Neuro-Symbolic Knowledge Representation and Reasoning

Uli Sattler
Professor in Computer Science
University of Manchester

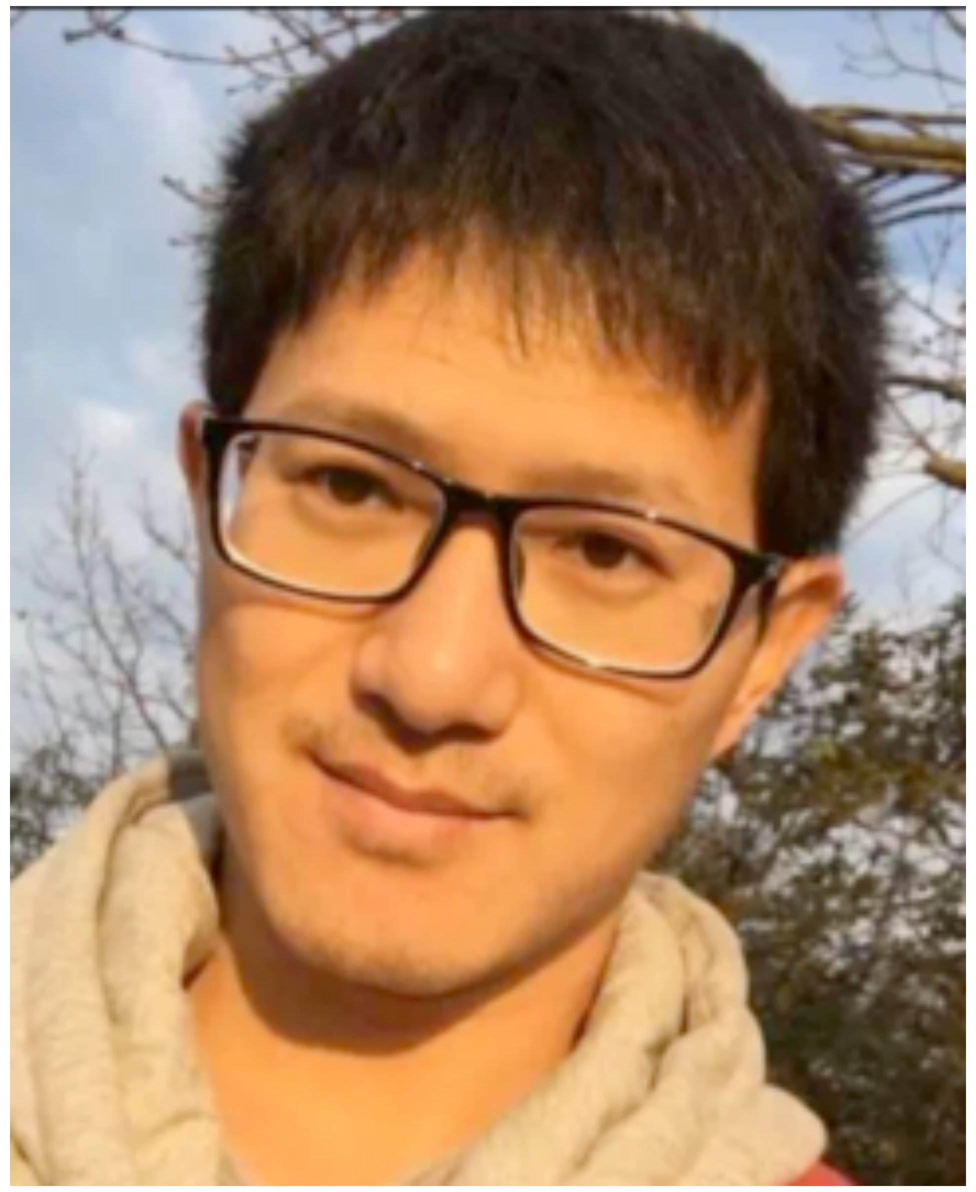
ESSAI 2024 Athens

Preamble

Uli Sattler
Professor in Computer Science
University of Manchester

ESSAI 2024 Athens

This course is



- introductory
- aimed at general computer scientist

- taught by
  - Jiaoyan Chen - days 3-5
  - Uli Sattler - days 1-2

- explores combination/integration/collaboration of
  - neural &
  - symbolic
    - approaches to knowledge representation, reasoning, ML, …

Overview of this course

Day	Topic	Concepts	Technologies
1	Knowledge Graphs	parsing/serialisation, queries, schemas, validation & reasoning	RDF(S), SPARQL, SHACL,
2	Ontologies	Facts & background knowledge, entailments, reasoning & materialisation	OWL, OWL API, Owlready, Protégé
3	Knowledge Graph Embeddings	Classis Es, literal-aware Es, variants, evaluation	TransE, TransR
4	Ontology Embeddings	Geometric embeddings, literal-aware OEs, soundness & completeness	ELEm, BoxEL, Box^2EL, OWL2Vec*, HiT
5	Applications & Outlook	Preprocessing, materialisation, evaluation	DeepOnto, mOWL

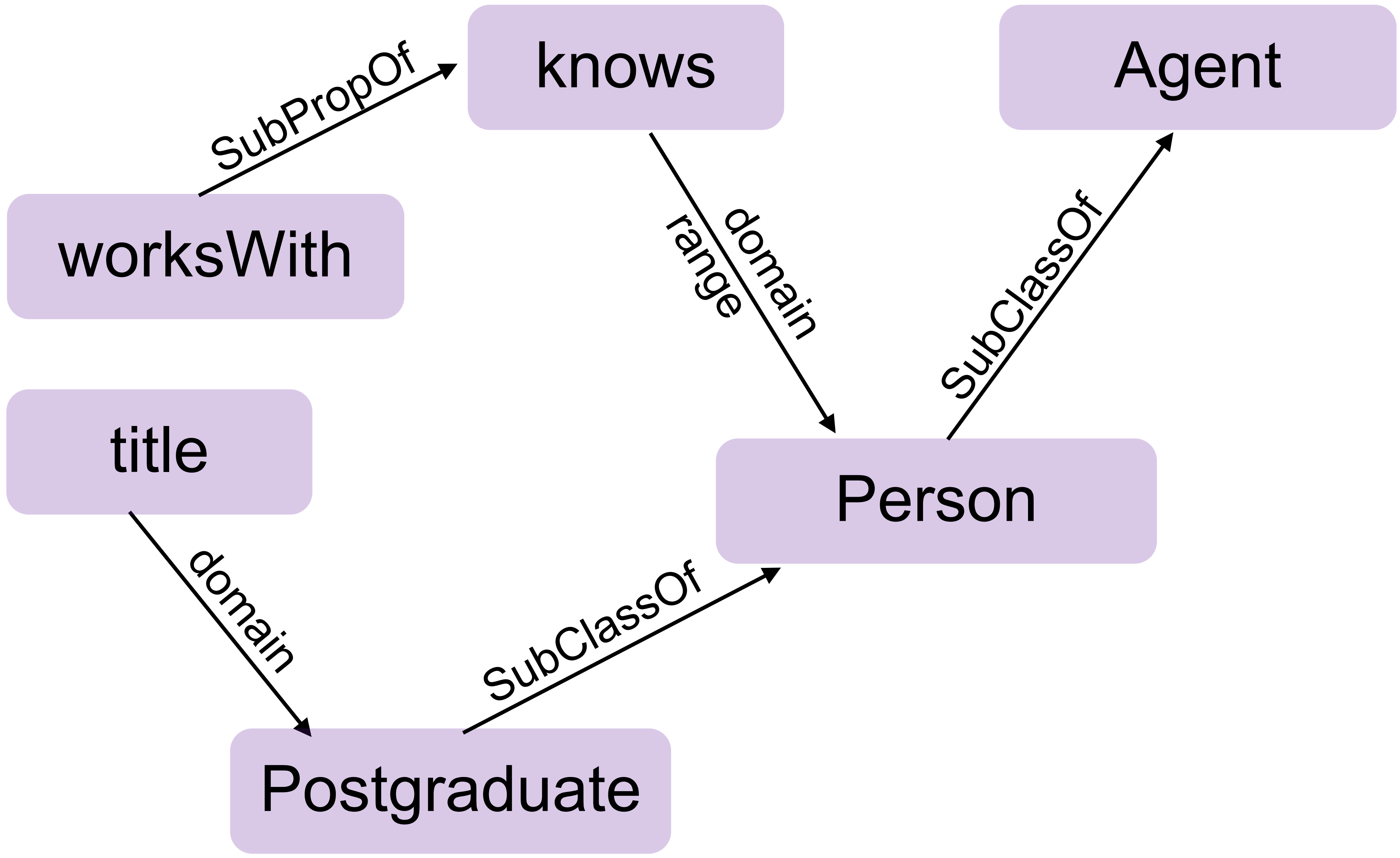
Brief Recap

Uli Sattler
Professor in Computer Science
University of Manchester

ESSAI 2024 Athens

Yesterday:

- Knowledge Graphs
  - RDF
  - factual and conceptual knowledge
- Querying of and Reasoning with KGs
  - SPARQL
  - RDFS
  - SHACL
  - Materialisation of reasoning results
    - making *explicit* the facts we know
    - helps us deal with *incomplete* KGs



Today:

- More on reasoning
  - OWL
    - what we can/can't say
    - why we should
  - materialisation…the many choices!

Day 2: Brief RDFS Recap and OWL Warm-Up

Uli Sattler
Professor in Computer Science
University of Manchester

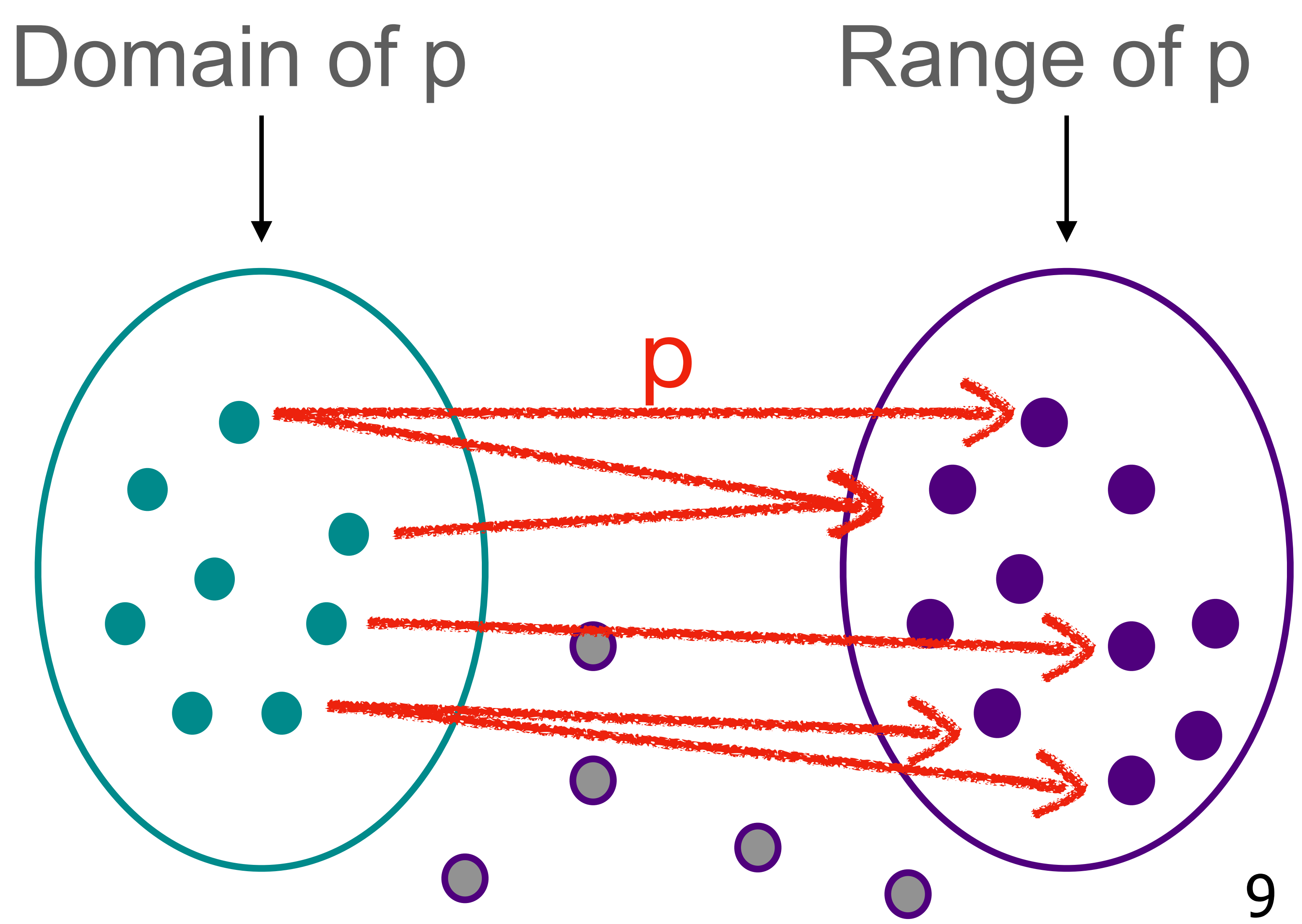
ESSAI 2024 Athens

The University of Manchester

In RDFS, we can state *conceptual knowledge*:

- rdfs:**subClassOf**
  - e.g. (foaf:Person, rdfs:subClassOf, foaf:Agent)
  - (ex:Woman, rdfs:subClassOf, foaf:Person)
- rdfs:**subPropertyOf**
  - e.g. (ex:worksWith, rdfs:subPropertyOf, foaf:knows)
- rdfs:**domain**
  - e.g. (ex:hasChild, rdfs:domain, foaf:Person)
    (foaf:currentProject, rdfs:domain, foaf:Person)
- rdfs:**range**
  - e.g. (ex:hasChild, rdfs:range, foaf:Person)
    (foaf:currentProject, rdfs:range, foaf:Project)

Only statements between *atoms*



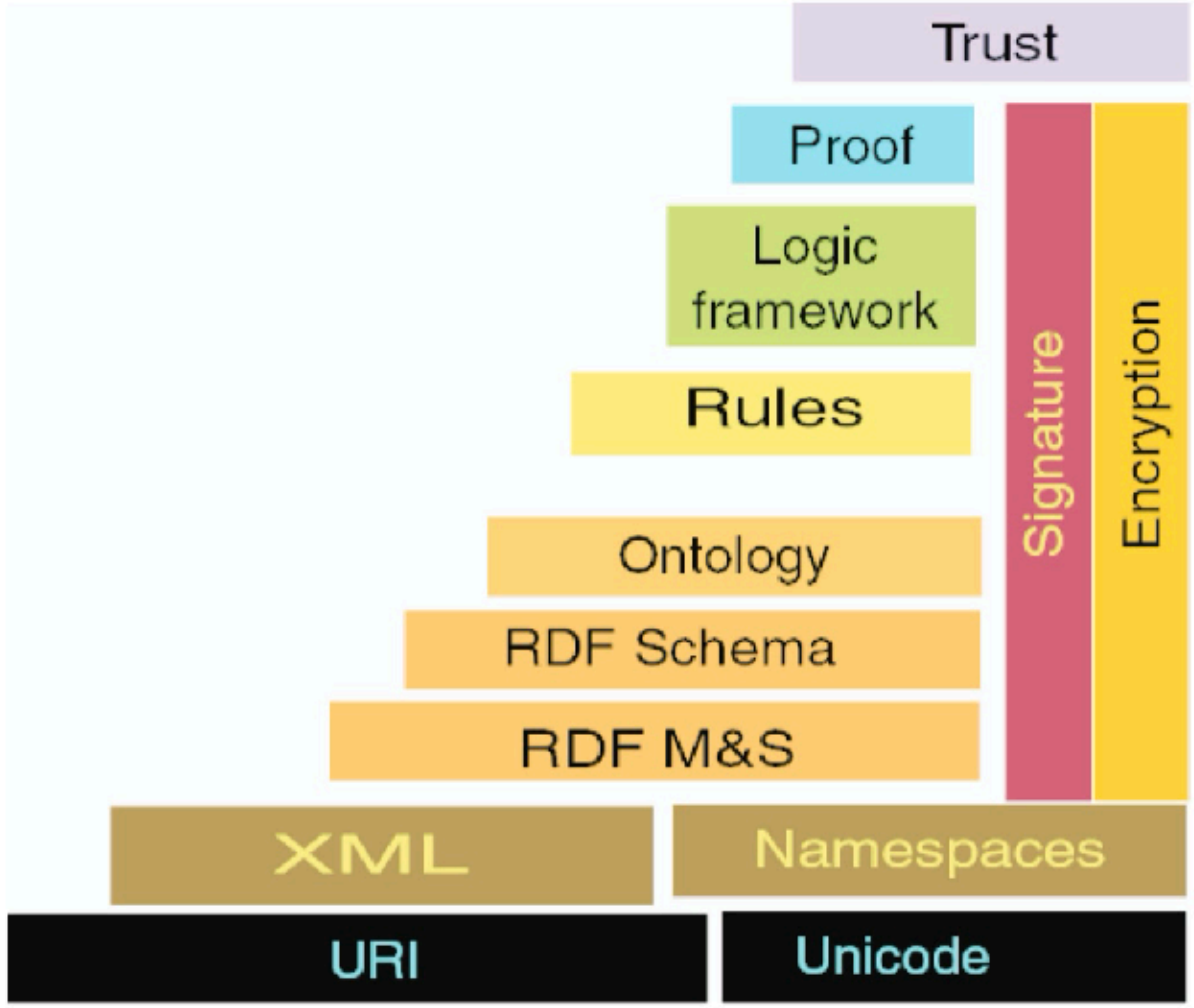
Background/Conceptual Knowledge

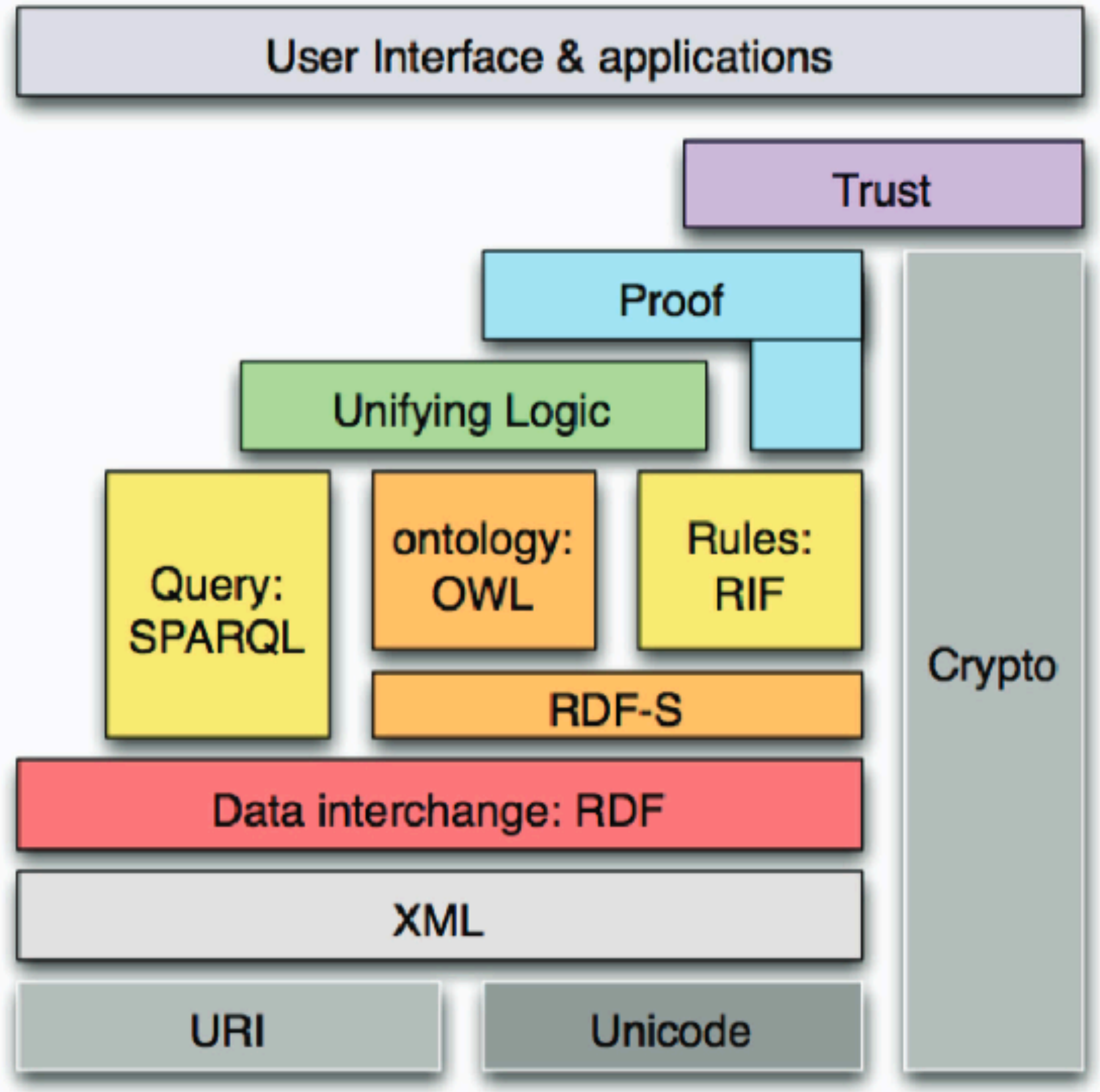
- can be *richer*
  - eg “hasChild is the inverse of isChildOf”
  - eg “isRelatedTo is transitive”
- may involve *expressions*
  - eg “Parents are those Persons who have a child”
  - eg “All children of a Person are also Persons”

- …we need a richer, more expressive language

The Semantic (Web) Layer Cake

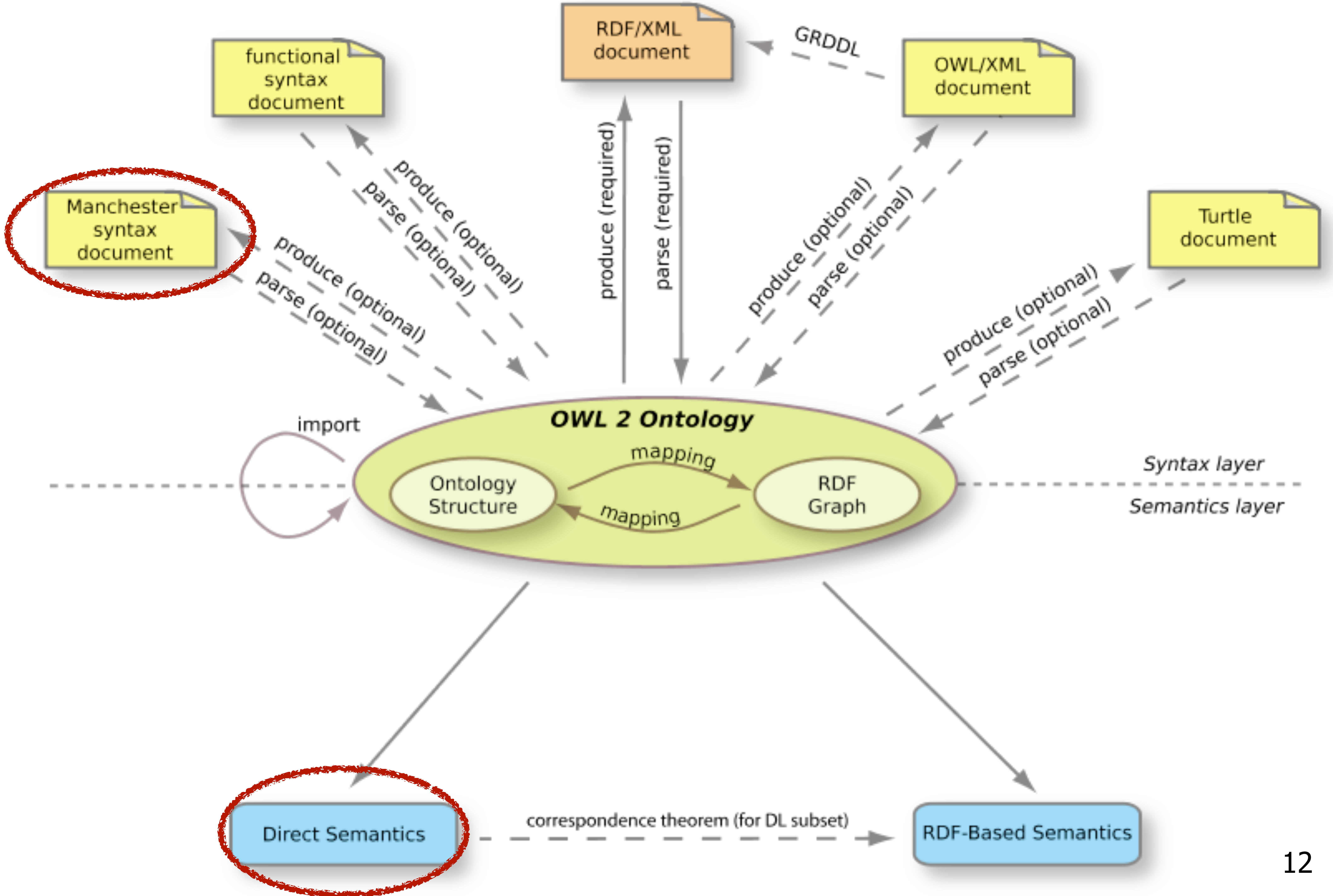
- To describe knowledge, we need a richer, more expressive language
  - background
  - conceptual
  - ontological
  - …
-





OWL is a *Web* Ontology Language

- entity names are IRIs
  - various syntaxes
    - RDF/XML
    - OWL/XML
    - Manchester syntax
    - …
- import mechanism
- version mechanism
- annotations of
  - entities
  - axioms
  - …



Different kinds of knowledge

RDFS

Controlled Vocabulary = {terms for concepts}

Taxonomy = CV + hierarchy

Classification system = Taxonomy + principles

Thesaurus = Taxonomy + more labels

Terminology = … + glossary/explanations

OWL

Ontology = … + logical axioms
+ well-defined semantics
+ reasoning
+ ….

Interlude

Concept

- denotes a set things
- class in RDF(S) or OWL
- 1 concept can have
  - more than 1 terms
    - Person, Human, Peo
  - different terms in diffe
    or contexts

Term

- is a string
  - in a language
- of many strings

In RDFS/OWL

- use class names as you like
- use annotations for
  - labels
  - preferred labels
  - labels in different languages/contexts

What is an OWL ontology?

- A set of *statements* about *entities*
  - eg, Person is a subclass of Agent
  - eg, worksFor is a subProperty of knows
- with a well-defined, logic-based *semantics*
  - directly or
  - via a translation:
    - $\forall x\,.\,\text{Person}(x) \Rightarrow \text{Agent}(x)$
    - $\forall x, y\,.\,\text{worksFor}(x, y) \Rightarrow \text{knows}(x, y)$
- and (usable) (reasoning) services required to
  - design, evolve, maintain, and use ontologies.

Inside an ontology, we find

- classes (unary relations)
  - describing sets of elements, eg, Agent, Person, Course
- properties (binary relations)
  - relating elements, eg, worksFor, knows, hasChild
- individuals
  - describing elements, eg, jchen, sattler
- axioms
  - constraining how we interpret the above:
  - eg, Person is a SubClass of Agent
  - eg, Employee is a SubClass of Person and works for some Company
- annotations
  - to record information about terms, axioms, …

Next:

- Dive into OWL
  - what we can see
  - how we can write it down
  - what it means
  - how to reason about it
- Back to materialisation of reasoning results

Day 2: OWL Syntax

Uli Sattler
Professor in Computer Science
University of Manchester

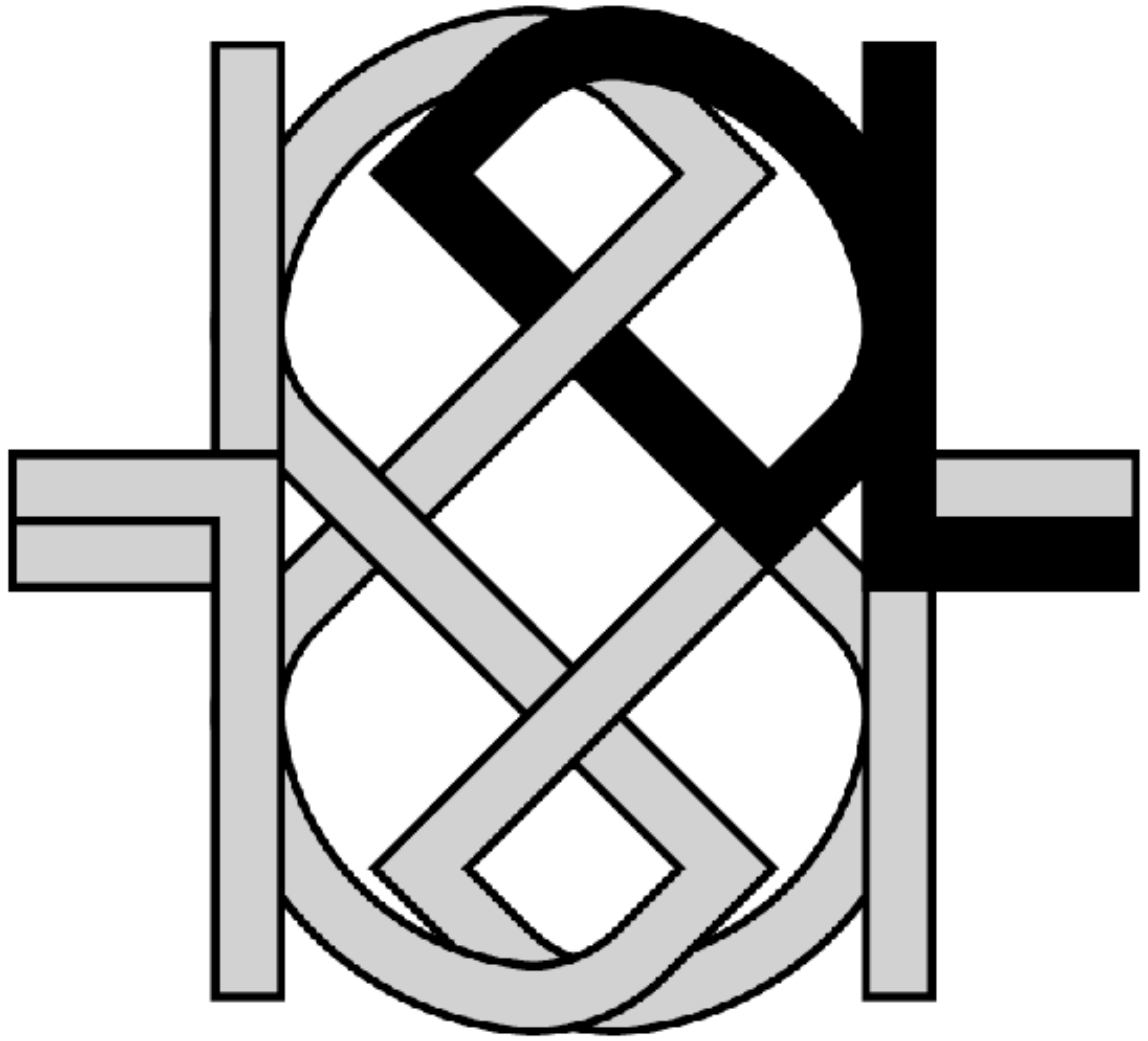
ESSAI 2024 Athens

OWL

- is based on *description logics*
  - a well-understood family of logic for KR&R
  - with *nice* properties:
    - decidable reasoning problems with
    - known computational complexity
- comes in 3 flavours/*profiles*
  - for different applications
- comes in many different syntaxes
  - so far, I used mainly “English”…

Description Logics

- decidable fragments of First Order Logic
  - more expressive than Boolean Logic
  - less than FOL
- closely related to
  - modal logics, guarded fragments
  - hybrid logics
- capture monotonic aspects of
  - frame-based systems
  - semantic networks
- complexity of reasoning ranges
  - from AC_0 via polynomial to ExpTime and NExpTime



OWL Axioms - an example

Manchester syntax

Inflammation	*SubClassOf*	Disease
HeartDisease	*EquivalentClass*	Disease *and* hasLoc *some* Heart
Endocarditis	*EquivalentClass*	Inflammation *and* hasLoc *some* Endocardium

- NCI Thesaurus
  - ~300K terms/classes
  - since 2000
  - since 2003 in OWL, monthly version, +800 terms/month
- …in OWL, published both
  - as a thesaurus ~ inferred concept hierarchy
  - in OWL, including underlying logical axioms, see BioPortal

Inflammation *SubClassOf* Disease

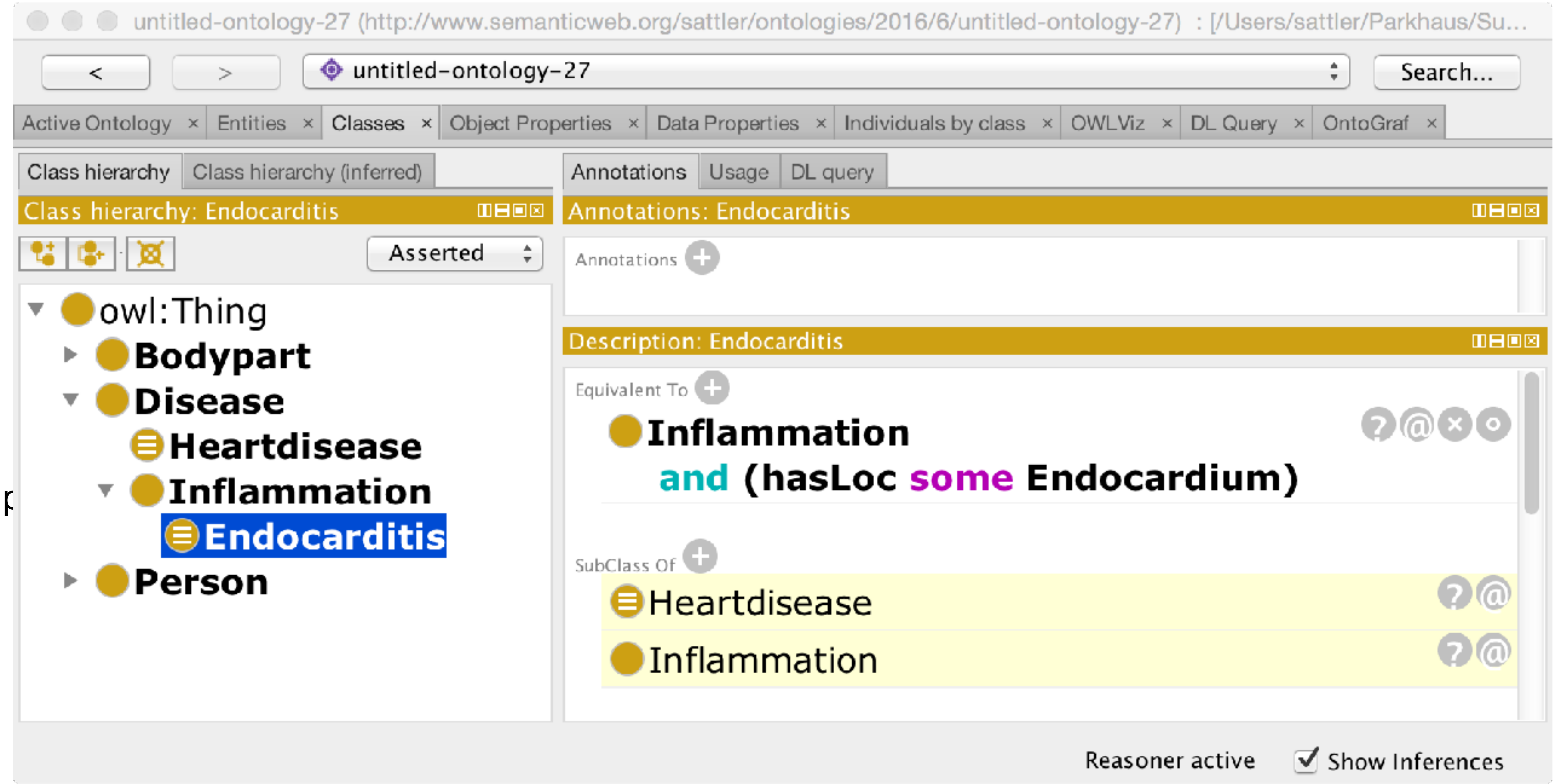
HeartDisease *EquivalentClass* Disease *and* hasLoc *some* Heart

Endocarditis *EquivalentClass* Inflammation *and* hasLoc *some* Endocardium

OWL Axioms - in Protégé

an OWL editor

see http://protege.stanford.edu/products.p



OWL Axioms in First Order Predicate Logic

Inflammation *SubClassOf* Disease

HeartDisease *EquivalentClass* Disease *and*
hasLoc *some* Heart

Endocarditis *EquivalentClass* Inflammation *and*
hasLoc *some* Endocardium

$$\forall x.\mathrm{Inflammation}(x) \Rightarrow \mathrm{Disease}(x)$$

$$\forall x.\mathrm{HeartDisease}(x) \Leftrightarrow \mathrm{Disease}(x) \wedge \exists y.(\mathrm{hasLoc}(x,y) \wedge \mathrm{Heart}(y))$$

$$\forall x.\mathrm{Endocarditis}(x) \Leftrightarrow \mathrm{Inflammation}(x) \wedge \exists y.(\mathrm{hasLoc}(x,y) \wedge \mathrm{Endocardium}(y))$$

OWL Axioms in Description Logic

Inflammation *SubClassOf* Disease

HeartDisease *EquivalentClass* Disease *and*
hasLoc *some* Heart

Endocarditis *EquivalentClass* Inflammation *and*
hasLoc *some* Endocardium

$$\text{Inflammation} \sqsubseteq \text{Disease}$$

$$\text{HeartDisease} \equiv \text{Disease} \sqcap \exists \text{hasLoc}.\text{Heart}$$

$$\text{Endocarditis} \equiv \text{Inflammation} \sqcap \exists \text{hasLoc}.\text{Endocardium}$$

Entities in OWL

Class Name

Class Expression

Inflammation *SubClassOf* Disease

Axiom

HeartDisease *EquivalentClass* Disease *and*
hasLoc *some* Heart

Endocarditis *EquivalentClass* Inflammation *and*
hasLoc *some* Endocardium

Bob *Types* Person *and*
(suffersFrom *some*
(Inflammation *and* (hasLoc some Endocardium)))

Property Name

Different kinds of OWL axioms

For a complete list, see
OWL 2 Primer
https://www.w3.org/TR/owl2-primer/

- about classes

Inflammation	*SubClassOf*	Disease
HeartDisease	*EquivalentClass*	Disease *and* hasLoc *some* Heart

- about properties

hasDaughter	*SubPropertyOf*	hasChild
hasPart	*InversePropertyOf*	isPartOf

- about individuals

Bob	*Types*	Person *and* (suffersFrom *some* Inflammation)
Bob	*Facts*	hasDaughter Mary

Day 2: OWL Entailments

Uli Sattler
Professor in Computer Science
University of Manchester

ESSAI 2024 Athens

Meaning of axioms…

- like in FOL, meaning/semantics
  is defined in terms of *an interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where
  - $\Delta^{\mathcal{I}}$ is a non-empty set, the *interpretation domain*
  - $\cdot^{\mathcal{I}}$ is a mapping that maps each
    - class name A to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
    - property name p to a binary relation $p^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
- plus
  - mechanism to interpret class expression, eg $A \sqcap B$
  - specification of what it means for $\mathcal{I}$ to satisfy axioms/an ontology …

Why so complicated?

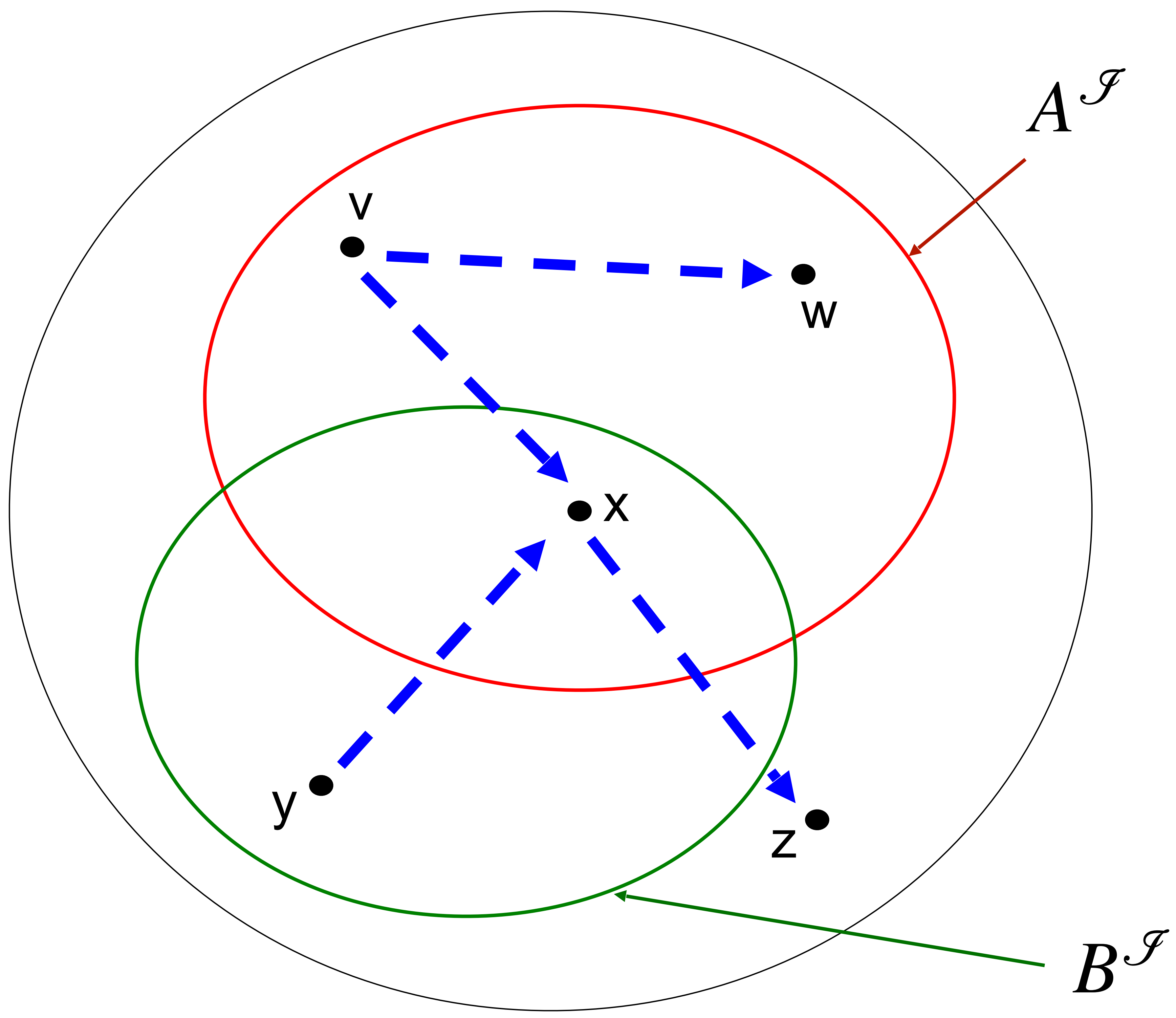
Eg to analyse algorithms

Eg as spec for reasoners

correctness complexity

We can *draw* interpretations

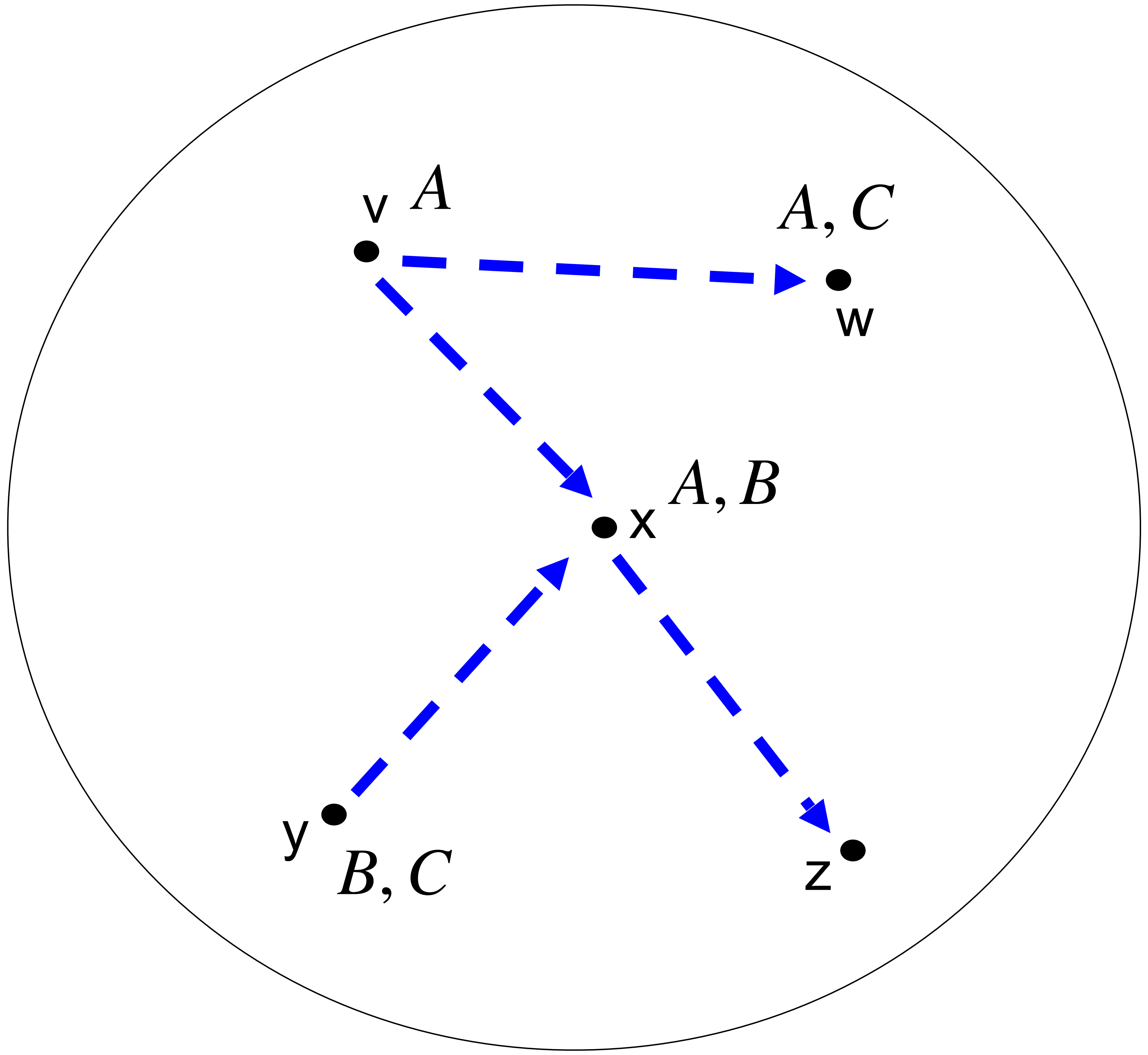
- $\Delta^{\mathcal{I}}$ = {v, w, x, y, z}
- $A^{\mathcal{I}}$ = {v, w, x}
- $B^{\mathcal{I}}$ = {x, y}
- $C^{\mathcal{I}}$ = {w, y}
- $D^{\mathcal{I}} = \emptyset$
- $p^{\mathcal{I}}$ = {(v, w), (v, x), (y, x), (x, z)}



$C^{\mathcal{I}}$???

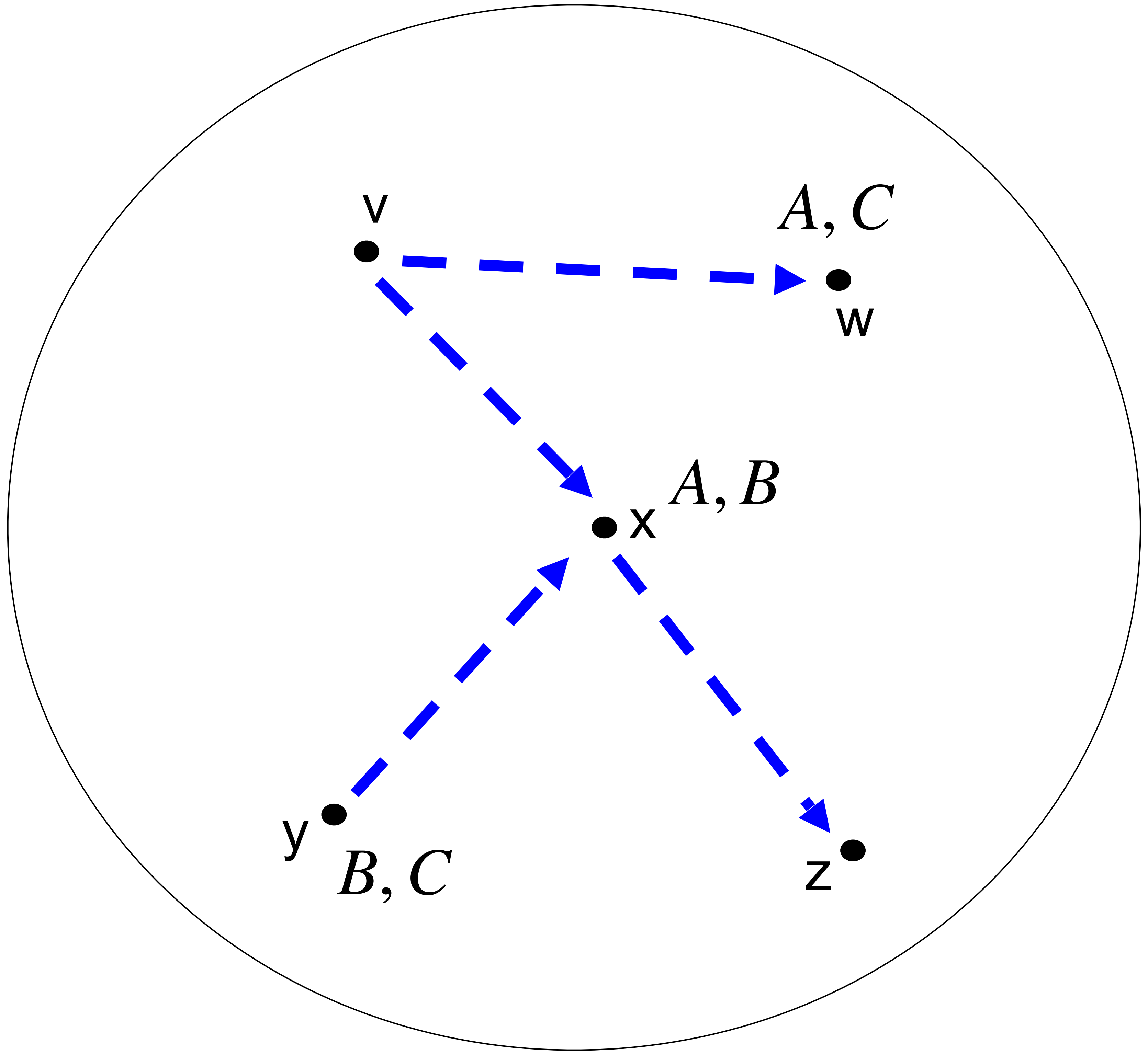
We can *draw* interpretations

- $\Delta^{\mathcal{I}}$ = {v, w, x, y, z}
- $A^{\mathcal{I}}$ = {v, w, x}
- $B^{\mathcal{I}}$ = {x, y}
- $C^{\mathcal{I}}$ = {w, y}
- $D^{\mathcal{I}} = \emptyset$
- $p^{\mathcal{I}}$ = {(v, w), (v, x), (y, x), (x, z)}



Drawing Interpretations

- helps understand semantics of OWL
- check/re-read the definition:
  - what size can the domain have?
  - what size are extensions?
  - which restrictions are on them?
  - what's a really small interpretation?
  - what's a really big interpretation?



Interpretations of Class Expressions

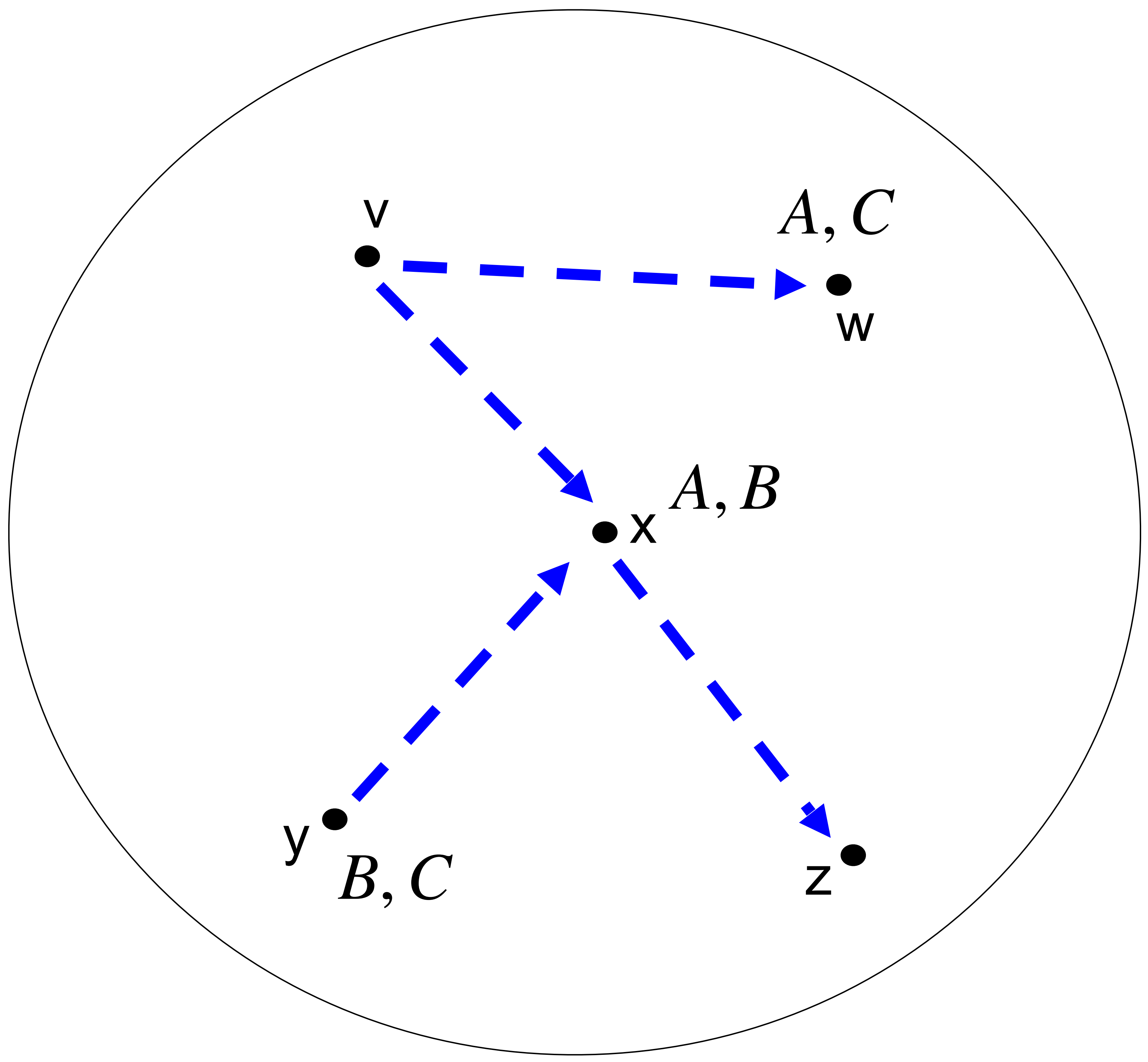
Constructor in Manchester Syntax	**Example**	**Interpretation**
Class name	*Human*	$Human^I \subseteq \Delta$
Thing	n/a	Δ
Nothing	n/a	$\emptyset$
and	*Human* and *Male*	$Human^I \cap Male^I$
or	*Doctor* or *Lawyer*	$Doctor^I \cup Lawyer^I$
not	not *Male*	$\Delta \setminus Male^I$

Interpretations of More Class Expressions

Constructor in Manchester Syntax	**Example**	**Interpretation**
some	*hasChild* some *Lawyer*	$\{e \in \Delta \mid$ there is some f: $(e,f) \in \mathit{hasChild}^I$ and $f \in \mathit{Lawyer}^I\}$
only	*hasChild* only *Doctor*	$\{e \in \Delta \mid$ for all $f \in \Delta$: if $(e,f) \in \mathit{hasChild}^I$ then $f \in \mathit{Doctor}^I\}$
min	*hasChild* min 2 *Tall*	$\{e \in \Delta \mid$ there are at least 2 $f \in \Delta$ with $(e,f) \in \mathit{hasChild}^I$ and $f \in \mathit{Tall}^I\}$
max	*hasChild* max 2 *Tall*	$\{e \in \Delta \mid$ there are at most 2 $f \in \Delta$ with $(e,f) \in \mathit{hasChild}^I$ and $f \in \mathit{Tall}^I\}$

Interpretation of Classes - let's see

- $(\text{not B})^{\mathcal{I}}$ =
- $(\text{A and B})^{\mathcal{I}}$ =
- $((\text{not A}) \text{ or B})^{\mathcal{I}}$ =
- $(\text{R some B})^{\mathcal{I}}$ =
- $(\text{R only B})^{\mathcal{I}}$ =
- $(\text{R some (R some A)})^{\mathcal{I}}$ =
- $(\text{R some not(A or B)})^{\mathcal{I}}$ =
- $(\text{R min 1.Thing})^{\mathcal{I}}$ =
- $(\text{R max 1.Thing})^{\mathcal{I}}$ =



Semantics of Axioms and Ontology

- An interpretation $\mathcal{I}$ *satisfies* an axiom of the form
  - C *SubClassOf* D if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - C *EquivalentTo* D if $C^{\mathcal{I}} = D^{\mathcal{I}}$
  - p *SubPropertyOf* q if $p^{\mathcal{I}} \subseteq q^{\mathcal{I}}$
  - …
  - x *Type* C if $x^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $x \; p \; y$ if $(x^{\mathcal{I}}, y^{\mathcal{I}}) \in p^{\mathcal{I}}$
- An interpretation $\mathcal{I}$ *satisfies* an ontology if $\mathcal{I}$ satisfies all axioms in it
  - we call $\mathcal{I}$ a *model* of the ontology

Open World Assumption

An ontology can have *many* models

Model of an ontology = a *fitting* interpretation

Entailments of an Ontology

Let $\mathcal{O}$ be an ontology, α an axiom, and A, B classes, b an individual name:

- $\mathcal{O}$ is *consistent* ______ if there exists some model $\mathcal{I}$ of $\mathcal{O}$
  - i.e., there is an interpretation that satisfies **all** axioms in $\mathcal{O}$
  - i.e., $\mathcal{O}$ isn't self contradictory
- $\mathcal{O}$ *entails* α (written $\mathcal{O} \models \alpha$) ______ if α is satisfied in **all** models of $\mathcal{O}$
  - i.e., α is a consequence of the axioms in $\mathcal{O}$
- A is *satisfiable* w.r.t. $\mathcal{O}$ ______ if $\mathcal{O} \not\models A$ SubClassOf Nothing
  - i.e., there is a model $\mathcal{I}$ of $\mathcal{O}$ with $A^{\mathcal{I}} \neq \emptyset$
- b is an instance of A w.r.t. $\mathcal{O}$ (written $\mathcal{O} \models b : A$) _ if $b^{\mathcal{I}} \in A^{\mathcal{I}}$ in every model $\mathcal{I}$ of $\mathcal{O}$

…let's see this in Protégé!?

Entailment Examples

Patient	*EquivalentClass*	Person *and* suffersFrom *some* Disease
Inflammation	*SubClassOf*	Disease
HeartDisease	*EquivalentClass*	Disease *and* hasLoc *some* Heart
Endocarditis	*EquivalentClass*	Inflammation *and* hasLoc *some* Endocardium
Endocardium	*SubClassOf*	Bodypart *and* isPartOf *some* Heart
hasLoc o isPartOf *SubPropertyOf* hasLoc		

⊨ Endocarditis *SubClassOf* HeartDisease

Entailment Examples

Patient *EquivalentClass* Person *and* suffersFrom *some* Disease

Inflammation *SubClassOf* Disease

HeartDisease *EquivalentClass* Disease *and* hasLoc *some* Heart

Endocarditis *EquivalentClass* Inflammation *and* hasLoc *some* Endocardium

Endocardium *SubClassOf* Bodypart *and* isPartOf *some* Heart

hasLoc o isPartOf *SubPropertyOf* hasLoc

Bob *Type* (Person *and*
suffersFrom *some* (Inflammation *and*
hasLoc *some* Endocardium))

⊨ Bob *Type* Patient

Entailment Examples

Patient *EquivalentClass* Person *and* suffersFrom *some* Disease

Inflammation *SubClassOf* Disease

HeartDisease *EquivalentClass* Disease *and* hasLoc *some* Heart

Endocarditis *EquivalentClass* Inflammation *and* hasLoc *some* Endocardium

Endocardium *SubClassOf* Bodypart *and* isPartOf *some* Heart

hasLoc o isPartOf *SubPropertyOf* hasLoc

Bob *Type* (Person *and*
suffersFrom *some* (Inflammation *and*
hasLoc *some* Endocardium))

⊨ Bob *Type* (Patient *and* suffersFrom *some* HeartDisease)

OWL Reasoning - how?

- OWL reasoners
  - implement *decision procedures* for consistency/entailments, and classify ontologies
- Protégé
  - interacts with reasoners via the OWL API
  - shows results as
    - inferred class hierarchy where
    - unsatisfiable classes are red and you get a
    - warning (red triangle) if O is inconsistent

OWL Reasoning - how?

- OWL reasoners
  - implement highly optimised algorithms which decide complex logical decision problems
  - for example
    - ELK
    - Hermit
    - JFact
    - Conclude
  - use via
    - OWL API
    - OWL2Ready
    - …

OWL Reasoning - why?

- Take a
  - *factual* KG $\mathscr{K}$
  - *rich* ontology $\mathscr{O}$
- ask reasoner to
  - find all entailments of $\mathscr{K} \cup \mathscr{O}$
  - add these to $\mathscr{K}$
    - so that $\mathscr{K}$ + entailments is *harmonised* & *complete*
- use $\mathscr{K}$ + entailments in downstream tasks

A KG can be completed in different ways

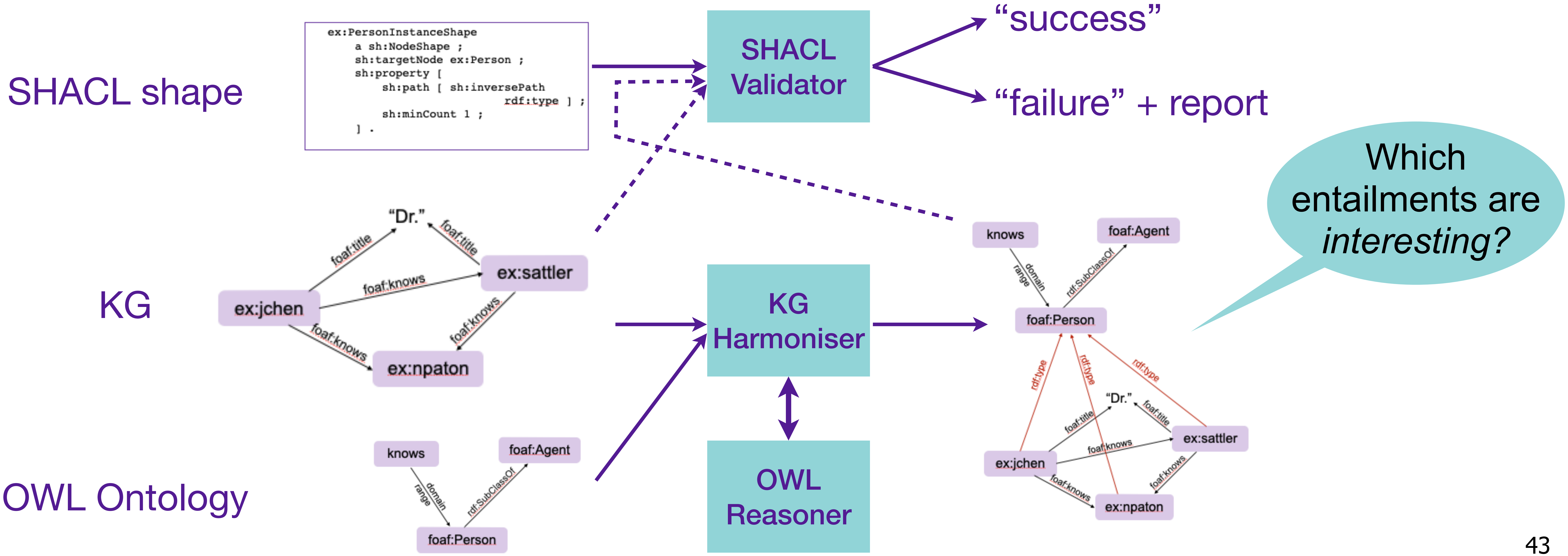
Graph embeddings depend on graph shape!

Only add *interesting* entailments

there are infinitely many entailments

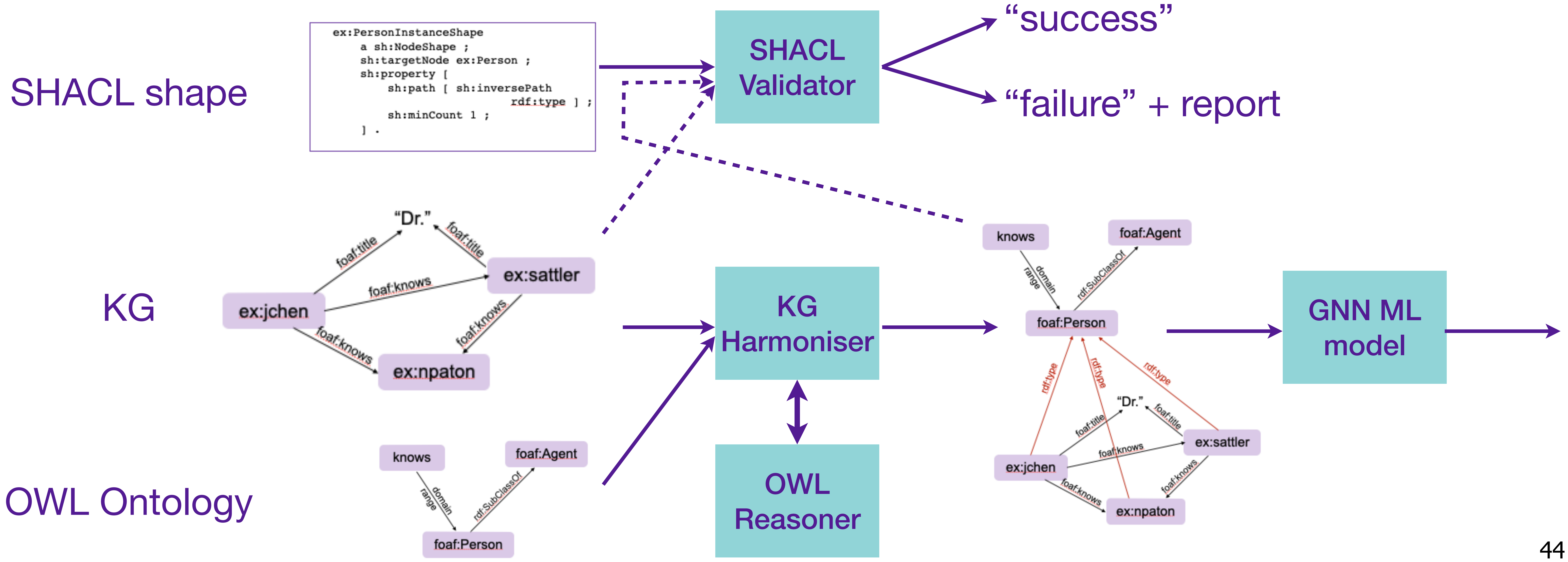
Materialising Ontology Consequences

- Slightly more involved



Materialising Ontology Consequences

- Slightly more involved - but far more powerful



Many things left out…

- how to reason exactly
- computational complexity of reasoning
- model theory of OWL and DL
- profiles
- modularity, module extraction
- explanation of entailments
- relationship with other formalisms
- query answering and rewriting
- …

Summary of today

- Factual KGs can be enhanced with rich conceptual knowledge
  - eg from OWL ontologies
    - with well-defined, logic based semantics and
    - powerful reasoners available
    - no need to reason yourself!
- Many ontologies are available
    - eg in Bioportal
  - that can be used to *harmonise* a factual knowledge graph
  - … in many different ways

Any questions?

Tomorrow: incorporating KG/ontologies in ML models

The End of Today's session

Polyglott persistence

- How can we **vary?**
  - Same core data model, same implementation
    - but different domain models
  - Same core data model, same domain model
    - different implementations, e.g., SQLite vs. MySQL
  - Same shape of core data model, same conceptual model
    - different formalisms!
      - Usually, but not always, implies different implementations
      - e.g. JSON and XML
- We can be **explicitly** or **implicitly** poly-
  - If we **encode** another data model into our home model
    - We are still poly-
    - But only implicitly so
    - Key Cost: Ad hoc implementation
  - If we split our domain model across multiple formalisms/implementations

Key point

- Understand your **domain**
  - What are you trying to represent and manipulate
- Understand your use case
  - including (frequent, relevant) queries, error sources,…
- Understand the **fit** between domain and data model(s)
  - To see where there are sufficiently good fits
- Understand your infrastructure

Question 1

Consider again the Conceptual Model you started to work on last week: can you

- finish/improve/extend it?
- add adjectives?
- add examples?

– format	– domain model	– robust
– formalism	– schema	– extensible
– core data model	– schema language	– scalable
– data model	– application	– self-describing
– database	– system	– valid
– external repr.	– internal repr.	– expressive
– …	– …	– verbose
		– …

Question 2

Consider a format for a reporting system for health & safety incidents, as exemplified by the printed example document:

- sketch a system for
  - gathering this data
  - reporting it monthly
- which kind of schema(s) would you use to describe it?
  - why?
- does this format make good use of XML's

Title Text

Good Bye!

- We hope you have learned a lot!
- It was a pleasure to work with you!
- Speak to us about projects
  - taster/MRes
  - MSc
- Enjoy the rest of your programme
  - COMP62421 query processing
  - COMP62342 rich modelling, inference semantic web, symbolic AI