Modelling Structured Domains With Logic

Despoina Magka, Boris Motik and Ian Horrocks

Department of Computer Science, University of Oxford

February 9, 2012

Outline

What is OWL?

What is OWL?

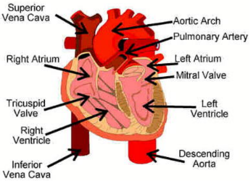
- Family of logic-based knowledge representation languages

What is OWL?

- Family of logic-based knowledge representation languages
- Web Ontology Language: a W3C standard, widely used in ontology-based applications, e.g. formal biomedical vocabularies

What is OWL?

- Family of logic-based knowledge representation languages
- Web Ontology Language: a W3C standard, widely used in ontology-based applications, e.g. formal biomedical vocabularies



What is OWL?

- Family of logic-based knowledge representation languages
- Web Ontology Language: a W3C standard, widely used in ontology-based applications, e.g. formal biomedical vocabularies

- Formal foundations of OWL provided by Description Logics

What is OWL?

- Family of logic-based knowledge representation languages
- Web Ontology Language: a W3C standard, widely used in ontology-based applications, e.g. formal biomedical vocabularies

- Formal foundations of OWL provided by Description Logics
- What are Description Logics?

What is OWL?

- Family of logic-based knowledge representation languages
- Web Ontology Language: a W3C standard, widely used in ontology-based applications, e.g. formal biomedical vocabularies

- Formal foundations of OWL provided by Description Logics
- What are Description Logics?
  $\rightsquigarrow$ Decidable fragments of first-order logic with well-understood computational properties

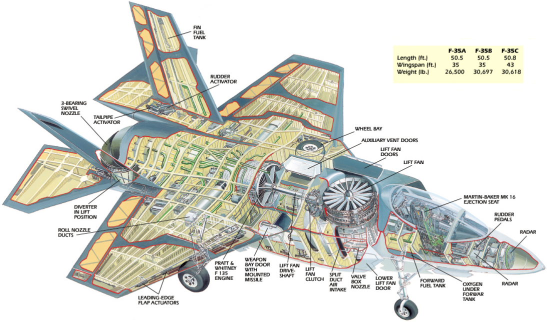
Modelling structured domains with OWL

- OWL used for the representation of complex structures:

Modelling structured domains with OWL

- OWL used for the representation of complex structures:

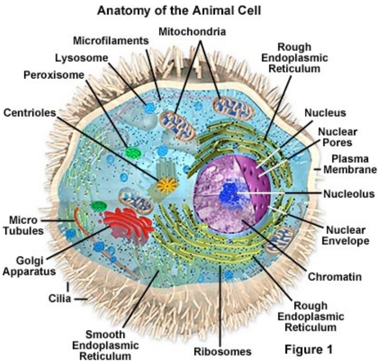
  - Aerospace



	F-35A	F-35B	F-35C
Length (ft.)	50.5	50.5	50.8
Wingspan (ft.)	35	35	43
Weight (lb.)	26,500	30,697	30,618

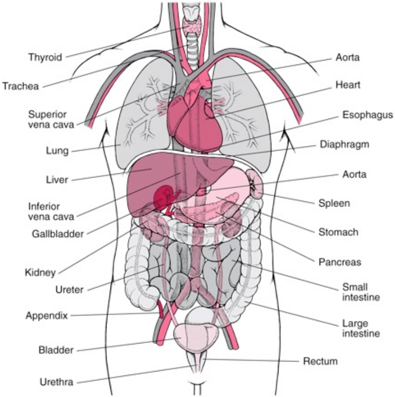
MODELLING STRUCTURED DOMAINS WITH OWL

- OWL used for the representation of complex structures:
  - Aerospace
  - Cellular biology



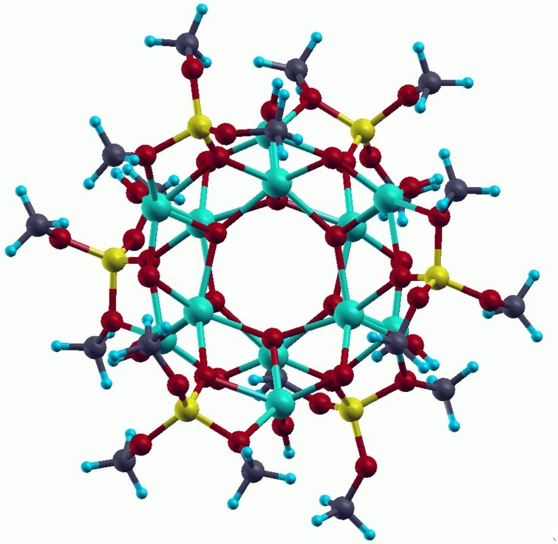
Modelling structured domains with OWL

- OWL used for the representation of complex structures:
  - Aerospace
  - Cellular biology
  - Human anatomy



MODELLING STRUCTURED DOMAINS WITH OWL

- OWL used for the representation of complex structures:
  - Aerospace
  - Cellular biology
  - Human anatomy
  - Molecules



THE CHEBI ONTOLOGY

- OWL ontology Chemical Entities of Biological Interest

THE CHEBI ONTOLOGY

- OWL ontology Chemical Entities of Biological Interest
  - Freely accessible dictionary of molecular entities

THE CHEBI ONTOLOGY

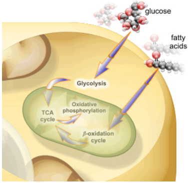
- OWL ontology Chemical Entities of Biological Interest
  - Freely accessible dictionary of molecular entities
  - High quality annotation and taxonomy of chemical compounds

THE CHEBI ONTOLOGY

- OWL ontology Chemical Entities of Biological Interest
  - Freely accessible dictionary of molecular entities
  - High quality annotation and taxonomy of chemical compounds
  - Interoperability between researchers

The ChEBI Ontology

- OWL ontology Chemical Entities of Biological Interest
  - Freely accessible dictionary of molecular entities
  - High quality annotation and taxonomy of chemical compounds
  - Interoperability between researchers
  - Drug discovery and elucidation of metabolic pathways



AUTOMATE CHEMICAL CLASSIFICATION

- ChEBI is manually incremented

Automate chemical classification

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries

Automate chemical classification

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year

Automate chemical classification

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000

Automate chemical classification

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000
- Each new molecule is subsumed by several chemical classes

AUTOMATE CHEMICAL CLASSIFICATION

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000
- Each new molecule is subsumed by several chemical classes
  - Is dinitrogen inorganic?

AUTOMATE CHEMICAL CLASSIFICATION

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000
- Each new molecule is subsumed by several chemical classes
  - Is dinitrogen inorganic?
  - Does cyclobutane contain a four-membered ring?

AUTOMATE CHEMICAL CLASSIFICATION

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000
- Each new molecule is subsumed by several chemical classes
  - Is dinitrogen inorganic?
  - Does cyclobutane contain a four-membered ring?
  - Is acetylene a hydrocarbon?

AUTOMATE CHEMICAL CLASSIFICATION

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000
- Each new molecule is subsumed by several chemical classes
  - Is dinitrogen inorganic?
  - Does cyclobutane contain a four-membered ring?
  - Is acetylene a hydrocarbon?
  - Does benzaldehyde contain a benzene ring?

Automate Chemical Classification

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000
- Each new molecule is subsumed by several chemical classes
  - Is dinitrogen inorganic?
  - Does cyclobutane contain a four-membered ring?
  - Is acetylene a hydrocarbon?
  - Does benzaldehyde contain a benzene ring?
- Speed up curating tasks with automated reasoning tools

AUTOMATE CHEMICAL CLASSIFICATION

- ChEBI is manually incremented
- Currently contains approx. 27,000 fully annotated entries
- Grows at a rate of 1,500 entries per curator per year
- Biologically interesting entities possibly > 1,000,000
- Each new molecule is subsumed by several chemical classes
  - Is dinitrogen inorganic? $\rightsquigarrow$Yes
  - Does cyclobutane contain a four-membered ring? $\rightsquigarrow$Yes
  - Is acetylene a hydrocarbon? $\rightsquigarrow$Yes
  - Does benzaldehyde contain a benzene ring? $\rightsquigarrow$Yes
- Speed up curating tasks with automated reasoning tools

(MIS)REPRESENTING RINGS WITH OWL

- Chemical compounds with rings are highly frequent

(Mis)representing rings with OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles

(MIS)REPRESENTING RINGS WITH OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL syntax does not offer access to variables

(MIS)REPRESENTING RINGS WITH OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL syntax does not offer access to variables

EXAMPLE

$\text{Uncle} \sqsubseteq \text{Male} \sqcap \exists \text{hasSibling}.\exists \text{hasChild}.(\text{Human})$

(Mis)representing rings with OWL

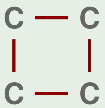
- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL has the tree-model property: each consistent OWL ontology has at least one tree-shaped model

(MIS)REPRESENTING RINGS WITH OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL has the tree-model property: each consistent OWL ontology has at least one tree-shaped model

EXAMPLE

$\text{Cyclobutane} \sqsubseteq \exists^{(=4)}\text{hasAtom}.(\text{Carbon} \sqcap \exists^{(=2)}\text{hasBond}.\text{Carbon})$

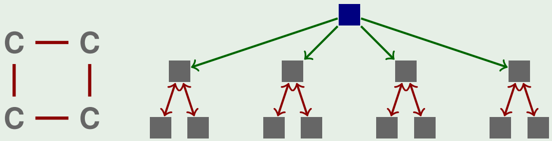


(Mis)representing rings with OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL has the tree-model property: each consistent OWL ontology has at least one tree-shaped model

Example

$\text{Cyclobutane} \sqsubseteq \exists^{(=4)}\text{hasAtom}.(\text{Carbon} \sqcap \exists^{(=2)}\text{hasBond}.\text{Carbon})$

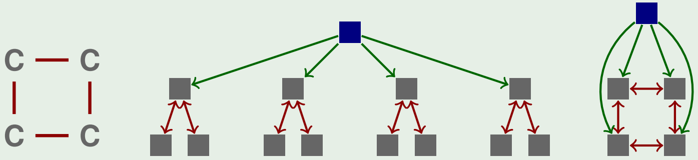


(MIS)REPRESENTING RINGS WITH OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL has the tree-model property: each consistent OWL ontology has at least one tree-shaped model

EXAMPLE

$$\text{Cyclobutane} \sqsubseteq \exists^{(=4)}\text{hasAtom}.(\text{Carbon} \sqcap \exists^{(=2)}\text{hasBond}.\text{Carbon})$$

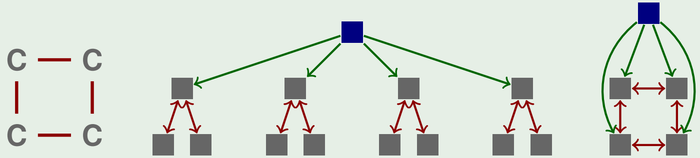


(MIS)REPRESENTING RINGS WITH OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL has the tree-model property: each consistent OWL ontology has at least one tree-shaped model

EXAMPLE

$\text{Cyclobutane} \sqsubseteq \exists^{(=4)}\text{hasAtom}.(\text{Carbon} \sqcap \exists^{(=2)}\text{hasBond}.\text{Carbon})$



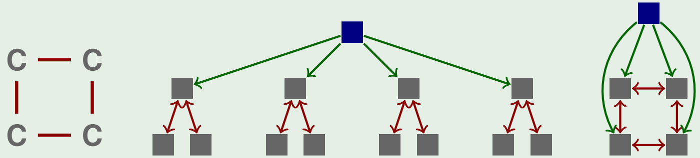
- OWL-based reasoning support

(MIS)REPRESENTING RINGS WITH OWL

- Chemical compounds with rings are highly frequent
- Fundamental inability of OWL to represent cycles
- OWL has the tree-model property: each consistent OWL ontology has at least one tree-shaped model

EXAMPLE

$$\text{Cyclobutane} \sqsubseteq \exists^{(=4)}\text{hasAtom}.(\text{Carbon} \sqcap \exists^{(=2)}\text{hasBond}.\text{Carbon})$$



- OWL-based reasoning support
  - Does cyclobutane contain a four-membered ring? ✗
  - Does benzaldehyde contain a benzene ring? ✗

OWL Extensions

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]

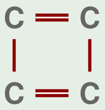
OWL Extensions

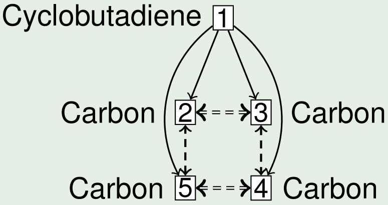
- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

OWL Extensions

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

Example

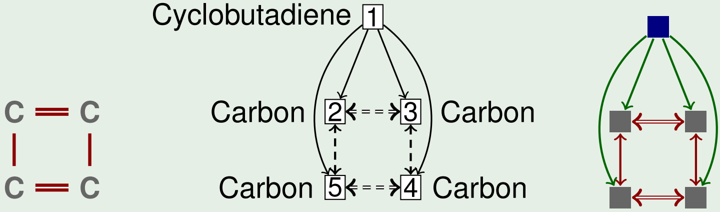




OWL Extensions

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

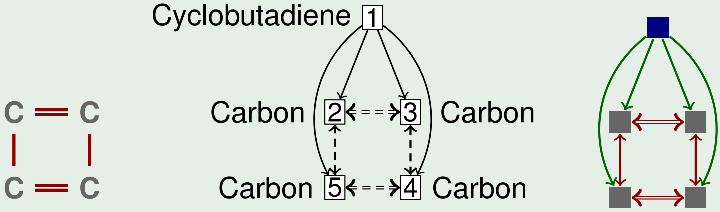
Example



OWL Extensions

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

Example

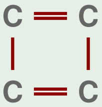


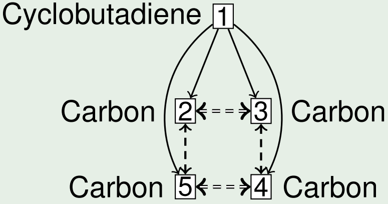
- Does cyclobutadiene have a conjugated four-membered ring?

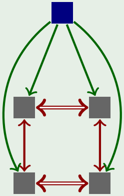
OWL Extensions

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

Example





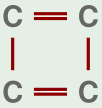


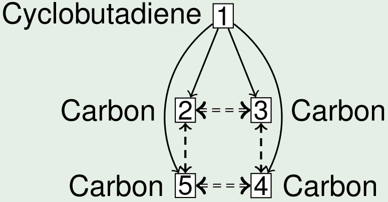
- Does cyclobutadiene have a conjugated four-membered ring? ✓

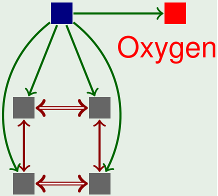
OWL Extensions

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

Example



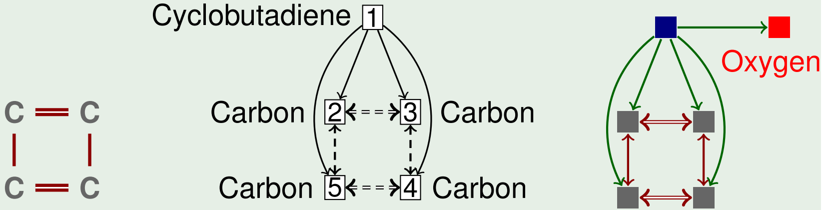




OWL EXTENSIONS

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

EXAMPLE

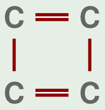


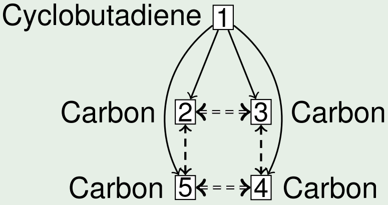
$\forall \text{hasAtom}.(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon}$

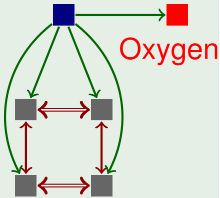
OWL EXTENSIONS

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

EXAMPLE







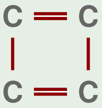
$\forall$hasAtom.(Carbon $\sqcup$ Hydrogen) $\sqsubseteq$ Hydrocarbon

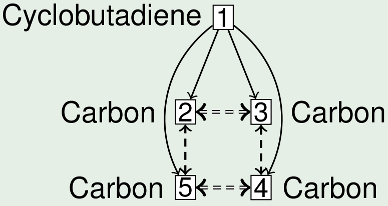
- Is cyclobutadiene a hydrocarbon?

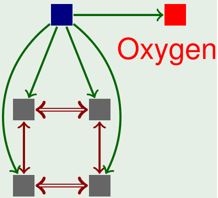
OWL Extensions

- Limitation of OWL to represent cycles (partially) remedied by extension of OWL with Description Graphs and rules [Motik et al., 2009]
- A Description Graph represents structures by means of a directed labeled graph

Example







$\forall \text{hasAtom}.(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon}$

- Is cyclobutadiene a hydrocarbon? ✕

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences.

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)
  Negation as failure $\leftrightarrow$ Closed-world assumption $\leftrightarrow$ Missing information treated as *false* (LP has it)

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)
  Negation as failure $\leftrightarrow$ Closed-world assumption $\leftrightarrow$ Missing information treated as *false* (LP has it)

Example

$$\mathsf{Icecream}(x) \wedge \neg\, \mathsf{BananaIscream}(x) \rightarrow \mathsf{Likes}(alice, x)$$

$$\mathsf{Icecream}(x) \wedge \textbf{not}\ \mathsf{BananaIscream}(x) \rightarrow \mathsf{Likes}(alice, x)$$

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)
  Negation as failure $\leftrightarrow$ Closed-world assumption $\leftrightarrow$ Missing information treated as *false* (LP has it)

Example

$$\mathsf{Icecream}(x) \wedge \neg\, \mathsf{BananaIscream}(x) \quad \rightarrow \quad \mathsf{Likes}(alice, x)$$

$$\mathcal{K} = \{\mathsf{Icecream}(gnds)\} \qquad \mathcal{K} \not\models \mathsf{Likes}(alice, gnds)$$

$$\mathsf{Icecream}(x) \wedge \mathbf{not}\ \mathsf{BananaIscream}(x) \quad \rightarrow \quad \mathsf{Likes}(alice, x)$$

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)
  Negation as failure $\leftrightarrow$ Closed-world assumption $\leftrightarrow$ Missing information treated as *false* (LP has it)

Example

$$\mathsf{Icecream}(x) \wedge \neg\, \mathsf{BananaIscream}(x) \quad \rightarrow \quad \mathsf{Likes}(alice, x)$$
$$\mathcal{K} = \{\mathsf{Icecream}(gnds)\} \qquad \mathcal{K} \not\models \mathsf{Likes}(alice, gnds)$$
$$\mathsf{Icecream}(x) \wedge \mathbf{not}\ \mathsf{BananaIscream}(x) \quad \rightarrow \quad \mathsf{Likes}(alice, x)$$
$$\mathcal{K} = \{\mathsf{Icecream}(gnds)\} \qquad \mathcal{K} \models \mathsf{Likes}(alice, gnds)$$

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)
  Negation as failure $\leftrightarrow$ Closed-world assumption $\leftrightarrow$ Missing information treated as *false* (LP has it)
- Double benefit:

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)
  Negation as failure $\leftrightarrow$ Closed-world assumption $\leftrightarrow$ Missing information treated as *false* (LP has it)
- Double benefit:
  - Encode chemical classes based on the absence of information (e.g. hydrocarbons, inorganic molecules, saturated compounds,...)

Switch to Logic Programming

- Shift from first-order logic semantics to logic programming semantics
- Replace classical negation with negation-as-failure to derive non-monotonic inferences. Two different negations?

  Classical negation $\leftrightarrow$ Open-world assumption $\leftrightarrow$ Missing information treated as *not known* (OWL has it)
  Negation as failure $\leftrightarrow$ Closed-world assumption $\leftrightarrow$ Missing information treated as *false* (LP has it)
- Double benefit:
  - Encode chemical classes based on the absence of information (e.g. hydrocarbons, inorganic molecules, saturated compounds,. . .)
  - Represent rings with adequate precision (no tree-model property)

Results overview

- Design of an expressive logic-based formalism for modelling structured entities that we call Description Graphs Logic Programs (DGLPs)

Results overview

- Design of an expressive logic-based formalism for modelling structured entities that we call Description Graphs Logic Programs (DGLPs)

- Ensure decidability of reasoning tasks by formulating sufficient restrictive conditions

Results overview

- Design of an expressive logic-based formalism for modelling structured entities that we call Description Graphs Logic Programs (DGLPs)
- Ensure decidability of reasoning tasks by formulating sufficient restrictive conditions
- Develop a prototypical implementation

RESULTS OVERVIEW

- Design of an expressive logic-based formalism for modelling structured entities that we call Description Graphs Logic Programs (DGLPs)

- Ensure decidability of reasoning tasks by formulating sufficient restrictive conditions

- Develop a prototypical implementation

- Encouraging results of a preliminary evaluation

Outline

WHAT IS A DGLP ONTOLOGY?

- The syntactic objects of a DGLP ontology:

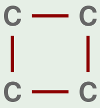
WHAT IS A DGLP ONTOLOGY?

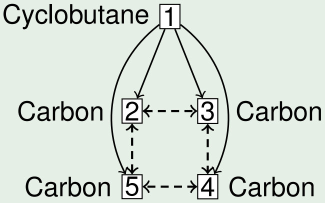
- The syntactic objects of a DGLP ontology:
  - Description graphs

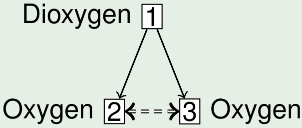
What is a DGLP ontology?

- The syntactic objects of a DGLP ontology:
  - Description graphs

Example







What is a DGLP ontology?

- The syntactic objects of a DGLP ontology:
  - Description graphs
  - Function-free FOL Horn rules

What is a DGLP ontology?

- The syntactic objects of a DGLP ontology:
  - Description graphs
  - Function-free FOL Horn rules

Example

$$\begin{array}{lcl} \mathsf{Bond}(x, y) & \rightarrow & \mathsf{Bond}(y, x) \\ \mathsf{SingleBond}(x, y) & \rightarrow & \mathsf{Bond}(x, y) \end{array}$$

WHAT IS A DGLP ONTOLOGY?

- The syntactic objects of a DGLP ontology:
  - Description graphs
  - Function-free FOL Horn rules

EXAMPLE

$$\begin{array}{lcl} \mathsf{Bond}(x, y) & \rightarrow & \mathsf{Bond}(y, x) \\ \mathsf{SingleBond}(x, y) & \rightarrow & \mathsf{Bond}(x, y) \end{array}$$

  - Rules with negation-as-failure

What is a DGLP ontology?

- The syntactic objects of a DGLP ontology:
  - Description graphs
  - Function-free FOL Horn rules

Example

$$\begin{array}{lcl} \mathsf{Bond}(x, y) & \rightarrow & \mathsf{Bond}(y, x) \\ \mathsf{SingleBond}(x, y) & \rightarrow & \mathsf{Bond}(x, y) \end{array}$$

  - Rules with negation-as-failure

Example

$$\begin{array}{lcl} \mathsf{HasAtom}(x, y) \wedge \mathsf{Carbon}(y) & \rightarrow & \mathsf{HasCarbon}(x) \\ \mathsf{Molecule}(x) \wedge \textbf{not } \mathsf{HasCarbon}(x) & \rightarrow & \mathsf{Inorganic}(x) \end{array}$$

What is a DGLP ontology?

- The syntactic objects of a DGLP ontology:
  - Description graphs
  - Function-free FOL Horn rules

Example

$$\begin{array}{lcl} \mathsf{Bond}(x,y) & \rightarrow & \mathsf{Bond}(y,x) \\ \mathsf{SingleBond}(x,y) & \rightarrow & \mathsf{Bond}(x,y) \end{array}$$

  - Rules with negation-as-failure

Example

$$\begin{array}{lcl} \mathsf{HasAtom}(x,y) \wedge \mathsf{Carbon}(y) & \rightarrow & \mathsf{HasCarbon}(x) \\ \mathsf{Molecule}(x) \wedge \textbf{not}\ \mathsf{HasCarbon}(x) & \rightarrow & \mathsf{Inorganic}(x) \end{array}$$

  - Facts

What is a DGLP ontology?

- The syntactic objects of a DGLP ontology:
  - Description graphs
  - Function-free FOL Horn rules

Example

$$\begin{array}{lcl} \mathrm{Bond}(x, y) & \rightarrow & \mathrm{Bond}(y, x) \\ \mathrm{SingleBond}(x, y) & \rightarrow & \mathrm{Bond}(x, y) \end{array}$$

  - Rules with negation-as-failure

Example

$$\begin{array}{lcl} \mathrm{HasAtom}(x, y) \wedge \mathrm{Carbon}(y) & \rightarrow & \mathrm{HasCarbon}(x) \\ \mathrm{Molecule}(x) \wedge \textbf{not } \mathrm{HasCarbon}(x) & \rightarrow & \mathrm{Inorganic}(x) \end{array}$$

  - Facts

Example

$\mathrm{Cyclobutane}(c_1), \quad \mathrm{Dinitrogen}(c_2), \ldots$

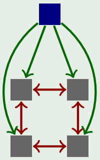
Semantics of DGs

- Translate DGs into logic programs with function symbols

Semantics of DGs

- Translate DGs into logic programs with function symbols

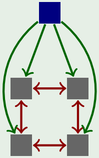
Example



Semantics of DGs

- Translate DGs into logic programs with function symbols

Example

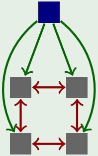


$$
\begin{aligned}
\mathrm{Cyclobutane}(x) &\rightarrow G_{cb}(x, f_1(x), f_2(x), f_3(x), f_4(x)) \\
G_{cb}(x, y_1, y_2, y_3, y_4) &\rightarrow \mathrm{Cyclobutane}(x) \wedge \\
&\quad \mathrm{Carbon}(y_1) \wedge \mathrm{Carbon}(y_2) \wedge \\
&\quad \mathrm{Carbon}(y_3) \wedge \mathrm{Carbon}(y_4) \wedge \\
&\quad \mathrm{HasAtom}(x, y_1) \wedge \mathrm{Bond}(y_1, y_2) \wedge \\
&\quad \mathrm{HasAtom}(x, y_2) \wedge \mathrm{Bond}(y_2, y_3) \wedge \\
&\quad \mathrm{HasAtom}(x, y_3) \wedge \mathrm{Bond}(y_3, y_4) \wedge \\
&\quad \mathrm{HasAtom}(x, y_4) \wedge \mathrm{Bond}(y_4, y_1)
\end{aligned}
$$

Semantics of DGs

- Translate DGs into logic programs with function symbols

Example



$$\begin{aligned}
\mathrm{Cyclobutane}(x) &\rightarrow G_{cb}(x, f_1(x), f_2(x), f_3(x), f_4(x)) \\
G_{cb}(x, y_1, y_2, y_3, y_4) &\rightarrow \mathrm{Cyclobutane}(x) \wedge \\
&\quad \mathrm{Carbon}(y_1) \wedge \mathrm{Carbon}(y_2) \wedge \\
&\quad \mathrm{Carbon}(y_3) \wedge \mathrm{Carbon}(y_4) \wedge \\
&\quad \mathrm{HasAtom}(x, y_1) \wedge \mathrm{Bond}(y_1, y_2) \wedge \\
&\quad \mathrm{HasAtom}(x, y_2) \wedge \mathrm{Bond}(y_2, y_3) \wedge \\
&\quad \mathrm{HasAtom}(x, y_3) \wedge \mathrm{Bond}(y_3, y_4) \wedge \\
&\quad \mathrm{HasAtom}(x, y_4) \wedge \mathrm{Bond}(y_4, y_1)
\end{aligned}$$

- Function symbols allow for schema-level reasoning

Classifying objects

Example

$\text{Molecule} \sqcap \forall \text{hasAtom}.(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon}$

Classifying objects

Example

$$\text{Molecule} \sqcap \forall \text{hasAtom}.(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon}$$

$$\Downarrow$$

Classifying objects

Example

$\mathsf{Molecule} \sqcap \forall \mathsf{hasAtom}.(\mathsf{Carbon} \sqcup \mathsf{Hydrogen}) \sqsubseteq \mathsf{Hydrocarbon}$

$$\Downarrow$$

$\mathsf{Molecule} \sqcap \neg\exists \mathsf{hasAtom}.((\neg\mathsf{Carbon}) \sqcap (\neg\mathsf{Hydrogen})) \sqsubseteq \mathsf{Hydrocarbon}$

CLASSIFYING OBJECTS

EXAMPLE

$\text{Molecule} \sqcap \forall \text{hasAtom}.(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon}$

$\Downarrow$

$\text{Molecule} \sqcap \neg\exists \text{hasAtom}.((\neg\text{Carbon}) \sqcap (\neg\text{Hydrogen})) \sqsubseteq \text{Hydrocarbon}$

$\Downarrow$

Classifying objects

Example

$\mathsf{Molecule} \sqcap \forall \mathsf{hasAtom}.(\mathsf{Carbon} \sqcup \mathsf{Hydrogen}) \sqsubseteq \mathsf{Hydrocarbon}$

$$\Downarrow$$

$\mathsf{Molecule} \sqcap \neg\exists \mathsf{hasAtom}.((\neg\mathsf{Carbon}) \sqcap (\neg\mathsf{Hydrogen})) \sqsubseteq \mathsf{Hydrocarbon}$

$$\Downarrow$$

$\mathsf{Molecule}(x) \wedge \mathsf{HasAtom}(x, y) \wedge \textbf{not } \mathsf{Carbon}(y) \wedge \textbf{not } \mathsf{Hydrogen}(y) \rightarrow \mathsf{NotHydroCarbon}(x)$

$\mathsf{Molecule}(x) \wedge \textbf{not } \mathsf{NotHydroCarbon}(x) \rightarrow \mathsf{HydroCarbon}(x)$

Classifying objects

Example

$\mathsf{Molecule} \sqcap \forall \mathsf{hasAtom}.(\mathsf{Carbon} \sqcup \mathsf{Hydrogen}) \sqsubseteq \mathsf{Hydrocarbon}$

$$\Downarrow$$

$\mathsf{Molecule} \sqcap \neg\exists \mathsf{hasAtom}.((\neg \mathsf{Carbon}) \sqcap (\neg \mathsf{Hydrogen})) \sqsubseteq \mathsf{Hydrocarbon}$

$$\Downarrow$$

$\mathsf{Molecule}(x) \wedge \mathsf{HasAtom}(x, y) \wedge \textbf{not } \mathsf{Carbon}(y) \wedge \textbf{not } \mathsf{Hydrogen}(y) \rightarrow \mathsf{NotHydroCarbon}(x)$

$\mathsf{Molecule}(x) \wedge \textbf{not } \mathsf{NotHydroCarbon}(x) \rightarrow \mathsf{HydroCarbon}(x)$

Classifying objects

Example

$\text{Molecule} \sqcap \forall \text{hasAtom}.(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon}$

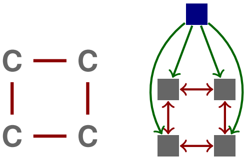
$\Downarrow$

$\text{Molecule} \sqcap \neg\exists \text{hasAtom}.((\neg\text{Carbon}) \sqcap (\neg\text{Hydrogen})) \sqsubseteq$
Hydrocarbon

$\Downarrow$

$\text{Molecule}(x) \wedge \text{HasAtom}(x, y) \wedge \textbf{not } \text{Carbon}(y) \wedge \textbf{not } \text{Hydrogen}(y) \rightarrow \text{NotHydroCarbon}(x)$

$\text{Molecule}(x) \wedge \textbf{not } \text{NotHydroCarbon}(x) \rightarrow \text{HydroCarbon}(x)$



Example

$\text{Molecule} \sqcap \forall \text{hasAtom}.(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon}$

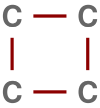
$\Downarrow$

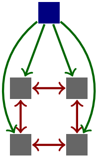
$\text{Molecule} \sqcap \neg\exists \text{hasAtom}.((\neg\text{Carbon}) \sqcap (\neg\text{Hydrogen})) \sqsubseteq \text{Hydrocarbon}$

$\Downarrow$

$\text{Molecule}(x) \wedge \text{HasAtom}(x, y) \wedge \textbf{not } \text{Carbon}(y) \wedge \textbf{not } \text{Hydrogen}(y) \rightarrow \text{NotHydroCarbon}(x)$

$\text{Molecule}(x) \wedge \textbf{not } \text{NotHydroCarbon}(x) \rightarrow \text{HydroCarbon}(x)$





- Is cyclobutane a hydrocarbon? ✓

Classifying objects

Example

$$\mathsf{Molecule}(\mathsf{x}) \wedge \bigwedge_{1 \le \mathsf{i} \le 4} \mathsf{HasAtom}(\mathsf{x}, \mathsf{y_i}) \wedge \bigwedge_{1 \le \mathsf{i} \le 3} \mathsf{Bond}(\mathsf{y_i}, \mathsf{y_{i+1}}) \wedge$$

$$\mathsf{Bond}(\mathsf{y_4}, \mathsf{y_1}) \bigwedge_{1 \le i < j \le 4} \textbf{not } \mathsf{y_i} = \mathsf{y_j}$$

$$\rightarrow \mathsf{MoleculeWith4MemberedRing}(\mathsf{x})$$

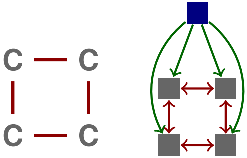
Classifying objects

Example

$$\mathsf{Molecule}(x) \wedge \bigwedge_{1 \leq i \leq 4} \mathsf{HasAtom}(x, y_i) \wedge \bigwedge_{1 \leq i \leq 3} \mathsf{Bond}(y_i, y_{i+1}) \wedge$$

$$\mathsf{Bond}(y_4, y_1) \bigwedge_{1 \leq i < j \leq 4} \textbf{not } y_i = y_j$$

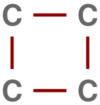
$$\rightarrow \mathsf{MoleculeWith4MemberedRing}(x)$$

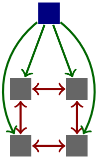


Classifying objects

Example

$$\mathrm{Molecule}(x) \wedge \bigwedge_{1\le i\le 4} \mathrm{HasAtom}(x, y_i) \wedge \bigwedge_{1\le i\le 3} \mathrm{Bond}(y_i, y_{i+1}) \wedge$$
$$\mathrm{Bond}(y_4, y_1) \bigwedge_{1\le i<j\le 4} \textbf{not } y_i = y_j$$
$$\rightarrow \mathrm{MoleculeWith4MemberedRing}(x)$$





- Does cyclobutane contain a four-membered ring? ✓

(UN)DECIDABILITY

- Logic programs with function symbols can axiomatise infinite structures

(UN)DECIDABILITY

- Logic programs with function symbols can axiomatise infinite structures
- Reasoning with DGLP ontologies is trivially undecidable

(UN)DECIDABILITY

- Logic programs with function symbols can axiomatise infinite structures
- Reasoning with DGLP ontologies is trivially undecidable

EXAMPLE

$$\begin{array}{lcl} A(x) & \rightarrow & G(x, f_1(x), f_2(x)) \\ G(x, y_1, y_2) & \rightarrow & A(y_1) \wedge A(y_2) \end{array}$$

$$\{A(a),\ G(a, f_1(a),\ f_2(a)),\ A(f_1(a)),\ A(f_2(a)),\ \ldots\}$$

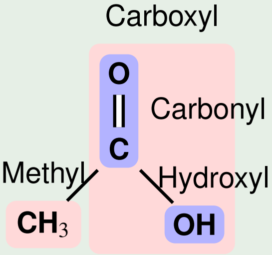
(UN)DECIDABILITY

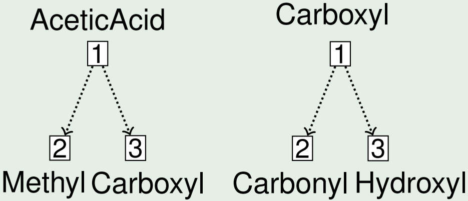
- Logic programs with function symbols can axiomatise infinite structures
- Reasoning with DGLP ontologies is trivially undecidable
- We are only interested in bounded structures

(Un)decidability

- Logic programs with function symbols can axiomatise infinite structures
- Reasoning with DGLP ontologies is trivially undecidable
- We are only interested in bounded structures

Example





Syntactic acyclicity conditions

- Problem extensively studied, e.g. in theory of databases

Syntactic acyclicity conditions

- Problem extensively studied, e.g. in theory of databases
- Various syntax-based acyclicity conditions

Syntactic acyclicity conditions

- Problem extensively studied, e.g. in theory of databases
- Various syntax-based acyclicity conditions
  - Weak acyclicity [Fagin et al., 2002]
  - Super-weak acyclicity [Marnette, 2009]
  - Joint acyclicity [Krötzsch et al., 2011]

Syntactic acyclicity conditions

- Problem extensively studied, e.g. in theory of databases
- Various syntax-based acyclicity conditions
  - Weak acyclicity [Fagin et al., 2002]
  - Super-weak acyclicity [Marnette, 2009]
  - Joint acyclicity [Krötzsch et al., 2011]

Example

$$\begin{aligned}
\mathrm{AceticAcid}(x) &\rightarrow G_{AA}(x, f_1(x), f_2(x)) \\
G_{AA}(x_1, x_2, x_3) &\rightarrow \mathrm{AceticAcid}(x_1) \wedge \mathrm{Methyl}(x_2) \wedge \mathrm{Carboxyl}(x_3) \wedge \\
&\quad\ \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3) \\
\mathrm{Methyl}(x_2) \wedge \mathrm{Carboxyl}(x_3) &\wedge \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3) \\
&\rightarrow G_{AA}(x_1, x_2, x_3) \\
\mathrm{Carboxyl}(x) &\rightarrow G_{cxl}(x, g_1(x), g_2(x)) \\
G_{cxl}(x_1, x_2, x_3) &\rightarrow \mathrm{Carboxyl}(x_1) \wedge \mathrm{Carbonyl}(x_2) \wedge \mathrm{Hydroxyl}(x_3) \wedge \\
&\quad\ \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3)
\end{aligned}$$

Syntactic acyclicity conditions

- Problem extensively studied, e.g. in theory of databases
- Various syntax-based acyclicity conditions
  - Weak acyclicity [Fagin et al., 2002]
  - Super-weak acyclicity [Marnette, 2009]
  - Joint acyclicity [Krötzsch et al., 2011]

Example

$$\mathrm{AceticAcid}(\underline{x}) \rightarrow G_{AA}(\underline{x}, f_1(x), f_2(x))$$

$$G_{AA}(x_1, x_2, x_3) \rightarrow \mathrm{AceticAcid}(x_1) \wedge \mathrm{Methyl}(x_2) \wedge \mathrm{Carboxyl}(x_3) \wedge \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3)$$

$$\mathrm{Methyl}(x_2) \wedge \mathrm{Carboxyl}(x_3) \wedge \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3) \rightarrow G_{AA}(x_1, x_2, x_3)$$

$$\mathrm{Carboxyl}(x) \rightarrow G_{cxl}(x, g_1(x), g_2(x))$$

$$G_{cxl}(x_1, x_2, x_3) \rightarrow \mathrm{Carboxyl}(x_1) \wedge \mathrm{Carbonyl}(x_2) \wedge \mathrm{Hydroxyl}(x_3) \wedge \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3)$$

Syntactic acyclicity conditions

- Problem extensively studied, e.g. in theory of databases
- Various syntax-based acyclicity conditions
  - Weak acyclicity [Fagin et al., 2002]
  - Super-weak acyclicity [Marnette, 2009]
  - Joint acyclicity [Krötzsch et al., 2011]

Example

$$\mathrm{AceticAcid}(x) \rightarrow \mathrm{G_{AA}}(x, f_1(x), f_2(x))$$

$$\mathrm{G_{AA}}(x_1, x_2, x_3) \rightarrow \mathrm{AceticAcid}(x_1) \wedge \mathrm{Methyl}(x_2) \wedge \mathrm{Carboxyl}(x_3) \wedge \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3)$$

$$\mathrm{Methyl}(x_2) \wedge \mathrm{Carboxyl}(x_3) \wedge \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3) \rightarrow \mathrm{G_{AA}}(x_1, x_2, x_3)$$

$$\mathrm{Carboxyl}(x) \rightarrow \mathrm{G_{cxl}}(x, g_1(x), g_2(x))$$

$$\mathrm{G_{cxl}}(x_1, x_2, x_3) \rightarrow \mathrm{Carboxyl}(x_1) \wedge \mathrm{Carbonyl}(x_2) \wedge \mathrm{Hydroxyl}(x_3) \wedge \mathrm{HasPart}(x_1, x_2) \wedge \mathrm{HasPart}(x_1, x_3)$$

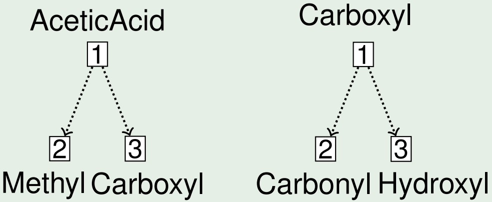
SEMANTIC ACYCLICITY

- Define a transitive and irreflexive graph ordering which specifies which graph instances may imply the existence of other graph instances

Semantic acyclicity

- Define a transitive and irreflexive graph ordering which specifies which graph instances may imply the existence of other graph instances

Example



AceticAcid $\prec$ Carboxyl

Semantic acyclicity

- Define a transitive and irreflexive graph ordering which specifies which graph instances may imply the existence of other graph instances

- Extentd the logic program with rules that are triggered by violation of the graph ordering

Semantic acyclicity

- Define a transitive and irreflexive graph ordering which specifies which graph instances may imply the existence of other graph instances

- Extentd the logic program with rules that are triggered by violation of the graph ordering

- If a repetitive construction of graph instances is detected during reasoning, then derive Cycle

Semantic acyclicity

- Define a transitive and irreflexive graph ordering which specifies which graph instances may imply the existence of other graph instances

- Extentd the logic program with rules that are triggered by violation of the graph ordering

- If a repetitive construction of graph instances is detected during reasoning, then derive Cycle

Example

$\mathrm{G}_{\mathrm{AA}}(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3) \wedge \mathrm{AceticAcid}(\mathrm{x}_2) \rightarrow \mathrm{Cycle}$

Semantic acyclicity

- Define a transitive and irreflexive graph ordering which specifies which graph instances may imply the existence of other graph instances
- Extentd the logic program with rules that are triggered by violation of the graph ordering
- If a repetitive construction of graph instances is detected during reasoning, then derive Cycle
- A DGLP ontology is semantically acyclic if it does not entail Cycle

Semantic acyclicity

- Define a transitive and irreflexive graph ordering which specifies which graph instances may imply the existence of other graph instances
- Extentd the logic program with rules that are triggered by violation of the graph ordering
- If a repetitive construction of graph instances is detected during reasoning, then derive Cycle
- A DGLP ontology is semantically acyclic if it does not entail Cycle
- DGLP ontology with acetic acid is semantically acyclic ✓

Technical results

- Decidability for negation-free DGLP ontologies

TECHNICAL RESULTS

- Decidability for negation-free DGLP ontologies
- Decidability for DGLP ontologies with stratified negation

TECHNICAL RESULTS

- Decidability for negation-free DGLP ontologies
- Decidability for DGLP ontologies with stratified negation
- DGLP ontologies with stratified negation capture a wide range of chemical classes:

Technical results

- Decidability for negation-free DGLP ontologies
- Decidability for DGLP ontologies with stratified negation
- DGLP ontologies with stratified negation capture a wide range of chemical classes:
  - Is dinitrogen inorganic?
  - Does cyclobutane contain a four-membered ring?
  - Is acetylene a hydrocarbon?
  - Does benzaldehyde contain a benzene ring?

Technical results

- Decidability for negation-free DGLP ontologies
- Decidability for DGLP ontologies with stratified negation
- DGLP ontologies with stratified negation capture a wide range of chemical classes:
  - Is dinitrogen inorganic? ✓
  - Does cyclobutane contain a four-membered ring? ✓
  - Is acetylene a hydrocarbon? ✓
  - Does benzaldehyde contain a benzene ring? ✓

Outline

EXPERIMENTAL SETTING

- Data extracted from ChEBI

EXPERIMENTAL SETTING

- Data extracted from ChEBI
- XSB logic programming engine

Experimental setting

- Data extracted from ChEBI
- XSB logic programming engine
- Chemical classes that were modelled:
  - Hydrocarbons
  - Inorganic molecules
  - Molecules with exactly two carbons
  - Molecules with a four-membered ring
  - Molecules with a benzene

Experimental setting

- Data extracted from ChEBI
- XSB logic programming engine
- Chemical classes that were modelled:
  - Hydrocarbons
  - Inorganic molecules
  - Molecules with exactly two carbons
  - Molecules with a four-membered ring
  - Molecules with a benzene
- Preliminary evaluation: size of the ontologies ranging from 10 to 70 molecules

Implementation results

- All DGLP ontologies were found acyclic

Implementation results

- All DGLP ontologies were found acyclic
- Molecules classified as expected

Implementation results

- All DGLP ontologies were found acyclic
- Molecules classified as expected

No mol.	T_1	T_2	T_3	T_4	T_5	Total time
10	< 0.01	< 0.01	< 0.01	0.36	0.02	2.47
20	< 0.01	< 0.01	0.02	2.07	0.21	10.66
30	0.01	< 0.01	0.03	2.23	0.23	13.85
40	0.01	< 0.01	0.04	2.58	0.29	19.06
50	0.01	0.01	0.06	3.55	0.41	27.15
60	0.04	0.02	0.51	109.88	21.68	300.84
70	0.06	0.03	0.75	172.14	35.08	447.12

- T_1: hydrocarbons, T_2: inorganic molecules
- T_3: molecules with exactly two carbons
- T_4: molecules with a four-membered ring
- T_5: molecules with a benzene

Implementation results

- All DGLP ontologies were found acyclic
- Molecules classified as expected

No mol.	T_1	T_2	T_3	T_4	T_5	Total time
10	< 0.01	< 0.01	< 0.01	0.36	0.02	2.47
20	< 0.01	< 0.01	0.02	2.07	0.21	10.66
30	0.01	< 0.01	0.03	2.23	0.23	13.85
40	0.01	< 0.01	0.04	2.58	0.29	19.06
50	0.01	0.01	0.06	3.55	0.41	27.15
60	0.04	0.02	0.51	109.88	21.68	300.84
70	0.06	0.03	0.75	172.14	35.08	447.12

- T_1: hydrocarbons, T_2: inorganic molecules
- T_3: molecules with exactly two carbons
- T_4: molecules with a four-membered ring
- T_5: molecules with a benzene

Outline

Results Overview

- Expressive and decidable formalism for modelling structured objects

Results Overview

- Expressive and decidable formalism for modelling structured objects

- Novel acyclicity condition for logic programs with restricted use of function symbols

Results Overview

- Expressive and decidable formalism for modelling structured objects
- Novel acyclicity condition for logic programs with restricted use of function symbols
- Development of a prototype for the automatic classification of chemical molecules

Results Overview

- Expressive and decidable formalism for modelling structured objects
- Novel acyclicity condition for logic programs with restricted use of function symbols
- Development of a prototype for the automatic classification of chemical molecules
- Encouraging results of a preliminary evaluation

Future directions

- Extend suggested DGLP formalism:

Future directions

- Extend suggested DGLP formalism:
  - Relax stratifiability criteria for negation
  - Disjunctive rule heads
  - Integrate logic programming rules with use of numerical values

FUTURE DIRECTIONS

- Extend suggested DGLP formalism:
  - Relax stratifiability criteria for negation
  - Disjunctive rule heads
  - Integrate logic programming rules with use of numerical values
- Optimise our prototype towards a fully-scalable classification system for structured objects

Future directions

- Extend suggested DGLP formalism:
  - Relax stratifiability criteria for negation
  - Disjunctive rule heads
  - Integrate logic programming rules with use of numerical values
- Optimise our prototype towards a fully-scalable classification system for structured objects
- Thank you for listening. Questions?