A Practical Introduction to Ontologies & OWL

Session 3: Additional Exercises: Common Errors and how to correct them

Michael Lutz

based on Slides from the Co-ode OWL Tutorial available from
http://www.co-ode.org/resources/tutorials/intro/

Overview

- Elephant Traps
  - Property Domain & Range
  - Property Characteristics – functional properties
  - Intersection
- Negation in OWL – ComplementOf
- Class expressions test

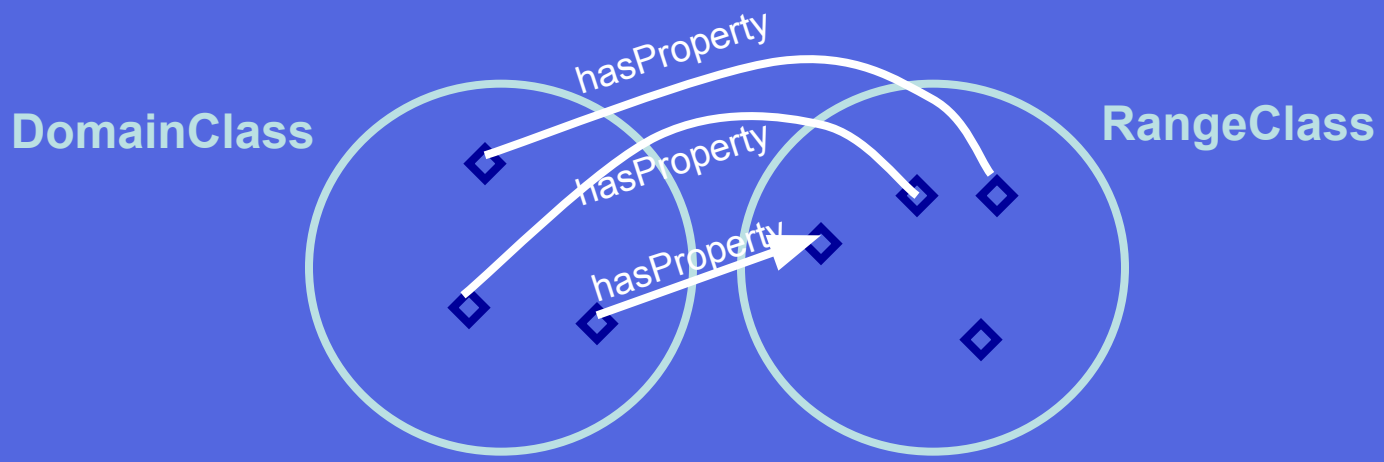
Elephant Traps

Common Errors in OWL generally include:

- Disjoint misuse – often used on defined classes by mistake
- Confusing AllValueFrom and SomeValuesFrom – some doesn't imply only, and only doesn't imply some
- Forgetting to close class descriptions
- Incorrect expectations of Domain and Range defined for properties
- Incorrect use of Functional Properties
- Using intersection (AND) instead of union (OR), where the members of the intersection are disjoint

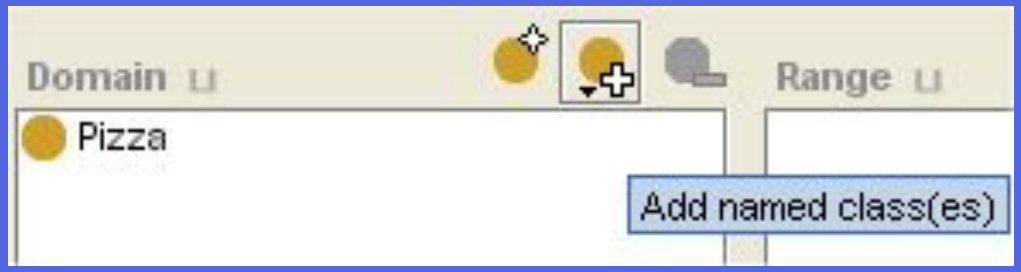
Property Domain & Range

- If a relation is:
  subject_individual □ hasProperty □ object_individual
- The domain is the class of the subject individual
- The range is the class of the object individual
  (or a datatype if hasProperty is a Datatype Property)



Setting a Domain & Range

- Setting a domain & range on a property has global implications
- Be careful not to over-constrain your ontology
- The domain & range can be set in the Properties Tab – just click **Add named class(es)**
  e.g. Setting a domain of **Pizza** on **hasBase**



- Using a Universal Restriction on a Class is like setting a local range

Semantics of Domain & Range

- Domain and Range are not used to restrict the interface
- They are used by the reasoner to infer additional information about individuals
- Any individual that uses a property with a domain set can be inferred to be a member of the domain class
- the same holds for range

Exercise 10: IceCream and Domain

Trap: Property Domain Reclassification

- Any Class that uses a property with a domain set in an existential restriction will be inferred to be a subclass of the domain class
- This is because all individuals in this class must have at least one relationship using this property – therefore, all members of this class must be members of the domain class
- If these classes are disjoint with the domain then they will come out inconsistent – another reason to check all your disjoints are set

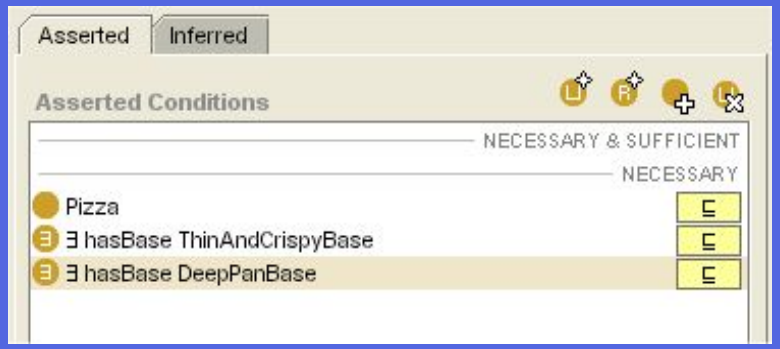
- The same does not apply to range

Property Characteristics

- **Inverses** – if property p has inverse property q, and p links A to B, then it can be inferred that q links B to A
- **Functional** – For a given individual, the property takes only one value.
- **Inverse functional** – The inverse of the property is functional.
- **Symmetric** – If a property links A to B then it can be inferred that it links B to A.
- **Transitive** – If a property links A to B and B to C then it can be inferred that it links A to C.

Functional Properties

- An individual can only have relationships with at most one other individual along a functional property, e.g. if hasBase is functional this means: “Every Pizza can have at most one PizzaBase”
- Description of DoubleBasePizza:



- The reasoner finds this inconsistent
- It looks like the interface is warning us that we can’t use the property more than once, but actually…

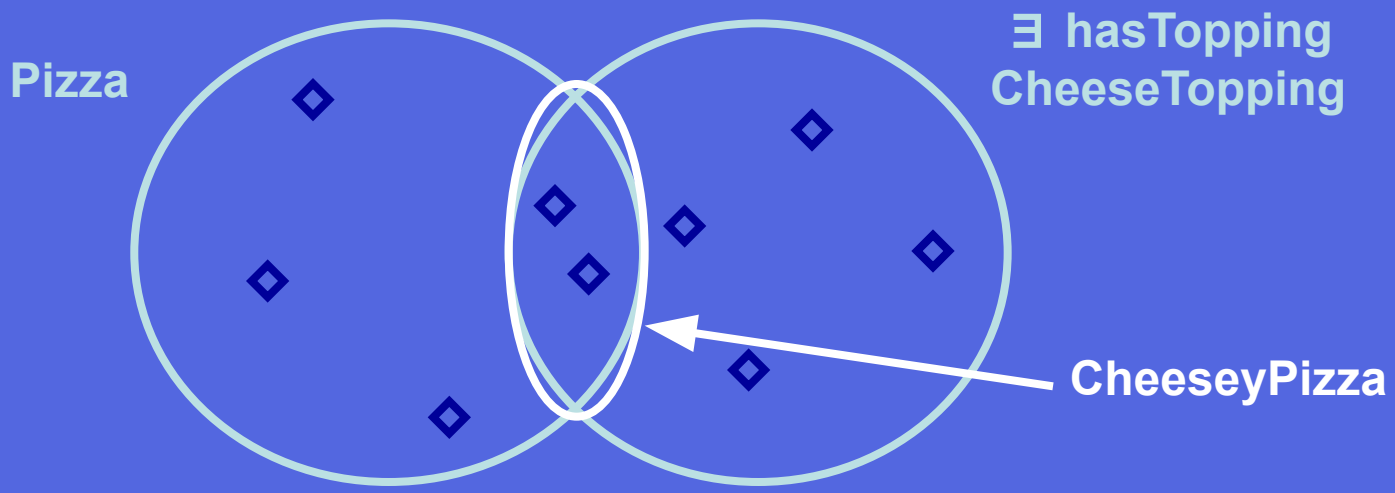
Exercise 11: Functional Properties

Trap: Functional Property Misuse

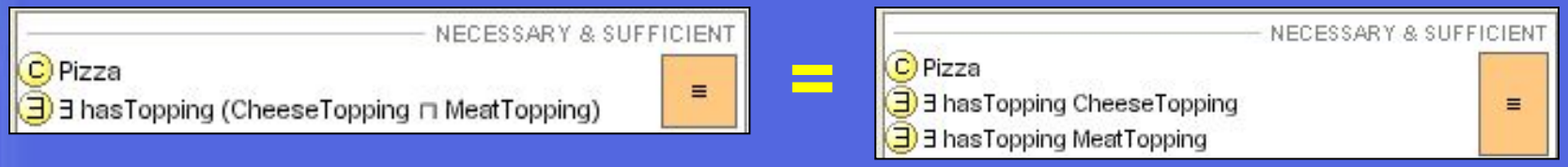
- If a property is functional and is used in several Existential restrictions on a class, the reasoner will infer that the filler classes must overlap
- If any of the fillers are disjoint from each other then this cannot be the case and therefore causes an inconsistency
- If they are not, no inconsistency is found!

Intersection Classes

- aka “conjunction”
- This AND That AND TheOther
- This ⊓ That ⊓ TheOther
- Each class description or definition is an intersection of the conditions in it
- CheeseyPizza ≡ Pizza AND ∃ hasTopping CheeseTopping



Intersection



- People often ask what the difference is between using 2 existential restrictions (which are, by default, in an intersection in the interface) and using a single restriction with a filler containing both the classes

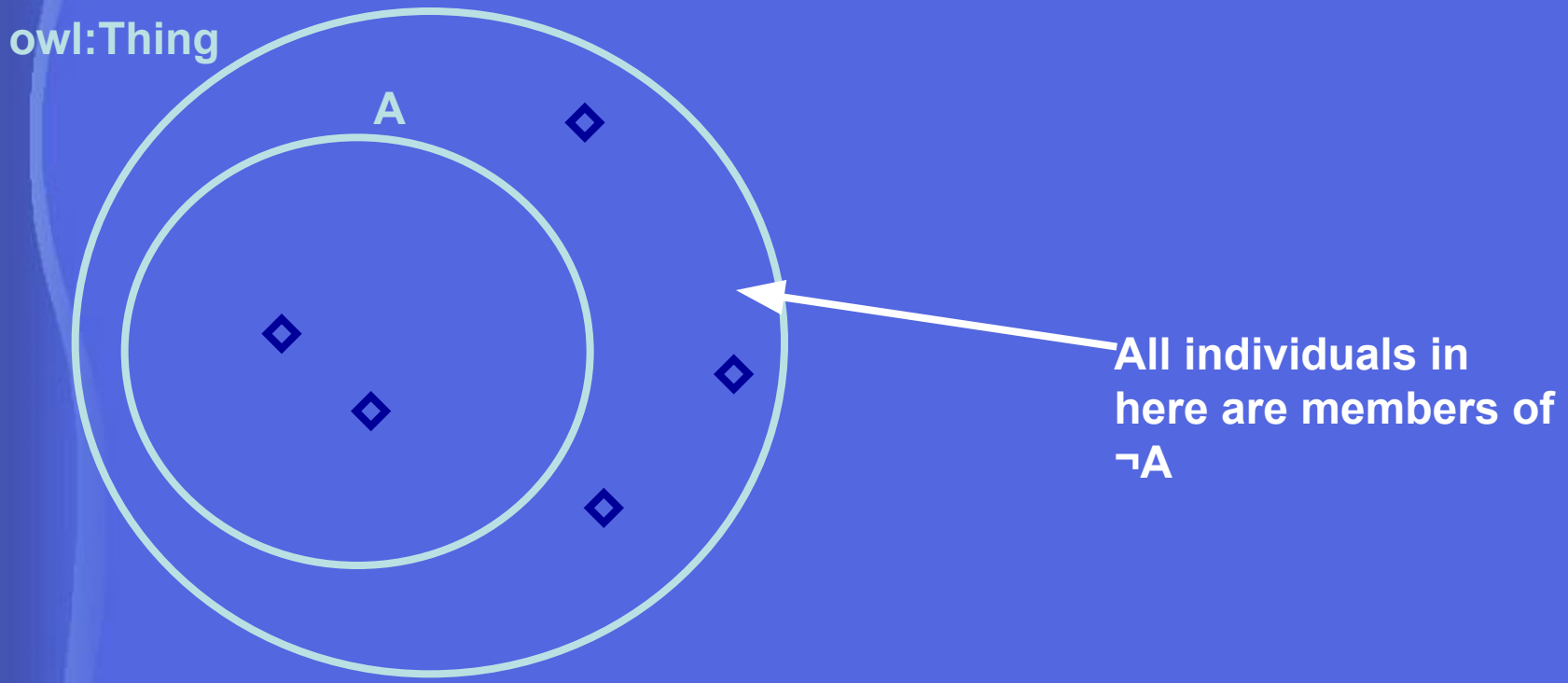
Trap: Intersection

There are 2 problems:

1. Often we paraphrase “AND” when we logically mean “OR”
   The filler “**CheeseTopping** AND **MeatTopping**” cannot contain any individuals as they are disjoint, and is therefore inconsistent
2. If we correct this to OR, it is still wrong as we’ve got a class description that can be fulfilled by a Pizza with a single topping – either Cheese or Meat. If we had 2 existential restrictions, there would have to be at least 2 (disjoint) toppings

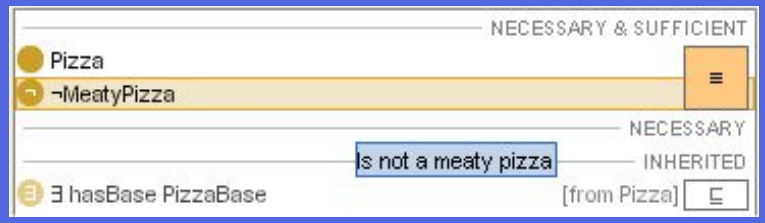
ComplementOf Classes

- aka “Negation” “Not”
- Not Something
- ¬ Something

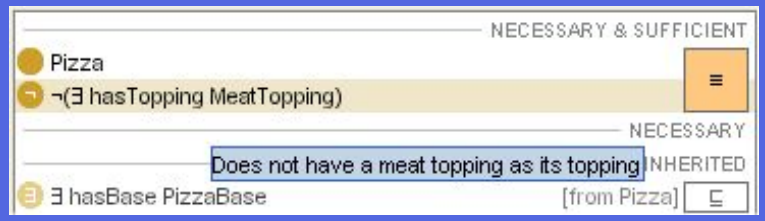


ComplementOf Classes

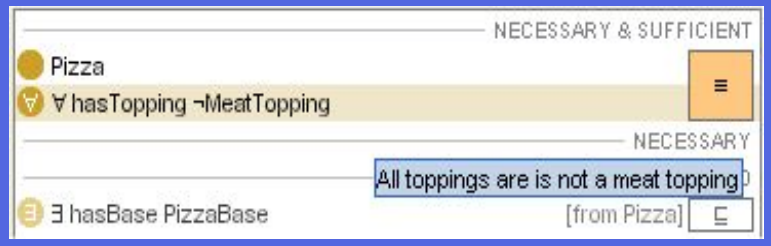
- Commonly used to model 3 things:
  - A is any C that is not B



  - A does not have some relation with B



  - A only has relations with things that are not B



Exercise 12: Variations of VeggiePizza

- The ontology used in this example will be available at: www.co-ode.org/ontologies/brokenPizza/

Summary

You should now be able to:

- Avoid some of the more common modelling errors in OWL
- Appreciate that all OWL statements are reasoned with and many mistakes are only caught because of disjoints
- Understand different characteristics of properties
- Spot various similar looking statements in OWL are very different