Lecture 12: Description Logics

Dragan Doder

Methods in AI Research

Outline

Outline

Fragments of FOL

Formal Ontologies

Introduction to DLs

Making Statements

DL Knowledge Bases & entailment

Limitations of FOL

First-order logic is very expressive. Why not:

- represent knowledge as a set of formulas Γ of FOL
- reason using the valid inferences: deduce those formulas α s.t. $\Gamma \models \alpha$?

Limitations of FOL

First-order logic is very expressive. Why not:

- represent knowledge as a set of formulas Γ of FOL
- reason using the valid inferences: deduce those formulas α s.t. $\Gamma \models \alpha$?

Theorem (Alan Turing)

There exists no Turing Machine (algorithm) that decides for any finite set of formulas Γ and any formula α whether $\Gamma \models \alpha$

Limitations of FOL

First-order logic is very expressive. Why not:

- represent knowledge as a set of formulas Γ of FOL
- reason using the valid inferences: deduce those formulas α s.t. $\Gamma \models \alpha$?

Theorem (Alan Turing)

There exists no Turing Machine (algorithm) that decides for any finite set of formulas Γ and any formula α whether $\Gamma \models \alpha$

FOL is undecidable.

Limitations of FOL

First-order logic is very expressive. Why not:

- represent knowledge as a set of formulas Γ of FOL
- reason using the valid inferences: deduce those formulas α s.t. $\Gamma \models \alpha$?

Theorem (Alan Turing)

There exists no Turing Machine (algorithm) that decides for any finite set of formulas Γ and any formula α whether $\Gamma \models \alpha$

FOL is undecidable.

FOL for KRR

Representation ✓

Reasoning ✗

Trade-off between expressiveness and efficiency

Definition

Logical system is decidable if membership in the set of logically valid formulas can be effectively determined.

FOL is semi-decidable:

- If a statement is valid, we can find it in finite time (in other words: there exists a proof we can check for correctness).
- But there is no effective procedure for checking that a sentence is not valid.

Trade-off between expressiveness and efficiency

Definition

Logical system is decidable if membership in the set of logically valid formulas can be effectively determined.

FOL is semi-decidable:

- If a statement is valid, we can find it in finite time (in other words: there exists a proof we can check for correctness).
- But there is no effective procedure for checking that a sentence is not valid.

Other extreme: Propositional logic

- decidable, good computational properties

Trade-off between expressiveness and efficiency

Definition

Logical system is decidable if membership in the set of logically valid formulas can be effectively determined.

FOL is semi-decidable:

- If a statement is valid, we can find it in finite time (in other words: there exists a proof we can check for correctness).
- But there is no effective procedure for checking that a sentence is not valid.

Other extreme: Propositional logic

- decidable, good computational properties
- but highly inexpressive

Trade-off between expressiveness and efficiency

Definition

Logical system is decidable if membership in the set of logically valid formulas can be effectively determined.

FOL is semi-decidable:

- If a statement is valid, we can find it in finite time (in other words: there exists a proof we can check for correctness).
- But there is no effective procedure for checking that a sentence is not valid.

Other extreme: Propositional logic

- decidable, good computational properties
- but highly inexpressive

Solution – "between" PL and FOL

Fragments

Idea: Restrict the set of FO formulas to a decidable subset.

Fragment of a logic:

- The set of formulas is a subset of those in the original logic.
- The semantics stays the same.

Fragments

Idea: Restrict the set of FO formulas to a decidable subset.

Fragment of a logic:

- The set of formulas is a subset of those in the original logic.
- The semantics stays the same.

Many logics of interest to AI correspond to a decidable fragment of FOL:

- epistemic logics
- temporal logics
- description logics

$\vdots$

Description logics

- represent and reason about terminological knowledge
  - concepts, roles, individuals
- a logical formalism for ontological modelling in the context of the Semantic Web
  - the Web Ontology Language (OWL) is based on DLs
- Open source reasoners and editors
  - The Protégé Ontology Editor. `http://protege.stanford.edu`

Outline

Fragments of FOL

Formal Ontologies

Introduction to DLs

Making Statements

DL Knowledge Bases & entailment

Ontologies

Explicit specification of a shared conceptualisation

Example (The student ontology)

- Employed students are students and employees
- Students are not taxpayers (do not pay taxes)
- Employed students are taxpayers (pay taxes)
- Employed students who are parents are not taxpayers (do not pay taxes)
- To work for is to be employed by
- John is an employed student, John works for IBM

Ontologies

Explicit specification of a shared conceptualisation

Example (The student ontology)

- Employed students are students and employees
- Students are not taxpayers (do not pay taxes)
- Employed students are taxpayers (pay taxes)
- Employed students who are parents are not taxpayers (do not pay taxes)
- To work for is to be employed by
- John is an employed student, John works for IBM

Ontologies

Explicit specification of a shared conceptualisation

Example (The student ontology)

- Employed students are students and employees
- Students are not taxpayers (do not pay taxes)
- Employed students are taxpayers (pay taxes)
- Employed students who are parents are not taxpayers (do not pay taxes)
- To work for is to be employed by
- John is an employed student, John works for IBM

classes relations individuals

Ontologies

Explicit specification of a shared conceptualisation

Example (The student ontology)

- Employed students are students and employees
- Students are not taxpayers (do not pay taxes)
- Employed students are taxpayers (pay taxes)
- Employed students who are parents are not taxpayers (do not pay taxes)
- To work for is to be employed by
- John is an employed student, John and IBM are in works for

classes relations individuals

specialisation and instantiation

Main ingredients in formal ontologies

A common vocabulary and a shared understanding

Main ingredients in formal ontologies

A common vocabulary and a shared understanding

Classes or concepts

- Describe concrete or abstract entities within the domain of interest
- E.g.: Employed student, Parent

Main ingredients in formal ontologies

A common vocabulary and a shared understanding

Classes or concepts

- Describe concrete or abstract entities within the domain of interest
- E.g.: Employed student, Parent

Relations or roles

- Describe relationships between concepts or attributes of a concept
- E.g.: work for someone, being employed by someone

Main ingredients in formal ontologies

A common vocabulary and a shared understanding

Classes or concepts

- Describe concrete or abstract entities within the domain of interest
- E.g.: Employed student, Parent

Relations or roles

- Describe relationships between concepts or attributes of a concept
- E.g.: work for someone, being employed by someone

Instances of classes and relations

- Name objects of the domain and denote representatives of a concept
- E.g.: John, John is an employed student, John works for IBM

Why Description Logics?

Expressivity

- Concepts ✓
- Relations ✓
- Instances ✓

DLs have all one needs to formalise ontologies!

Why Description Logics?

Expressivity

- Concepts ✓
- Relations ✓
- Instances ✓

DLs have all one needs to formalise ontologies!

Computational properties

- Amenability to implementation ✓
- Decidability ✓
- Good trade-off between expressivity and complexity ✓

Most DL-based systems satisfy all of these!

Why Description Logics?

Expressivity

- Concepts ✓
- Relations ✓
- Instances ✓

DLs have all one needs to formalise ontologies!

Computational properties

- Amenability to implementation ✓
- Decidability ✓
- Good trade-off between expressivity and complexity ✓

Most DL-based systems satisfy all of these!

Available tools

protégé

FaCT++

Pellet

HermiT

CEL

...

Outline

Elements of the language (domain dependent)

Atomic concept names

- $\mathsf{C} =_{\text{def}} \{A_1, \ldots, A_n\}$ (Special concepts: $\top$, $\bot$)
- Intuition: basic classes of a domain of interest
- Student, Employee, Parent

Elements of the language (domain dependent)

Atomic concept names

- $\mathsf{C} =_{\text{def}} \{A_1, \ldots, A_n\}$ (Special concepts: $\top$, $\bot$)
- Intuition: basic classes of a domain of interest
- Student, Employee, Parent

Atomic role names

- $\mathsf{R} =_{\text{def}} \{r_1, \ldots, r_m\}$
- Intuition: basic relations between concepts
- worksFor, empBy

Elements of the language (domain dependent)

Atomic concept names

- C $=_{\text{def}} \{A_1, \ldots, A_n\}$ (Special concepts: $\top$, $\bot$)
- Intuition: basic classes of a domain of interest
- Student, Employee, Parent

Atomic role names

- R $=_{\text{def}} \{r_1, \ldots, r_m\}$
- Intuition: basic relations between concepts
- worksFor, empBy

Individual names

- I $=_{\text{def}} \{a_1, \ldots, a_l\}$
- Intuition: names of objects in the domain
- john, mary, ibm

Elements of the language (domain independent)

Boolean constructors

- Concept negation: $\neg$ (class complement)
- Concept conjunction: $\sqcap$ (class intersection)
- Concept disjunction: $\sqcup$ (class union)

Elements of the language (domain independent)

Boolean constructors

- Concept negation: $\neg$ (class complement)
- Concept conjunction: $\sqcap$ (class intersection)
- Concept disjunction: $\sqcup$ (class union)

Role restrictions

- Existential restriction: $\exists$ (at least one relationship)
- Value restriction: $\forall$ (all relationships)

Elements of the language (domain independent)

Boolean constructors

- Concept negation: $\neg$ (class complement)
- Concept conjunction: $\sqcap$ (class intersection)
- Concept disjunction: $\sqcup$ (class union)

Role restrictions

- Existential restriction: $\exists$ (at least one relationship)
- Value restriction: $\forall$ (all relationships)

Further constructors: cardinality constraints, inverse roles, ... (if needed)

Building concepts

Definition (Complex concepts)

- $\top$ and $\bot$ are concepts
- Every concept name $A \in \mathsf{C}$ is a concept
- If C and D are concepts and $r \in \mathsf{R}$, then

$\neg C$	(complement of C)	$\exists r.C$	(existential restriction)
$C \sqcap D$	(intersection of C and D)	$\forall r.C$	(value restriction)
$C \sqcup D$	(union of C and D)		

  are all concepts
- Nothing else is a concept (at least for now)

Building concepts

Full negation

- Negation of arbitrary concepts
- Intuition: the complement of a concept
- E.g.: $\neg\neg\text{Student}$ $\quad$ $\neg(\text{Student} \sqcap \text{Parent})$

Building concepts

Full negation

- Negation of arbitrary concepts
- Intuition: the complement of a concept
- E.g.: $\neg\neg\text{Student}$ $\quad \neg(\text{Student} \sqcap \text{Parent})$

Atomic negation

- Some DLs only allow negation of concept names
- Good complexity results
- E.g.: $\neg\text{Student}$ $\quad \neg\text{Parent}$

Building concepts

Concept conjunction

- Intuition: the intersection of two concepts
- E.g.: $\mathsf{Student} \sqcap \mathsf{Parent}$

Building concepts

Concept conjunction

- Intuition: the intersection of two concepts
- E.g.: Student $\sqcap$ Parent

Concept disjunction

- Intuition: the union of two concepts
- E.g.: Employee $\sqcup$ Student

Building concepts

Concept conjunction

- Intuition: the intersection of two concepts
- E.g.: Student $\sqcap$ Parent

Concept disjunction

- Intuition: the union of two concepts
- E.g.: Employee $\sqcup$ Student

So far we have seen the Boolean fragment of our concept language

- At least as expressive as classical propositional logic

Building concepts

Existential restriction

- Intuition: there is some link with a concept
- E.g.: $\exists$pays.Tax

  Those individuals that pay some tax

Building concepts

Existential restriction

- Intuition: there is some link with a concept
- E.g.: $\exists$pays.Tax

  Those individuals that pay some tax

Value restriction

- Intuition: all links with a concept
- E.g.: $\forall$empBy.Company

  Those individuals that work only for companies

Building concepts

Existential restriction

- Intuition: there is some link with a concept
- E.g.: $\exists \text{pays}.\text{Tax}$

  Those individuals that pay some tax

Value restriction

- Intuition: all links with a concept
- E.g.: $\forall \text{empBy}.\text{Company}$

  Those individuals that work only for companies

So far we have got $\mathcal{ALC}$ (Attributive Language with Complement)

Note: *A description logic primer* describes $\mathcal{SROIQ}$ language
In this course we focus on $\mathcal{ALC}$, which is a fragment of $\mathcal{SROIQ}$

Language

Different flavours

- $\mathcal{ALC}$: $C ::= \top \mid \bot \mid \mathsf{C} \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall r.C \mid \exists r.C$
- $\mathcal{ALCQ}$: $C ::= \ldots \mid \geq nr.C \mid \leq nr.C$
- $\mathcal{EL}$, DL-Lite, $\mathcal{SHIQ}$, $\mathcal{SHOQ}$, $\mathcal{SROIQ}$ (basis of OWL 2), ...

Language

Different flavours

- $\mathcal{ALC}$: $C \ ::= \ \top \ | \ \bot \ | \ \mathsf{C} \ | \ \neg C \ | \ C \sqcap C \ | \ C \sqcup C \ | \ \forall r.C \ | \ \exists r.C$
- $\mathcal{ALCQ}$: $C \ ::= \ \dots \ | \ \geq nr.C \ | \ \leq nr.C$
- $\mathcal{EL}$, DL-Lite, $\mathcal{SHIQ}$, $\mathcal{SHOQ}$, $\mathcal{SROIQ}$ (basis of OWL 2), ...

Language

Different flavours

- $\mathcal{ALC}$: $C \ ::= \ \top \mid \bot \mid \mathsf{C} \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall r.C \mid \exists r.C$
- $\mathcal{ALCQ}$: $C \ ::= \ \ldots \mid \geq nr.C \mid \leq nr.C$
- $\mathcal{EL}$, DL-Lite, $\mathcal{SHIQ}$, $\mathcal{SHOQ}$, $\mathcal{SROIQ}$ (basis of OWL 2), ...

Example

$\neg(\mathsf{Student} \sqcap \mathsf{Parent})$	$\mathsf{Student} \sqcap \neg\exists \mathsf{pays}.\mathsf{Tax}$
$\exists \mathsf{empBy}.\mathsf{Company}$	$\mathsf{EmpStud} \sqcap \exists \mathsf{pays}.\mathsf{Tax}$
$\mathsf{Employee} \sqcup \mathsf{Student} \sqcap \exists \mathsf{worksFor}.\mathsf{Parent}$	$\forall \mathsf{worksFor}.\mathsf{Company}$

Language

Different flavours

- $\mathcal{ALC}$: $C \ ::= \ \top \ | \ \bot \ | \ \mathsf{C} \ | \ \neg C \ | \ C \sqcap C \ | \ C \sqcup C \ | \ \forall r.C \ | \ \exists r.C$
- $\mathcal{ALCQ}$: $C \ ::= \ \dots \ | \ \geq nr.C \ | \ \leq nr.C$
- $\mathcal{EL}$, DL-Lite, $\mathcal{SHIQ}$, $\mathcal{SHOQ}$, $\mathcal{SROIQ}$ (basis of OWL 2), ...

Example

$\neg(\mathsf{Student} \sqcap \mathsf{Parent})$	$\mathsf{Student} \sqcap \neg\exists \mathsf{pays}.\mathsf{Tax}$
$\exists \mathsf{empBy}.\mathsf{Company}$	$\mathsf{EmpStud} \sqcap \exists \mathsf{pays}.\mathsf{Tax}$
$\mathsf{Employee} \sqcup \mathsf{Student} \sqcap \exists \mathsf{worksFor}.\mathsf{Parent}$	$\forall \mathsf{worksFor}.\mathsf{Company}$

With $\mathcal{L}_{\mathcal{ALC}}$ we denote the concept language of $\mathcal{ALC}$

Translating concepts to FOL formulas

Example

The concept:
Student that does not pay any taxes and works only for companies.

- DL: Student $\sqcap$ $\neg\exists$pays.Tax $\sqcap$ $\forall$worksFor.Company
- FOL:
  $Student(x) \wedge \neg(\exists y)(pays(x, y) \wedge Tax(y)) \wedge (\forall y)(worksFor(x, y) \rightarrow Company(y))$

Translating concepts to FOL formulas

Example

The concept:

Student that does not pay any taxes and works only for companies.

- DL: Student $\sqcap \neg\exists$pays.Tax $\sqcap \forall$worksFor.Company
- FOL:
  $Student(x) \land \neg(\exists y)(pays(x,y) \land Tax(y)) \land (\forall y)(worksFor(x,y) \rightarrow Company(y))$

Translation:

- $\sqcap, \sqcup, \neg \implies \land, \lor, \neg$
- $r \implies$ binary relation symbols
- $C \implies$ unary relation symbol
- $\forall r.C \implies \forall y(r(x,y) \rightarrow C(y))$
- $\exists r.C \implies \exists y(r(x,y) \land C(y))$

Semantics

Definition (Interpretation)

Tuple $\mathcal{I} =_{\text{def}} \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where

- $\Delta^{\mathcal{I}}$ is a domain (set of objects)
- $\cdot^{\mathcal{I}}$ is an interpretation function such that

$$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \qquad r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \qquad a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$$

Semantics

Definition (Interpretation)

Tuple $\mathcal{I} =_{\text{def}} \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where

- $\Delta^{\mathcal{I}}$ is a domain (set of objects)
- $\cdot^{\mathcal{I}}$ is an interpretation function such that

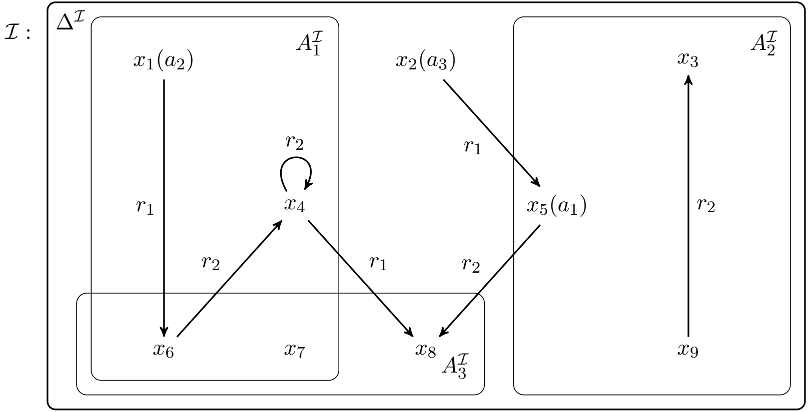
$$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \qquad r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \qquad a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$$

Example

Let $\mathsf{C} = \{A_1, A_2, A_3\}$, $\mathsf{R} = \{r_1, r_2\}$, $\mathsf{I} = \{a_1, a_2, a_3\}$. Let $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ where:

- $\Delta^{\mathcal{I}} = \{x_i \mid 1 \leq i \leq 9\}$, $a_1^{\mathcal{I}} = x_5$, $a_2^{\mathcal{I}} = x_1$, $a_3^{\mathcal{I}} = x_2$
- $A_1^{\mathcal{I}} = \{x_1, x_4, x_6, x_7\}$, $A_2^{\mathcal{I}} = \{x_3, x_5, x_9\}$, $A_3^{\mathcal{I}} = \{x_6, x_7, x_8\}$
- $r_1^{\mathcal{I}} = \{(x_1, x_6), (x_4, x_8), (x_2, x_5)\}$, $r_2^{\mathcal{I}} = \{(x_4, x_4), (x_6, x_4), (x_5, x_8), (x_9, x_3)\}$

Semantics


$\mathcal{I}:$
$\Delta^{\mathcal{I}}$
$A_1^{\mathcal{I}}$
$A_2^{\mathcal{I}}$
$A_3^{\mathcal{I}}$
$x_1(a_2)$
$x_2(a_3)$
x_3
x_4
$x_5(a_1)$
x_6
x_7
x_8
x_9
r_1
r_2
r_2
r_1
r_1
r_2
r_2

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \quad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \quad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \qquad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \qquad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \qquad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \quad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \quad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \qquad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \qquad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \qquad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \quad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \quad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Semantics

Extending DL interpretations

$$\top^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} =_{\text{def}} \emptyset \quad (\neg C)^{\mathcal{I}} =_{\text{def}} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad (C \sqcup D)^{\mathcal{I}} =_{\text{def}} C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$$

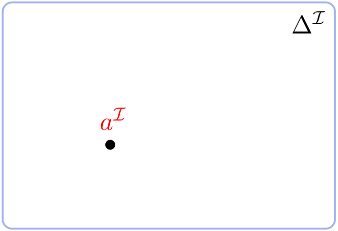
$$(\forall r.C)^{\mathcal{I}} =_{\text{def}} \{x \in \Delta^{\mathcal{I}} \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$$

Notation: $r^{\mathcal{I}}(x) = \{y \in \Delta^{\mathcal{I}} \mid (x, y) \in r^{\mathcal{I}}\}$

Definition (Concept Satisfiability)

A concept C is satisfiable if there is $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ s.t. $C^{\mathcal{I}} \neq \emptyset$

Semantics



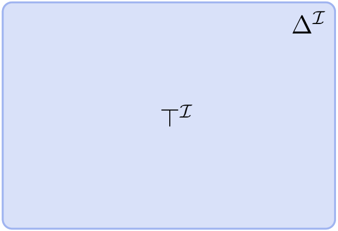
Individual names

- At most one element of $\Delta^{\mathcal{I}}$

Semantics

The 'top' concept

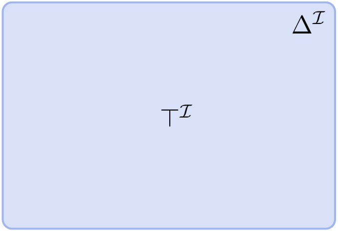
- Everything is in $\top^{\mathcal{I}}$
- Also called Thing



Semantics

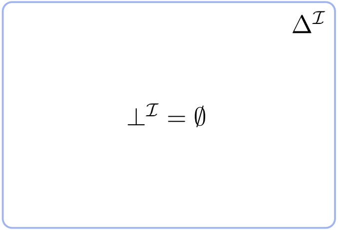
The ‘top’ concept

- Everything is in $\top^{\mathcal{I}}$
- Also called Thing



The ‘bottom’ concept

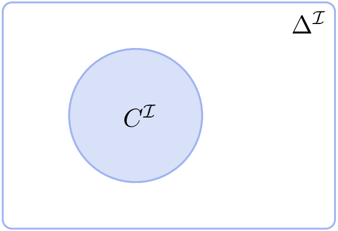
- $\bot^{\mathcal{I}}$ is in everything
- Also called Nothing



Semantics

Arbitrary concept

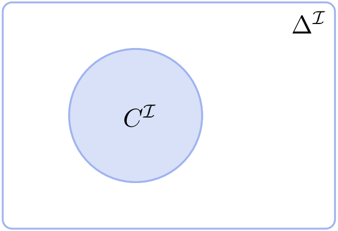
- A class in the domain
- $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$



Semantics

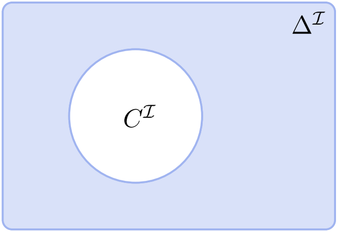
Arbitrary concept

- A class in the domain
- $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$



Concept negation

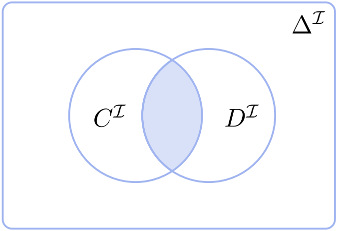
- The complement of a concept
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$



Semantics

Concept conjunction

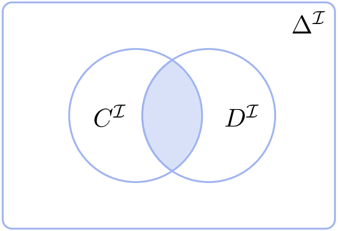
- The intersection of two classes
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$



Semantics

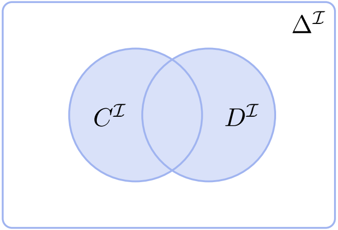
Concept conjunction

- The intersection of two classes
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$



Concept disjunction

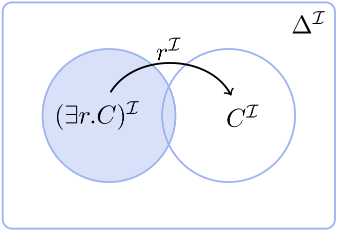
- The union of two classes
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$



Semantics

Existential restriction

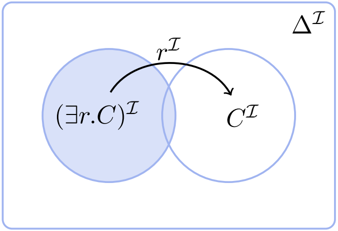
- At least one value of a class
- $(\exists r.C)^{\mathcal{I}} = \{x \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$



Semantics

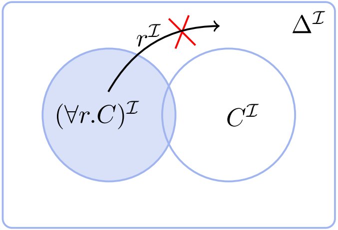
Existential restriction

- At least one value of a class
- $(\exists r.C)^{\mathcal{I}} = \{x \mid r^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\}$



Value restriction

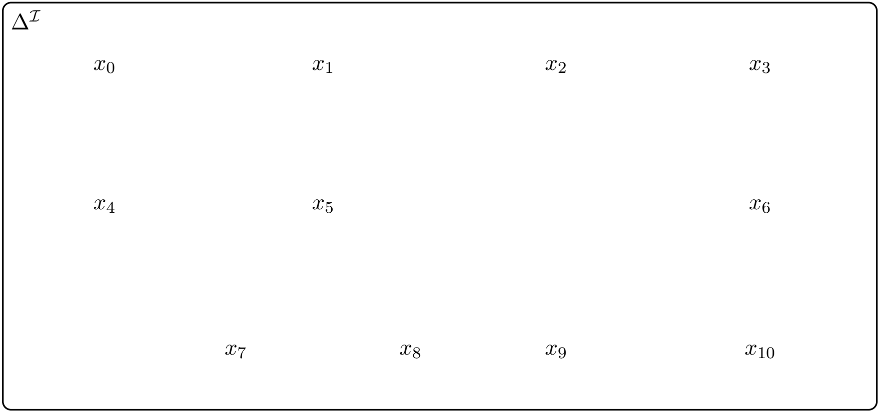
- All values of a class
- $(\forall r.C)^{\mathcal{I}} = \{x \mid r^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\}$



Semantics

An interpretation is a complete description of the world

$\mathcal{I}$:



Semantics

An interpretation is a complete description of the world

$\mathcal{I}:$ $\Delta^{\mathcal{I}}$

x_0 x_1 x_2(mary) x_3

x_4 x_5 x_6

x_7 x_8 x_9 x_{10}

Semantics

An interpretation is a complete description of the world

$\mathcal{I}$: $\Delta^{\mathcal{I}}$

x_0 x_1 x_2(mary) x_3

x_4 x_5(john) x_6

x_7 x_8 x_9 x_{10}

Semantics

An interpretation is a complete description of the world

$\mathcal{I}$: $\Delta^{\mathcal{I}}$

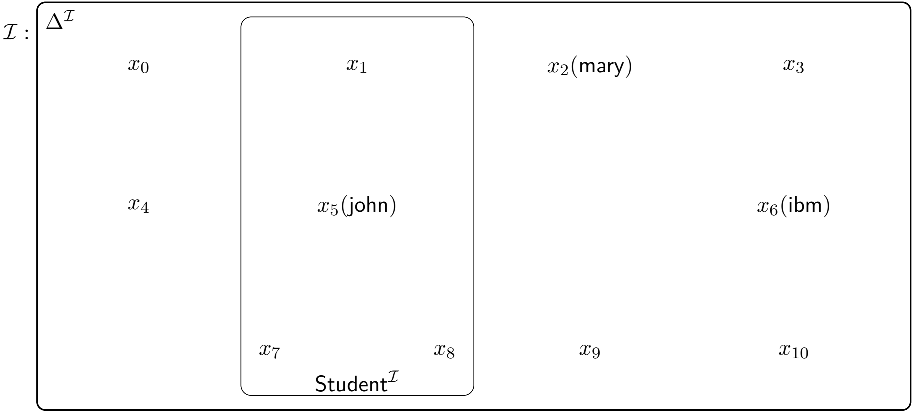
x_0 x_1 x_2(mary) x_3

x_4 x_5(john) x_6(ibm)

x_7 x_8 x_9 x_{10}

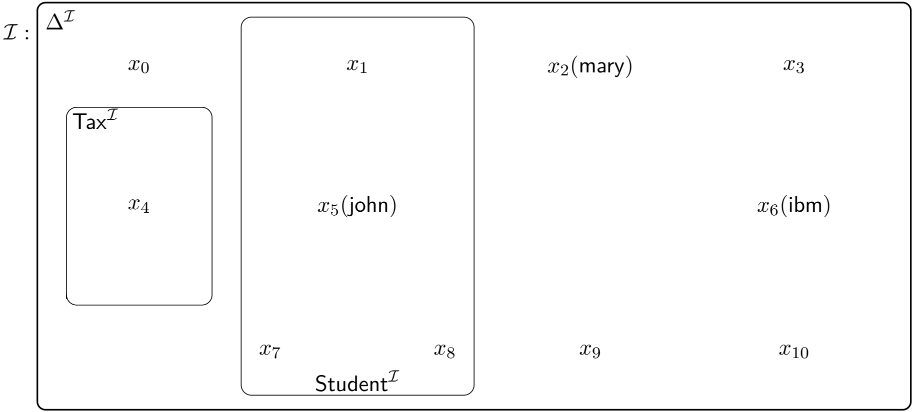
Semantics

An interpretation is a complete description of the world



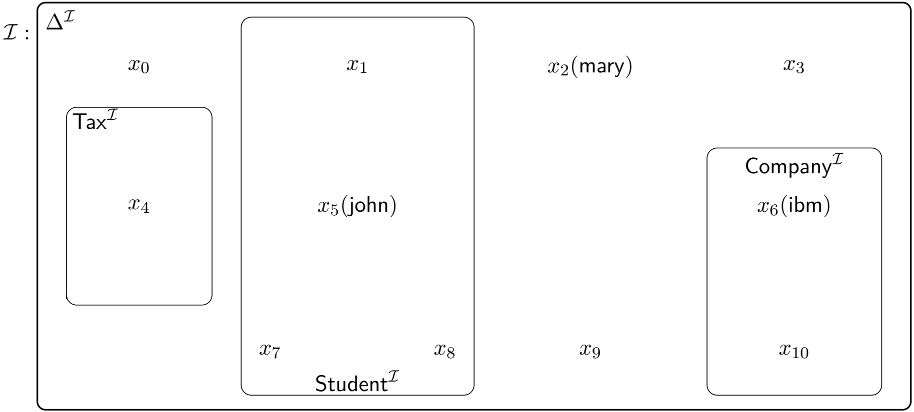
Semantics

An interpretation is a complete description of the world



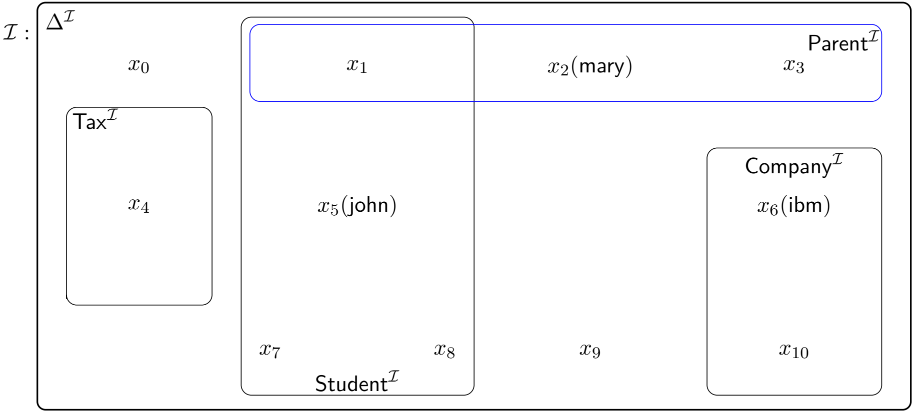
Semantics

An interpretation is a complete description of the world



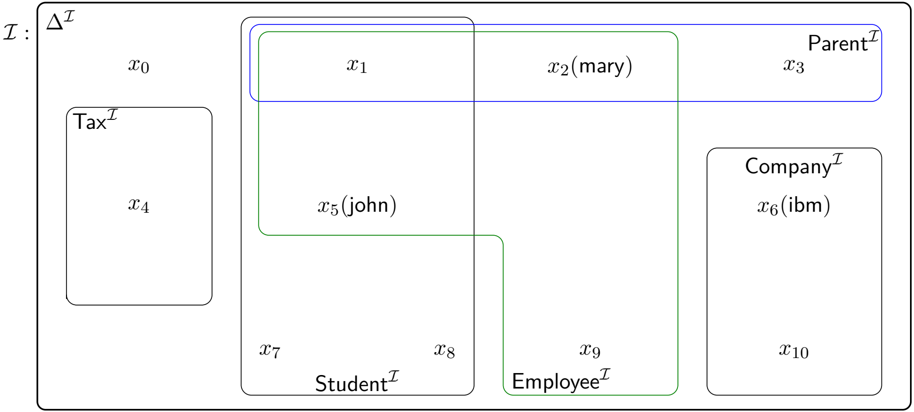
Semantics

An interpretation is a complete description of the world



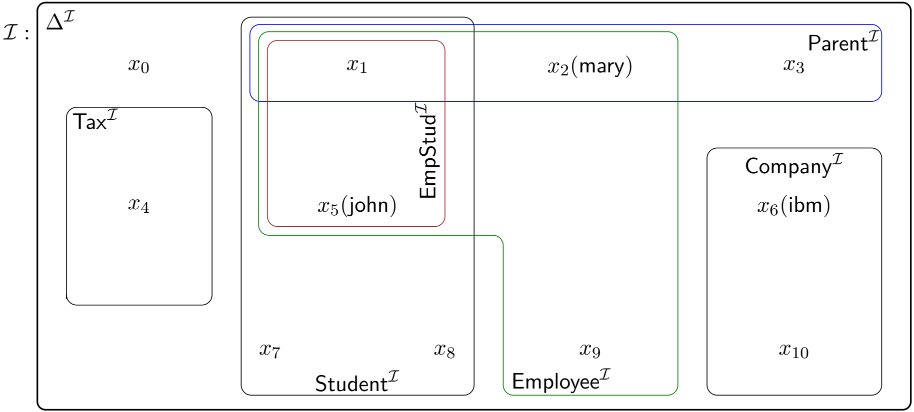
Semantics

An interpretation is a complete description of the world



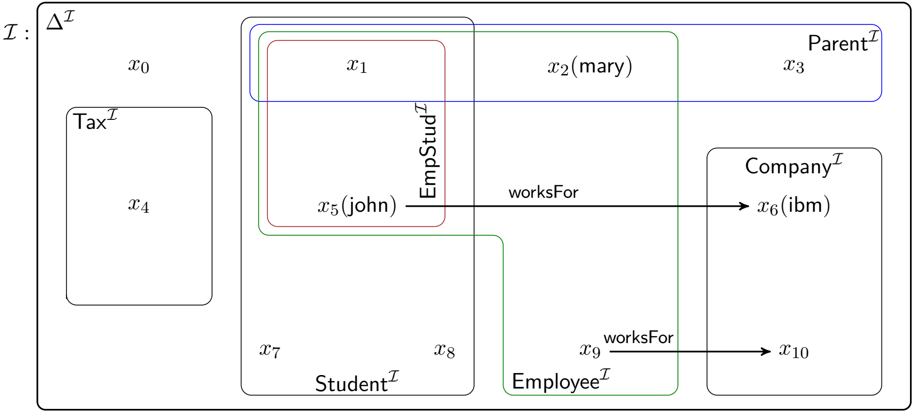
Semantics

An interpretation is a complete description of the world



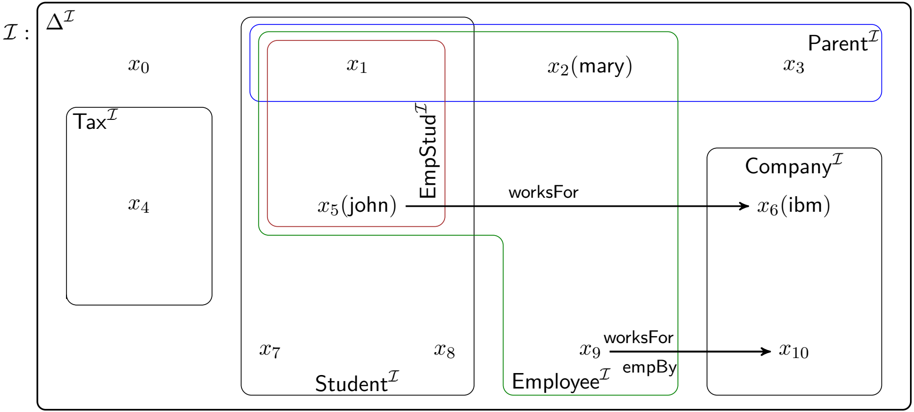
Semantics

An interpretation is a complete description of the world



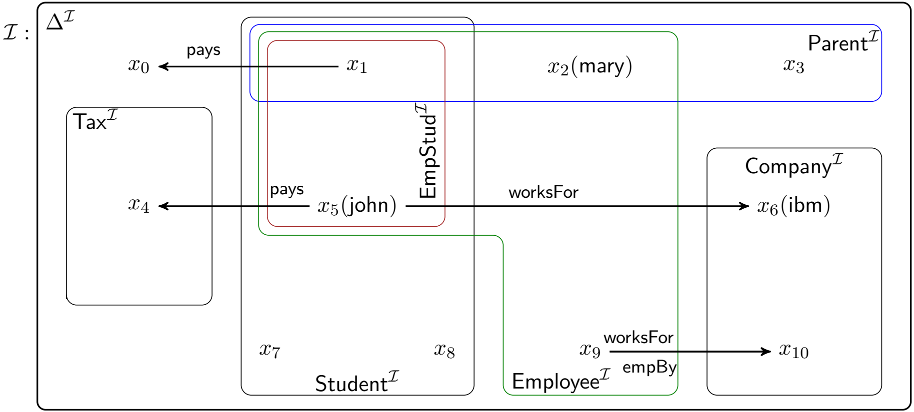
Semantics

An interpretation is a complete description of the world



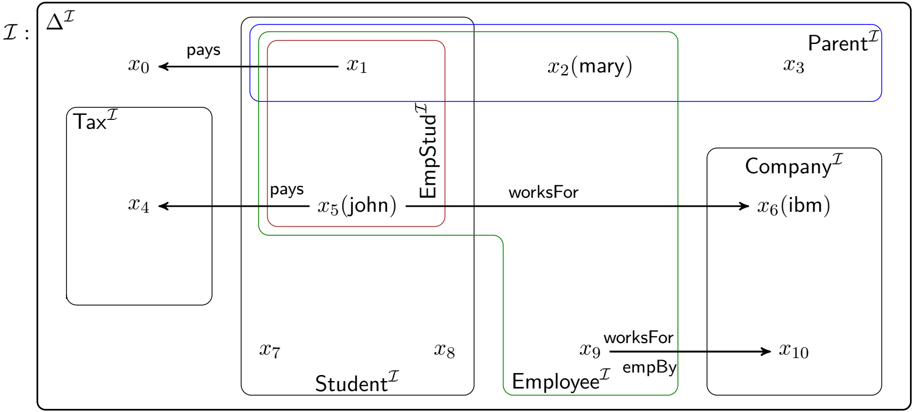
Semantics

An interpretation is a complete description of the world



Semantics

An interpretation is a complete description of the world



$$((\text{EmpStud} \sqcup \text{Parent}) \sqcap \exists \text{pays}.\top)^{\mathcal{I}} = \{x_1, x_5\}$$

Outline

Motivation

Concept language of $\mathcal{ALC}$

$\top,\ \bot$	(constants)
A	(atomic concept)
$\neg C$	(complement of C)
$C \sqcap D$	(intersection of C and D)
$C \sqcup D$	(union of C and D)
$\exists r.C$	(existential restriction)
$\forall r.C$	(value restriction)

Motivation

Concept language of $\mathcal{ALC}$

$\top,\ \bot$	(constants)
A	(atomic concept)
$\neg C$	(complement of C)
$C \sqcap D$	(intersection of C and D)
$C \sqcup D$	(union of C and D)
$\exists r.C$	(existential restriction)
$\forall r.C$	(value restriction)

Something is missing

Motivation

Concept language of $\mathcal{ALC}$

$\top,\ \bot$	(constants)
A	(atomic concept)
$\neg C$	(complement of C)
$C \sqcap D$	(intersection of C and D)
$C \sqcup D$	(union of C and D)
$\exists r.C$	(existential restriction)
$\forall r.C$	(value restriction)

Something is missing

- The central notion in logic: C '$\rightarrow$' D

Motivation

Concept language of $\mathcal{ALC}$

$\top, \bot$	(constants)
A	(atomic concept)
$\neg C$	(complement of C)
$C \sqcap D$	(intersection of C and D)
$C \sqcup D$	(union of C and D)
$\exists r.C$	(existential restriction)
$\forall r.C$	(value restriction)

Something is missing

- The central notion in logic: C '$\rightarrow$' D
- What would C '$\rightarrow$' D mean here?

Motivation

Concept language of $\mathcal{ALC}$

$\top,\ \bot$	(constants)
A	(atomic concept)
$\neg C$	(complement of C)
$C \sqcap D$	(intersection of C and D)
$C \sqcup D$	(union of C and D)
$\exists r.C$	(existential restriction)
$\forall r.C$	(value restriction)

Something is missing

- The central notion in logic: C '$\rightarrow$' D
- What would C '$\rightarrow$' D mean here? (We already have $\neg C \sqcup D$)

Motivation

Concept language of $\mathcal{ALC}$

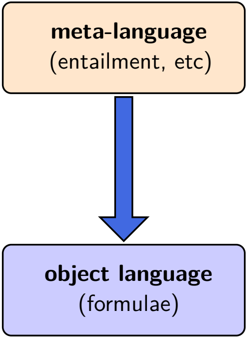
$\top$, $\bot$	(constants)
A	(atomic concept)
$\neg C$	(complement of C)
$C \sqcap D$	(intersection of C and D)
$C \sqcup D$	(union of C and D)
$\exists r.C$	(existential restriction)
$\forall r.C$	(value restriction)

Something is missing

- The central notion in logic: C '$\rightarrow$' D
- What would C '$\rightarrow$' D mean here? (We already have $\neg C \sqcup D$)
- DLs have a version of '$\rightarrow$' that is very special

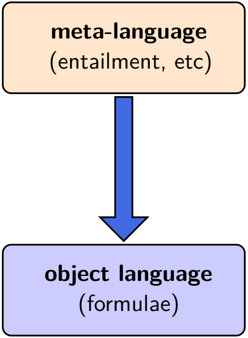
Statements

In many logics

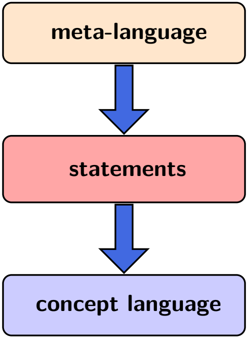


Statements

In many logics



In DLs



- Two levels of language
- Two notions of ‘entailment’
- Two notions of ‘satisfaction’

Making statements

Subsumption

- Concept inclusion
- Employed students are students
- Employed students are employees

Making statements

Subsumption

- Concept inclusion
- Employed students are students
- Employed students are employees

Instantiation or assertions

- Concept and role membership
- John is an employed student (John instantiates employed student)
- John works for IBM (John and IBM instantiate works for)

Making statements

Subsumption

- Concept inclusion
- Employed students are students
- Employed students are employees

Instantiation or assertions

- Concept and role membership
- John is an employed student (John instantiates employed student)
- John works for IBM (John and IBM instantiate works for)

Statements talk about concepts, roles and individuals

Making statements

Subsumption

- Concept inclusion
- Employed students are students
- Employed students are employees

Instantiation or assertions

- Concept and role membership
- John is an employed student (John instantiates employed student)
- John works for IBM (John and IBM instantiate works for)

Statements talk about concepts, roles and individuals

They are not concepts! They are in the 'in-between' language

Subsumption statements

$$C \sqsubseteq D$$

Intuition

- D subsumes C (or C is subsumed by D)
- C is more specific than D (or D is more general than C)
- Formalise one aspect of is-a relations

Subsumption statements

$$C \sqsubseteq D$$

Intuition

- D subsumes C (or C is subsumed by D)
- C is more specific than D (or D is more general than C)
- Formalise one aspect of is-a relations

Example

- $\mathsf{EmpStud} \sqsubseteq \mathsf{Student} \sqcap \mathsf{Employee}$, $\mathsf{Employee} \sqsubseteq \exists \mathsf{worksFor}.\top$
- $\mathsf{EmpStud} \sqsubseteq \exists \mathsf{pays}.\mathsf{Tax}$, $\mathsf{Student} \sqcap \neg \mathsf{Employee} \sqsubseteq \neg \exists \mathsf{pays}.\mathsf{Tax}$

Subsumption statements

$$C \sqsubseteq D$$

Intuition

- D subsumes C (or C is subsumed by D)
- C is more specific than D (or D is more general than C)
- Formalise one aspect of is-a relations

Example

- EmpStud $\sqsubseteq$ Student $\sqcap$ Employee, Employee $\sqsubseteq$ $\exists$worksFor.$\top$
- EmpStud $\sqsubseteq$ $\exists$pays.Tax, Student $\sqcap$ $\neg$Employee $\sqsubseteq$ $\neg\exists$pays.Tax

Central notion in DL terminologies (taxonomies)

Subsumption statements

$$C \sqsubseteq D$$

Semantics

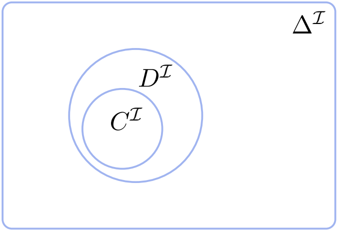
- $\mathcal{I} \Vdash C \sqsubseteq D$ ($\mathcal{I}$ satisfies $C \sqsubseteq D$)   if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- First level of 'entailment': all C-objects are D-objects

Subsumption statements

$$C \sqsubseteq D$$

Semantics

- $\mathcal{I} \Vdash C \sqsubseteq D$ ($\mathcal{I}$ satisfies $C \sqsubseteq D$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- First level of 'entailment': all C-objects are D-objects



Subsumption statements

$$C \equiv D$$

Concept equivalence

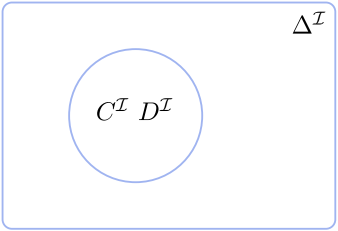
- Just an abbreviation for $C \sqsubseteq D$ and $D \sqsubseteq C$
- $\mathcal{I} \Vdash C \equiv D$ if $\mathcal{I} \Vdash C \sqsubseteq D$ and $\mathcal{I} \Vdash D \sqsubseteq C$
- $\mathcal{I} \Vdash C \equiv D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$

Subsumption statements

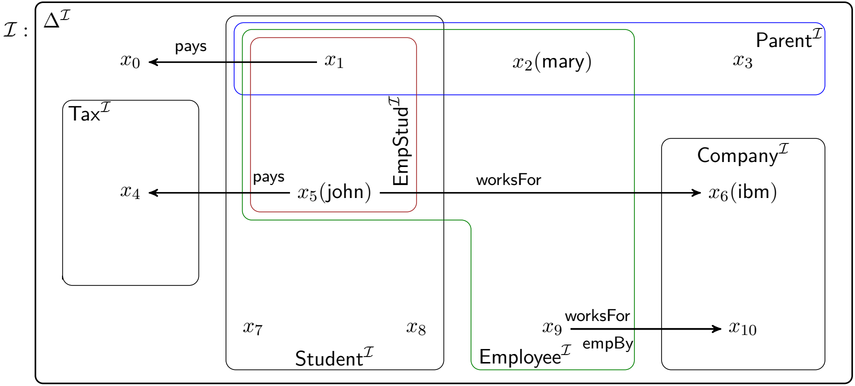
$$C \equiv D$$

Concept equivalence

- Just an abbreviation for $C \sqsubseteq D$ and $D \sqsubseteq C$
- $\mathcal{I} \Vdash C \equiv D$ if $\mathcal{I} \Vdash C \sqsubseteq D$ and $\mathcal{I} \Vdash D \sqsubseteq C$
- $\mathcal{I} \Vdash C \equiv D$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$



Example



- $\mathcal{I} \Vdash$ EmpStud $\sqsubseteq$ Student $\sqcap$ Employee
- $\mathcal{I} \Vdash \exists$worksFor.$\top \sqsubseteq$ Employee

Assertions

$$a : C \qquad\qquad (a, b) : r$$

Notation in *"A description logic primer"*: $C(a) \qquad r(a, b)$

Intuition

- a is an instance of C
- a and b are related via r (or (a, b) is an instance of r)
- Formalise another aspect of is-a relations

Assertions

$$a : C \qquad\qquad (a, b) : r$$

Notation in *"A description logic primer"*: $C(a) \qquad r(a, b)$

Intuition

- a is an instance of C
- a and b are related via r (or (a, b) is an instance of r)
- Formalise another aspect of is-a relations

Example

- $\mathsf{john} : \mathsf{EmpStud}, \quad \mathsf{mary} : \mathsf{Parent} \sqcap \neg\exists \mathsf{worksFor}.\top \sqcap \neg\exists \mathsf{pays}.\mathsf{Tax}$
- $(\mathsf{john}, \mathsf{ibm}) : \mathsf{worksFor}$

Assertions

$$a : C \qquad\qquad (a, b) : r$$

Notation in *"A description logic primer"*: $C(a) \qquad r(a, b)$

Intuition

- a is an instance of C
- a and b are related via r (or (a, b) is an instance of r)
- Formalise another aspect of is-a relations

Example

- $\mathsf{john} : \mathsf{EmpStud}, \qquad \mathsf{mary} : \mathsf{Parent} \sqcap \neg\exists \mathsf{worksFor}.\top \sqcap \neg\exists \mathsf{pays}.\mathsf{Tax}$
- $(\mathsf{john}, \mathsf{ibm}) : \mathsf{worksFor}$

Central notion in DL 'databases'

Assertions

$$a : C \qquad\qquad (a, b) : r$$

Semantics

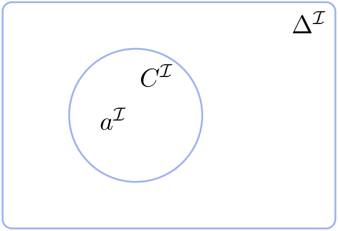
- $\mathcal{I} \Vdash a : C$ ($\mathcal{I}$ satisfies $a : C$) if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I} \Vdash (a, b) : r$ ($\mathcal{I}$ satisfies $(a, b) : r$) if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$
- First level of 'satisfaction': a is a 'model' of C, (a, b) is a 'model' of r

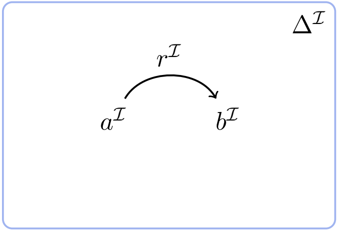
Assertions

$$a : C \qquad (a, b) : r$$

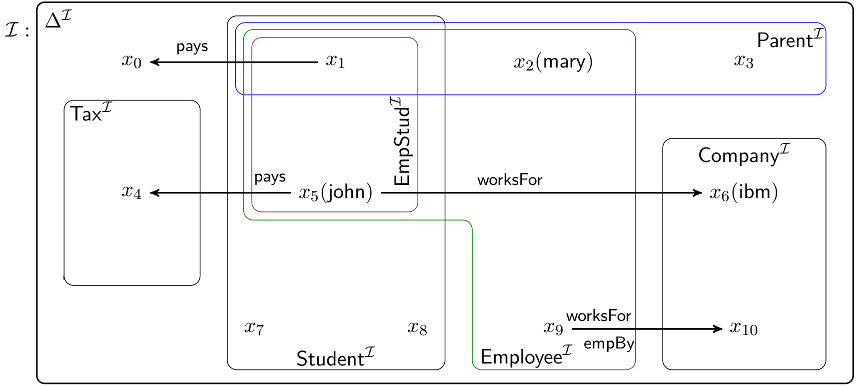
Semantics

- $\mathcal{I} \Vdash a : C$ ($\mathcal{I}$ satisfies $a : C$) if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I} \Vdash (a, b) : r$ ($\mathcal{I}$ satisfies $(a, b) : r$) if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$
- First level of 'satisfaction': a is a 'model' of C, (a, b) is a 'model' of r





Example



- $\mathcal{I} \Vdash$ john : Employee $\sqcap \exists$pays.Tax
- $\mathcal{I} \Vdash$ (john, ibm) : worksFor

Subsumptions and assertions

Validity

- Let α denote a statement
- $\models \alpha$ (α is valid) if $\mathcal{I} \Vdash \alpha$ for every $\mathcal{I}$

Subsumptions and assertions

Validity

- Let α denote a statement
- $\models \alpha$ (α is valid) if $\mathcal{I} \Vdash \alpha$ for every $\mathcal{I}$

Example

- $\models \neg(C \sqcap D) \equiv (\neg C \sqcup \neg D)$
- $\models \forall r.(C \sqcap D) \sqsubseteq \forall r.C$
- $\not\models \exists r.\top \sqsubseteq \exists r.C$
- $\models a : C \sqcup \neg C$
- $\models \neg(C \sqcup D) \equiv (\neg C \sqcap \neg D)$
- $\not\models \forall r.C \sqsubseteq \forall r.(C \sqcap D)$
- $\models \exists r.C \sqsubseteq \exists r.\top$
- $\not\models (a, b) : r$

Subsumptions and assertions

Validity

- Let α denote a statement
- $\models \alpha$ (α is valid) if $\mathcal{I} \Vdash \alpha$ for every $\mathcal{I}$

Example

- $\models \neg(C \sqcap D) \equiv (\neg C \sqcup \neg D)$
- $\models \neg(C \sqcup D) \equiv (\neg C \sqcap \neg D)$
- $\models \forall r.(C \sqcap D) \sqsubseteq \forall r.C$
- $\not\models \forall r.C \sqsubseteq \forall r.(C \sqcap D)$
- $\not\models \exists r.\top \sqsubseteq \exists r.C$
- $\models \exists r.C \sqsubseteq \exists r.\top$
- $\models a : C \sqcup \neg C$
- $\not\models (a, b) : r$

Watch out: Statements can be valid; concepts cannot!

Translating statements to FOL formulas

Note: If C is a concept, then its translation to FOL is a formula with one (unbounded) variable x:

$$C \Longrightarrow C(x)$$

- $C \sqsubseteq D \Longrightarrow \forall x(C(x) \to D(x))$
- $a : C \Longrightarrow C(a)$
- $(a, b) : r \Longrightarrow r(a, b)$

Outline

Fragments of FOL

Formal Ontologies

Introduction to DLs

Making Statements

DL Knowledge Bases & entailment

TBoxes and ABoxes

Intensional knowledge

- Set of subsumption statements
- Intuition: provide definitions of concepts (a terminology)
- Called the TBox (terminological box). Notation: $\mathcal{T}$

Extensional knowledge

- Concept and role assertions
- Intuition: provide an instantiation of concepts and roles (a 'database')
- Called the ABox (assertion box). Notation: $\mathcal{A}$

Definition (Knowledge base)

A DL knowledge base (a.k.a. ontology) is a tuple $\mathcal{KB} =_{\text{def}} \langle \mathcal{T}, \mathcal{A} \rangle$

Knowledge bases

Example (The student ontology in DL)

$$\mathcal{T} = \left\{ \begin{array}{c} \mathsf{EmpStud} \equiv \mathsf{Student} \sqcap \mathsf{Employee}, \\ \mathsf{Student} \sqcap \neg\mathsf{Employee} \sqsubseteq \neg\exists \mathsf{pays}.\mathsf{Tax}, \\ \mathsf{EmpStud} \sqcap \neg\mathsf{Parent} \sqsubseteq \exists \mathsf{pays}.\mathsf{Tax}, \\ \mathsf{EmpStud} \sqcap \mathsf{Parent} \sqsubseteq \neg\exists \mathsf{pays}.\mathsf{Tax}, \\ \exists \mathsf{worksFor}.\mathsf{Company} \sqsubseteq \mathsf{Employee} \end{array} \right\}$$

$$\mathcal{A} = \left\{ \begin{array}{c} \mathsf{ibm} : \mathsf{Company}, \\ \mathsf{mary} : \mathsf{Parent}, \\ \mathsf{john} : \mathsf{EmpStud}, \\ (\mathsf{john}, \mathsf{ibm}) : \mathsf{worksFor} \end{array} \right\}$$

classes
relations
individuals

Knowledge bases

Semantics

- $\mathcal{I} \Vdash \mathcal{T}$ if $\mathcal{I} \Vdash C \sqsubseteq D$ for every $C \sqsubseteq D \in \mathcal{T}$
- $\mathcal{I} \Vdash \mathcal{A}$ if:
  - $\mathcal{I} \Vdash a : C$ for every $a : C \in \mathcal{A}$, and
  - $\mathcal{I} \Vdash (a, b) : r$ for every $(a, b) : r \in \mathcal{A}$

Knowledge bases

Semantics

- $\mathcal{I} \Vdash \mathcal{T}$ if $\mathcal{I} \Vdash C \sqsubseteq D$ for every $C \sqsubseteq D \in \mathcal{T}$
- $\mathcal{I} \Vdash \mathcal{A}$ if:
  - $\mathcal{I} \Vdash a : C$ for every $a : C \in \mathcal{A}$, and
  - $\mathcal{I} \Vdash (a, b) : r$ for every $(a, b) : r \in \mathcal{A}$

Moreover

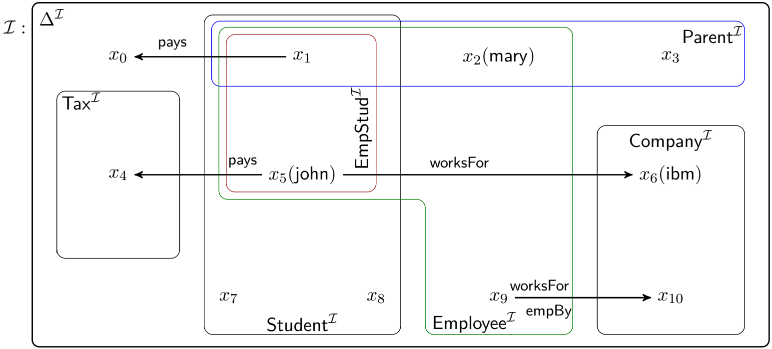
- If $\mathcal{I} \Vdash \mathcal{T} \cup \mathcal{A}$, then $\mathcal{I}$ is a model of $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$
- $\mathcal{KB}$ is satisfiable if it has a model

Example

- Let $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$, where:

$$\mathcal{T} = \left\{ \begin{array}{c} \mathsf{EmpStud} \equiv \mathsf{Student} \sqcap \mathsf{Employee}, \\ \mathsf{Student} \sqcap \neg\mathsf{Employee} \sqsubseteq \neg\exists\mathsf{pays}.\mathsf{Tax}, \\ \mathsf{EmpStud} \sqcap \neg\mathsf{Parent} \sqsubseteq \exists\mathsf{pays}.\mathsf{Tax}, \\ \mathsf{EmpStud} \sqcap \mathsf{Parent} \sqsubseteq \neg\exists\mathsf{pays}.\mathsf{Tax}, \\ \exists\mathsf{worksFor}.\mathsf{Company} \sqsubseteq \mathsf{Employee} \end{array} \right\} \qquad \mathcal{A} = \left\{ \begin{array}{c} \mathsf{ibm} : \mathsf{Company}, \\ \mathsf{mary} : \mathsf{Parent}, \\ \mathsf{john} : \mathsf{EmpStud}, \\ (\mathsf{john}, \mathsf{ibm}) : \mathsf{worksFor} \end{array} \right\}$$

- The interpretation $\mathcal{I}$ below is a model of $\mathcal{KB}$ (check)



What does follow from a KB?

Entailment from KBs

- Defined on the level of statements (not concepts)

Obvious definition of entailment

- $\mathcal{KB} \models \alpha$ if $\mathcal{I} \Vdash \alpha$ for every $\mathcal{I}$ s.t. $\mathcal{I} \Vdash \mathcal{KB}$

What does follow from a KB?

Example

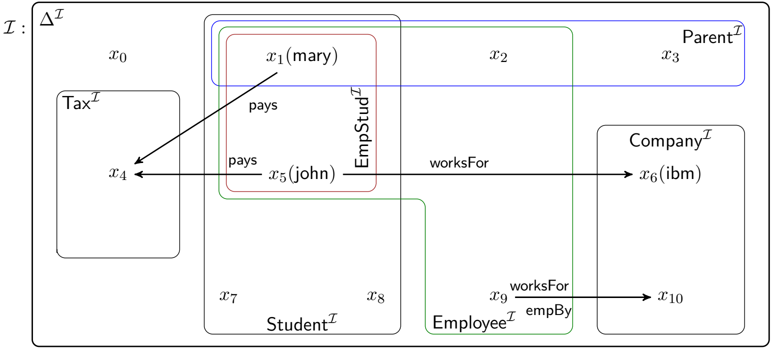
$$\mathcal{T} = \left\{ \begin{array}{c} \mathsf{EmpStud} \equiv \mathsf{Student} \sqcap \mathsf{Employee}, \\ \mathsf{Student} \sqcap \neg\mathsf{Employee} \sqsubseteq \neg\exists \mathsf{pays}.\mathsf{Tax}, \\ \mathsf{EmpStud} \sqcap \neg\mathsf{Parent} \sqsubseteq \exists \mathsf{pays}.\mathsf{Tax}, \\ \mathsf{EmpStud} \sqcap \mathsf{Parent} \sqsubseteq \neg\exists \mathsf{pays}.\mathsf{Tax}, \\ \exists \mathsf{worksFor}.\mathsf{Company} \sqsubseteq \mathsf{Employee} \end{array} \right\} \quad \mathcal{A} = \left\{ \begin{array}{c} \mathsf{ibm} : \mathsf{Company}, \\ \mathsf{mary} : \mathsf{Parent}, \\ \mathsf{john} : \mathsf{EmpStud}, \\ (\mathsf{john}, \mathsf{ibm}) : \mathsf{worksFor} \end{array} \right\}$$

- $\mathcal{KB} \models \mathsf{Student} \sqcap \exists \mathsf{worksFor}.\mathsf{Company} \sqcap \neg\mathsf{Parent} \sqsubseteq \mathsf{EmpStud} \sqcap \exists \mathsf{pays}.\mathsf{Tax}$
- $\mathcal{KB} \models \mathsf{john} : \mathsf{Student} \sqcap \exists \mathsf{worksFor}.\mathsf{Company}$
- $\mathcal{KB} \not\models \mathsf{mary} : \neg\exists \mathsf{pays}.\mathsf{Tax}$

What does follow from a KB?

Example

- $\mathcal{KB} \not\models$ mary : $\neg\exists$pays.Tax



Open-world assumption: Truth of non-derivable statements is just unknown

Thanks to

Slides based on ESSLLI 2018 course on description logic
by Ivan Varzinczak, Université d'Artois, France

ESSLLI 2021 in Utrecht!