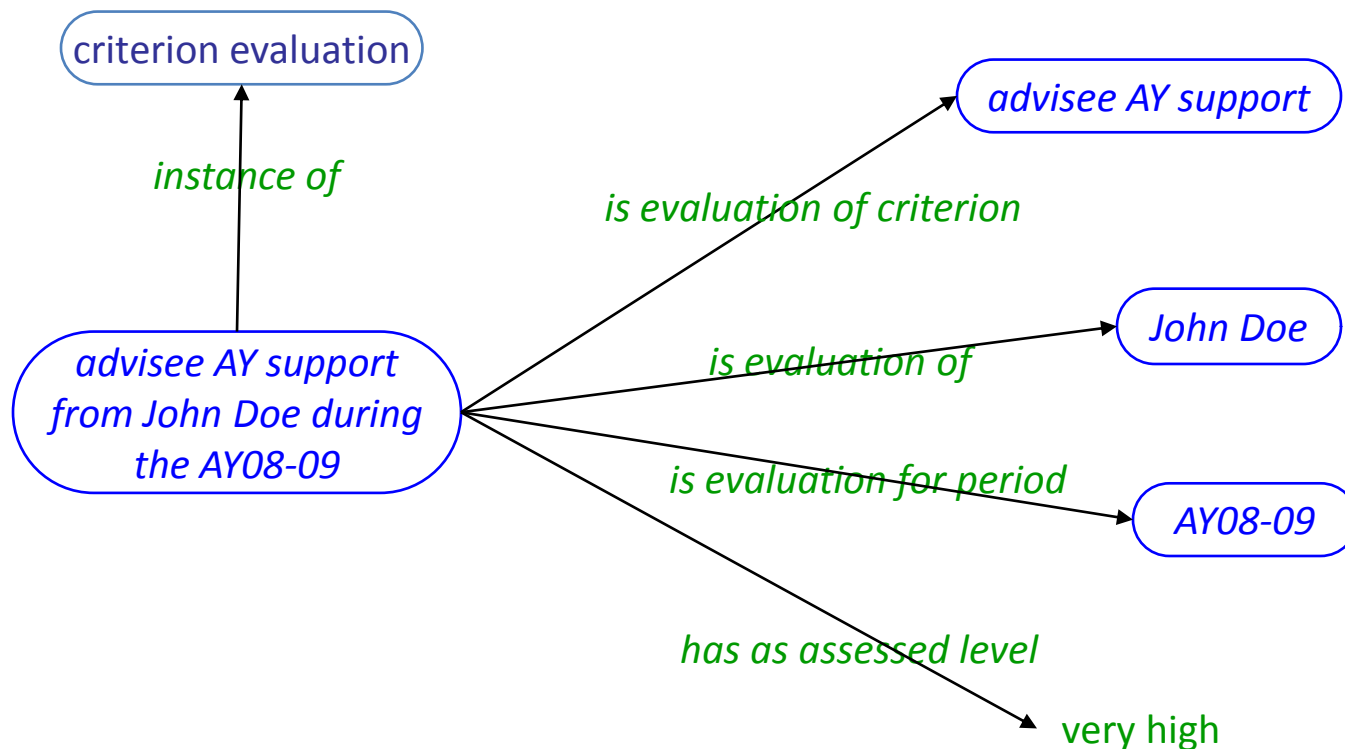
Using the Ontology Design Pattern



N-ary Features

The assessed level of advisee AY support from John Doe during the AY08-09 is very high.

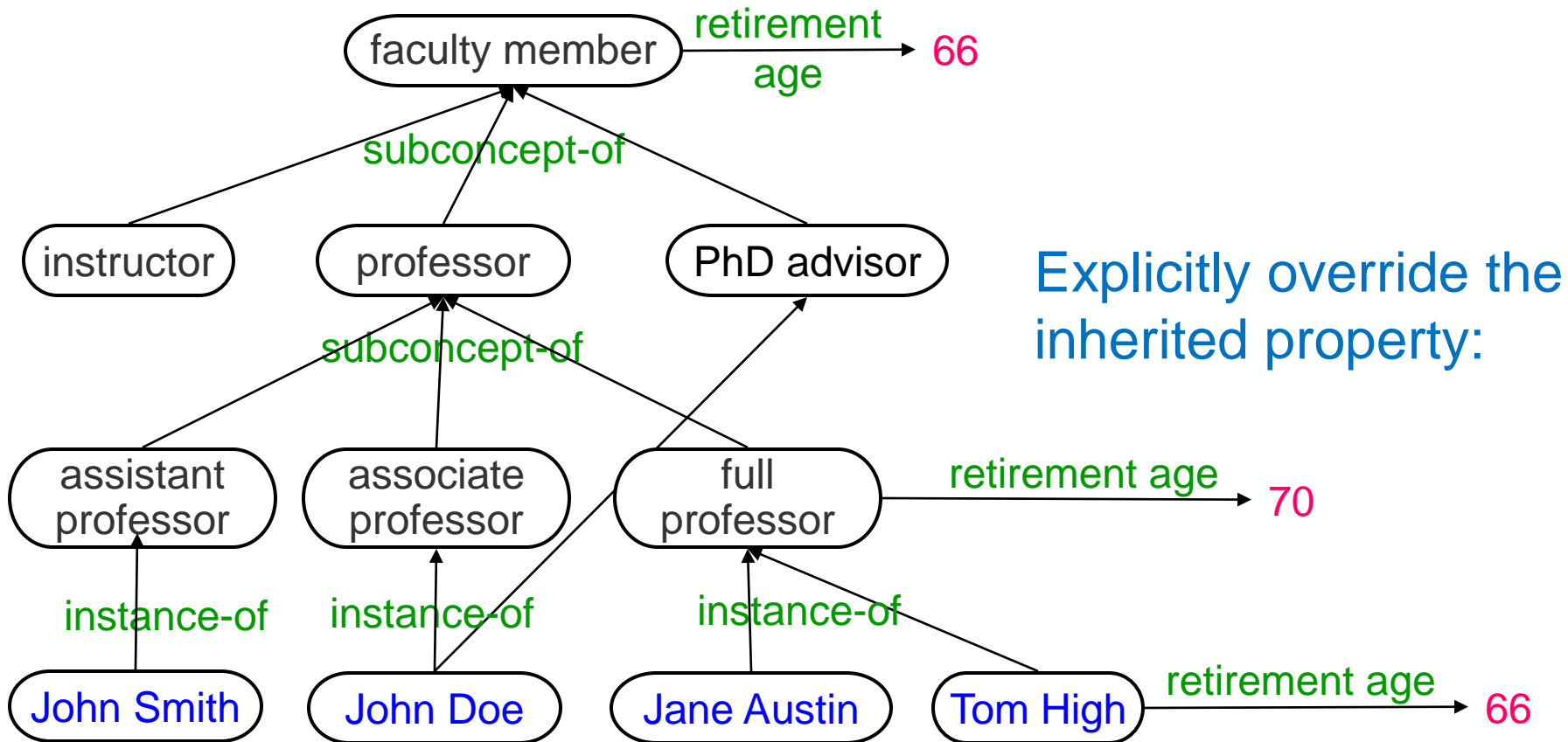
critterion evaluation (advisee AY support, John Doe, AY08-09, very high)



Default Inheritance

Properties associated with concepts in a hierarchy are assumed to be true of all subconcepts and instances.

How can we deal with exceptions (i.e. sub-concepts or instances that do not have the inherited property)?

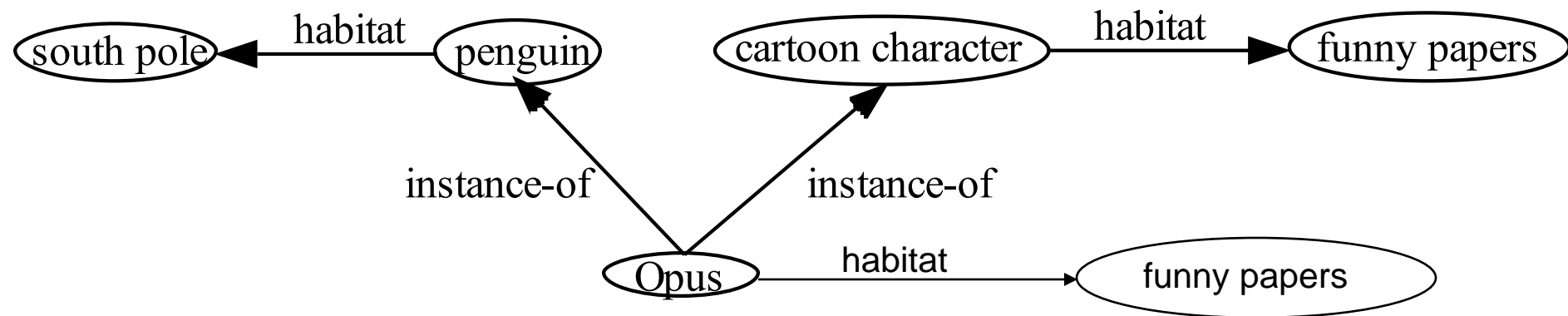


Multiple Inheritance

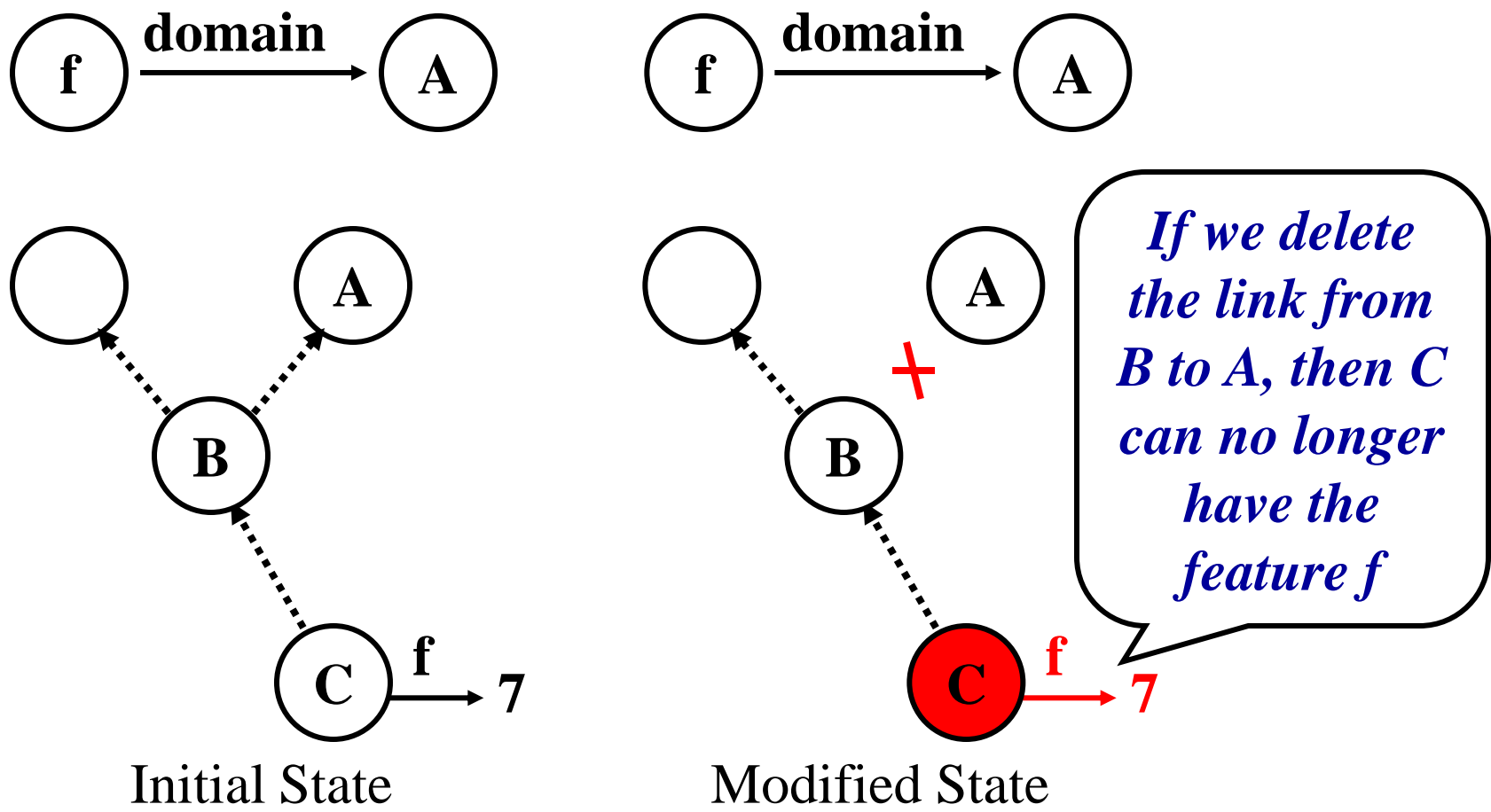
An object (instance or concept) may inherit properties from several super-concepts.

How can we deal with the inheritance of contradictory properties?

Explicitly state the property of the object, instead of inheriting it:



Complexity of Ontology Management: Illustration



Explain why.

Because C is no longer in the domain of f.

Hands-on: Object Browser

Use the Object Browser to build a generalization hierarchy containing the following information:

course **subconcept-of** object

operating systems course **subconcept-of** course

artificial intelligence course **subconcept-of** course

CS571 **instance-of** operating systems course

CS580 **instance-of** artificial intelligence course

Use the Ontology tools of Disciple to develop an object ontology that represents the following information:

The color of Apple1 is red.

The color of Apple2 is green.

Apple1 is an apple.

Apple2 is an apple.

Apples are fruits.

Hint: Define object concepts, object features and instances.