Logik in Protégé

Wissensorganisation

Prof. Dr. Spree

Präsentation von:

Rembert Wohlers

David Maus

Hochschule für Angewandte Wissenschaften Hamburg

Hamburg University of Applied Sciences

Gliederung

- Was ist Logik?
- Formale und traditionelle Logik
- Aussagenlogik
- Prädikatenlogik
- Necessary & Sufficient Conditions
- Restriction Types

Was ist Logik?

- „Als Logik bezeichnet man die Wissenschaft von den Gesetzen und Formen des Denkens. Die Logik kann als eine Ethik des Denkens bezeichnet werden.“ (Quelle: Lexikon der Philosophie 2009)

- Logik = vernünftiges Schlussfolgern

- Logik ist ein Begriff aus der Philosophie, Mathematik und Informatik.

Formale und traditionelle Logik

- In der formalen Logik werden die Gesetze des abgeleiteten Wissens studiert, das aus früher bestimmten und geprüften Wahrheiten gewonnen wird, ohne in jedem konkreten Fall direkt auf die Erfahrung zurückzugreifen.

- Die traditionelle Logik untersucht die allgemeinen Gesetze der Logik. Sie untersucht die allgemeinen Formen des Denkens wie Urteil und Begriff sowie die Formen der Verknüpfung der Gedanken im Schluss

Quelle: Lexikon der Philosophie 2009

Aussagenlogik

- befasst sich mit Aussagen und deren Verknüpfung durch Junktoren.
- Jede klassische Aussage wird in „wahr oder „falsch“ zugeordnet.
- Nichtklassiche Logiksysteme haben nicht das Prinzip der Zeideutigkeit

Prädikatenlogik

- Familie logischer Systeme
- Überprüfen und formalisieren Argumente auf Gültigkeit
- formale und nicht formale Logik

Beispiel

- Aussagenlogik:

  „Es regnet“ und „die Erde ist eine Scheibe“

- elementare Aussagen: lassen sich nicht in weitere Teilaussagen zerlegen.

Beispiel

- Prädikatenlogik:

  „_ist eine Scheibe“ Leerstelle: Die Erde

- wird wahr wenn in die Leerstelle ein Eigenname eingesetzt wird

- elementare Aussagen werden hinsichtlich ihrer inneren Struktur untersucht

Necessary

- Falls etwas ein Teil der Klasse ist, dann ist es notwendig, dass die Bedingungen erfüllt sind.

  -> Primitive Class
- Keine Aussage darüber, dass es so sein muss.

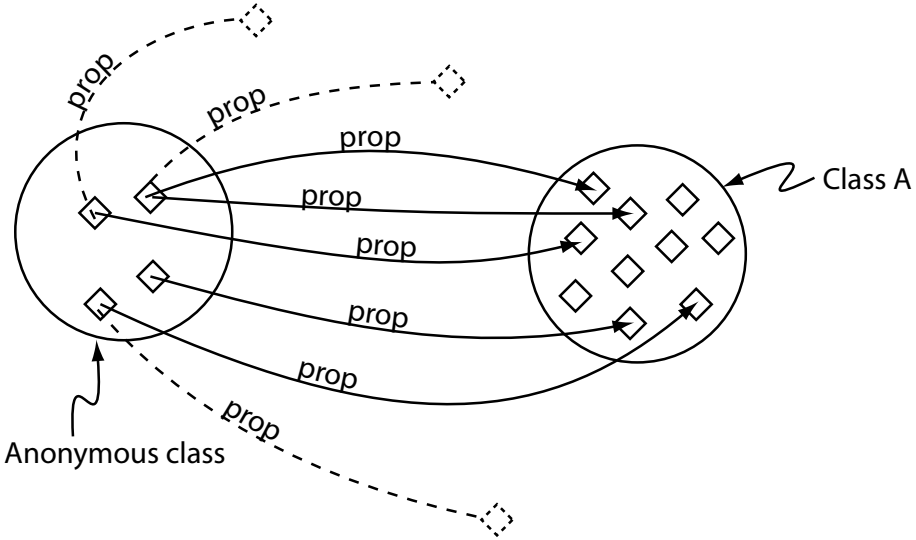
- Beispiel: Erstellen der Unterklasse „Käsepizza“ von „Pizza“, die mindestens einen Belag der Klasse „Käse“ hat

Necessary & Sufficient

- -> Defined Class

- Beispiel: Drag&Drop der Bedingungen in das Necessary & Sufficient-Feld

someValuesFrom – Existential Restriction

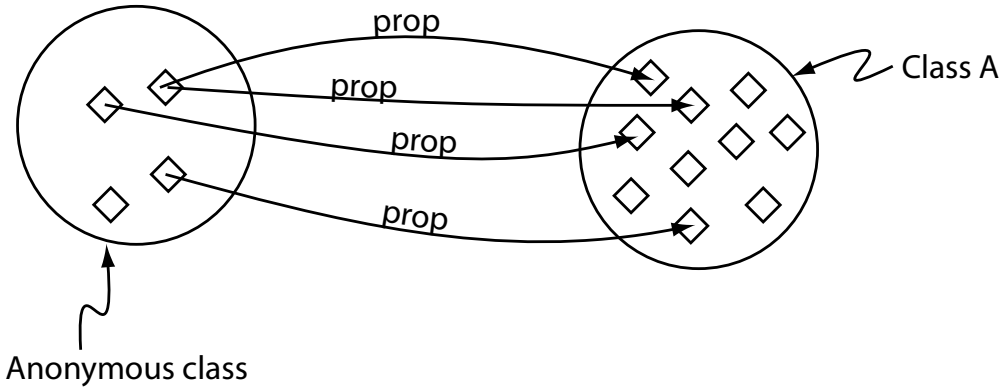


- Beispiel: ∃ hatBelag Morzarella

someValuesFrom – Existential Restriction

- Aufgaben:
  1. Erstelle eine Pizza Margherita
  2. Die Pizza Margherita hat einen Morzarellabelag
  3. Die Pizza Margherita hat einen Tomatenbelag

allValuesFrom – Universial Restriction



- Beispiel: ∀ hatBelag Morzarella

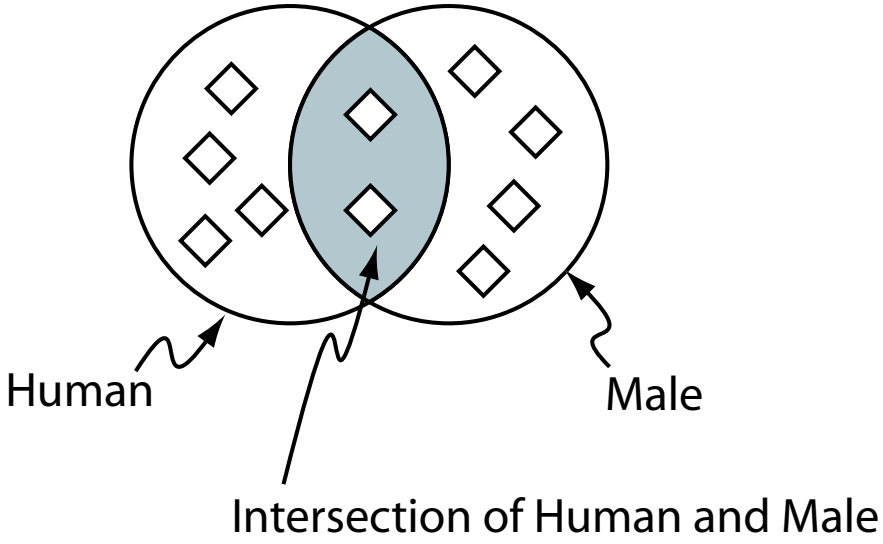
Kombination

- Wenn universal, dann auch existentiell !
- ∀ hatBelag Morzarella

-> ∃ hatBelag Morzarella

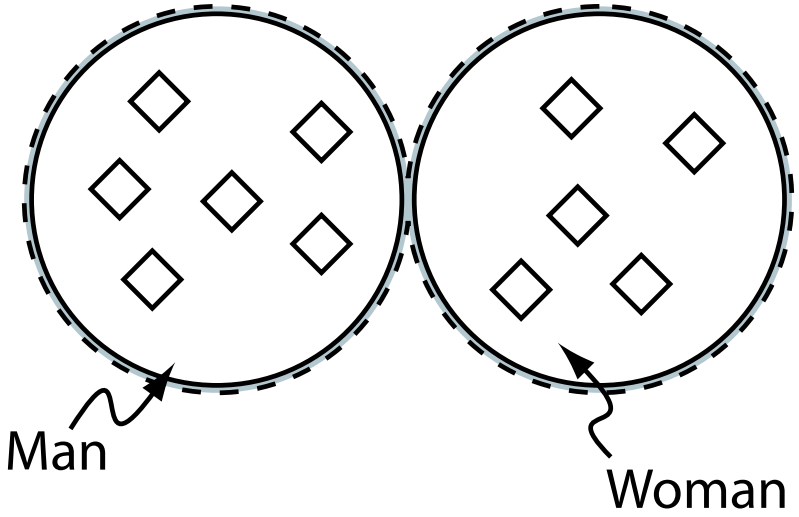
Intersection Class – AND

- Beispiel: Human ⊓ Male



Union Class – OR

- Beispiel: Man ⊔ Woman



Aufgabe

- Die Instanzen der Klasse Margherita dürfen nur Käse und Tomate als Belag haben

Lösung


NECESSARY & SUFFICIENT
NECESSARY
Pizza
hatBelag some Tomaten
hatBelag some Mozzarella
hatBelag only (Mozzarella or Tomaten)
hatBelag some Belag
[from Pizza]
hatTeig some Teig
[from Pizza]

Minimale Kardinalität – ≥

- Minimale Anzahl der Beziehungen
- Beispiel: interestingPizza ≥ hatBelag 3

Maximale Kardinalität – ≤

- Maximale Anzahl der Beziehungen
- Beispiel: cheapPizza ≤ hatBelag 2

Kardinalität - =

- exakte Anzahl der Beziehungen
- Beispiel: Margherita = hatBelag 2

complementOf – ¬

- beinhaltet Instanzen, die nicht in der Klasse sind, zu der sie komplementär ist
- Beispiel: PizzaFungi not (hatBelag some Salami)

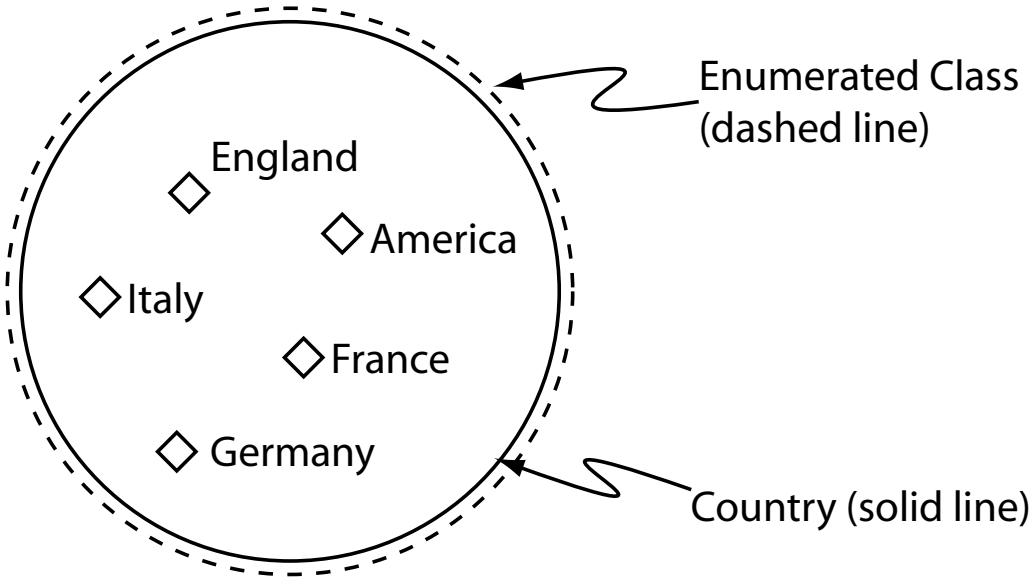
- Aufgabe:
  1. Vegetarische Pizza erstellen
  2. Vegetarische Pizza hat kein Fleisch als Belag
  3. Beschränkungen als necessary & sufficient deklarieren

Lösung


NECESSARY & SUFFICIENT
not (hatBelag some Fleisch)
≡
NECESSARY
Pizza
⊆
hatBelag some Belag
[from Pizza]
⊆
hatTeig some Teig
[from Pizza]
⊆

enumeration – {...}

- Beispiel: Country {America England France Germany Italy}



enumeration – {...}

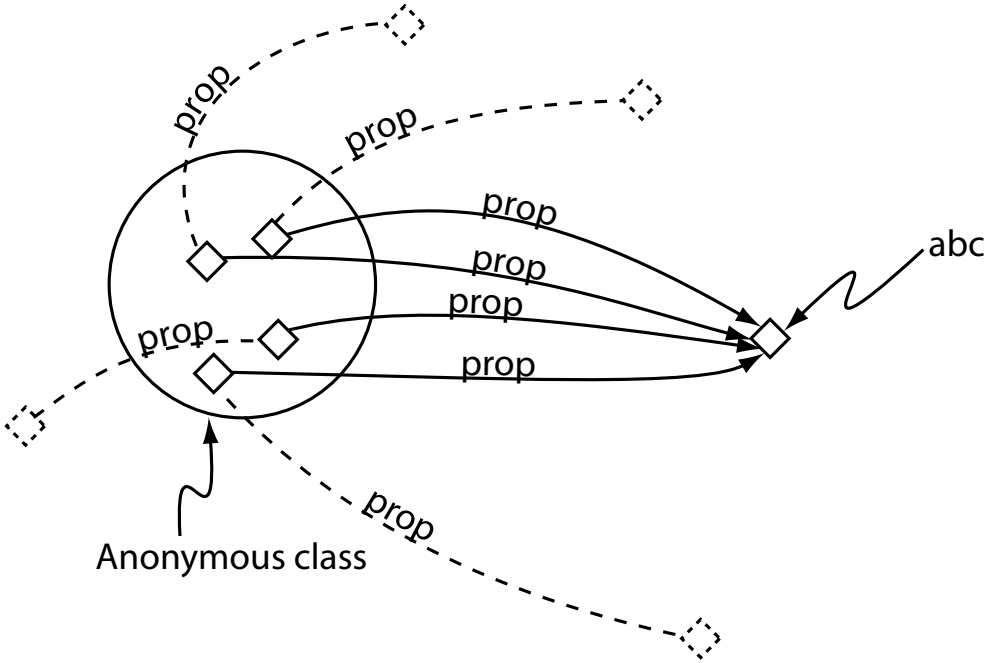
- Aufgabe:
  1. Erstelle die Klasse „Land“
  2. Erstelle die Instanzen Amerika Deutschland Italien
  2. Definiere die Klasse durch diese Instanzen
  3. Beschränkungen als necessary & sufficient deklarieren

Lösung


NECESSARY & SUFFICIENT
{Amerika Italien Deutschland}
≡
NECESSARY
owl:Thing
⊑

hasValue – ∋

- Beispiel: prop ∋ abc



hasValue

- Aufgabe:
  1. Erstelle die Eigenschaft „hatHeimatland“
  2. Erstelle eine Einschränkung, dass Mozzarella Italien als Heimatland hat.
  3. Beschränkungen als necessary & sufficient deklarieren

Lösung


NECESSARY & SUFFICIENT
hatHeimatland has Italien
=
NECESSARY
Käse
⊑

Quellen

- A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0: www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf - Abruf 2009-12-05

- Phillex: Lexikon der Philosophie. http://www.phillex.de/ - Abruf 2009-12-05

- Protege 4 Pizzas 10 Minutes. http://protegewiki.stanford.edu/index.php/Protege4Pizzas10Minutes - Abruf 2009-12-05

Logik in Protégé

OWL Element	Symbol	Key	Example	Meaning of example
allValuesFrom	∀	*	children ∀ Male	All children must be of type Male
someValuesFrom	∃	?	children ∃ Lawyer	At least one child must be of type Lawyer
hasValue	∋	$	rich ∋ true	The rich property must have the value true
cardinality	=	=	children = 3	There must be exactly 3 children
minCardinality	≥	>	children ≥ 3	There must be at least 3 children
maxCardinality	≤	<	children ≤ 3	There must be at most 3 children
complementOf	¬	!	¬ Parent	Anything that is not of type Parent
intersectionOf	⊓	&	Human ⊓ Male	All Humans that are Male
unionOf	⊔	|	Doctor ⊔ Lawyer	Anything that is either Doctor or Lawyer
enumeration	{...}	{ }	{male female}	The individuals male or female

Vielen Dank