Description Logics

Petr Křemen
petr.kremen@fel.cvut.cz

FEL ČVUT

Our plan

Towards Description Logics

$\mathcal{ALC}$ Language

Towards Description Logics

Let's review our knowledge about FOPL [2]

- What is a *term*, *axiom/formula*, *theory*, *model*, *universal closure*, *resolution*, *logical consequence* ?
- What is an open-world assumption (OWA)/closed-world assumption (CWA) ?
- What is the difference between a predicate (relation) and a predicate symbol ?
- What does it mean, when saying that FOPL is *undecidable* ?
- What does it mean, when saying that FOPL is *monotonic* ?
- What is the idea behind *Deduction Theorem*, *Soundness*, *Completeness* ?

[2] First Order Predicate Logic

Let's review our knowledge about FOPL [2]

- What is a *term*, *axiom/formula*, *theory*, *model*, *universal closure*, *resolution*, *logical consequence* ?
- What is an open-world assumption (OWA)/closed-world assumption (CWA) ?
- What is the difference between a predicate (relation) and a predicate symbol ?
- What does it mean, when saying that FOPL is *undecidable* ?
- What does it mean, when saying that FOPL is *monotonic* ?
- What is the idea behind *Deduction Theorem*, *Soundness*, *Completeness* ?

[2] First Order Predicate Logic

Let's review our knowledge about FOPL [2]

- What is a *term*, *axiom/formula*, *theory*, *model*, *universal closure*, *resolution*, *logical consequence* ?
- What is an open-world assumption (OWA)/closed-world assumption (CWA) ?
- What is the difference between a predicate (relation) and a predicate symbol ?
- What does it mean, when saying that FOPL is *undecidable* ?
- What does it mean, when saying that FOPL is *monotonic* ?
- What is the idea behind *Deduction Theorem*, *Soundness*, *Completeness* ?

[2] First Order Predicate Logic

Let's review our knowledge about FOPL [2]

- What is a *term*, *axiom/formula*, *theory*, *model*, *universal closure*, *resolution*, *logical consequence* ?
- What is an open-world assumption (OWA)/closed-world assumption (CWA) ?
- What is the difference between a predicate (relation) and a predicate symbol ?
- What does it mean, when saying that FOPL is *undecidable* ?
- What does it mean, when saying that FOPL is *monotonic* ?
- What is the idea behind *Deduction Theorem*, *Soundness*, *Completeness* ?

[2] First Order Predicate Logic

Let's review our knowledge about FOPL [2]

- What is a *term*, *axiom/formula*, *theory*, *model*, *universal closure*, *resolution*, *logical consequence* ?
- What is an open-world assumption (OWA)/closed-world assumption (CWA) ?
- What is the difference between a predicate (relation) and a predicate symbol ?
- What does it mean, when saying that FOPL is *undecidable* ?
- What does it mean, when saying that FOPL is *monotonic* ?
- What is the idea behind *Deduction Theorem*, *Soundness*, *Completeness* ?

[2] First Order Predicate Logic

Let's review our knowledge about FOPL [2]

- What is a *term*, *axiom/formula*, *theory*, *model*, *universal closure*, *resolution*, *logical consequence* ?
- What is an open-world assumption (OWA)/closed-world assumption (CWA) ?
- What is the difference between a predicate (relation) and a predicate symbol ?
- What does it mean, when saying that FOPL is *undecidable* ?
- What does it mean, when saying that FOPL is *monotonic* ?
- What is the idea behind *Deduction Theorem*, *Soundness*, *Completeness* ?

[2] First Order Predicate Logic

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
    - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains.
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
  - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains.
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology.

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
  - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains.
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology.

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
  - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology

laboratory Gerstner

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
  - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains.
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
  - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains.
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology.

laboratory Gerstner

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
  - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains.
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology.

Isn't FOPL enough ?

- Why do we speak about modal logics, description logics, etc. ?
  - ☹ FOPL is undecidable – many logical consequences cannot be verified in finite time.
  - We often do not need full expressiveness of FOL.
- Well, we have Prolog – wide-spread and optimized implementation of FOPL, right ?
  - ☹ Prolog is not an implementation of FOPL – OWA vs. CWA, negation as failure, problems in expressing disjunctive knowledge, etc.
- Well, relational databases are also not enough ?
  - RDBMS accept CWA and support just finite domains.
  - RDBMS are not flexible enough – DB model change is complicated that adding/removing an axiom from an ontology.

Technologies sketched so far aren't enough ?

- Semantic networks and Frames
  - Lack well defined (declarative) semantics
  - What is the semantiics of a "slot" in a frame (relation in semantic networks) ? The slot **must/might** be filled **once/multiple times** ?
- Conceptual graphs are beyond FOPL (thus undecidable).
- What are description logics (DLs)?
  - logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
  - první jazyky vznikly jako snaha o formalizaci sémantických sítí a rámců. První implementace v 80's – systémy KL-ONE, KAON, Classic .

Technologies sketched so far aren't enough ?

- Semantic networks and Frames
  - Lack well defined (declarative) semantics
  - What is the semantiics of a "slot" in a frame (relation in semantic networks) ? The slot **must/might** be filled **once/multiple times** ?
- Conceptual graphs are beyond FOPL (thus undecidable).
- What are description logics (DLs)?

Technologies sketched so far aren't enough ?

- Semantic networks and Frames
  - Lack well defined (declarative) semantics
  - What is the semantiics of a "slot" in a frame (relation in semantic networks) ? The slot **must/might** be filled **once/multiple times** ?
- Conceptual graphs are beyond FOPL (thus undecidable).
- What are description logics (DLs)?
  - logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL
  - první jazyky vznikly jako snaha o formalizaci sémantických sítí a rámců. První implementace v 80's – systémy KL-ONE, KAON, Classic .

laboratory Gerstner

Technologies sketched so far aren't enough ?

- Semantic networks and Frames
  - Lack well defined (declarative) semantics
  - What is the semantiics of a "slot" in a frame (relation in semantic networks) ? The slot **must/might** be filled **once/multiple times** ?
- Conceptual graphs are beyond FOPL (thus undecidable).
- What are description logics (DLs)?
  - logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
  - první jazyky vznikly jako snaha o formalizaci sémantických sítí a rámců. První implementace v 80's – systémy KL-ONE, KAON, Classic.

Technologies sketched so far aren't enough ?

- Semantic networks and Frames
  - Lack well defined (declarative) semantics
  - What is the semantiics of a "slot" in a frame (relation in semantic networks) ? The slot **must/might** be filled **once/multiple times** ?
- Conceptual graphs are beyond FOPL (thus undecidable).
- What are description logics (DLs)?
  - logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
  - první jazyky vznikly jako snaha o formalizaci sémantických sítí a rámců. První implementace v 80's – systémy KL-ONE, KAON, Classic .

Technologies sketched so far aren't enough ?

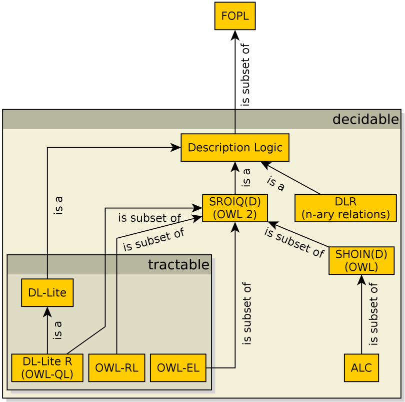
- Semantic networks and Frames
  - Lack well defined (declarative) semantics
  - What is the semantiics of a "slot" in a frame (relation in semantic networks) ? The slot **must/might** be filled **once/multiple times** ?
- Conceptual graphs are beyond FOPL (thus undecidable).
- What are description logics (DLs)?
  - logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
  - první jazyky vznikly jako snaha o formalizaci sémantických sítí a rámců. První implementace v 80's – systémy KL-ONE, KAON, Classic .

Technologies sketched so far aren't enough ?

- Semantic networks and Frames
  - Lack well defined (declarative) semantics
  - What is the semantiics of a "slot" in a frame (relation in semantic networks) ? The slot **must/might** be filled **once/multiple times** ?
- Conceptual graphs are beyond FOPL (thus undecidable).
- What are description logics (DLs)?
  - logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
  - první jazyky vznikly jako snaha o formalizaci sémantických sítí a rámců. První implementace v 80's – systémy KL-ONE, KAON, Classic .

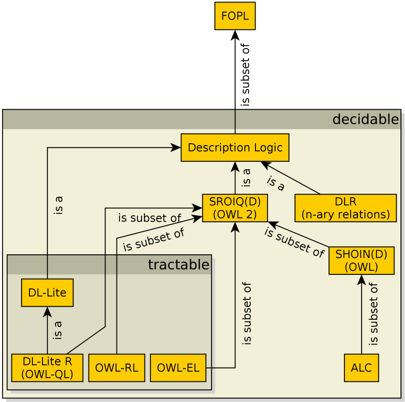
What are Description Logics ?

- family of logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
- first languages emerged as an experiment of giving formal semantics to semantic networks and frames. First implementations in 80's – KL-ONE, KAON, Classic.
- 90's $\mathcal{ALC}$
- 2004 $\mathcal{SHOIN}(\mathcal{D})$ – OWL
- 2009 $\mathcal{SROIQ}(\mathcal{D})$ – OWL 2



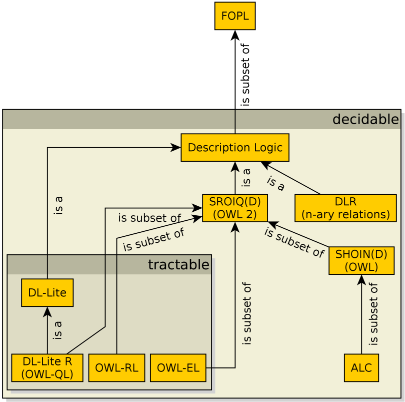
What are Description Logics ?

- family of logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
- first languages emerged as an experiment of giving formal semantics to semantic networks and frames. First implementations in 80's – KL-ONE, KAON, Classic.
- 90's $\mathcal{ALC}$
- 2004 $\mathcal{SHOIN}(\mathcal{D})$ – OWL
- 2009 $\mathcal{SROIQ}(\mathcal{D})$ – OWL 2



What are Description Logics ?

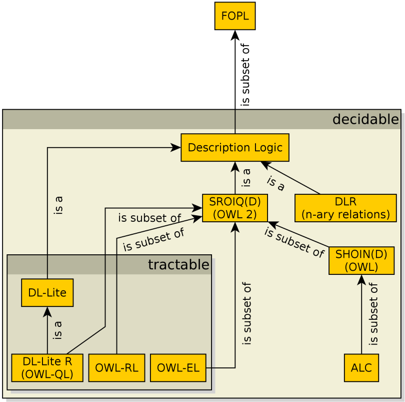
- family of logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
- first languages emerged as an experiment of giving formal semantics to semantic networks and frames. First implementations in 80's – KL-ONE, KAON, Classic.
- 90's $\mathcal{ALC}$
- 2004 $\mathcal{SHOIN}(\mathcal{D})$ – OWL
- 2009 $\mathcal{SROIQ}(\mathcal{D})$ – OWL 2



laboratory Gerstner

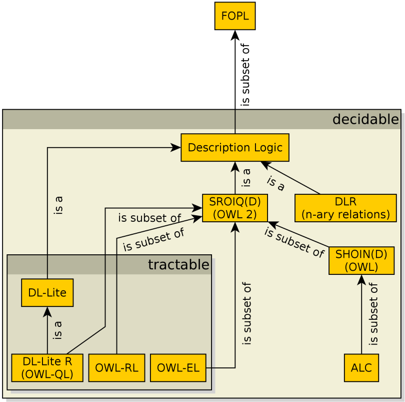
What are Description Logics ?

- family of logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
- first languages emerged as an experiment of giving formal semantics to semantic networks and frames. First implementations in 80's – KL-ONE, KAON, Classic.
- 90's $\mathcal{ALC}$
- 2004 $\mathcal{SHOIN}(\mathcal{D})$ – OWL
- 2009 $\mathcal{SROIQ}(\mathcal{D})$ – OWL 2



What are Description Logics ?

- family of logic-based languages for modeling *terminological knowledge*, *incomplete knowledge*. Almost exclusively, DLs are decidable subsets of FOPL.
- first languages emerged as an experiment of giving formal semantics to semantic networks and frames. First implementations in 80's – KL-ONE, KAON, Classic.
- 90's $\mathcal{ALC}$
- 2004 $\mathcal{SHOIN}(\mathcal{D})$ – OWL
- 2009 $\mathcal{SROIQ}(\mathcal{D})$ – OWL 2



laboratory Gerstner

$\mathcal{ALC}$ Language

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

laboratory Gerstner

Concepts and Roles

- Basic building blocks of DLs are :

  (atomic) concepts - representing (named) *unary predicates* / classes, e.g. *Parent*, or $Person \sqcap \exists hasChild \cdot Person$.

  (atomic) roles - represent (named) *binary predicates* / relations, e.g. *hasChild*

  individuals - represent ground terms / individuals, e.g. *JOHN*

- Theory $\mathcal{K}$ (in OWL refered as Ontology) of DLs consists of a

  TBOX $\mathcal{T}$ - representing axioms generally valid in the domain, e.g. $\mathcal{T} = \{Man \sqsubseteq Person\}$

  ABOX $\mathcal{A}$ - representing a particular relational structure (data), e.g. $\mathcal{A} = \{Man(JOHN)\}$

- DLs differ in their expressive power (concept/role constructors, axiom types).

Semantics, Interpretation

- as $\mathcal{ALC}$ is a subset of FOPL, let's define semantics analogously (and restrict interpretation function where applicable):
- **Interpretation** is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is an interpretation domain and $\cdot^{\mathcal{I}}$ is an interpretation function.
- Having *atomic* concept A, *atomic* role R and individual a, then

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

Semantics, Interpretation

- as $\mathcal{ALC}$ is a subset of FOPL, let's define semantics analogously (and restrict interpretation function where applicable):
- **Interpretation** is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is an interpretation domain and $\cdot^{\mathcal{I}}$ is an interpretation function.
- Having *atomic* concept A, *atomic* role R and individual a, then

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

Semantics, Interpretation

- as $\mathcal{ALC}$ is a subset of FOPL, let's define semantics analogously (and restrict interpretation function where applicable):
- **Interpretation** is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is an interpretation domain and $\cdot^{\mathcal{I}}$ is an interpretation function.
- Having *atomic* concept A, *atomic* role R and individual a, then

$$\begin{aligned} A^{\mathcal{I}} &\subseteq \Delta^{\mathcal{I}} \\ R^{\mathcal{I}} &\subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \\ a^{\mathcal{I}} &\in \Delta^{\mathcal{I}} \end{aligned}$$

$\mathcal{ALC}$ (= attributive language with complements)

Having concepts C, D, atomic concept A and atomic role R, then for interpretation $\mathcal{I}$:

concept	*concept*$^{\mathcal{I}}$	*description*
$\top$	$\Delta^{\mathcal{I}}$	(universal concept)
$\bot$	$\emptyset$	(unsatisfiable concept)
$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	(negation)
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	(intersection)
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	(union)
$\forall R \cdot C$	$\{a \mid \forall b\,((a,b) \in R^{\mathcal{I}} \Rightarrow b \in C^{\mathcal{I}})\}$	(universal restriction)
$\exists R \cdot C$	$\{a \mid \exists b\,((a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}})\}$	(existential restriction)

TBOX

axiom	$\mathcal{I} \models$ axiom iff	*description*
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$	(inclusion)
$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$	(equivalence)

ABOX (UNA = unique name assumption[3])

axiom	$\mathcal{I} \models$ axiom iff	*description*
$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$	(concept assertion)
$R(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$	(role assertion)

[3] two different individuals denote two different domain elements

$\mathcal{ALC}$ (= attributive language with complements)

Having concepts C, D, atomic concept A and atomic role R, then for interpretation $\mathcal{I}$:

concept	*concept*$^{\mathcal{I}}$	*description*
$\top$	$\Delta^{\mathcal{I}}$	(universal concept)
$\bot$	$\emptyset$	(unsatisfiable concept)
$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	(negation)
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	(intersection)
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	(union)
$\forall R \cdot C$	$\{a \mid \forall b\,((a,b) \in R^{\mathcal{I}} \Rightarrow b \in C^{\mathcal{I}})\}$	(universal restriction)
$\exists R \cdot C$	$\{a \mid \exists b\,((a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}})\}$	(existential restriction)

TBOX

axiom	$\mathcal{I} \models$ axiom iff	*description*
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$	(inclusion)
$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$	(equivalence)

ABOX (UNA = unique name assumption[3])

axiom	$\mathcal{I} \models$ axiom iff	*description*
$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$	(concept assertion)
$R(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$	(role assertion)

[3] two different individuals denote two different domain elements

$\mathcal{ALC}$ (= attributive language with complements)

Having concepts C, D, atomic concept A and atomic role R, then for interpretation $\mathcal{I}$:

concept	*concept*$^{\mathcal{I}}$	*description*
$\top$	$\Delta^{\mathcal{I}}$	(universal concept)
$\bot$	$\emptyset$	(unsatisfiable concept)
$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	(negation)
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	(intersection)
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$	(union)
$\forall R \cdot C$	$\{a \mid \forall b\,((a,b) \in R^{\mathcal{I}} \Rightarrow b \in C^{\mathcal{I}})\}$	(universal restriction)
$\exists R \cdot C$	$\{a \mid \exists b\,((a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}})\}$	(existential restriction)

TBOX

axiom	$\mathcal{I} \models$ axiom iff	*description*
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$	(inclusion)
$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$	(equivalence)

ABOX (UNA = unique name assumption[3])

axiom	$\mathcal{I} \models$ axiom iff	*description*
$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$	(concept assertion)
$R(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$	(role assertion)

[3] two different individuals denote two different domain elements

Logical Consequence

For an arbitrary set S of axioms (resp. theory $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $S = \mathcal{T} \cup \mathcal{A}$), then

- $\mathcal{I} \models S$ if $\mathcal{I} \models \alpha$ for all $\alpha \in S$ ($\mathcal{I}$ is a model of S, resp. $\mathcal{K}$)
- $S \models \beta$ if $\mathcal{I} \models \beta$ whenever $\mathcal{I} \models S$ (β is a logical consequence of S, resp. $\mathcal{K}$)
- S is consistent, if S has at least one model

laboratory Gerstner

Logical Consequence

For an arbitrary set S of axioms (resp. theory $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $S = \mathcal{T} \cup \mathcal{A}$), then

- $\mathcal{I} \models S$ if $\mathcal{I} \models \alpha$ for all $\alpha \in S$ ($\mathcal{I}$ is a model of S, resp. $\mathcal{K}$)
- $S \models \beta$ if $\mathcal{I} \models \beta$ whenever $\mathcal{I} \models S$ (β is a logical consequence of S, resp. $\mathcal{K}$)
- S is consistent, if S has at least one model

laboratory Gerstner

Logical Consequence

For an arbitrary set S of axioms (resp. theory $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $S = \mathcal{T} \cup \mathcal{A}$), then

- $\mathcal{I} \models S$ if $\mathcal{I} \models \alpha$ for all $\alpha \in S$ ($\mathcal{I}$ is a model of S, resp. $\mathcal{K}$)
- $S \models \beta$ if $\mathcal{I} \models \beta$ whenever $\mathcal{I} \models S$ (β is a logical consequence of S, resp. $\mathcal{K}$)
- S is consistent, if S has at least one model

laboratory Gerstner

$\mathcal{ALC}$ – Example

Example

Consider an information system for genealogical data. Information integration from various sources is crucial – databases, information systems with *different data models*. As an integration layer, let's use a description logic theory. Let's have atomic concepts *Person*, *Man*, *GrandParent* and atomic role *hasChild*.

- How to express a set of persons that have just men as their descendants, if any ?
  - $Person \sqcap \forall hasChild \cdot Man$
- How to define concept *GrandParent* ?
  - $GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top$
- How does the previous axiom look like in FOPL ?

$$\forall x \left(GrandParent(x) \equiv \left(Person(x) \wedge \exists y \left(hasChild(x, y) \wedge \exists z \left(hasChild(y, z) \right) \right) \right) \right)$$

$\mathcal{ALC}$ – Example

Example

Consider an information system for genealogical data. Information integration from various sources is crucial – databases, information systems with *different data models*. As an integration layer, let's use a description logic theory. Let's have atomic concepts *Person*, *Man*, *GrandParent* and atomic role *hasChild*.

- How to express a set of persons that have just men as their descendants, if any ?
  - $Person \sqcap \forall hasChild \cdot Man$
- How to define concept *GrandParent* ?
  - $GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top$
- How does the previous axiom look like in FOPL ?

$$\forall x \left(GrandParent(x) \equiv \left(Person(x) \wedge \exists y \left(hasChild(x, y) \wedge \exists z \left(hasChild(y, z) \right) \right) \right) \right)$$

$\mathcal{ALC}$ – Example

Example

Consider an information system for genealogical data. Information integration from various sources is crucial – databases, information systems with *different data models*. As an integration layer, let's use a description logic theory. Let's have atomic concepts *Person*, *Man*, *GrandParent* and atomic role *hasChild*.

- How to express a set of persons that have just men as their descendants, if any ?
  - $Person \sqcap \forall hasChild \cdot Man$
- How to define concept *GrandParent* ?
  - $GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top$
- How does the previous axiom look like in FOPL ?

$$\forall x\,(GrandParent(x) \equiv (Person(x) \wedge \exists y\,(hasChild(x, y) \wedge \exists z\,(hasChild(y, z)))))$$

Example

Consider an information system for genealogical data. Information integration from various sources is crucial – databases, information systems with *different data models*. As an integration layer, let's use a description logic theory. Let's have atomic concepts *Person*, *Man*, *GrandParent* and atomic role *hasChild*.

- How to express a set of persons that have just men as their descendants, if any ?
  - $Person \sqcap \forall hasChild \cdot Man$
- How to define concept *GrandParent* ?
  - $GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top$
- How does the previous axiom look like in FOPL ?

$$\forall x \left(GrandParent(x) \equiv \left(Person(x) \wedge \exists y \left(hasChild(x, y) \wedge \exists z \left(hasChild(y, z) \right) \right) \right) \right)$$

$\mathcal{ALC}$ – Example

Example

Consider an information system for genealogical data. Information integration from various sources is crucial – databases, information systems with *different data models*. As an integration layer, let's use a description logic theory. Let's have atomic concepts *Person*, *Man*, *GrandParent* and atomic role *hasChild*.

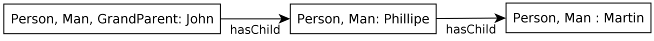
- How to express a set of persons that have just men as their descendants, if any ?
  - $Person \sqcap \forall hasChild \cdot Man$
- How to define concept *GrandParent* ?
  - $GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top$
- How does the previous axiom look like in FOPL ?

$$\forall x\,(GrandParent(x) \equiv (Person(x) \wedge \exists y\,(hasChild(x,y) \\ \wedge \exists z\,(hasChild(y,z)))))$$

Interpretation – Example

Example

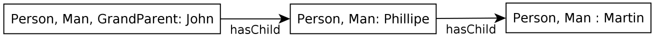
- Consider an ontology $\mathcal{K}_1 = (\{GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top\}, \{GrandParent(JOHN)\})$, modelem $\mathcal{K}_1$ může být např. interpretace $\mathcal{I}_1$:
  - $\Delta^{\mathcal{I}_1} = Man^{\mathcal{I}_1} = Person^{\mathcal{I}_1} = \{John, Phillipe, Martin\}$
  - $hasChild^{\mathcal{I}_1} = \{(John, Phillipe), (Phillipe, Martin)\}$
  - $GrandParent^{\mathcal{I}_1} = \{John\}$
  - $JOHN^{\mathcal{I}_1} = \{John\}$
- this model is finite and has the form of a tree with the root in the node *Jan* :



Interpretation – Example

Example

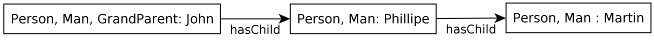
- Consider an ontology $\mathcal{K}_1 = (\{GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top\}, \{GrandParent(JOHN)\})$, modelem $\mathcal{K}_1$ může být např. interpretace $\mathcal{I}_1$:
  - $\Delta^{\mathcal{I}_1} = Man^{\mathcal{I}_1} = Person^{\mathcal{I}_1} = \{John, Phillipe, Martin\}$
  - $hasChild^{\mathcal{I}_1} = \{(John, Phillipe), (Phillipe, Martin)\}$
  - $GrandParent^{\mathcal{I}_1} = \{John\}$
  - $JOHN^{\mathcal{I}_1} = \{John\}$
- this model is finite and has the form of a tree with the root in the node *Jan* :



Interpretation – Example

Example

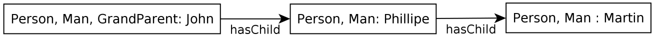
- Consider an ontology $\mathcal{K}_1 = (\{GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top\}, \{GrandParent(JOHN)\})$, modelem $\mathcal{K}_1$ může být např. interpretace $\mathcal{I}_1$:
  - $\Delta^{\mathcal{I}_1} = Man^{\mathcal{I}_1} = Person^{\mathcal{I}_1} = \{John, Phillipe, Martin\}$
  - $hasChild^{\mathcal{I}_1} = \{(John, Phillipe), (Phillipe, Martin)\}$
  - $GrandParent^{\mathcal{I}_1} = \{John\}$
  - $JOHN^{\mathcal{I}_1} = \{John\}$
- this model is finite and has the form of a tree with the root in the node *Jan* :



Interpretation – Example

Example

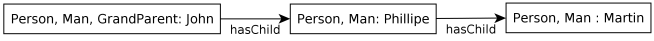
- Consider an ontology $\mathcal{K}_1 = (\{GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top\}, \{GrandParent(JOHN)\})$, modelem $\mathcal{K}_1$ může být např. interpretace $\mathcal{I}_1$:
  - $\Delta^{\mathcal{I}_1} = Man^{\mathcal{I}_1} = Person^{\mathcal{I}_1} = \{John, Phillipe, Martin\}$
  - $hasChild^{\mathcal{I}_1} = \{(John, Phillipe), (Phillipe, Martin)\}$
  - $GrandParent^{\mathcal{I}_1} = \{John\}$
  - $JOHN^{\mathcal{I}_1} = \{John\}$
- this model is finite and has the form of a tree with the root in the node *Jan* :



Interpretation – Example

Example

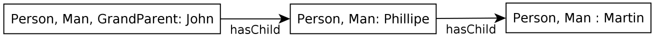
- Consider an ontology $\mathcal{K}_1 = (\{GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top\}, \{GrandParent(JOHN)\})$, modelem $\mathcal{K}_1$ může být např. interpretace $\mathcal{I}_1$:
  - $\Delta^{\mathcal{I}_1} = Man^{\mathcal{I}_1} = Person^{\mathcal{I}_1} = \{John, Phillipe, Martin\}$
  - $hasChild^{\mathcal{I}_1} = \{(John, Phillipe), (Phillipe, Martin)\}$
  - $GrandParent^{\mathcal{I}_1} = \{John\}$
  - $JOHN^{\mathcal{I}_1} = \{John\}$
- this model is finite and has the form of a tree with the root in the node *Jan* :



Interpretation – Example

Example

- Consider an ontology $\mathcal{K}_1 = (\{GrandParent \equiv Person \sqcap \exists hasChild \cdot \exists hasChild \cdot \top\}, \{GrandParent(JOHN)\})$, modelem $\mathcal{K}_1$ může být např. interpretace $\mathcal{I}_1$:
  - $\Delta^{\mathcal{I}_1} = Man^{\mathcal{I}_1} = Person^{\mathcal{I}_1} = \{John, Phillipe, Martin\}$
  - $hasChild^{\mathcal{I}_1} = \{(John, Phillipe), (Phillipe, Martin)\}$
  - $GrandParent^{\mathcal{I}_1} = \{John\}$
  - $JOHN^{\mathcal{I}_1} = \{John\}$
- this model is finite and has the form of a tree with the root in the node *Jan* :



Shape of DL Models

The last example revealed several important properties of DL models:

TMP (tree model property), if every satisfiable concept[4] C of the language has a model in the shape of a *rooted tree*.

FMP (finite model property), if every consistent theory $\mathcal{K}$ of the language has a *finite model*.

Both properties represent important characteristics of a DL that directly influence inferencing (see next lecture).

In particular (generalized) TMP is a characteristics that is shared by most DLs and significantly reduces their computational complexity.

[4] Concept is satisfiable, if at least one model interprets it as a non-empty set

Shape of DL Models

The last example revealed several important properties of DL models:

TMP (tree model property), if every satisfiable concept[4] C of the language has a model in the shape of a *rooted tree*.

FMP (finite model property), if every consistent theory $\mathcal{K}$ of the language has a *finite model*.

Both properties represent important characteristics of a DL that directly influence inferencing (see next lecture).

In particular (generalized) TMP is a characteristics that is shared by most DLs and significantly reduces their computational complexity.

[4] Concept is satisfiable, if at least one model interprets it as a non-empty set

Shape of DL Models

The last example revealed several important properties of DL models:

TMP (tree model property), if every satisfiable concept[4] C of the language has a model in the shape of a *rooted tree*.

FMP (finite model property), if every consistent theory $\mathcal{K}$ of the language has a *finite model*.

Both properties represent important characteristics of a DL that directly influence inferencing (see next lecture).

In particular (generalized) TMP is a characteristics that is shared by most DLs and significantly reduces their computational complexity.

[4] Concept is satisfiable, if at least one model interprets it as a non-empty set

Shape of DL Models

The last example revealed several important properties of DL models:

TMP (tree model property), if every satisfiable concept[4] C of the language has a model in the shape of a *rooted tree*.

FMP (finite model property), if every consistent theory $\mathcal{K}$ of the language has a *finite model*.

Both properties represent important characteristics of a DL that directly influence inferencing (see next lecture).

In particular (generalized) TMP is a characteristics that is shared by most DLs and significantly reduces their computational complexity.

[4] Concept is satisfiable, if at least one model interprets it as a non-empty set

Shape of DL Models

The last example revealed several important properties of DL models:

TMP (tree model property), if every satisfiable concept[4] C of the language has a model in the shape of a *rooted tree*.

FMP (finite model property), if every consistent theory $\mathcal{K}$ of the language has a *finite model*.

Both properties represent important characteristics of a DL that directly influence inferencing (see next lecture).

In particular (generalized) TMP is a characteristics that is shared by most DLs and significantly reduces their computational complexity.

[4] Concept is satisfiable, if at least one model interprets it as a non-empty set

Example

Example

primitive concept
defined concept

$$
\begin{aligned}
\mathit{Woman} &\equiv \mathit{Person} \sqcap \mathit{Female} \\
\mathit{Man} &\equiv \mathit{Person} \sqcap \neg \mathit{Woman} \\
\mathit{Mother} &\equiv \mathit{Woman} \sqcap \exists \mathit{hasChild} \cdot \mathit{Person} \\
\mathit{Father} &\equiv \mathit{Man} \sqcap \exists \mathit{hasChild} \cdot \mathit{Person} \\
\mathit{Parent} &\equiv \mathit{Father} \sqcup \mathit{Mother} \\
\mathit{Grandmother} &\equiv \mathit{Mother} \sqcap \exists \mathit{hasChild} \cdot \mathit{Parent} \\
\mathit{MotherWithoutDaughter} &\equiv \mathit{Mother} \sqcap \forall \mathit{hasChild} \cdot \neg \mathit{Woman} \\
\mathit{Wife} &\equiv \mathit{Woman} \sqcap \exists \mathit{hasHusband} \cdot \mathit{Man}
\end{aligned}
$$

Example – CWA × OWA

Example

ABOX

$hasChild(JOCASTA, OEDIPUS)$ $\quad$ $hasChild(JOCASTA, POLYNEIKES)$
$hasChild(OEDIPUS, POLYNEIKES)$ $\quad$ $hasChild(POLYNEIKES, THERSANDROS)$
$Patricide(OEDIPUS)$ $\quad$ $\neg Patricide(THERSANDROS)$

Edges represent role assertions of *hasChild*; colors distinguish concepts instances – *Patricide* a $\neg$*Patricide*

Q1 $(\exists hasChild \cdot (Patricide \sqcap \exists hasChild \cdot \neg Patricide))(JOCASTA)$,

Q2 Find individuals x such that $\mathcal{K} \models C(x)$, where C is

$$\neg Patricide \sqcap \exists hasChild^{-} \cdot (Patricide \sqcap \exists hasChild^{-}) \cdot \{JOCASTA\}$$

What is the difference, when considering CWA ?

Example

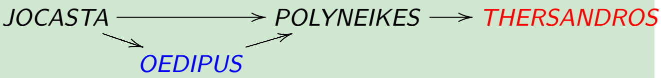
ABOX

$hasChild(JOCASTA, OEDIPUS)$ $\quad$ $hasChild(JOCASTA, POLYNEIKES)$

$hasChild(OEDIPUS, POLYNEIKES)$ $\quad$ $hasChild(POLYNEIKES, THERSANDROS)$

$Patricide(OEDIPUS)$ $\quad$ $\neg Patricide(THERSANDROS)$

Edges represent role assertions of *hasChild*; colors distinguish concepts instances – *Patricide* a $\neg$*Patricide*



Q1 $(\exists hasChild \cdot (Patricide \sqcap \exists hasChild \cdot \neg Patricide))(JOCASTA)$,

$$JOCASTA \longrightarrow \bullet \longrightarrow \bullet$$

Q2 Find individuals x such that $\mathcal{K} \models C(x)$, where C is

$$\neg Patricide \sqcap \exists hasChild^{-} \cdot (Patricide \sqcap \exists hasChild^{-}) \cdot \{JOCASTA\}$$

What is the difference, when considering CWA ?

$$JOCASTA \longrightarrow \bullet \longrightarrow \times$$

Example – CWA × OWA

Example

ABOX

$hasChild(JOCASTA, OEDIPUS)$ $hasChild(JOCASTA, POLYNEIKES)$
$hasChild(OEDIPUS, POLYNEIKES)$ $hasChild(POLYNEIKES, THERSANDROS)$
$Patricide(OEDIPUS)$ $\neg Patricide(THERSANDROS)$

Edges represent role assertions of *hasChild*; colors distinguish concepts instances – *Patricide* a $\neg$*Patricide*

$JOCASTA \longrightarrow POLYNEIKES \longrightarrow THERSANDROS$

$JOCASTA \longrightarrow OEDIPUS \longrightarrow POLYNEIKES$

Q1 $(\exists hasChild \cdot (Patricide \sqcap \exists hasChild \cdot \neg Patricide))(JOCASTA)$,

$JOCASTA \longrightarrow \bullet \longrightarrow \bullet$

Q2 Find individuals x such that $\mathcal{K} \models C(x)$, where C is

$\neg Patricide \sqcap \exists hasChild^{-} \cdot (Patricide \sqcap \exists hasChild^{-}) \cdot \{JOCASTA\}$

What is the difference, when considering CWA ?

$JOCASTA \longrightarrow \bullet \longrightarrow \times$

Example – CWA × OWA

Example

ABOX

$hasChild(JOCASTA, OEDIPUS)$ $hasChild(JOCASTA, POLYNEIKES)$
$hasChild(OEDIPUS, POLYNEIKES)$ $hasChild(POLYNEIKES, THERSANDROS)$
$Patricide(OEDIPUS)$ $\neg Patricide(THERSANDROS)$

Edges represent role assertions of *hasChild*; colors distinguish concepts instances – *Patricide* a $\neg$*Patricide*

$$JOCASTA \longrightarrow POLYNEIKES \longrightarrow THERSANDROS$$

$$JOCASTA \searrow OEDIPUS \nearrow POLYNEIKES$$

Q1 $(\exists hasChild \cdot (Patricide \sqcap \exists hasChild \cdot \neg Patricide))(JOCASTA)$,

$$JOCASTA \longrightarrow \bullet \longrightarrow \bullet$$

Q2 Find individuals x such that $\mathcal{K} \models C(x)$, where C is

$$\neg Patricide \sqcap \exists hasChild^{-} \cdot (Patricide \sqcap \exists hasChild^{-}) \cdot \{JOCASTA\}$$

What is the difference, when considering CWA ?

$$JOCASTA \longrightarrow \bullet \longrightarrow x$$