Classical Algorithms for Reasoning and Explanation in Description Logics

Birte Glimm and Yevgeny Kazakov

University of Ulm

| 20.09.2019

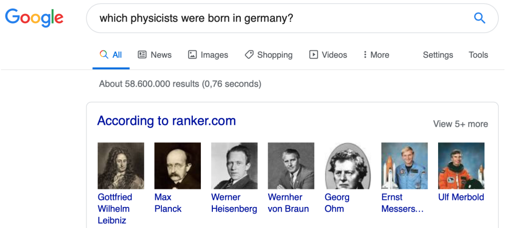
German Physicists

Which physicists were born in Germany?

German Physicists

Which physicists were born in Germany?

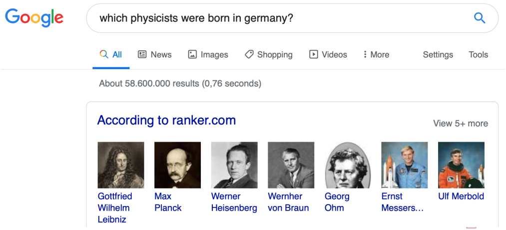
- Google can answer this question astonishingly well!



German Physicists

Which physicists were born in Germany?

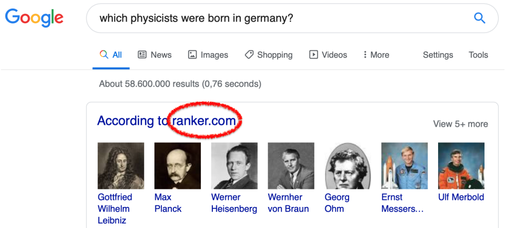
- Google can answer this question astonishingly well!
  - although it misses Angela Merkel



German Physicists

Which physicists were born in Germany?

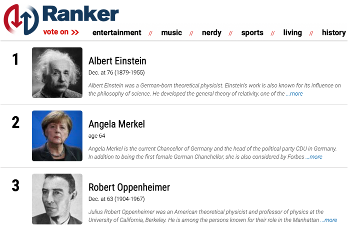
- Google can answer this question astonishingly well!
  - although it misses Angela Merkel
- How does it know?



German Physicists

Which physicists were born in Germany?

- Google can answer this question astonishingly well!
  - although it misses Angela Merkel
- How does it know?
  - there is a Web page for that!



German Physicists

Which physicists were born in Germany?

- Google can answer this question astonishingly well!
  - although it misses Angela Merkel
- How does it know?
  - there is a Web page for that!
- In general, Web search is based on matching keywords

Science and technology in Germany - Wikipedia

https://en.wikipedia.org › wiki › Science_and_technology_in_Germany ▾

German science **have been** very significant and research and development efforts form an ... They **were** preceded by such key **physicists** as Hermann von Helmholtz, Joseph von Fraunhofer, and Gabriel Daniel Fahrenheit, among others . Wilhelm ... Wladimir Köppen (1846–1940) **was** an eclectic Russian-**born** botanist and ...

German Physicists

Which physicists were born in Germany?

- Google can answer this question astonishingly well!
  - although it misses Angela Merkel
- How does it know?
  - there is a Web page for that!
- In general, Web search is based on matching keywords
- There is no guarantee that the answer will be found
  - even if there is a Web page with the answer

German Physicists

Which physicists were born in Germany?

- Google can answer this question astonishingly well!
  - although it misses Angela Merkel
- How does it know?
  - there is a Web page for that!
- In general, Web search is based on matching keywords
- There is no guarantee that the answer will be found
  - even if there is a Web page with the answer
- In some applications wrong/missed results cannot be tolerated
  - medicine, banking, autonomous driving,. . .

Description Logics

- Description Logics (DLs) are formal languages designed for knowledge representation

Description Logics

- Description Logics (DLs) are formal languages designed for knowledge representation
- The basic principle is similar to Wikipedia:
  - knowledge is described and curated in a single place (ontology) by domain experts

Description Logics

- Description Logics (DLs) are formal languages designed for knowledge representation
- The basic principle is similar to Wikipedia:
  - knowledge is described and curated in a single place (ontology) by domain experts
- But the knowledge description is formal:
  - use formulas instead of text
  - each formula represents a piece of information
  - like in mathematics, it is well-defined what each formula means

Description Logics

- Description Logics (DLs) are formal languages designed for knowledge representation
- The basic principle is similar to Wikipedia:
  - knowledge is described and curated in a single place (ontology) by domain experts
- But the knowledge description is formal:
  - use formulas instead of text
  - each formula represents a piece of information
  - like in mathematics, it is well-defined what each formula means
- Main advantage: an answer can be obtained by combinding several sources of information (formulas)

Description Logics

- Example:

F1 = "Albert Einstein was a physicist"
F2 = "Albert Einstein was born in Ulm"
F3 = "Ulm is a city in Germany"

⇒ "Albert Einstein was a German Physicists"

DLs @ RW

- Reasoning Web summer school hosted many courses on DLs:
  - DL introduction (@RW 2007, 2009, 2011, 2013)
  - lightweight DLs (@RW 2010)
  - query answering (@RW 2012, 2014, 2015)
  - non-standard reasoning (@RW 2015, 2016)

DLs @ RW

- Reasoning Web summer school hosted many courses on DLs:
  - DL introduction (@RW 2007, 2009, 2011, 2013)
  - lightweight DLs (@RW 2010)
  - query answering (@RW 2012, 2014, 2015)
  - non-standard reasoning (@RW 2015, 2016)
- In this course: a detailed account of core DL algorithms:
  - reasoning: tableau-based procedures
  - explantion: axiom-pinpointing methods

DLs @ RW

- Reasoning Web summer school hosted many courses on DLs:
  - DL introduction (@RW 2007, 2009, 2011, 2013)
  - lightweight DLs (@RW 2010)
  - query answering (@RW 2012, 2014, 2015)
  - non-standard reasoning (@RW 2015, 2016)
- In this course: a detailed account of core DL algorithms:
  - reasoning: tableau-based procedures
  - explantion: axiom-pinpointing methods
- Main focus: correctness, complexity, optimizations

Outline

Outline

Vocabulary of $\mathcal{ALC}$

- The vocabulary of DL $\mathcal{ALC}$ consists of:
  - Concept names (atomic concepts): A, B,...
  - Role names (atomic roles): R, S, H,...
  - Individual names (individuals): a, b, c,...
  - Logical symbols: $\top$, $\bot$, $\neg$, $\sqcap$, $\sqcup$, $\forall$, $\exists$.

Vocabulary of $\mathcal{ALC}$

- The vocabulary of DL $\mathcal{ALC}$ consists of:
  - Concept names (atomic concepts): A, B,...
  - Role names (atomic roles): R, S, H,...
  - Individual names (individuals): a, b, c,...
  - Logical symbols: $\top$, $\bot$, $\neg$, $\sqcap$, $\sqcup$, $\forall$, $\exists$.
- Concepts represent sets of things:
  - *Human* – set of all human beings
  - *Male* – the set of all male (not necessarily human) beings
  - *Country* – the set of all countries

Vocabulary of $\mathcal{ALC}$

- The vocabulary of DL $\mathcal{ALC}$ consists of:
  - Concept names (atomic concepts): A, B,...
  - Role names (atomic roles): R, S, H,...
  - Individual names (individuals): a, b, c,...
  - Logical symbols: $\top$, $\bot$, $\neg$, $\sqcap$, $\sqcup$, $\forall$, $\exists$.
- Concepts represent sets of things:
  - *Human* – set of all human beings
  - *Male* – the set of all male (not necessarily human) beings
  - *Country* – the set of all countries
- Roles represent relations between things:
  - *hasChild* – holds between parents and their children
  - *hasLocation* – holds between objects and their locations

Vocabulary of $\mathcal{ALC}$

- The vocabulary of DL $\mathcal{ALC}$ consists of:
  - Concept names (atomic concepts): A, B,...
  - Role names (atomic roles): R, S, H,...
  - Individual names (individuals): a, b, c,...
  - Logical symbols: $\top$, $\bot$, $\neg$, $\sqcap$, $\sqcup$, $\forall$, $\exists$.
- Concepts represent sets of things:
  - *Human* – set of all human beings
  - *Male* – the set of all male (not necessarily human) beings
  - *Country* – the set of all countries
- Roles represent relations between things:
  - *hasChild* – holds between parents and their children
  - *hasLocation* – holds between objects and their locations
- Individuals represent concrete (unique) objects:
  - *germany* – the country of Germany
  - *john* – the person John

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:
  - $Male \sqcap Human$ – the set of male humans

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:
  - $\mathit{Male} \sqcap \mathit{Human}$ – the set of male humans
  - $\mathit{Male} \sqcup \mathit{Female}$ – the union of male and female beings

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:
  - $Male \sqcap Human$ – the set of male humans
  - $Male \sqcup Female$ – the union of male and female beings
  - $\neg Male$ – the set of non-male beings

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:
  - $\mathit{Male} \sqcap \mathit{Human}$ – the set of male humans
  - $\mathit{Male} \sqcup \mathit{Female}$ – the union of male and female beings
  - $\neg \mathit{Male}$ – the set of non-male beings
  - $\neg \mathit{Male} \sqcap \mathit{Human}$ – non-male humans

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:
  - $\exists hasChild.Male$ – all beings that have a male child

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:
  - $\exists hasChild.Male$ – all beings that have a male child
  - $\forall hasChild.Female$ – all beings that have only female children

Complex Concepts of $\mathcal{ALC}$

- Complex concepts are built using concept constructors:
  - $\top$ (top) is a concept that represents all objects in the world
  - $\bot$ (bottom) is a concept that has no member objects
  - $C \sqcap D$ (conjunction) are the common objects of C and D
  - $C \sqcup D$ (disjunction) is the union of objects in C and D
  - $\neg C$ (negation) are all objects that are not in C
  - $\exists R.C$ (existential restriction) are all objects that are related via role R to some object in C
  - $\forall R.C$ (universal restriction) are all objects that are related via role R to only objects in C
- Examples:
  - $\exists hasChild.Male$ – all beings that have a male child
  - $\forall hasChild.Female$ – all beings that have only female children
  - $Male \sqcap \forall hasChild.\neg Male$ – all male beings all of whose children are not male

Axioms of $\mathcal{ALC}$

- Description Logic axioms postulate facts about concepts, roles, or individuals:
  - $C \sqsubseteq D$ (concept inclusion) states that every member of C is also a member of D
  - $C \equiv D$ (concept equivalence) states that C and D have the same members
  - $C(a)$ (concept assertion) states that the individual a is a member of C
  - $R(a, b)$ (role assertion) states that the individuals a and b are connected by the role R

Axioms of $\mathcal{ALC}$

- Description Logic axioms postulate facts about concepts, roles, or individuals:
  - $C \sqsubseteq D$ (concept inclusion) states that every member of C is also a member of D
  - $C \equiv D$ (concept equivalence) states that C and D have the same members
  - $C(a)$ (concept assertion) states that the individual a is a member of C
  - $R(a, b)$ (role assertion) states that the individuals a and b are connected by the role R
- Examples:

Axioms of $\mathcal{ALC}$

- Description Logic axioms postulate facts about concepts, roles, or individuals:
  - $C \sqsubseteq D$ (concept inclusion) states that every member of C is also a member of D
  - $C \equiv D$ (concept equivalence) states that C and D have the same members
  - $C(a)$ (concept assertion) states that the individual a is a member of C
  - $R(a, b)$ (role assertion) states that the individuals a and b are connected by the role R
- Examples:
  - $Human \sqsubseteq Dead \sqcup Alive$ – every human is either dead or alive

Axioms of $\mathcal{ALC}$

- Description Logic axioms postulate facts about concepts, roles, or individuals:
  - $C \sqsubseteq D$ (concept inclusion) states that every member of C is also a member of D
  - $C \equiv D$ (concept equivalence) states that C and D have the same members
  - $C(a)$ (concept assertion) states that the individual a is a member of C
  - $R(a, b)$ (role assertion) states that the individuals a and b are connected by the role R
- Examples:
  - $\mathit{Human} \sqsubseteq \mathit{Dead} \sqcup \mathit{Alive}$ – every human is either dead or alive
  - $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$ – parents are exactly those that have some child

Axioms of $\mathcal{ALC}$

- Description Logic axioms postulate facts about concepts, roles, or individuals:
  - $C \sqsubseteq D$ (concept inclusion) states that every member of C is also a member of D
  - $C \equiv D$ (concept equivalence) states that C and D have the same members
  - $C(a)$ (concept assertion) states that the individual a is a member of C
  - $R(a, b)$ (role assertion) states that the individuals a and b are connected by the role R
- Examples:
  - $Human \sqsubseteq Dead \sqcup Alive$ – every human is either dead or alive
  - $Parent \equiv \exists hasChild.\top$ – parents are exactly those that have some child
  - $Male(john)$ – john is male

Axioms of $\mathcal{ALC}$

- Description Logic axioms postulate facts about concepts, roles, or individuals:
  - $C \sqsubseteq D$ (concept inclusion) states that every member of C is also a member of D
  - $C \equiv D$ (concept equivalence) states that C and D have the same members
  - $C(a)$ (concept assertion) states that the individual a is a member of C
  - $R(a, b)$ (role assertion) states that the individuals a and b are connected by the role R
- Examples:
  - $Human \sqsubseteq Dead \sqcup Alive$ – every human is either dead or alive
  - $Parent \equiv \exists hasChild.\top$ – parents are exactly those that have some child
  - $Male(john)$ – john is male
  - $bornIn(einstein, ulm)$ – Albert Einstein was born in Ulm

$\mathcal{ALC}$ Knowledge Bases

- An $\mathcal{ALC}$ knowledge base (or ontology) is a finite set $\mathcal{O}$ of axioms.

$\mathcal{ALC}$ Knowledge Bases

- An $\mathcal{ALC}$ knowledge base (or ontology) is a finite set $\mathcal{O}$ of axioms.
- Typically consist of two parts:
  - TBox (terminological axioms): concept inclusions and equivalences
  - ABox (assertion axioms): concept and role assertions

$\mathcal{ALC}$ Knowledge Bases

- An $\mathcal{ALC}$ knowledge base (or ontology) is a finite set $\mathcal{O}$ of axioms.
- Typically consist of two parts:
  - TBox (terminological axioms): concept inclusions and equivalences
  - ABox (assertion axioms): concept and role assertions
- Example: take $\mathcal{O}$ consisting of three axioms:

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$

(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$

(ax3) $\mathit{hasChild}(\mathit{john}, \mathit{mary})$

$\mathcal{ALC}$ Knowledge Bases

- An $\mathcal{ALC}$ knowledge base (or ontology) is a finite set $\mathcal{O}$ of axioms.
- Typically consist of two parts:
  - TBox (terminological axioms): concept inclusions and equivalences
  - ABox (assertion axioms): concept and role assertions
- Example: take $\mathcal{O}$ consisting of three axioms:

(ax1) $Parent \equiv \exists hasChild.\top$
(ax2) $GrandParent \equiv \exists hasChild.Parent$
(ax3) $hasChild(john, mary)$

- Then its TBox $= \{(ax1), (ax2)\}$, its ABox $= \{(ax3)\}$

Outline

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations
- A DL interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations
- A DL interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
- $\Delta^{\mathcal{I}}$ (the domain of $\mathcal{I}$) is an arbitrary non-empty set, and

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations
- A DL interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
- $\Delta^{\mathcal{I}}$ (the domain of $\mathcal{I}$) is an arbitrary non-empty set, and
- $\cdot^{\mathcal{I}}$ (the interpretation function) is a mapping that assigns:

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations
- A DL interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
- $\Delta^{\mathcal{I}}$ (the domain of $\mathcal{I}$) is an arbitrary non-empty set, and
- $\cdot^{\mathcal{I}}$ (the interpretation function) is a mapping that assigns:
  - to every concept name A a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations
- A DL interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
- $\Delta^{\mathcal{I}}$ (the domain of $\mathcal{I}$) is an arbitrary non-empty set, and
- $\cdot^{\mathcal{I}}$ (the interpretation function) is a mapping that assigns:
  - to every concept name A a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
  - to every role name R a relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations
- A DL interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
- $\Delta^{\mathcal{I}}$ (the domain of $\mathcal{I}$) is an arbitrary non-empty set, and
- $\cdot^{\mathcal{I}}$ (the interpretation function) is a mapping that assigns:
  - to every concept name A a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
  - to every role name R a relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
  - to every individual a an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$

Interpretation

- $\mathcal{ALC}$ has a model-theoretic Semantics
  - the meaning of concepts and axioms is defined using interpretations
- A DL interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
- $\Delta^{\mathcal{I}}$ (the domain of $\mathcal{I}$) is an arbitrary non-empty set, and
- $\cdot^{\mathcal{I}}$ (the interpretation function) is a mapping that assigns:
  - to every concept name A a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
  - to every role name R a relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
  - to every individual a an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
- Example: define $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ as follows:
  - $\Delta^{\mathcal{I}} = \{a, b\}$
  - $Human^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$
  - $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$
  - $john^{\mathcal{I}} = a$, $mary^{\mathcal{I}} = b$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{Male}^{\mathcal{I}} = \{a\}$, $\mathit{Female}^{\mathcal{I}} = \{b\}$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \;\&\; y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y. \; (x, y) \in R^{\mathcal{I}} \;\Rightarrow\; y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \&\ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow\ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{Male}^{\mathcal{I}} = \{a\}$, $\mathit{Female}^{\mathcal{I}} = \{b\}$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$
  - $(\mathit{Male} \sqcap \mathit{Female})^{\mathcal{I}} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^\mathcal{I}$ can be recursively extended to complex concepts as follows:
  - $\top^\mathcal{I} = \Delta^\mathcal{I}$
  - $\bot^\mathcal{I} = \emptyset$
  - $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$
  - $(C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$
  - $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$
  - $(\exists R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \exists y : (x, y) \in R^\mathcal{I} \ \& \ y \in C^\mathcal{I}\}$
  - $(\forall R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \forall y. \ (x, y) \in R^\mathcal{I} \ \Rightarrow \ y \in C^\mathcal{I}\}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $Male^\mathcal{I} = \{a\}$, $Female^\mathcal{I} = \{b\}$, $hasChild^\mathcal{I} = \{\langle a, b\rangle\}$. Then:
  - $\top^\mathcal{I} = \{a, b\}$
  - $(Male \sqcap Female)^\mathcal{I} = \emptyset$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$
  - $(Male \sqcap Female)^{\mathcal{I}} = \emptyset$
  - $(Male \sqcup Female)^{\mathcal{I}} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$
  - $(Male \sqcap Female)^{\mathcal{I}} = \emptyset$
  - $(Male \sqcup Female)^{\mathcal{I}} = \{a, b\}$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \;\&\; y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \;\Rightarrow\; y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$
  - $(Male \sqcap Female)^{\mathcal{I}} = \emptyset$
  - $(Male \sqcup Female)^{\mathcal{I}} = \{a, b\}$
  - $(\neg Male)^{\mathcal{I}} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{Male}^{\mathcal{I}} = \{a\}$, $\mathit{Female}^{\mathcal{I}} = \{b\}$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$
  - $(\mathit{Male} \sqcap \mathit{Female})^{\mathcal{I}} = \emptyset$
  - $(\mathit{Male} \sqcup \mathit{Female})^{\mathcal{I}} = \{a, b\}$
  - $(\neg \mathit{Male})^{\mathcal{I}} = \{b\}$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$
  - $(Male \sqcap Female)^{\mathcal{I}} = \emptyset$
  - $(Male \sqcup Female)^{\mathcal{I}} = \{a, b\}$
  - $(\neg Male)^{\mathcal{I}} = \{b\}$
  - $(Male \sqcap \neg Female)^{\mathcal{I}} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \;\&\; y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \;\Rightarrow\; y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $\top^{\mathcal{I}} = \{a, b\}$
  - $(Male \sqcap Female)^{\mathcal{I}} = \emptyset$
  - $(Male \sqcup Female)^{\mathcal{I}} = \{a, b\}$
  - $(\neg Male)^{\mathcal{I}} = \{b\}$
  - $(Male \sqcap \neg Female)^{\mathcal{I}} = \{a\}$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \&\ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow\ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^{\mathcal{I}} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \;\&\; y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \;\Rightarrow\; y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^{\mathcal{I}} = \{a\}$

Interpretation of Complex Concepts

- The interpretation function $\cdot^\mathcal{I}$ can be recursively extended to complex concepts as follows:
  - $\top^\mathcal{I} = \Delta^\mathcal{I}$
  - $\bot^\mathcal{I} = \emptyset$
  - $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$
  - $(C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$
  - $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$
  - $(\exists R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \exists y : (x, y) \in R^\mathcal{I} \;\&\; y \in C^\mathcal{I}\}$
  - $(\forall R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \forall y.\ (x, y) \in R^\mathcal{I} \;\Rightarrow\; y \in C^\mathcal{I}\}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $Male^\mathcal{I} = \{a\}$, $Female^\mathcal{I} = \{b\}$, $hasChild^\mathcal{I} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^\mathcal{I} = \{a\}$
  - $(\forall hasChild.Female)^\mathcal{I} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^\mathcal{I}$ can be recursively extended to complex concepts as follows:
  - $\top^\mathcal{I} = \Delta^\mathcal{I}$
  - $\bot^\mathcal{I} = \emptyset$
  - $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$
  - $(C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$
  - $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$
  - $(\exists R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \exists y : (x, y) \in R^\mathcal{I} \ \&\ y \in C^\mathcal{I}\}$
  - $(\forall R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \forall y.\ (x, y) \in R^\mathcal{I} \ \Rightarrow\ y \in C^\mathcal{I}\}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $Male^\mathcal{I} = \{a\}$, $Female^\mathcal{I} = \{b\}$, $hasChild^\mathcal{I} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^\mathcal{I} = \{a\}$
  - $(\forall hasChild.Female)^\mathcal{I} = \{a, b\}$ (!!!)

Interpretation of Complex Concepts

- The interpretation function $\cdot^\mathcal{I}$ can be recursively extended to complex concepts as follows:
  - $\top^\mathcal{I} = \Delta^\mathcal{I}$
  - $\bot^\mathcal{I} = \emptyset$
  - $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$
  - $(C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$
  - $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$
  - $(\exists R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \exists y : (x, y) \in R^\mathcal{I} \ \&\ y \in C^\mathcal{I}\}$
  - $(\forall R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \forall y.\ (x, y) \in R^\mathcal{I} \ \Rightarrow\ y \in C^\mathcal{I}\}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $Male^\mathcal{I} = \{a\}$, $Female^\mathcal{I} = \{b\}$, $hasChild^\mathcal{I} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^\mathcal{I} = \{a\}$
  - $(\forall hasChild.Female)^\mathcal{I} = \{a, b\}$ (!!!)
  - $(\forall hasChild.Male)^\mathcal{I} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \;\&\; y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \;\Rightarrow\; y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^{\mathcal{I}} = \{a\}$
  - $(\forall hasChild.Female)^{\mathcal{I}} = \{a, b\}$ (!!!)
  - $(\forall hasChild.Male)^{\mathcal{I}} = \{b\}$ (!!!)

Interpretation of Complex Concepts

- The interpretation function $\cdot^{\mathcal{I}}$ can be recursively extended to complex concepts as follows:
  - $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
  - $\bot^{\mathcal{I}} = \emptyset$
  - $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in R^{\mathcal{I}} \ \& \ y \in C^{\mathcal{I}}\}$
  - $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y.\ (x, y) \in R^{\mathcal{I}} \ \Rightarrow \ y \in C^{\mathcal{I}}\}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^{\mathcal{I}} = \{a\}$
  - $(\forall hasChild.Female)^{\mathcal{I}} = \{a, b\}$ (!!!)
  - $(\forall hasChild.Male)^{\mathcal{I}} = \{b\}$ (!!!)
  - $(\exists hasChild.\forall hasChild.\bot)^{\mathcal{I}} =$

Interpretation of Complex Concepts

- The interpretation function $\cdot^\mathcal{I}$ can be recursively extended to complex concepts as follows:
  - $\top^\mathcal{I} = \Delta^\mathcal{I}$
  - $\bot^\mathcal{I} = \emptyset$
  - $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$
  - $(C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$
  - $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$
  - $(\exists R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \exists y : (x, y) \in R^\mathcal{I} \ \&\ y \in C^\mathcal{I}\}$
  - $(\forall R.C)^\mathcal{I} = \{x \in \Delta^\mathcal{I} \mid \forall y.\ (x, y) \in R^\mathcal{I} \ \Rightarrow\ y \in C^\mathcal{I}\}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $Male^\mathcal{I} = \{a\}$, $Female^\mathcal{I} = \{b\}$, $hasChild^\mathcal{I} = \{\langle a, b\rangle\}$. Then:
  - $(\exists hasChild.Female)^\mathcal{I} = \{a\}$
  - $(\forall hasChild.Female)^\mathcal{I} = \{a, b\}$ (!!!)
  - $(\forall hasChild.Male)^\mathcal{I} = \{b\}$ (!!!)
  - $(\exists hasChild.\forall hasChild.\bot)^\mathcal{I} = \{a\}$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D \quad \text{iff} \quad C^\mathcal{I} \subseteq D^\mathcal{I}$
  - $\mathcal{I} \models C \equiv D \quad \text{iff} \quad C^\mathcal{I} = D^\mathcal{I}$
  - $\mathcal{I} \models C(a) \quad \text{iff} \quad a^\mathcal{I} \in C^\mathcal{I}$
  - $\mathcal{I} \models R(a, b) \quad \text{iff} \quad (a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models C(a)$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{Male}^{\mathcal{I}} = \{a\}$, $\mathit{Female}^{\mathcal{I}} = \{b\}$, and $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$, $\mathit{john}^{\mathcal{I}} = a$, $\mathit{mary}^{\mathcal{I}} = b$. Then:

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models C(a)$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, and $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$, $john^{\mathcal{I}} = a$, $mary^{\mathcal{I}} = b$. Then:
  - $\mathcal{I}$ $\quad Male \sqsubseteq Female$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^\mathcal{I} \subseteq D^\mathcal{I}$
  - $\mathcal{I} \models C \equiv D$ iff $C^\mathcal{I} = D^\mathcal{I}$
  - $\mathcal{I} \models C(a)$ iff $a^\mathcal{I} \in C^\mathcal{I}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $\mathit{Male}^\mathcal{I} = \{a\}$, $\mathit{Female}^\mathcal{I} = \{b\}$, and $\mathit{hasChild}^\mathcal{I} = \{\langle a, b\rangle\}$, $\mathit{john}^\mathcal{I} = a$, $\mathit{mary}^\mathcal{I} = b$. Then:
  - $\mathcal{I} \not\models \mathit{Male} \sqsubseteq \mathit{Female}$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^\mathcal{I} \subseteq D^\mathcal{I}$
  - $\mathcal{I} \models C \equiv D$ iff $C^\mathcal{I} = D^\mathcal{I}$
  - $\mathcal{I} \models C(a)$ iff $a^\mathcal{I} \in C^\mathcal{I}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $\mathit{Male}^\mathcal{I} = \{a\}$, $\mathit{Female}^\mathcal{I} = \{b\}$, and $\mathit{hasChild}^\mathcal{I} = \{\langle a, b\rangle\}$, $\mathit{john}^\mathcal{I} = a$, $\mathit{mary}^\mathcal{I} = b$. Then:
  - $\mathcal{I} \not\models \mathit{Male} \sqsubseteq \mathit{Female}$
  - $\mathcal{I} \quad \mathit{Male} \equiv \neg\mathit{Female}$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models C(a)$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, and $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$, $john^{\mathcal{I}} = a$, $mary^{\mathcal{I}} = b$. Then:
  - $\mathcal{I} \not\models Male \sqsubseteq Female$
  - $\mathcal{I} \models Male \equiv \neg Female$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^\mathcal{I} \subseteq D^\mathcal{I}$
  - $\mathcal{I} \models C \equiv D$ iff $C^\mathcal{I} = D^\mathcal{I}$
  - $\mathcal{I} \models C(a)$ iff $a^\mathcal{I} \in C^\mathcal{I}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $Male^\mathcal{I} = \{a\}$, $Female^\mathcal{I} = \{b\}$, and $hasChild^\mathcal{I} = \{\langle a, b\rangle\}$, $john^\mathcal{I} = a$, $mary^\mathcal{I} = b$. Then:
  - $\mathcal{I} \not\models Male \sqsubseteq Female$
  - $\mathcal{I} \models Male \equiv \neg Female$
  - $\mathcal{I} \quad (\exists hasChild.Male)(john)$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models C(a)$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, and $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$, $john^{\mathcal{I}} = a$, $mary^{\mathcal{I}} = b$. Then:
  - $\mathcal{I} \not\models Male \sqsubseteq Female$
  - $\mathcal{I} \models Male \equiv \neg Female$
  - $\mathcal{I} \not\models (\exists hasChild.Male)(john)$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D \quad$ iff $\quad C^\mathcal{I} \subseteq D^\mathcal{I}$
  - $\mathcal{I} \models C \equiv D \quad$ iff $\quad C^\mathcal{I} = D^\mathcal{I}$
  - $\mathcal{I} \models C(a) \quad$ iff $\quad a^\mathcal{I} \in C^\mathcal{I}$
  - $\mathcal{I} \models R(a,b) \quad$ iff $\quad (a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$
- Example: let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $Male^\mathcal{I} = \{a\}$, $Female^\mathcal{I} = \{b\}$, and $hasChild^\mathcal{I} = \{\langle a, b\rangle\}$, $john^\mathcal{I} = a$, $mary^\mathcal{I} = b$. Then:
  - $\mathcal{I} \not\models Male \sqsubseteq Female$
  - $\mathcal{I} \models Male \equiv \neg Female$
  - $\mathcal{I} \not\models (\exists hasChild.Male)(john)$
  - $\mathcal{I} \quad hasChild(mary, john)$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models C(a)$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\mathcal{I} \models R(a, b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, and $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$, $john^{\mathcal{I}} = a$, $mary^{\mathcal{I}} = b$. Then:
  - $\mathcal{I} \not\models Male \sqsubseteq Female$
  - $\mathcal{I} \models Male \equiv \neg Female$
  - $\mathcal{I} \not\models (\exists hasChild.Male)(john)$
  - $\mathcal{I} \not\models hasChild(mary, john)$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D \quad \text{iff} \quad C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D \quad \text{iff} \quad C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models C(a) \quad \text{iff} \quad a^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\mathcal{I} \models R(a, b) \quad \text{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $Male^{\mathcal{I}} = \{a\}$, $Female^{\mathcal{I}} = \{b\}$, and $hasChild^{\mathcal{I}} = \{\langle a, b\rangle\}$, $john^{\mathcal{I}} = a$, $mary^{\mathcal{I}} = b$. Then:
  - $\mathcal{I} \not\models Male \sqsubseteq Female$
  - $\mathcal{I} \models Male \equiv \neg Female$
  - $\mathcal{I} \not\models (\exists hasChild.Male)(john)$
  - $\mathcal{I} \not\models hasChild(mary, john)$
  - $\mathcal{I} \quad (\forall hasChild.\neg Male)(mary)$

Interpretation of Axioms

- An interpretation can either satisfy an axiom ($\mathcal{I} \models \alpha$) or violate it ($\mathcal{I} \not\models \alpha$):
  - $\mathcal{I} \models C \sqsubseteq D \quad \text{iff} \quad C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - $\mathcal{I} \models C \equiv D \quad \text{iff} \quad C^{\mathcal{I}} = D^{\mathcal{I}}$
  - $\mathcal{I} \models C(a) \quad \text{iff} \quad a^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $\mathcal{I} \models R(a, b) \quad \text{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- Example: let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{Male}^{\mathcal{I}} = \{a\}$, $\mathit{Female}^{\mathcal{I}} = \{b\}$, and $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$, $\mathit{john}^{\mathcal{I}} = a$, $\mathit{mary}^{\mathcal{I}} = b$. Then:
  - $\mathcal{I} \not\models \mathit{Male} \sqsubseteq \mathit{Female}$
  - $\mathcal{I} \models \mathit{Male} \equiv \neg\mathit{Female}$
  - $\mathcal{I} \not\models (\exists \mathit{hasChild}.\mathit{Male})(\mathit{john})$
  - $\mathcal{I} \not\models \mathit{hasChild}(\mathit{mary}, \mathit{john})$
  - $\mathcal{I} \models (\forall \mathit{hasChild}.\neg\mathit{Male})(\mathit{mary})$

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent
- Example: is the following ontology satisfiable?

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent
- Example: is the following ontology satisfiable?

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$

- Satisfiable in a "trivial" interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\mathit{Parent}^{\mathcal{I}} = \mathit{GrandParent}^{\mathcal{I}} = \mathit{hasChild}^{\mathcal{I}} = \emptyset$.

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent
- Example: is the following ontology satisfiable?

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$

- Satisfiable in a "trivial" interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\mathit{Parent}^{\mathcal{I}} = \mathit{GrandParent}^{\mathcal{I}} = \mathit{hasChild}^{\mathcal{I}} = \emptyset$.

  Indeed:
  $(\exists \mathit{hasChild}.\top)^{\mathcal{I}} \equiv \emptyset = \mathit{Parent}^{\mathcal{I}}$ and
  $(\exists \mathit{hasChild}.\mathit{Parent})^{\mathcal{I}} = \emptyset = \mathit{GrandParent}^{\mathcal{I}}$

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent
- Example: is the following ontology satisfiable?

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
(ax3) $(\mathit{GrandParent} \sqcap \neg \mathit{Parent})(\mathit{john})$

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent
- Example: is the following ontology satisfiable?

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
(ax3) $(\mathit{GrandParent} \sqcap \neg \mathit{Parent})(\mathit{john})$

- The trivial interpretation does not satisfy the last axiom!

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent
- Example: is the following ontology satisfiable?

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
(ax3) $(\mathit{GrandParent} \sqcap \neg \mathit{Parent})(\mathit{john})$

- The trivial interpretation does not satisfy the last axiom! $\mathit{john}^{\mathcal{I}} \notin \emptyset = (\mathit{GrandParent} \sqcap \neg \mathit{Parent})^{\mathcal{I}}$

Models

- Interpretations that satisfy axioms are of a special interest
  - because they "agree" with the requirements imposed by axioms
- $\mathcal{I}$ is called a model of $\mathcal{O}$ ($\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \alpha$ for each $\alpha \in \mathcal{O}$
- $\mathcal{O}$ is satisfiable if there exists at least one model $\mathcal{I}$ of $\mathcal{O}$
- Otherwise, $\mathcal{O}$ is unsatisfiable or inconsistent
- Example: is the following ontology satisfiable?

(ax1) $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
(ax2) $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
(ax3) $(\mathit{GrandParent} \sqcap \neg \mathit{Parent})(\mathit{john})$

- The trivial interpretation does not satisfy the last axiom! $\mathit{john}^{\mathcal{I}} \notin \emptyset = (\mathit{GrandParent} \sqcap \neg \mathit{Parent})^{\mathcal{I}}$
- Proving unsatisfiability is harder: one has to prove that $\mathcal{I} \not\models \mathcal{O}$ for every interpretation $\mathcal{I}$!

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$ is not satisfiable because $\bot^{\mathcal{I}} = \emptyset$ by definition

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$ is not satisfiable because $\bot^{\mathcal{I}} = \emptyset$ by definition
  - $\forall R.\bot$

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$ is not satisfiable because $\bot^{\mathcal{I}} = \emptyset$ by definition
  - $\forall R.\bot$ is satisfiable in every $\mathcal{I}$ in which $R^{\mathcal{I}} = \emptyset$

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$ is not satisfiable because $\bot^{\mathcal{I}} = \emptyset$ by definition
  - $\forall R.\bot$ is satisfiable in every $\mathcal{I}$ in which $R^{\mathcal{I}} = \emptyset$
  - $\exists R.\top$ w.r.t. $\mathcal{O} = \{\top \sqsubseteq \exists R.\bot\}$

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^\mathcal{I} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^\mathcal{I} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$ is not satisfiable because $\bot^\mathcal{I} = \emptyset$ by definition
  - $\forall R.\bot$ is satisfiable in every $\mathcal{I}$ in which $R^\mathcal{I} = \emptyset$
  - $\exists R.\top$ w.r.t. $\mathcal{O} = \{\top \sqsubseteq \exists R.\bot\}$ is not satisfiable because $\mathcal{O}$ does not have a model (why?)

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$ is not satisfiable because $\bot^{\mathcal{I}} = \emptyset$ by definition
  - $\forall R.\bot$ is satisfiable in every $\mathcal{I}$ in which $R^{\mathcal{I}} = \emptyset$
  - $\exists R.\top$ w.r.t. $\mathcal{O} = \{\top \sqsubseteq \exists R.\bot\}$ is not satisfiable because $\mathcal{O}$ does not have a model (why?)
  - A w.r.t. $\mathcal{O} = \{A \sqsubseteq \neg A\}$?

Satisfiability of Concepts

- In addition to satisfiability of axioms, one can be interested in satisfiability of concepts
- A (possibly complex) concept C is satisfiable if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A (possibly complex) concept C is satisfiable w.r.t. $\mathcal{O}$, if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$.
- Which of the following concepts are satisfiable?
  - $\bot$ is not satisfiable because $\bot^{\mathcal{I}} = \emptyset$ by definition
  - $\forall R.\bot$ is satisfiable in every $\mathcal{I}$ in which $R^{\mathcal{I}} = \emptyset$
  - $\exists R.\top$ w.r.t. $\mathcal{O} = \{\top \sqsubseteq \exists R.\bot\}$ is not satisfiable because $\mathcal{O}$ does not have a model (why?)
  - A w.r.t. $\mathcal{O} = \{A \sqsubseteq \neg A\}$? (tricky!)

Entailment

- One is often interested in logical consequences of an ontology

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^{\mathcal{I}} \subseteq B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^{\mathcal{I}} \subseteq B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^{\mathcal{I}} \subseteq B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^{\mathcal{I}} \subseteq B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^\mathcal{I} \subseteq B^\mathcal{I} \subseteq C^\mathcal{I}$
- Example: $\mathcal{O} = \{A \sqsubseteq \exists R.B,\ B \sqsubseteq C\} \models A \sqsubseteq \exists R.C$
  - $\forall x \in A^\mathcal{I}\ \exists y : \langle x, y \rangle \in R^\mathcal{I}\ \&\ y \in B^\mathcal{I} \subseteq C^\mathcal{I}$

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^{\mathcal{I}} \subseteq B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$
- Example: $\mathcal{O} = \{A \sqsubseteq \exists R.B,\ B \sqsubseteq C\} \models A \sqsubseteq \exists R.C$
  - $\forall x \in A^{\mathcal{I}}\ \exists y : \langle x, y\rangle \in R^{\mathcal{I}}\ \&\ y \in B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^\mathcal{I} \subseteq B^\mathcal{I} \subseteq C^\mathcal{I}$
- Example: $\mathcal{O} = \{A \sqsubseteq \exists R.B,\ B \sqsubseteq C\} \models A \sqsubseteq \exists R.C$
  - $\forall x \in A^\mathcal{I}\ \exists y : \langle x, y \rangle \in R^\mathcal{I}\ \&\ y \in B^\mathcal{I} \subseteq C^\mathcal{I}$

Entailment

- One is often interested in logical consequences of an ontology
- An ontology $\mathcal{O}$ entails an axiom α ($\mathcal{O} \models \alpha$) if for every $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$ we have $\mathcal{I} \models \alpha$.
- Example: $\mathcal{O} = \{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$
  - if $\mathcal{I} \models \mathcal{O}$ then $A^\mathcal{I} \subseteq B^\mathcal{I} \subseteq C^\mathcal{I}$
- Example: $\mathcal{O} = \{A \sqsubseteq \exists R.B,\ B \sqsubseteq C\} \models A \sqsubseteq \exists R.C$
  - $\forall x \in A^\mathcal{I}\ \exists y : \langle x, y\rangle \in R^\mathcal{I}\ \&\ y \in B^\mathcal{I} \subseteq C^\mathcal{I}$

Outline

Ontology Engineering

- How to create a useful ontology?

Ontology Engineering

- How to create a useful ontology?
  1. It should be detailed enough to capture the intended application domain in a precise way
  2. It should be error-free

Ontology Engineering

- How to create a useful ontology?
  1. It should be detailed enough to capture the intended application domain in a precise way
  2. It should be error-free
- Both of these conditions are hard to achieve in the same time
  - the more axioms are added, the higher is a chance of an error

Ontology Engineering

- How to create a useful ontology?
  1. It should be detailed enough to capture the intended application domain in a precise way
  2. It should be error-free
- Both of these conditions are hard to achieve in the same time
  - the more axioms are added, the higher is a chance of an error
- We should aim at detecting as much errors as possible automatically

Modeling Errors

- What can be regarded as a modeling error?

Modeling Errors

- What can be regarded as a modeling error?
  1. Inconsistency of an ontology $\mathcal{O}$.

Modeling Errors

- What can be regarded as a modeling error?
  1. Inconsistency of an ontology $\mathcal{O}$.

     Example: $\mathcal{O} =$

     1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
     2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
     3. $(\mathit{GrandParent} \sqcap \neg \mathit{Parent})(\mathit{john})$

Modeling Errors

- What can be regarded as a modeling error?
  1. Inconsistency of an ontology $\mathcal{O}$.
  2. Unsatisfiability of an atomic concept: $\mathcal{O} \models A \sqsubseteq \bot$

Modeling Errors

- What can be regarded as a modeling error?
  1. Inconsistency of an ontology $\mathcal{O}$.
  2. Unsatisfiability of an atomic concept: $\mathcal{O} \models A \sqsubseteq \bot$

     Example: $\mathcal{O} =$

     1. $\mathit{Parent} \sqsubseteq \mathit{GrandParent}$
     2. $\mathit{Parent} \sqcap \mathit{GandParent} \sqsubseteq \bot$

     $\models \mathit{Parent} \sqsubseteq \bot$

Modeling Errors

- What can be regarded as a modeling error?
  1. Inconsistency of an ontology $\mathcal{O}$.
  2. Unsatisfiability of an atomic concept: $\mathcal{O} \models A \sqsubseteq \bot$

     Example: $\mathcal{O} =$

     1. $\mathit{Parent} \sqsubseteq \mathit{GrandParent}$
     2. $\mathit{Parent} \sqcap \mathit{GandParent} \sqsubseteq \bot$

     $\models \; \mathit{Parent} \sqsubseteq \bot$

Modeling Errors

- What can be regarded as a modeling error?
  1. Inconsistency of an ontology $\mathcal{O}$.
  2. Unsatisfiability of an atomic concept: $\mathcal{O} \models A \sqsubseteq \bot$
  3. Unexpected consequence: $\mathcal{O} \models C \sqsubseteq D$

Modeling Errors

- What can be regarded as a modeling error?
  1. Inconsistency of an ontology $\mathcal{O}$.
  2. Unsatisfiability of an atomic concept: $\mathcal{O} \models A \sqsubseteq \bot$
  3. Unexpected consequence: $\mathcal{O} \models C \sqsubseteq D$

     Example: $\mathcal{O} =$

     1. $\mathit{HappyParent} \equiv \forall \mathit{hasChild}.\mathit{Happy}$
     2. $\mathit{NotParent} \equiv \neg\exists \mathit{hasChild}.\top$

     $\models \mathit{NotParent} \sqsubseteq \mathit{HappyParent}$

Standard Reasoning Problems

- Ontology satisfiability checking:
  - Given: $\mathcal{O}$ an ontology
  - Return: yes if $\mathcal{O}$ is satisfiable, and no otherwise

Standard Reasoning Problems

- Ontology satisfiability checking:
  - Given: $\mathcal{O}$ an ontology
  - Return: yes if $\mathcal{O}$ is satisfiable, and no otherwise
- Concept satisfiability checking:
  - Given: $\mathcal{O}$ an ontology, C a concept
  - Return: yes if C is satisfiable w.r.t. $\mathcal{O}$, and no otherwise

Standard Reasoning Problems

- Ontology satisfiability checking:
  - Given: $\mathcal{O}$ an ontology
  - Return: yes if $\mathcal{O}$ is satisfiable, and no otherwise
- Concept satisfiability checking:
  - Given: $\mathcal{O}$ an ontology, C a concept
  - Return: yes if C is satisfiable w.r.t. $\mathcal{O}$, and no otherwise
- Concept subsumption checking:
  - Given: $\mathcal{O}$ an ontology, C, D concepts
  - Return: yes if $\mathcal{O} \models C \sqsubseteq D$ and no otherwise

Standard Reasoning Problems

- Ontology satisfiability checking:
  - Given: $\mathcal{O}$ an ontology
  - Return: yes if $\mathcal{O}$ is satisfiable, and no otherwise
- Concept satisfiability checking:
  - Given: $\mathcal{O}$ an ontology, C a concept
  - Return: yes if C is satisfiable w.r.t. $\mathcal{O}$, and no otherwise
- Concept subsumption checking:
  - Given: $\mathcal{O}$ an ontology, C, D concepts
  - Return: yes if $\mathcal{O} \models C \sqsubseteq D$ and no otherwise
- Instance checking:
  - Given: $\mathcal{O}$ an ontology, C a concept, a an individual
  - Return: yes if $\mathcal{O} \models C(a)$ and no otherwise

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Satisfiability checking: is $\mathcal{O}$ satisfiable? Yes:
  - Take $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{john}^{\mathcal{I}} = a$, $\mathit{mary}^{\mathcal{I}} = b$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$, $\mathit{Parent}^{\mathcal{I}} = \{a\}$, $\mathit{GrandParent}^{\mathcal{I}} = \emptyset$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Satisfiability checking: is $\mathcal{O}$ satisfiable? Yes:
  - Take $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{john}^{\mathcal{I}} = a$, $\mathit{mary}^{\mathcal{I}} = b$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$, $\mathit{Parent}^{\mathcal{I}} = \{a\}$, $\mathit{GrandParent}^{\mathcal{I}} = \emptyset$.
- Concept satisfiability: is *Parent* satisfiable w.r.t. $\mathcal{O}$? Yes

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Satisfiability checking: is $\mathcal{O}$ satisfiable? Yes:
  - Take $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{john}^{\mathcal{I}} = a$, $\mathit{mary}^{\mathcal{I}} = b$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$, $\mathit{Parent}^{\mathcal{I}} = \{a\}$, $\mathit{GrandParent}^{\mathcal{I}} = \emptyset$.
- Concept satisfiability: is *Parent* satisfiable w.r.t. $\mathcal{O}$? Yes
- Concept satisfiability: is *GrandParent* satisfiable w.r.t. $\mathcal{O}$?

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Satisfiability checking: is $\mathcal{O}$ satisfiable? Yes:
  - Take $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{a, b\}$, $\mathit{john}^{\mathcal{I}} = a$, $\mathit{mary}^{\mathcal{I}} = b$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle a, b\rangle\}$, $\mathit{Parent}^{\mathcal{I}} = \{a\}$, $\mathit{GrandParent}^{\mathcal{I}} = \emptyset$.
- Concept satisfiability: is Parent satisfiable w.r.t. $\mathcal{O}$? Yes
- Concept satisfiability: is $\mathit{GrandParent}$ satisfiable w.r.t. $\mathcal{O}$? Yes
  - Take $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\Delta^{\mathcal{I}} = \{x, a, b\}$, $\mathit{john}^{\mathcal{I}} = a$, $\mathit{mary}^{\mathcal{I}} = b$, $\mathit{hasChild}^{\mathcal{I}} = \{\langle x, a\rangle, \langle a, b\rangle\}$, $\mathit{Parent}^{\mathcal{I}} = \{x, a\}$, $\mathit{GrandParent}^{\mathcal{I}} = \{x\}$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Satisfiability checking: is $\mathcal{O}$ satisfiable? Yes:
  - Take $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{a, b\}$, $\mathit{john}^\mathcal{I} = a$, $\mathit{mary}^\mathcal{I} = b$, $\mathit{hasChild}^\mathcal{I} = \{\langle a, b\rangle\}$, $\mathit{Parent}^\mathcal{I} = \{a\}$, $\mathit{GrandParent}^\mathcal{I} = \emptyset$.
- Concept satisfiability: is Parent satisfiable w.r.t. $\mathcal{O}$? Yes
- Concept satisfiability: is $\mathit{GrandParent}$ satisfiable w.r.t. $\mathcal{O}$? Yes
  - Take $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ with $\Delta^\mathcal{I} = \{x, a, b\}$, $\mathit{john}^\mathcal{I} = a$, $\mathit{mary}^\mathcal{I} = b$, $\mathit{hasChild}^\mathcal{I} = \{\langle x, a\rangle, \langle a, b\rangle\}$, $\mathit{Parent}^\mathcal{I} = \{x, a\}$, $\mathit{GrandParent}^\mathcal{I} = \{x\}$.
- Concept subsumption: $\mathcal{O} \models \mathit{Parent} \sqsubseteq \mathit{GrandParent}$? No:
  - $\mathcal{I} \not\models \mathit{Parent} \sqsubseteq \mathit{GrandParent}$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Concept subsumption: $\mathcal{O} \models \mathit{GrandParent} \sqsubseteq \mathit{Parent}$? Yes:

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Concept subsumption: $\mathcal{O} \models \mathit{GrandParent} \sqsubseteq \mathit{Parent}$? Yes:
  - Take any interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Concept subsumption: $\mathcal{O} \models \mathit{GrandParent} \sqsubseteq \mathit{Parent}$? Yes:
  - Take any interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$.
  - Take any $x \in \mathit{GrandParent}^{\mathcal{I}} = (\exists \mathit{hasChild}.\mathit{Parent})^{\mathcal{I}}$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $Parent \equiv \exists hasChild.\top$
  2. $GrandParent \equiv \exists hasChild.Parent$
  3. $hasChild(john, mary)$
- Concept subsumption: $\mathcal{O} \models GrandParent \sqsubseteq Parent$? Yes:
  - Take any interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$.
  - Take any $x \in GrandParent^{\mathcal{I}} = (\exists hasChild.Parent)^{\mathcal{I}}$.
  - Hence, $\exists y : \langle x, y \rangle \in hashChild^{\mathcal{I}}$ & $y \in Parent^{\mathcal{I}}$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $Parent \equiv \exists hasChild.\top$
  2. $GrandParent \equiv \exists hasChild.Parent$
  3. $hasChild(john, mary)$
- Concept subsumption: $\mathcal{O} \models GrandParent \sqsubseteq Parent$? Yes:
  - Take any interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$.
  - Take any $x \in GrandParent^{\mathcal{I}} = (\exists hasChild.Parent)^{\mathcal{I}}$.
  - Hence, $\exists y : \langle x, y \rangle \in hashChild^{\mathcal{I}}$ & $y \in Parent^{\mathcal{I}}$.
  - Trivially, $y \in \Delta^{\mathcal{I}} = \top^{\mathcal{I}}$, so $x \in (\exists hasChild.\top)^{\mathcal{I}} = Parent^{\mathcal{I}}$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Concept subsumption: $\mathcal{O} \models \mathit{GrandParent} \sqsubseteq \mathit{Parent}$? Yes:
  - Take any interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$.
  - Take any $x \in \mathit{GrandParent}^{\mathcal{I}} = (\exists \mathit{hasChild}.\mathit{Parent})^{\mathcal{I}}$.
  - Hence, $\exists y : \langle x, y \rangle \in \mathit{hashChild}^{\mathcal{I}}$ & $y \in \mathit{Parent}^{\mathcal{I}}$.
  - Trivially, $y \in \Delta^{\mathcal{I}} = \top^{\mathcal{I}}$, so $x \in (\exists \mathit{hasChild}.\top)^{\mathcal{I}} = \mathit{Parent}^{\mathcal{I}}$.
  - Since $x \in \mathit{GrandParent}^{\mathcal{I}}$ was arbitrary, we proved that $\mathit{GrandParent}^{\mathcal{I}} \subseteq \mathit{Parent}^{\mathcal{I}}$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $Parent \equiv \exists hasChild.\top$
  2. $GrandParent \equiv \exists hasChild.Parent$
  3. $hasChild(john, mary)$
- Concept subsumption: $\mathcal{O} \models GrandParent \sqsubseteq Parent$? Yes:
  - Take any interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$.
  - Take any $x \in GrandParent^{\mathcal{I}} = (\exists hasChild.Parent)^{\mathcal{I}}$.
  - Hence, $\exists y : \langle x, y\rangle \in hashChild^{\mathcal{I}}$ & $y \in Parent^{\mathcal{I}}$.
  - Trivially, $y \in \Delta^{\mathcal{I}} = \top^{\mathcal{I}}$, so $x \in (\exists hasChild.\top)^{\mathcal{I}} = Parent^{\mathcal{I}}$.
  - Since $x \in GrandParent^{\mathcal{I}}$ was arbitrary, we proved that $GrandParent^{\mathcal{I}} \subseteq Parent^{\mathcal{I}}$.
  - Since $\mathcal{I} \models \mathcal{O}$ was arbitrary, we proved that $\mathcal{O} \models GrandParent \sqsubseteq Parent$.

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Instance checking: what are the instances of *Parent*?
  - $\mathit{Parent}(\mathit{john})$
  - $\mathit{Parent}(\mathit{mary})$

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Instance checking: what are the instances of *Parent*?
  - $\mathcal{O} \models \mathit{Parent}(\mathit{john})$ – See the paper
  - $\mathit{Parent}(\mathit{mary})$

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Instance checking: what are the instances of *Parent*?
  - $\mathcal{O} \models \mathit{Parent}(\mathit{john})$ – See the paper
  - $\mathcal{O} \not\models \mathit{Parent}(\mathit{mary})$ – $\mathit{mary}^{\mathcal{I}} \notin \mathit{Parent}^{\mathcal{I}}$

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $Parent \equiv \exists hasChild.\top$
  2. $GrandParent \equiv \exists hasChild.Parent$
  3. $hasChild(john, mary)$
- Instance checking: what are the instances of $Parent$?
  - $\mathcal{O} \models Parent(john)$ – See the paper
  - $\mathcal{O} \not\models Parent(mary)$ – $mary^{\mathcal{I}} \notin Parent^{\mathcal{I}}$
- Instance checking: what are the instances of $\neg Parent$?
  - $(\neg Parent)(john)$
  - $(\neg Parent)(mary)$

Standard Reasoning Problems: Examples

- Example: ontology $\mathcal{O}$:
  1. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$
  2. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$
  3. $\mathit{hasChild}(\mathit{john}, \mathit{mary})$
- Instance checking: what are the instances of *Parent*?
  - $\mathcal{O} \models \mathit{Parent}(\mathit{john})$ – See the paper
  - $\mathcal{O} \not\models \mathit{Parent}(\mathit{mary})$ – $\mathit{mary}^{\mathcal{I}} \notin \mathit{Parent}^{\mathcal{I}}$
- Instance checking: what are the instances of $\neg\mathit{Parent}$?
  - $\mathcal{O} \not\models (\neg\mathit{Parent})(\mathit{john})$ – Exercise
  - $\mathcal{O} \not\models (\neg\mathit{Parent})(\mathit{mary})$ – Exercise

Outline

Reduction between Decision Problems

- There are quite many reasoning problems for DLs. Do we really need to find algorithms for each of them independently?

Reduction between Decision Problems

- There are quite many reasoning problems for DLs. Do we really need to find algorithms for each of them independently?
- Use a reduction!
  - if we solve one problem, we solve all of them!

Reduction between Decision Problems

- Recall: a decision problem (for the input set X) is a mapping $P : X \to yes, no$.

Reduction between Decision Problems

- Recall: a decision problem (for the input set X) is a mapping $P : X \to yes, no$.

Definition (Reduction)

$P_1 : X \to \{yes, no\}$ is reducible to $P_2 : Y \to \{yes, no\}$ if $\exists$ an algorithm $R : X \to Y$ (a reduction) such that $\forall x \in X$:

- if $P_1(x) = yes$ then $P_2(R(x)) = yes$
- if $P_1(x) = no$ then $P_2(R(x)) = no$

Reduction between Decision Problems

- Recall: a decision problem (for the input set X) is a mapping $P : X \to yes, no$.

Definition (Reduction)

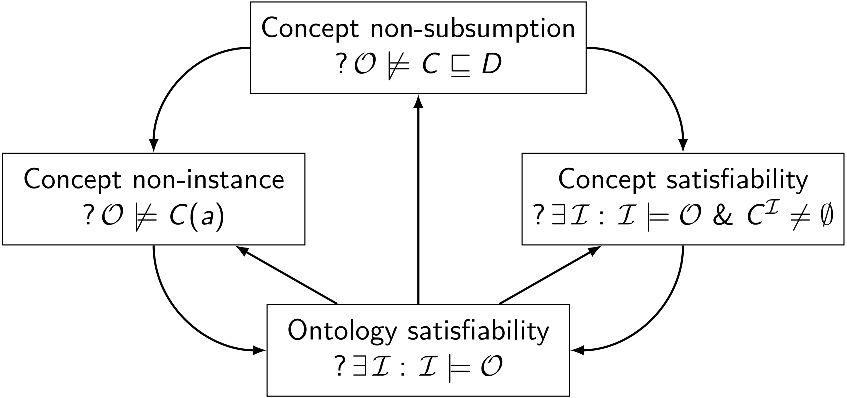
$P_1 : X \to \{yes, no\}$ is reducible to $P_2 : Y \to \{yes, no\}$ if $\exists$ an algorithm $R : X \to Y$ (a reduction) such that $\forall x \in X$:

- if $P_1(x) = yes$ then $P_2(R(x)) = yes$
- if $P_1(x) = no$ then $P_2(R(x)) = no$

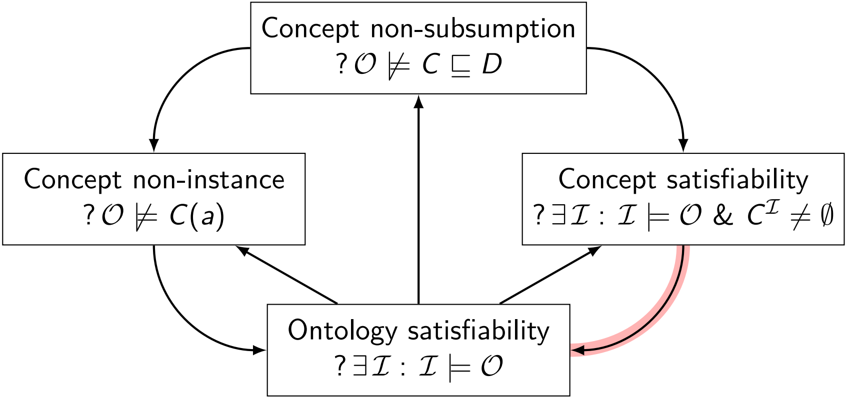
If R is polynomial then P_1 is polynomially reducible to P_2.

- polynomial reductions are of a most interest

Reduction between Reasoning Problems



Concept Satisfiability $\Rightarrow$ Ontology Satisfiability



Concept Satisfiability ⇒ Ontology Satisfiability

Lemma

C is satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{O} \cup \{C(a)\}$ is satisfiable for every a not appearing in $\mathcal{O}$.

Concept Satisfiability ⇒ Ontology Satisfiability

Lemma

C is satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{O} \cup \{C(a)\}$ is satisfiable for every a not appearing in $\mathcal{O}$.

Proof.

- $(\Rightarrow)$: if C is satisfiable w.r.t. $\mathcal{O}$ then there exists $\mathcal{I} \models \mathcal{O}$ such that $C^{\mathcal{I}} \neq \emptyset$. That is, there exists some $x \in C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$.

Concept Satisfiability ⇒ Ontology Satisfiability

Lemma

C is satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{O} \cup \{C(a)\}$ is satisfiable for every a not appearing in $\mathcal{O}$.

Proof.

- $(\Rightarrow)$: if C is satisfiable w.r.t. $\mathcal{O}$ then there exists $\mathcal{I} \models \mathcal{O}$ such that $C^{\mathcal{I}} \neq \emptyset$. That is, there exists some $x \in C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$.

  Let $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be an interpretation such that:
  - $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}}$
  - $A^{\mathcal{J}} = A^{\mathcal{I}}$, $R^{\mathcal{J}} = R^{\mathcal{I}}$ for every atomic A and R
  - $i^{\mathcal{J}} = i^{\mathcal{I}}$ for every individual $i \neq a$
  - $a^{\mathcal{J}} = x \in C^{\mathcal{I}} \in \Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$

Concept Satisfiability $\Rightarrow$ Ontology Satisfiability

Lemma

C is satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{O} \cup \{C(a)\}$ is satisfiable for every a not appearing in $\mathcal{O}$.

Proof.

- $(\Rightarrow)$: if C is satisfiable w.r.t. $\mathcal{O}$ then there exists $\mathcal{I} \models \mathcal{O}$ such that $C^{\mathcal{I}} \neq \emptyset$. That is, there exists some $x \in C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$.

  Let $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be an interpretation such that:
  - $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}}$
  - $A^{\mathcal{J}} = A^{\mathcal{I}}$, $R^{\mathcal{J}} = R^{\mathcal{I}}$ for every atomic A and R
  - $i^{\mathcal{J}} = i^{\mathcal{I}}$ for every individual $i \neq a$
  - $a^{\mathcal{J}} = x \in C^{\mathcal{I}} \in \Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$

  Then $\mathcal{J} \models \mathcal{O}$ and $\mathcal{J} \models C(a)$.

Concept Satisfiability $\Rightarrow$ Ontology Satisfiability

Lemma

C is satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{O} \cup \{C(a)\}$ is satisfiable for every a not appearing in $\mathcal{O}$.

Proof.

- $(\Leftarrow)$: If $\mathcal{O} \cup \{C(a)\}$ is satisfiable then there exists a model $\mathcal{I} \models \mathcal{O} \cup \{C(a)\}$.

Concept Satisfiability $\Rightarrow$ Ontology Satisfiability

Lemma

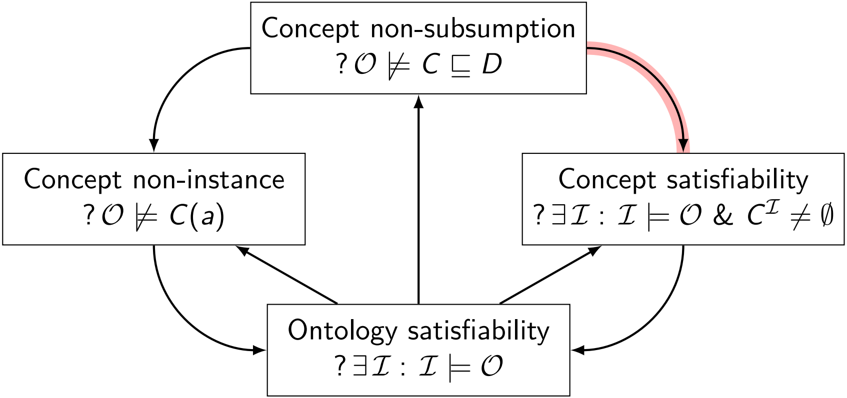
C is satisfiable w.r.t. $\mathcal{O}$ iff $\mathcal{O} \cup \{C(a)\}$ is satisfiable for every a not appearing in $\mathcal{O}$.

Proof.

- $(\Leftarrow)$: If $\mathcal{O} \cup \{C(a)\}$ is satisfiable then there exists a model $\mathcal{I} \models \mathcal{O} \cup \{C(a)\}$.

  Then $\mathcal{I} \models \mathcal{O}$ and $a^{\mathcal{I}} \in C^{\mathcal{I}} \neq \emptyset$. $\square$

Concept Non-Subsumption $\Rightarrow$ Concept Unsatisfiability



Concept Non-Subsumption $\Rightarrow$ Concept Unsatisfiability

Lemma

$\mathcal{O} \not\models C \sqsubseteq D$ iff $C \sqcap \neg D$ is satisfiable w.r.t. $\mathcal{O}$.

Concept Non-Subsumption $\Rightarrow$ Concept Unsatisfiability

Lemma

$\mathcal{O} \not\models C \sqsubseteq D$ iff $C \sqcap \neg D$ is satisfiable w.r.t. $\mathcal{O}$.

Proof.

- $(\Rightarrow)$: If $\mathcal{O} \not\models C \sqsubseteq D$ then $\exists \mathcal{I} \models \mathcal{O}$: $\mathcal{I} \not\models C \sqsubseteq D$.

Concept Non-Subsumption $\Rightarrow$ Concept Unsatisfiability

Lemma

$\mathcal{O} \not\models C \sqsubseteq D$ iff $C \sqcap \neg D$ is satisfiable w.r.t. $\mathcal{O}$.

Proof.

- $(\Rightarrow)$: If $\mathcal{O} \not\models C \sqsubseteq D$ then $\exists \mathcal{I} \models \mathcal{O}$: $\mathcal{I} \not\models C \sqsubseteq D$.
  Then $C^{\mathcal{I}} \not\subseteq D^{\mathcal{I}}$. Hence $\exists x \in (C^{\mathcal{I}} \setminus D^{\mathcal{I}}) = (C \sqcap \neg D)^{\mathcal{I}}$.

Concept Non-Subsumption $\Rightarrow$ Concept Unsatisfiability

Lemma

$\mathcal{O} \not\models C \sqsubseteq D$ iff $C \sqcap \neg D$ is satisfiable w.r.t. $\mathcal{O}$.

Proof.

- $(\Leftarrow)$: If $C \sqcap \neg D$ is satisfiable w.r.t. $\mathcal{O}$ then $\exists \mathcal{I}$: $(C \sqcap \neg D)^{\mathcal{I}} \neq \emptyset$.

Concept Non-Subsumption $\Rightarrow$ Concept Unsatisfiability

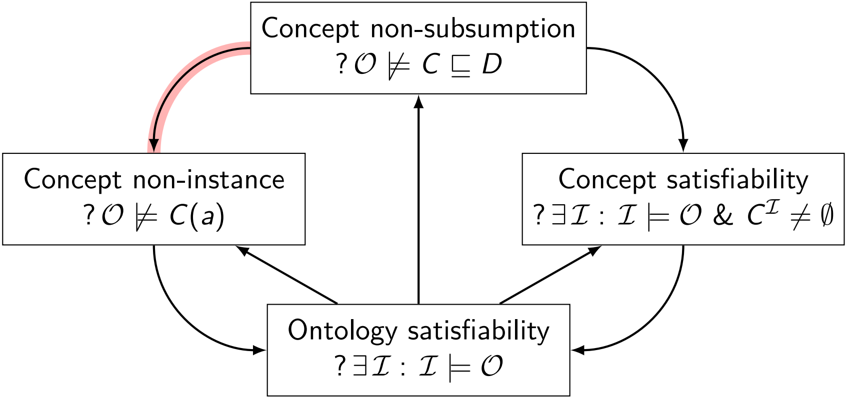
Lemma

$\mathcal{O} \not\models C \sqsubseteq D$ iff $C \sqcap \neg D$ is satisfiable w.r.t. $\mathcal{O}$.

Proof.

- $(\Leftarrow)$: If $C \sqcap \neg D$ is satisfiable w.r.t. $\mathcal{O}$ then $\exists \mathcal{I}$: $(C \sqcap \neg D)^{\mathcal{I}} \neq \emptyset$. Then $C^{\mathcal{I}} \not\subseteq D^{\mathcal{I}}$. $\square$

Concept Subsumption $\Rightarrow$ Concept Instance

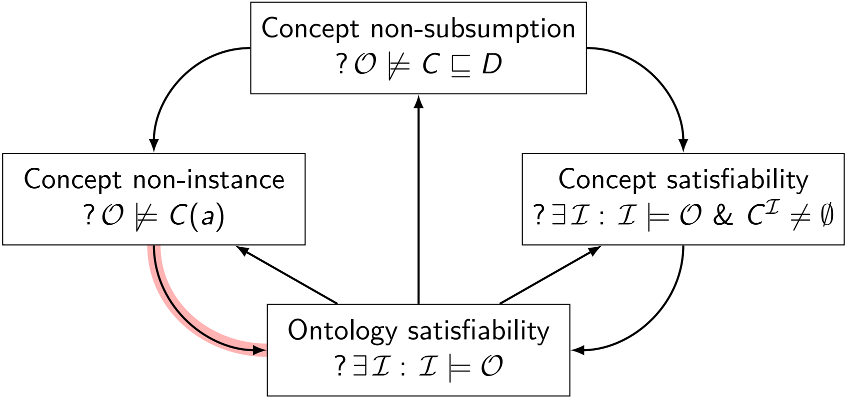


Concept Subsumption $\Rightarrow$ Concept Instance

Lemma

$\mathcal{O} \models C \sqsubseteq D$ iff $\mathcal{O} \cup \{C(a)\} \models D(a)$ for every a not appearing in $\mathcal{O}$.

Concept Non-Instance ⇒ Ontology Satisfiability

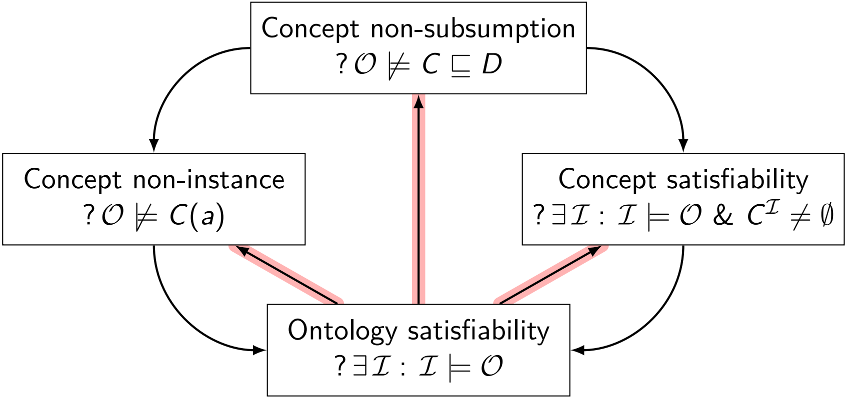


Concept Non-Instance $\Rightarrow$ Ontology Satisfiability

Lemma

$\mathcal{O} \not\models C(a)$ iff $\mathcal{O} \cup \{(\neg C)(a)\}$ is satisfiable.

Ontology Satisfiability $\Rightarrow$ Everything Else



Ontology Satisfiability $\Rightarrow$ Everything Else

Lemma

Then the following conditions are equivalent:

1. $\mathcal{O}$ *is unsatisfiable,*
2. $\top$ *is unsatisfiable w.r.t.* $\mathcal{O}$,
3. $\mathcal{O} \models \top \sqsubseteq \bot$,
4. $\mathcal{O} \models (\bot)(a)$ *for every* a,
5. $\mathcal{O} \models (\bot)(a)$ *for some* a.

Ontology Satisfiability $\Rightarrow$ Everything Else

Corollary

All standard reasoning problems are reducible to each other in polynomial time.

Outline

Description Logics

Tableau Procedures

- Deciding Concept Satisfiability
- Correctness of the Tableau Procedure
- Termination and Complexity Analysis
- Tableau with TBoxes
- Blocking

Axiom Pinpointing

Conclusions

Outline

Tableau

- We first focus first on pure concept satisfiability
  - Given: C a concept
  - Return: yes if C is satisfiable, and no otherwise

Tableau

- We first focus first on pure concept satisfiability
  - Given: C a concept
  - Return: yes if C is satisfiable, and no otherwise
- To check satisfiability of C, we build a tableau, which represents an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$

Tableau

- We first focus first on pure concept satisfiability
  - Given: C a concept
  - Return: yes if C is satisfiable, and no otherwise
- To check satisfiability of C, we build a tableau, which represents an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A tableau is a directed labeled graph $T = (V, E, L)$ (most commonly, a tree) in which:

Tableau

- We first focus first on pure concept satisfiability
  - Given: C a concept
  - Return: yes if C is satisfiable, and no otherwise
- To check satisfiability of C, we build a tableau, which represents an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A tableau is a directed labeled graph $T = (V, E, L)$ (most commonly, a tree) in which:
  - nodes V represent domain elements,

Tableau

- We first focus first on pure concept satisfiability
  - Given: C a concept
  - Return: yes if C is satisfiable, and no otherwise
- To check satisfiability of C, we build a tableau, which represents an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A tableau is a directed labeled graph $T = (V, E, L)$ (most commonly, a tree) in which:
  - nodes V represent domain elements,
  - edges E represent pairs from role interpretations.

Tableau

- We first focus first on pure concept satisfiability
  - Given: C a concept
  - Return: yes if C is satisfiable, and no otherwise
- To check satisfiability of C, we build a tableau, which represents an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- A tableau is a directed labeled graph $T = (V, E, L)$ (most commonly, a tree) in which:
  - nodes V represent domain elements,
  - edges E represent pairs from role interpretations.
  - labeling function L assigns:

    to each $v \in V \quad \mapsto \quad$ a set of concepts $L(v)$

    to each $e \in E \quad \mapsto \quad$ a set of roles $L(e)$

Tableau: Example

Interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$:

- $\Delta^{\mathcal{I}} = \{a, b\}$
- $Child^{\mathcal{I}} = \{a, b\}$
- $Dog^{\mathcal{I}} = \{b\}$
- $likes^{\mathcal{I}} = \{(a, b), (b, b)\}$

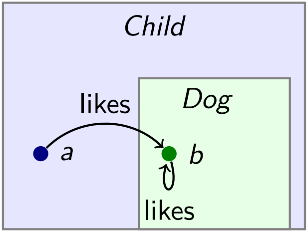


Tableau: Example

Interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$:

- $\Delta^{\mathcal{I}} = \{a, b\}$
- $Child^{\mathcal{I}} = \{a, b\}$
- $Dog^{\mathcal{I}} = \{b\}$
- $likes^{\mathcal{I}} = \{(a, b), (b, b)\}$

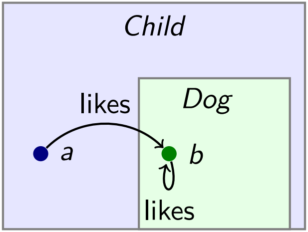
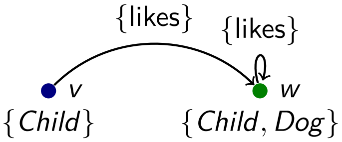


Tableau $T = (V, E, L)$:

- $V = \{v, w\}$
- $E = \{\langle v, w\rangle, \langle w, w\rangle\}$
- $L(v) = \{Child\}$
- $L(w) = \{Child, Dog\}$
- $L\langle v, w\rangle = L\langle w, w\rangle = \{likes\}$



Negation Normal Form

Before constructing a tableau for a concept C, it is first converted into a suitable normal form

Negation Normal Form

Before constructing a tableau for a concept C, it is first converted into a suitable normal form

Definition

C is in Negation Normal Form (short NNF) if $\neg$ in C appears only in front of atomic concepts.

Negation Normal Form

Before constructing a tableau for a concept C, it is first converted into a suitable normal form

Definition

C is in Negation Normal Form (short NNF) if $\neg$ in C appears only in front of atomic concepts.

- Example: in NNF: $\forall R.(\neg A \sqcup \exists S.\neg B)$
  not in NNF: $\neg\exists R.A, \quad \forall R.\neg(A \sqcap B), \quad A \sqcap \exists R.\neg\top$

Negation Normal Form

Before constructing a tableau for a concept C, it is first converted into a suitable normal form

Definition

C is in Negation Normal Form (short NNF) if $\neg$ in C appears only in front of atomic concepts.

- Example: in NNF: $\forall R.(\neg A \sqcup \exists S.\neg B)$
  not in NNF: $\neg\exists R.A$, $\forall R.\neg(A \sqcap B)$, $A \sqcap \exists R.\neg\top$
- Transformation to NNF: "pushing negation inwards":
  - $\neg(C \sqcap D) \quad \Rightarrow \quad (\neg C) \sqcup (\neg D)$
  - $\neg(C \sqcup D) \quad \Rightarrow \quad (\neg C) \sqcap (\neg D)$
  - $\neg(\exists R.C) \quad \Rightarrow \quad \forall R.(\neg C)$
  - $\neg(\forall R.C) \quad \Rightarrow \quad \exists R.(\neg C)$
  - $\neg\neg C \quad \Rightarrow \quad C$
  - $\neg\top \quad \Rightarrow \quad \bot$, $\neg\bot \quad \Rightarrow \quad \top$

Tableau Expansion Rules

- To check satisfiability of a concept C in NNF, create a node x and set $L(x) = \{C\}$. We call it tableau initialization rule.

$x \bullet C$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

Tableau Expansion Rules

- To check satisfiability of a concept C in NNF, create a node x and set $L(x) = \{C\}$. We call it tableau initialization rule.

$$x \bullet C$$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$$v \bullet \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$$

Tableau Expansion Rules

- $\sqcap$-Rule:
  if $(A \sqcap B) \in L(x)$ and $\{A, B\} \not\subseteq L(x)$
  then update $L(x) := L(x) \cup \{A, B\}$

$x \bullet A \sqcap B \quad \rightsquigarrow \quad x \bullet A \sqcap B, A, B$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$\exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$v \bullet$

Tableau Expansion Rules

- $\sqcap$-Rule:
  if $(A \sqcap B) \in L(x)$ and $\{A, B\} \not\subseteq L(x)$
  then update $L(x) := L(x) \cup \{A, B\}$

$x \bullet A \sqcap B \quad \rightsquigarrow \quad x \bullet A \sqcap B, A, B$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$v \bullet \exists R.A \sqcap (\forall R.(\neg A) \sqcup B), \exists R.A, \forall R.(\neg A) \sqcup B$

Tableau Expansion Rules

- $\sqcup$-Rule:
  if $(A \sqcup B) \in L(x)$ and $\{A, B\} \cap L(x) = \emptyset$
  then update $L(x) := L(x) \cup \{A\}$ or $L(x) := L(x) \cup \{B\}$

$x \bullet A \sqcup B \quad \rightsquigarrow \quad x \bullet A \sqcup B, A$

$x \bullet A \sqcup B \quad \rightsquigarrow \quad x \bullet A \sqcup B, B$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$v \bullet \quad \exists R.A \sqcap (\forall R.(\neg A) \sqcup B), \; \exists R.A, \; \forall R.(\neg A) \sqcup B$

Tableau Expansion Rules

- $\sqcup$-Rule:
  if $(A \sqcup B) \in L(x)$ and $\{A, B\} \cap L(x) = \emptyset$
  then update $L(x) := L(x) \cup \{A\}$ or $L(x) := L(x) \cup \{B\}$

$x \bullet A \sqcup B$ $\rightsquigarrow$ $x \bullet A \sqcup B, A$

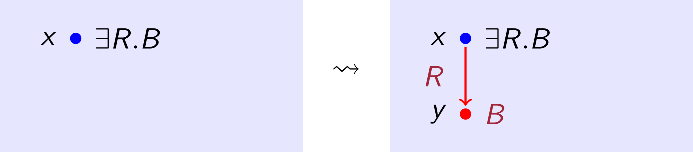
$x \bullet A \sqcup B$ $\rightsquigarrow$ $x \bullet A \sqcup B, B$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$v \bullet$ $\exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$, $\exists R.A$, $\forall R.(\neg A) \sqcup B$, $\forall R.(\neg A)$

Tableau Expansion Rules

- $\exists$-Rule:
  if $(\exists R.B) \in L(x)$ and $B \notin L(y)$ for all y with $R \in L\langle x, y\rangle$
  then create a new y and set $L\langle x, y\rangle := \{R\}$ and $L(y) := \{B\}$



Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

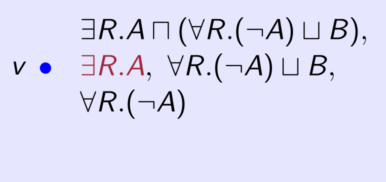
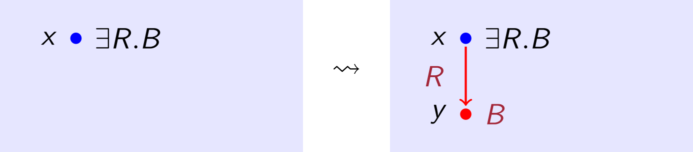


Tableau Expansion Rules

- $\exists$-Rule:
  if $(\exists R.B) \in L(x)$ and $B \notin L(y)$ for all y with $R \in L\langle x, y\rangle$
  then create a new y and set $L\langle x, y\rangle := \{R\}$ and $L(y) := \{B\}$



Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

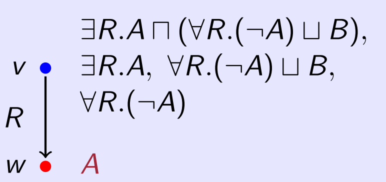
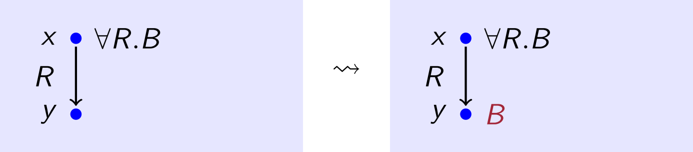


Tableau Expansion Rules

- $\forall$-Rule:
  if $(\forall R.B) \in L(x)$ and $R \in L\langle x, y\rangle$, $B \notin L(y)$ for some $y \in V$
  then update $L(y) := L(y) \cup \{B\}$



Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

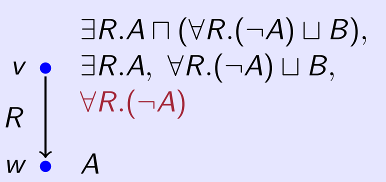


Tableau Expansion Rules

- $\forall$-Rule:
  if $(\forall R.B) \in L(x)$ and $R \in L\langle x, y\rangle$, $B \notin L(y)$ for some $y \in V$
  then update $L(y) := L(y) \cup \{B\}$

$$\begin{tikzcd} x \arrow[d, "R"'] & \forall R.B \\ y & \end{tikzcd} \quad \rightsquigarrow \quad \begin{tikzcd} x \arrow[d, "R"'] & \forall R.B \\ y & B \end{tikzcd}$$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$$\begin{tikzcd} v \arrow[d, "R"'] & \exists R.A \sqcap (\forall R.(\neg A) \sqcup B),\ \exists R.A,\ \forall R.(\neg A) \sqcup B,\ \forall R.(\neg A) \\ w & A,\ \neg A \end{tikzcd}$$

Tableau Expansion Rules

- $\bot$-Rule:
  if $\{A, \neg A\} \subseteq L(x)$ and $\bot \notin L(x)$
  then update $L(x) := L(x) \cup \{\bot\}$

$x \bullet A, \neg A \quad \rightsquigarrow \quad x \bullet A, \neg A, \bot$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$$
\begin{array}{lcl}
v & \bullet & \exists R.A \sqcap (\forall R.(\neg A) \sqcup B), \\
 & & \exists R.A,\ \forall R.(\neg A) \sqcup B, \\
 & & \forall R.(\neg A) \\
R & \downarrow & \\
w & \bullet & A,\ \neg A
\end{array}
$$

Tableau Expansion Rules

- $\bot$-Rule:
  if $\{A, \neg A\} \subseteq L(x)$ and $\bot \notin L(x)$
  then update $L(x) := L(x) \cup \{\bot\}$

$$x \bullet\ A,\ \neg A \quad \rightsquigarrow \quad x \bullet\ A,\ \neg A,\ \bot$$

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

$$\begin{array}{ll} v \bullet & \exists R.A \sqcap (\forall R.(\neg A) \sqcup B),\ \exists R.A,\ \forall R.(\neg A) \sqcup B,\ \forall R.(\neg A) \\ \downarrow R & \\ w \bullet & A,\ \neg A,\ \bot \end{array}$$

Tableau Expansion Rules

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$ (two tableau expansions)

v: $\exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$, $\exists R.A$, $\forall R.(\neg A) \sqcup B$, $\forall R.(\neg A)$

$v \xrightarrow{R} w$

w: A, $\neg A$, $\perp$

v: $\exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$, $\exists R.A$, $\forall R.(\neg A) \sqcup B$, B

$v \xrightarrow{R} w$

w: A

Outline

Completeness of Tableau

Definition

A tableau $T = (V, E, L)$ contains a clash if $\perp \in L(x)$ for some node $x \in V$. A tableau is clash-free if it does not contain a clash.

Completeness of Tableau

Definition

A tableau $T = (V, E, L)$ contains a clash if $\perp \in L(x)$ for some node $x \in V$. A tableau is clash-free if it does not contain a clash.

Theorem (Completeness)

If an $\mathcal{ALC}$ concept C is satisfiable then the tableau rules can be always applied in such a way that a clash is never produced.

Completeness of Tableau

Definition

A tableau $T = (V, E, L)$ contains a clash if $\bot \in L(x)$ for some node $x \in V$. A tableau is clash-free if it does not contain a clash.

Theorem (Completeness)

If an $\mathcal{ALC}$ concept C is satisfiable then the tableau rules can be always applied in such a way that a clash is never produced.

Proof Idea.

Use a model of C to guide construction of the tableau. □

Completeness: Proof Idea

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

Model $\mathcal{I}$:

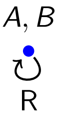
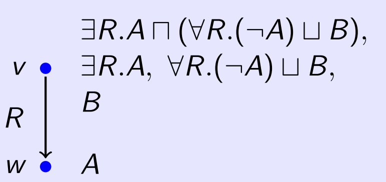


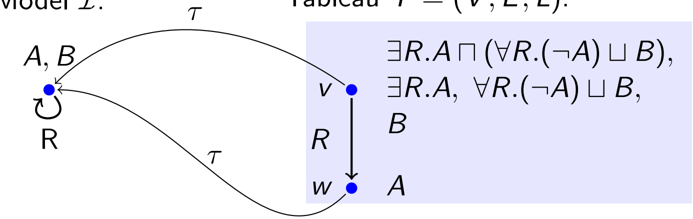
Tableau $T = (V, E, L)$:



Completeness: Proof Idea

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

Model $\mathcal{I}$: Tableau $T = (V, E, L)$:

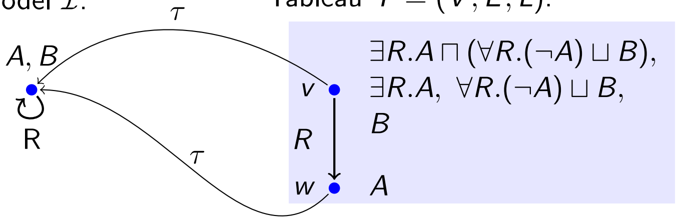


- For every created node x we assign the corresponding element $\tau(x) \in \Delta^{\mathcal{I}}$ of the model $\mathcal{I}$.

Completeness: Proof Idea

Example: $C = \exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$

Model $\mathcal{I}$: Tableau $T = (V, E, L)$:



- For every created node x we assign the corresponding element $\tau(x) \in \Delta^{\mathcal{I}}$ of the model $\mathcal{I}$.
- We can always apply the rules such that:
  1. if $D \in L(x)$ then $\tau(x) \in D^{\mathcal{I}}$, and
  2. if $R \in L(x, y)$ then $\langle \tau(x), \tau(y) \rangle \in R^{\mathcal{I}}$.

Soundness of Tableau

Theorem (Soundness)

If there exists a clash-free tableau $T = (V, E, L)$ *such that* $C \in L(v)$ *for some* $v \in V$*, then* C *is satisfiable.*

Soundness of Tableau

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable.

Soundness of Tableau

Definition

A tableau is fully expanded if all expansion rules are applied to every node.

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable.

Soundness of Tableau

Definition

A tableau is fully expanded if all expansion rules are applied to every node.

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ *such that* $C \in L(v)$ *for some* $v \in V$*, then* C *is satisfiable.*

Proof Idea.

Build a model from the tableau. □

Soundness: Proof Idea

Tableau $T = (V, E, L)$:

$$\begin{tikzcd} v \arrow[d, "R"'] & \exists R.A \sqcap (\forall R.(\neg A) \sqcup B), \\ & \exists R.A,\ \forall R.(\neg A) \sqcup B, \\ & B \\ w & A \end{tikzcd}$$

Soundness: Proof Idea

Tableau $T = (V, E, L)$:

v: $\exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$, $\exists R.A$, $\forall R.(\neg A) \sqcup B$, B

$$v \xrightarrow{R} w$$

w: A

Model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$:

- $\Delta^{\mathcal{I}} = \{v, w\}$
- $A^{\mathcal{I}} = \{w\}$
- $B^{\mathcal{I}} = \{v\}$
- $R^{\mathcal{I}} = \{\langle v, w\rangle\}$

Soundness: Proof Idea

Tableau $T = (V, E, L)$:

$$
\begin{array}{ll}
v & L(v) = \{\exists R.A \sqcap (\forall R.(\neg A) \sqcup B),\ \exists R.A,\ \forall R.(\neg A) \sqcup B,\ B\} \\
\downarrow R & \\
w & L(w) = \{A\}
\end{array}
$$

Model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$:

- $\Delta^{\mathcal{I}} = \{v, w\}$
- $A^{\mathcal{I}} = \{w\}$
- $B^{\mathcal{I}} = \{v\}$
- $R^{\mathcal{I}} = \{\langle v, w\rangle\}$

- The model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is defined from $T = (V, E, L)$ by:
  - $\Delta^{\mathcal{I}} = V$
  - $A^{\mathcal{I}} = \{x \mid A \in L(x)\}$
  - $R^{\mathcal{I}} = \{\langle x, y\rangle \mid R \in L(x, y)\}$

Soundness: Proof Idea

Tableau $T = (V, E, L)$:

v: $\exists R.A \sqcap (\forall R.(\neg A) \sqcup B)$, $\exists R.A$, $\forall R.(\neg A) \sqcup B$, B

$v \xrightarrow{R} w$

w: A

Model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$:

- $\Delta^{\mathcal{I}} = \{v, w\}$
- $A^{\mathcal{I}} = \{w\}$
- $B^{\mathcal{I}} = \{v\}$
- $R^{\mathcal{I}} = \{\langle v, w\rangle\}$

- The model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is defined from $T = (V, E, L)$ by:
  - $\Delta^{\mathcal{I}} = V$
  - $A^{\mathcal{I}} = \{x \mid A \in L(x)\}$
  - $R^{\mathcal{I}} = \{\langle x, y\rangle \mid R \in L(x, y)\}$
- By structural induction it can be shown that:

$$\boxed{\text{if } D \in L(x) \text{ then } x \in D^{\mathcal{I}}}$$

Outline

Termination

Three possible outcomes of tableau rules application:

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ in this case the concept is satisfiable

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ in this case the concept is satisfiable
2. Every attempt to apply the rules eventually results in a clash.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ in this case the concept is satisfiable
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ in this case the concept is unsatisfiable

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ in this case the concept is satisfiable
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ in this case the concept is unsatisfiable
3.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ in this case the concept is satisfiable
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ in this case the concept is unsatisfiable
3. The rules can be applied forever without producing a clash.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ in this case the concept is satisfiable
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ in this case the concept is unsatisfiable
3. The rules can be applied forever without producing a clash.
   $\Rightarrow$ we will never find out if the concept is satisfiable or not

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ in this case the concept is satisfiable
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ in this case the concept is unsatisfiable
3. The rules can be applied forever without producing a clash.
   $\Rightarrow$ we will never find out if the concept is satisfiable or not

Is outcome 3 possible?

Properties of Tableau Expansion Rules

- $x \bullet C$
- $x \bullet A \sqcap B \rightsquigarrow A, B$ $\quad$ $x \bullet A \sqcup B \rightsquigarrow A \mid B$
- $$\begin{tikzcd} x \bullet \exists R.B \arrow[d, "\rightsquigarrow R"'] \\ y \bullet \rightsquigarrow B \end{tikzcd} \qquad \begin{tikzcd} x \bullet \forall R.B \arrow[d, "R"'] \\ y \bullet \rightsquigarrow B \end{tikzcd}$$
- $x \bullet A, \neg A \rightsquigarrow \perp$

Properties of Tableau Expansion Rules

- $x \bullet C$
- $x \bullet A \sqcap B \rightsquigarrow A, B$ $\qquad$ $x \bullet A \sqcup B \rightsquigarrow A \mid B$
- $\begin{array}{rl} x & \bullet\ \exists R.B \\ \rightsquigarrow R & \downarrow \\ y & \bullet \rightsquigarrow B \end{array}$ $\qquad$ $\begin{array}{rl} x & \bullet\ \forall R.B \\ R & \downarrow \\ y & \bullet \rightsquigarrow B \end{array}$
- $x \bullet A, \neg A \rightsquigarrow \bot$

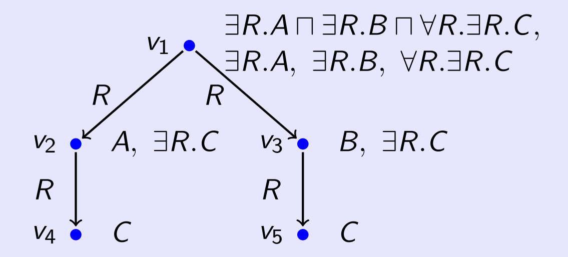
1. Each new concept in the label is a sub-concept of the concept to which the rule is applied, or $\bot$

Properties of Tableau Expansion Rules

- $x \bullet C$
- $x \bullet A \sqcap B \rightsquigarrow A, B$ $\quad$ $x \bullet A \sqcup B \rightsquigarrow A \mid B$
- $\begin{array}{rl} x \bullet & \exists R.B \\ \rightsquigarrow R \downarrow & \\ y \bullet & \rightsquigarrow B \end{array}$ $\quad$ $\begin{array}{rl} x \bullet & \forall R.B \\ R \downarrow & \\ y \bullet & \rightsquigarrow B \end{array}$
- $x \bullet A, \neg A \rightsquigarrow \bot$

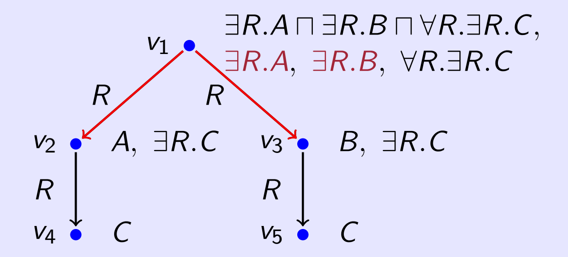
1. Each new concept in the label is a sub-concept of the concept to which the rule is applied, or $\bot$
2. There can be at most one predecessor of every node

Properties of Tableau



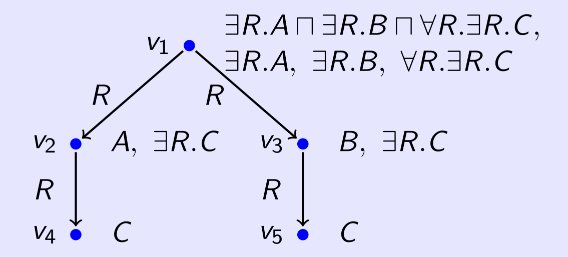
1. Tableau is a tree: each non-root node has a single predecessor

Properties of Tableau



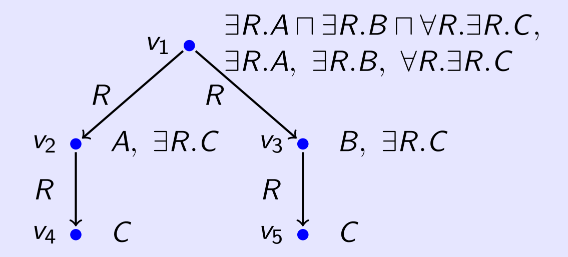
1. Tableau is a tree: each non-root node has a single predecessor
2. The number of node children $\leq$ the number of concepts of the form $\exists R.D$ in the label

Properties of Tableau



1. Tableau is a tree: each non-root node has a single predecessor
2. The number of node children $\leq$ the number of concepts of the form $\exists R.D$ in the label
3. Each concept in the label is a sub-concepts of the original concept (in the root) or $\perp$

Properties of Tableau



1. Tableau is a tree: each non-root node has a single predecessor
2. The number of node children $\leq$ the number of concepts of the form $\exists R.D$ in the label
3. Each concept in the label is a sub-concepts of the original concept (in the root) or $\perp$
4. The depth of the tree is bounded by the maximal quantifier depth of concepts – the maximal number of nested quantifiers

Properties of Tableau

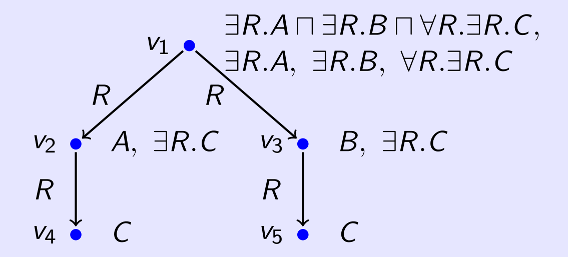


tableau contains $\leq \|C\| \times \|C\|^{qd(C)}$ concepts overall

1. Tableau is a tree: each non-root node has a single predecessor
2. The number of node children $\leq$ the number of concepts of the form $\exists R.D$ in the label
3. Each concept in the label is a sub-concepts of the original concept (in the root) or $\perp$
4. The depth of the tree is bounded by the maximal quantifier depth of concepts – the maximal number of nested quantifiers

Complexity of the Tableau Procedure

- The tableau expansion rules are non-deterministic due to the $\sqcup$-Rule:

$$x \bullet A \sqcup B \rightsquigarrow A \mid B$$

Complexity of the Tableau Procedure

- The tableau expansion rules are non-deterministic due to the $\sqcup$-Rule:

$$x \bullet A \sqcup B \rightsquigarrow A \mid B$$

- Each tableau expansion run terminates in exponential time

Complexity of the Tableau Procedure

- The tableau expansion rules are non-deterministic due to the $\sqcup$-Rule:

  $$x \bullet A \sqcup B \rightsquigarrow A \mid B$$

- Each tableau expansion run terminates in exponential time
- Hence, $\mathcal{ALC}$ concept satisfiability can be decided in non-deterministic exponential time (NExpTime)

Complexity of the Tableau Procedure

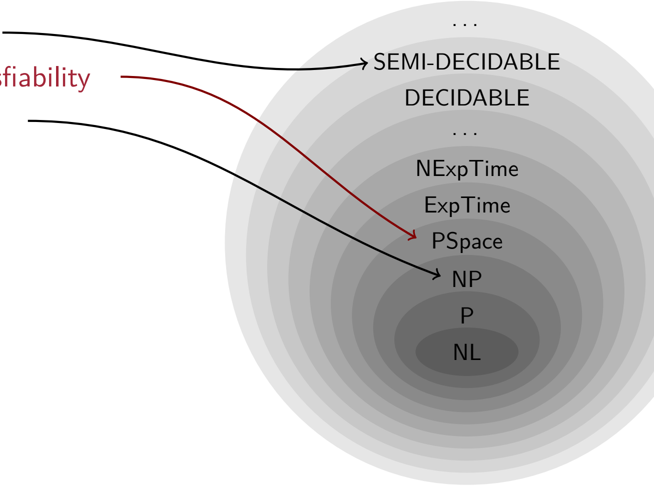
- The tableau expansion rules are non-deterministic due to the $\sqcup$-Rule:

  $$x \bullet A \sqcup B \rightsquigarrow A \mid B$$

- Each tableau expansion run terminates in exponential time
- Hence, $\mathcal{ALC}$ concept satisfiability can be decided in non-deterministic exponential time (NExpTime)
- In fact, it is possible to decide concept satisfiability in polynomial space (PSpace $\subseteq$ ExpTime $\subseteq$ NExpTime)

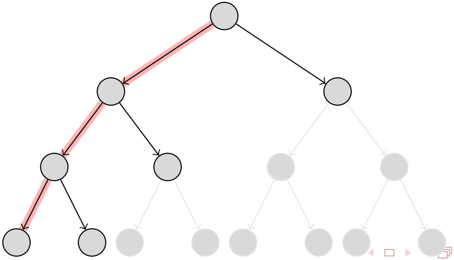
Complexity of the Tableau Procedure

- First-Order Logic
- $\mathcal{ALC}$ Concept Satisfiability
- Propositional Logic



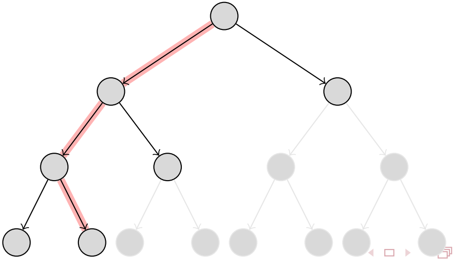
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



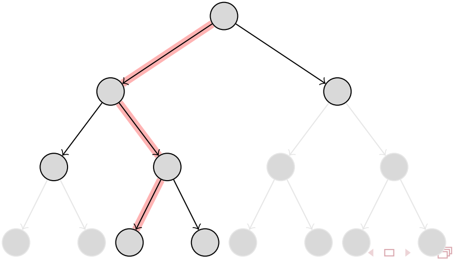
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



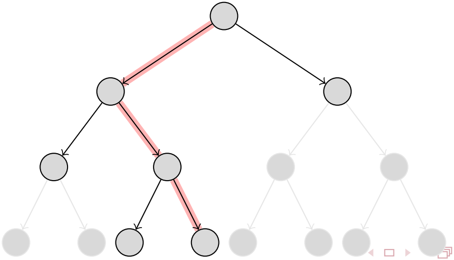
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



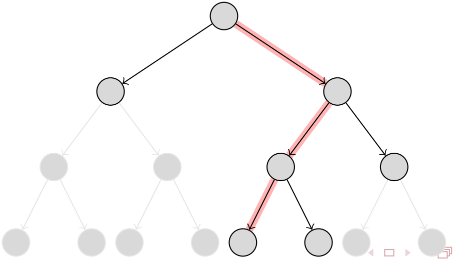
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



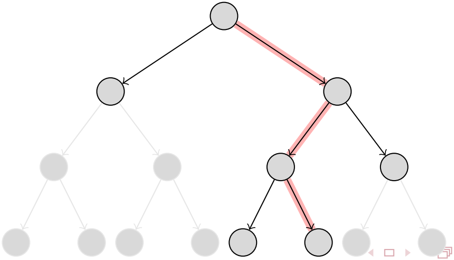
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



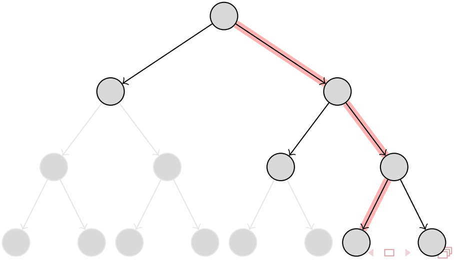
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



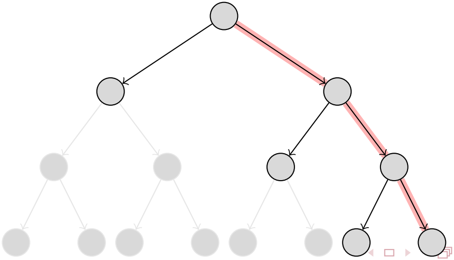
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



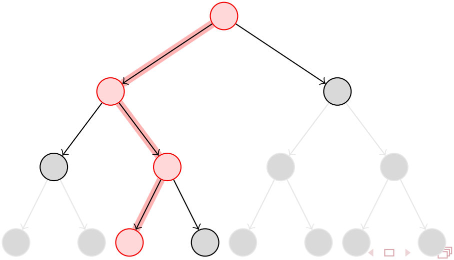
PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.



PSpace Tableau Procedure

- Expand the tableau depth-first and keep only one branch in memory.
- Once all nodes on a branch are fully expanded, nothing new can be added to these nodes anymore.



Outline

Concept Satisfiability w.r.t. TBox Axioms

- Our goal is to extend the tableau procedure so that we can test satisfiability of a concept C w.r.t. an ontology $\mathcal{O}$

Concept Satisfiability w.r.t. TBox Axioms

- Our goal is to extend the tableau procedure so that we can test satisfiability of a concept C w.r.t. an ontology $\mathcal{O}$
- Recall that C is satisfiable w.r.t. $\mathcal{O}$ if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$

Concept Satisfiability w.r.t. TBox Axioms

- Our goal is to extend the tableau procedure so that we can test satisfiability of a concept C w.r.t. an ontology $\mathcal{O}$
- Recall that C is satisfiable w.r.t. $\mathcal{O}$ if there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$ and $\mathcal{I} \models \mathcal{O}$
- For simplicity, assume that $\mathcal{O}$ contains only of TBox axioms:
  - concept inclusions $C \sqsubseteq D$
  - concept equivalences $C \equiv D$

Normal Form of TBox Axioms

- As for concepts, we first need to normalize TBox axioms
  - convert to the form $\top \sqsubseteq C$ where C is in NNF

Normal Form of TBox Axioms

- As for concepts, we first need to normalize TBox axioms
  - convert to the form $\top \sqsubseteq C$ where C is in NNF
- The conversion is easy:

Normal Form of TBox Axioms

- As for concepts, we first need to normalize TBox axioms
  - convert to the form $\top \sqsubseteq C$ where C is in NNF
- The conversion is easy:

  1. $C \equiv D \quad \rightsquigarrow \quad C \sqsubseteq D, \quad D \sqsubseteq C$

Normal Form of TBox Axioms

- As for concepts, we first need to normalize TBox axioms
  - convert to the form $\top \sqsubseteq C$ where C is in NNF
- The conversion is easy:
  1. $C \equiv D \quad \rightsquigarrow \quad C \sqsubseteq D, \quad D \sqsubseteq C$
  2. $C \sqsubseteq D \quad \rightsquigarrow \quad \top \sqsubseteq \neg C \sqcup D$

Normal Form of TBox Axioms

- As for concepts, we first need to normalize TBox axioms
  - convert to the form $\top \sqsubseteq C$ where C is in NNF
- The conversion is easy:
  1. $C \equiv D \quad \rightsquigarrow \quad C \sqsubseteq D, \quad D \sqsubseteq C$
  2. $C \sqsubseteq D \quad \rightsquigarrow \quad \top \sqsubseteq \neg C \sqcup D$
  3. $\top \sqsubseteq C \quad \rightsquigarrow \quad \top \sqsubseteq NNF(C)$

Normal Form of TBox Axioms

- As for concepts, we first need to normalize TBox axioms
  - convert to the form $\top \sqsubseteq C$ where C is in NNF
- The conversion is easy:
  1. $C \equiv D \quad \rightsquigarrow \quad C \sqsubseteq D, \quad D \sqsubseteq C$
  2. $C \sqsubseteq D \quad \rightsquigarrow \quad \top \sqsubseteq \neg C \sqcup D$
  3. $\top \sqsubseteq C \quad \rightsquigarrow \quad \top \sqsubseteq NNF(C)$
- Note that $\mathcal{I} \models \top \sqsubseteq C$ if and only if $C^{\mathcal{I}} = \Delta^{\mathcal{I}}$.
  - or $d \in C^{\mathcal{I}}$ for every $d \in \Delta^{\mathcal{I}}$

Tableau Rules for TBox Axioms

To take TBox axioms into account, we need to add one more rule:

$\sqcap$-Rule: if $(A \sqcap B) \in L(x)$ and $\{A, B\} \not\subseteq L(x)$
then update $L(x) := L(x) \cup \{A, B\}$

$\sqcup$-Rule: if $(A \sqcup B) \in L(x)$ and $\{A, B\} \cap L(x) = \emptyset$
then update $L(x) := L(x) \cup \{A\}$ or $L(x) := L(x) \cup \{B\}$

$\exists$-Rule: if $(\exists R.B) \in L(x)$ and $B \notin L(y)$ for all y with $R \in L(x, y)$
then create a new y and set $L(x, y) := \{R\}$ and $L(y) := \{B\}$

$\forall$-Rule: if $(\forall R.B) \in L(x)$ and $R \in L(x, y)$, $B \notin L(y)$ for some $y \in V$
then update $L(y) := L(y) \cup \{B\}$

$\bot$-Rule: if $\{A, \neg A\} \subseteq L(x)$ and $\bot \notin L(x)$
then update $L(x) := L(x) \cup \{\bot\}$

$\boxed{\top\text{-Rule:}}$ if $\top \sqsubseteq C \in \mathcal{O}$ and $C \notin L(x)$
then update $L(x) := L(x) \cup \{C\}$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:

  $A \sqcap \forall R.B \sqsubseteq \exists R.A$

  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$

  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - tableau initialization: $L(v) := \{A\}$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - tableau initialization: $L(v) := \{A\}$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v) := L(v) \cup \{(\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A\}$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v) := L(v) \cup \{(\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A\}$

$A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$

v •

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v) := L(v) \cup \{\neg A \sqcup \exists R.\neg B\}$

$A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$

v

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v) := L(v) \cup \{\neg A \sqcup \exists R.\neg B\}$

$A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$

$v \bullet \quad \neg A \sqcup \exists R.\neg B$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v) := L(v) \cup \{\exists R.\neg B\}$

$A, \ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$

v • $\neg A \sqcup \exists R.\neg B$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v) := L(v) \cup \{\exists R.\neg B\}$

v • $A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
$\neg A \sqcup \exists R.\neg B,\ \exists R.\neg B$

Example

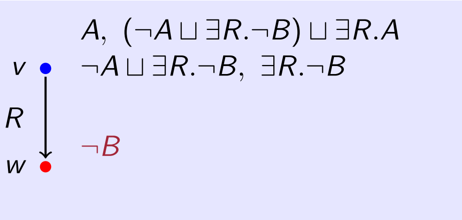
- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\exists$-Rule: $L(v, w) := \{R\}$, $L(w) := \{\neg B\}$

$A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$

v • $\neg A \sqcup \exists R.\neg B,\ \exists R.\neg B$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\exists$-Rule: $L(v, w) := \{R\}$, $L(w) := \{\neg B\}$



Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:

  $A \sqcap \forall R.B \sqsubseteq \exists R.A$

  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$

  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(w) := L(w) \cup \{(\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A\}$

$$\begin{array}{ll} & A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A \\ v & \neg A \sqcup \exists R.\neg B,\ \exists R.\neg B \\ \downarrow R & \\ w & \neg B \end{array}$$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(w) := L(w) \cup \{(\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A\}$

$$\begin{array}{ll} & A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A \\ v & \neg A \sqcup \exists R.\neg B,\ \exists R.\neg B \\ \Big\downarrow R & \\ w & \neg B,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A \end{array}$$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:

$$
\begin{aligned}
&A \sqcap \forall R.B \sqsubseteq \exists R.A \\
&\quad\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A \\
&\quad\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A
\end{aligned}
$$

- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(w) := L(w) \cup \{\neg A \sqcup \exists R.\neg B\}$

v: $A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$, $\neg A \sqcup \exists R.\neg B,\ \exists R.\neg B$

$v \xrightarrow{R} w$

w: $\neg B,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(w) := L(w) \cup \{\neg A \sqcup \exists R.\neg B\}$

v: $A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
$\neg A \sqcup \exists R.\neg B,\ \exists R.\neg B$

$v \xrightarrow{R} w$

w: $\neg B,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
$\neg A \sqcup \exists R.\neg B$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\rightsquigarrow \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(w) := L(w) \cup \{\neg A\}$

v: $A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
$\neg A \sqcup \exists R.\neg B,\ \exists R.\neg B$

$v \xrightarrow{R} w$

w: $\neg B,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
$\neg A \sqcup \exists R.\neg B$

Example

- Consider $\mathcal{O} = \{A \sqcap \forall R.B \sqsubseteq \exists R.A\}$.
- Normalization:
  $A \sqcap \forall R.B \sqsubseteq \exists R.A$
  $\leadsto \quad \top \sqsubseteq \neg(A \sqcap \forall R.B) \sqcup \exists R.A$
  $\leadsto \quad \top \sqsubseteq (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:

v: $A,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
$\neg A \sqcup \exists R.\neg B,\ \exists R.\neg B$

$v \xrightarrow{R} w$

w: $\neg B,\ (\neg A \sqcup \exists R.\neg B) \sqcup \exists R.A$
$\neg A \sqcup \exists R.\neg B,\ \neg A$

Completeness of Tableau for TBoxes

Theorem (Completeness)

If an $\mathcal{ALC}$ concept C is satisfiable w.r.t. $\mathcal{O}$ then the tableau rules can be always applied in such a way that a clash is never produced.

Completeness of Tableau for TBoxes

Theorem (Completeness)

If an $\mathcal{ALC}$ concept C is satisfiable w.r.t. $\mathcal{O}$ then the tableau rules can be always applied in such a way that a clash is never produced.

Proof.

As before, we build the tableau $T = (V, E, L)$ by applying the rules to mimic $\mathcal{I} \models \mathcal{O}$ such that $C^{\mathcal{I}} \neq \emptyset$. $\square$

Soundness of Tableau for TBoxes

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ *such that* $C \in L(v)$ *for some* $v \in V$*, then* C *is satisfiable w.r.t.* $\mathcal{O}$.

Soundness of Tableau for TBoxes

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable w.r.t. $\mathcal{O}$.

Proof.

As in the case without $\mathcal{O}$, we define an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ from $T = (V, E, L)$ with $\Delta^{\mathcal{I}} = V$ and prove that:

if $D \in L(x)$ then $x \in D^{\mathcal{I}}$

This implies that $C^{\mathcal{I}} \neq \emptyset$.

Soundness of Tableau for TBoxes

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable w.r.t. $\mathcal{O}$.

Proof.

As in the case without $\mathcal{O}$, we define an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ from $T = (V, E, L)$ with $\Delta^\mathcal{I} = V$ and prove that:

if $D \in L(x)$ then $x \in D^\mathcal{I}$

This implies that $C^\mathcal{I} \neq \emptyset$. Now we also prove that $\mathcal{I} \models \mathcal{O}$:

Soundness of Tableau for TBoxes

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable w.r.t. $\mathcal{O}$.

Proof.

As in the case without $\mathcal{O}$, we define an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ from $T = (V, E, L)$ with $\Delta^{\mathcal{I}} = V$ and prove that:

$$\boxed{\text{if } D \in L(x) \text{ then } x \in D^{\mathcal{I}}}$$

This implies that $C^{\mathcal{I}} \neq \emptyset$. Now we also prove that $\mathcal{I} \models \mathcal{O}$:

Take any $\top \sqsubseteq D \in \mathcal{O}$ and $x \in \top^{\mathcal{I}} = \Delta^{\mathcal{I}} = V$.

Soundness of Tableau for TBoxes

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable w.r.t. $\mathcal{O}$.

Proof.

As in the case without $\mathcal{O}$, we define an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ from $T = (V, E, L)$ with $\Delta^{\mathcal{I}} = V$ and prove that:

$$\boxed{\text{if } D \in L(x) \text{ then } x \in D^{\mathcal{I}}}$$

This implies that $C^{\mathcal{I}} \neq \emptyset$. Now we also prove that $\mathcal{I} \models \mathcal{O}$:

Take any $\top \sqsubseteq D \in \mathcal{O}$ and $x \in \top^{\mathcal{I}} = \Delta^{\mathcal{I}} = V$.
Then $D \in L(x)$ because $\top$-Rule is applied to x.

Soundness of Tableau for TBoxes

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable w.r.t. $\mathcal{O}$.

Proof.

As in the case without $\mathcal{O}$, we define an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ from $T = (V, E, L)$ with $\Delta^{\mathcal{I}} = V$ and prove that:

$$\boxed{\text{if } D \in L(x) \text{ then } x \in D^{\mathcal{I}}}$$

This implies that $C^{\mathcal{I}} \neq \emptyset$. Now we also prove that $\mathcal{I} \models \mathcal{O}$:

Take any $\top \sqsubseteq D \in \mathcal{O}$ and $x \in \top^{\mathcal{I}} = \Delta^{\mathcal{I}} = V$.
Then $D \in L(x)$ because $\top$-Rule is applied to x.
Then $x \in D^{\mathcal{I}}$.

Soundness of Tableau for TBoxes

Theorem (Soundness)

If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable w.r.t. $\mathcal{O}$.

Proof.

As in the case without $\mathcal{O}$, we define an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ from $T = (V, E, L)$ with $\Delta^{\mathcal{I}} = V$ and prove that:

$$\boxed{\text{if } D \in L(x) \text{ then } x \in D^{\mathcal{I}}}$$

This implies that $C^{\mathcal{I}} \neq \emptyset$. Now we also prove that $\mathcal{I} \models \mathcal{O}$:

Take any $\top \sqsubseteq D \in \mathcal{O}$ and $x \in \top^{\mathcal{I}} = \Delta^{\mathcal{I}} = V$.
Then $D \in L(x)$ because $\top$-Rule is applied to x.
Then $x \in D^{\mathcal{I}}$. Hence $\mathcal{I} \models \top \sqsubseteq D$. $\square$

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ In this case the concept is satisfiable.
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ In this case the concept is unsatisfiable.
3. The rules can be applied forever without producing a clash.
   $\Rightarrow$ We will never find out if the concept is satisfiable or not.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ In this case the concept is satisfiable.
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ In this case the concept is unsatisfiable.
3. The rules can be applied forever without producing a clash.
   $\Rightarrow$ We will never find out if the concept is satisfiable or not.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ In this case the concept is satisfiable.
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ In this case the concept is unsatisfiable.
3. The rules can be applied forever without producing a clash.
   $\Rightarrow$ We will never find out if the concept is satisfiable or not.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ In this case the concept is satisfiable.
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ In this case the concept is unsatisfiable.
3. The rules can be applied forever without producing a clash.
   $\Rightarrow$ We will never find out if the concept is satisfiable or not.

Termination

Three possible outcomes of tableau rules application:

1. Tableau can be fully expanded without producing clash.
   $\Rightarrow$ In this case the concept is satisfiable.
2. Every attempt to apply the rules eventually results in a clash.
   $\Rightarrow$ In this case the concept is unsatisfiable.
3. The rules can be applied forever without producing a clash.
   $\Rightarrow$ We will never find out if the concept is satisfiable or not.

Unfortunately, with TBoxes, the outcome 3 becomes possible!

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - tableau initialization: $L(v_0) := \{A\}$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - tableau initialization: $L(v_0) := \{A\}$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v_0) := L(v_0) \cup \{\neg A \sqcup \exists R.A\}$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v_0) := L(v_0) \cup \{\neg A \sqcup \exists R.A\}$

v_0 • $A,\ \neg A \sqcup \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v_0) := L(v_0) \cup \{\exists R.A\}$

v_0 • A, $\neg A \sqcup \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v_0) := L(v_0) \cup \{\exists R.A\}$

v_0 • $A, \neg A \sqcup \exists R.A, \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\exists$-Rule: $L\langle v_0, v_1\rangle := \{R\}$, $L(v_1) := \{A\}$

v_0 $A, \neg A \sqcup \exists R.A, \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\exists$-Rule: $L\langle v_0, v_1\rangle := \{R\}$, $L(v_1) := \{A\}$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \leadsto \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v_1) := L(v_1) \cup \{\neg A \sqcup \exists R.A\}$

$$\begin{tikzcd} v_0 \bullet \arrow[d, "R"'] & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ v_1 \bullet & A \end{tikzcd}$$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v_1) := L(v_1) \cup \{\neg A \sqcup \exists R.A\}$

$$\begin{array}{lcl} v_0 & \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ R & \downarrow & \\ v_1 & \bullet & A,\ \neg A \sqcup \exists R.A \end{array}$$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v_1) := L(v_1) \cup \{\exists R.A\}$

v_0 $A,\ \neg A \sqcup \exists R.A,\ \exists R.A$

R $\downarrow$

v_1 $A,\ \neg A \sqcup \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v_1) := L(v_1) \cup \{\exists R.A\}$

$$\begin{array}{lll} v_0 & \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ R & \downarrow & \\ v_1 & \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \end{array}$$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\exists$-Rule: $L\langle v_1, v_2\rangle := \{R\}$, $L(v_2) := \{A\}$

$$\begin{array}{ll} v_0 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ R \big\downarrow & \\ v_1 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \end{array}$$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\exists$-Rule: $L\langle v_1, v_2\rangle := \{R\}$, $L(v_2) := \{A\}$

$$\begin{array}{lll} v_0 & \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ R & \downarrow & \\ v_1 & \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ R & \downarrow & \\ v_2 & \bullet & A \end{array}$$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v_2) := L(v_2) \cup \{\neg A \sqcup \exists R.A\}$

v_0 $A,\ \neg A \sqcup \exists R.A,\ \exists R.A$

$\downarrow R$

v_1 $A,\ \neg A \sqcup \exists R.A,\ \exists R.A$

$\downarrow R$

v_2 A

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\top$-Rule: $L(v_2) := L(v_2) \cup \{\neg A \sqcup \exists R.A\}$

v_0 $A,\ \neg A \sqcup \exists R.A,\ \exists R.A$

$\downarrow R$

v_1 $A,\ \neg A \sqcup \exists R.A,\ \exists R.A$

$\downarrow R$

v_2 $A,\ \neg A \sqcup \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v_2) := L(v_2) \cup \{\exists R.A\}$

v_0 $A,\ \neg A \sqcup \exists R.A,\ \exists R.A$

$\downarrow R$

v_1 $A,\ \neg A \sqcup \exists R.A,\ \exists R.A$

$\downarrow R$

v_2 $A,\ \neg A \sqcup \exists R.A$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\sqcup$-Rule: $L(v_2) := L(v_2) \cup \{\exists R.A\}$

$$
\begin{array}{ll}
v_0 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
\downarrow R & \\
v_1 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
\downarrow R & \\
v_2 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A
\end{array}
$$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
  - $\exists$-Rule: ...

$$\begin{array}{ll} v_0 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ \downarrow R & \\ v_1 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ \downarrow R & \\ v_2 & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ \vdots & \end{array}$$

Non-Termination: Example

- Consider $\mathcal{O} = \{A \sqsubseteq \exists R.A\}$
- Normalization:
  $A \sqsubseteq \exists R.A \quad \rightsquigarrow \quad \top \sqsubseteq \neg A \sqcup \exists R.A$
- Let's check satisfiability of A w.r.t. $\mathcal{O}$:
- This process can continue forever!

$$
\begin{tikzcd}
v_0 \arrow[d, "R"'] & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
v_1 \arrow[d, "R"'] & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
v_2 \arrow[d, dotted] & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
{} &
\end{tikzcd}
$$

Outline

Blocking

- Notice that the labels of the nodes repeat:

$$
\begin{tikzcd}
v_0 \arrow[d, "R"'] & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
v_1 \arrow[d, dotted] & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
{}
\end{tikzcd}
$$

Blocking

- Notice that the labels of the nodes repeat:

$$\begin{array}{ll} v_0 \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ R \downarrow & \\ v_1 \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ \vdots & \end{array}$$

- We can block further expansion for such repetitions

Blocking

- Notice that the labels of the nodes repeat:

$$
\begin{array}{ll}
v_0 \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
R \downarrow & \\
v_1 \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\
\vdots &
\end{array}
$$

- We can block further expansion for such repetitions

Definition (Blocking)

A node $v \in V$ is blocked if there exists an ancestor node $w \in V$ of v such that $L(v) \subseteq L(w)$. (We say that v is blocked by w.)

Blocking

- Notice that the labels of the nodes repeat:

$$\begin{array}{lll} v_0 & \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ R & \downarrow & \\ v_1 & \bullet & A,\ \neg A \sqcup \exists R.A,\ \exists R.A \\ & \vdots & \end{array}$$

- We can block further expansion for such repetitions

Definition (Blocking)

A node $v \in V$ is blocked if there exists an ancestor node $w \in V$ of v such that $L(v) \subseteq L(w)$. (We say that v is blocked by w.)

- Above v_1 is blocked by v_0, but v_0 is not blocked by v_1

Soundness with Blocking

Definition

A tableau is fully expanded if all expansion rules are applied to every non-blocked node.

Soundness with Blocking

Definition

A tableau is fully expanded if all expansion rules are applied to every non-blocked node.

Theorem (Soundness)

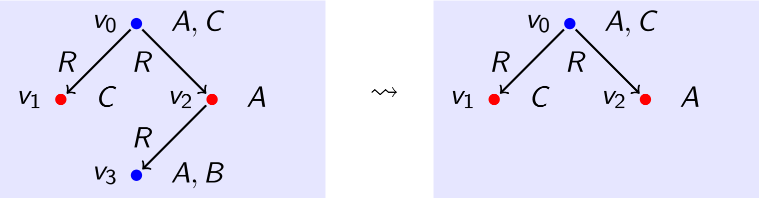
If there exists a clash-free fully expanded tableau $T = (V, E, L)$ such that $C \in L(v)$ for some $v \in V$, then C is satisfiable w.r.t. $\mathcal{O}$.

Direct Blocking and Blocking

- We can extend the blocking condition to prevent further unnecessary rule applications:

Direct Blocking and Blocking

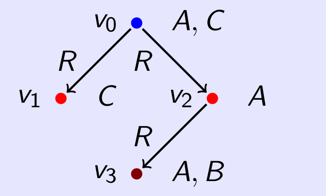
- We can extend the blocking condition to prevent further unnecessary rule applications:
- Example:



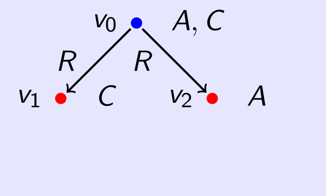
  - v_3 is not blocked, but it is a descendant of a blocked node v_2
  - no rules need to be applied as v_3 can be simply removed

Direct Blocking and Blocking

- We can extend the blocking condition to prevent further unnecessary rule applications:
- Example:



$\rightsquigarrow$

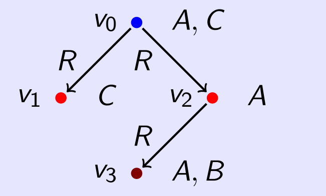


Definition (Direct Blocking, Blocking)

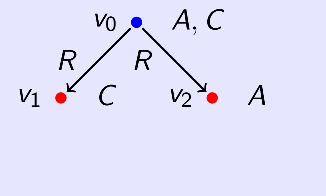
A node $v \in V$ is directly blocked if there exists an ancestor node $w \in V$ of v such that $L(v) \subseteq L(w)$.

Direct Blocking and Blocking

- We can extend the blocking condition to prevent further unnecessary rule applications:
- Example:



$\rightsquigarrow$



Definition (Direct Blocking, Blocking)

A node $v \in V$ is directly blocked if there exists an ancestor node $w \in V$ of v such that $L(v) \subseteq L(w)$.

A node $v \in V$ is blocked if it is either directly blocked or one of its ancestor nodes is directly blocked.

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$
  - The number of different subsets of these concepts is 2^n

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$
  - The number of different subsets of these concepts is 2^n
  - So, if a path in a tableau contains more than 2^n nodes, then at least one node on this path is directly blocked

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$
  - The number of different subsets of these concepts is 2^n
  - So, if a path in a tableau contains more than 2^n nodes, then at least one node on this path is directly blocked
  - Hence, all tableau nodes with level $2^n + 1$ must be blocked

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$
  - The number of different subsets of these concepts is 2^n
  - So, if a path in a tableau contains more than 2^n nodes, then at least one node on this path is directly blocked
  - Hence, all tableau nodes with level $2^n + 1$ must be blocked
  - Thus, we can never create a node with level $2^n + 2$

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$
  - The number of different subsets of these concepts is 2^n
  - So, if a path in a tableau contains more than 2^n nodes, then at least one node on this path is directly blocked
  - Hence, all tableau nodes with level $2^n + 1$ must be blocked
  - Thus, we can never create a node with level $2^n + 2$
  - So, the depth of the tableau is always bounded by $2^n + 1$

Complexity

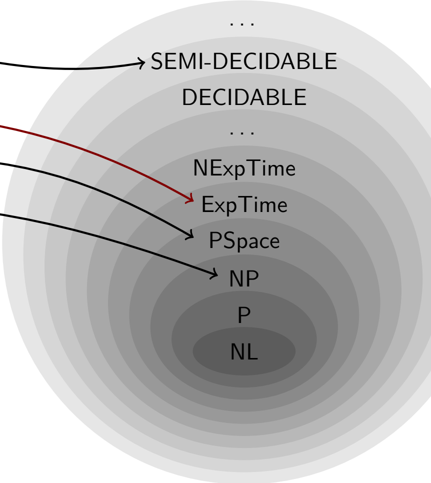
- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$
  - The number of different subsets of these concepts is 2^n
  - So, if a path in a tableau contains more than 2^n nodes, then at least one node on this path is directly blocked
  - Hence, all tableau nodes with level $2^n + 1$ must be blocked
  - Thus, we can never create a node with level $2^n + 2$
  - So, the depth of the tableau is always bounded by $2^n + 1$
  - So, the maximal size of the tableau is n^{2^n} – doubly exponential

Complexity

- What is the size of the largest clash-free tableau one can obtain without applying the rules to blocked nodes?
  - Let n be the number of sub-concepts occurring in C or $\mathcal{O}$
  - The number of different subsets of these concepts is 2^n
  - So, if a path in a tableau contains more than 2^n nodes, then at least one node on this path is directly blocked
  - Hence, all tableau nodes with level $2^n + 1$ must be blocked
  - Thus, we can never create a node with level $2^n + 2$
  - So, the depth of the tableau is always bounded by $2^n + 1$
  - So, the maximal size of the tableau is n^{2^n} – doubly exponential
- However, the optimal complexity for $\mathcal{ALC}$ concept satisfiability w.r.t. TBoxes, is "only" ExpTime.

Complexity

- First-Order Logic
- $\mathcal{ALC}$ Concept Satisfiability w.r.t. TBox
- $\mathcal{ALC}$ Concept Satisfiability
- Propositional Logic



Outline

Description Logics

Tableau Procedures

Axiom Pinpointing

- Justifications
- The Reiter's Hitting Set Tree Algorithm
- Axiom Pinpointing using SAT Solvers

Conclusions

Outline

Motivation

- Reasoning algorithms (such as tableau) can detect the presence of modeling errors: answer yes or no

$$\mathcal{O} \overset{?}{\models} \top \sqsubseteq \bot$$

Motivation

- Reasoning algorithms (such as tableau) can detect the presence of modeling errors: answer yes or no
- How to determine what causes the error?
  - existing ontologies contain hundreds of thousands of axioms
  - an inconsistency is rarely caused by more than a few axioms

...

1254. $\mathit{Parent} \equiv \exists \mathit{hasChild}.\top$

...

2456. $\mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}$

...

7312. $(\mathit{GrandParent} \sqcap \neg \mathit{Parent})(\mathit{John})$

...

Motivation

- Reasoning algorithms (such as tableau) can detect the presence of modeling errors: answer yes or no
- How to determine what causes the error?
  - existing ontologies contain hundreds of thousands of axioms
  - an inconsistency is rarely caused by more than a few axioms
- The axiom pinpointing algorithms can be used to narrow down the axioms responsible for the error
  - using a series of entailment tests

$$
\begin{aligned}
&\dots\\
1254.\ & \mathit{Parent} \equiv \exists \mathit{hasChild}.\top\\
&\dots\\
2456.\ & \mathit{GrandParent} \equiv \exists \mathit{hasChild}.\mathit{Parent}\\
&\dots\\
7312.\ & (\mathit{GrandParent} \sqcap \neg \mathit{Parent})(\mathit{John})\\
&\dots
\end{aligned}
$$

Justifications

- A justification for an entailment $\mathcal{O} \models \alpha$ is a a minimal subset of axioms $J \subseteq \mathcal{O}$ such that $J \models \alpha$
  - minimal means that for every $J' \subsetneq J$, we have $J' \not\models \alpha$.

Justifications

- A justification for an entailment $\mathcal{O} \models \alpha$ is a a minimal subset of axioms $J \subseteq \mathcal{O}$ such that $J \models \alpha$
  - minimal means that for every $J' \subsetneq J$, we have $J' \not\models \alpha$.
- Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Justifications:

Justifications

- A justification for an entailment $\mathcal{O} \models \alpha$ is a a minimal subset of axioms $J \subseteq \mathcal{O}$ such that $J \models \alpha$
  - minimal means that for every $J' \subsetneq J$, we have $J' \not\models \alpha$.
- Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Justifications:
  - $J_1 = \{A \sqsubseteq B,\ B \sqsubseteq C\}$,

Justifications

- A justification for an entailment $\mathcal{O} \models \alpha$ is a a minimal subset of axioms $J \subseteq \mathcal{O}$ such that $J \models \alpha$
  - minimal means that for every $J' \subsetneq J$, we have $J' \not\models \alpha$.
- Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Justifications:
  - $J_1 = \{A \sqsubseteq B,\ B \sqsubseteq C\}$,
  - $J_2 = \{A \sqsubseteq C\}$,

Justifications

- A justification for an entailment $\mathcal{O} \models \alpha$ is a a minimal subset of axioms $J \subseteq \mathcal{O}$ such that $J \models \alpha$
  - minimal means that for every $J' \subsetneq J$, we have $J' \not\models \alpha$.
- Example: $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Justifications:
  - $J_1 = \{A \sqsubseteq B,\ B \sqsubseteq C\}$,
  - $J_2 = \{A \sqsubseteq C\}$,
  - $J_3 = \{A \sqsubseteq B, A \sqcap B \sqsubseteq \bot\}$.

The Number of Justifications

- How many justifications an entailment may have?
  - there can be exponentially-many justifications!

The Number of Justifications

- How many justifications an entailment may have?
  - there can be exponentially-many justifications!
- Example:
  $\{A_0 \sqsubseteq B \sqcap A_1,\ A_1 \sqsubseteq B \sqcap A_2, \ldots, A_{n-1} \sqsubseteq B \sqcap A_n,$
  $A_0 \sqsubseteq C \sqcap A_1,\ A_1 \sqsubseteq C \sqcap A_2, \ldots, A_{n-1} \sqsubseteq C \sqcap A_n\} \models A_0 \sqsubseteq A_n$

The Number of Justifications

- How many justifications an entailment may have?
  - there can be exponentially-many justifications!
- Example:
  $\{A_0 \sqsubseteq B \sqcap A_1, A_1 \sqsubseteq B \sqcap A_2, \ldots, A_{n-1} \sqsubseteq B \sqcap A_n,$
  $A_0 \sqsubseteq C \sqcap A_1, A_1 \sqsubseteq C \sqcap A_2, \ldots, A_{n-1} \sqsubseteq C \sqcap A_n\} \models A_0 \sqsubseteq A_n$
- There are 2^n justifications:
  - for every i choose either $A_{i-1} \sqsubseteq B \sqcap A_i$ or $A_{i-1} \sqsubseteq C \sqcap A_i$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\}$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\}$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \qquad A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{A \sqcap B \sqsubseteq \bot\}$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \qquad A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{A \sqcap B \sqsubseteq \bot\} \not\models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{A \sqsubseteq C\}$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
  - $\{A \sqsubseteq C\} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad A \sqsubseteq C \quad \} \models A \sqsubseteq C$

Computing One Justification

- How to find a justification?
  - remove axioms one by one so long the entailment still holds
- Example: $\{ \quad A \sqsubseteq C \quad \} \models A \sqsubseteq C$
- Result: $J_2 = \{A \sqsubseteq C\}$

Computing All Justifications

- Notice that the justification returned by the previous algorithm depends on the order in which axioms are considered.

Computing All Justifications

- Notice that the justification returned by the previous algorithm depends on the order in which axioms are considered.
- Example: removal in the reversed order of axioms:

  $\{A \sqcap B \sqsubseteq \bot, A \sqsubseteq C, B \sqsubseteq C, A \sqsubseteq B\} \models A \sqsubseteq C$

  Gives: $J_1 = \{B \sqsubseteq C, A \sqsubseteq B\}$

Computing All Justifications

- Notice that the justification returned by the previous algorithm depends on the order in which axioms are considered.
- Example: removal in the reversed order of axioms:

  $\{A \sqcap B \sqsubseteq \bot, A \sqsubseteq C, B \sqsubseteq C, A \sqsubseteq B\} \models A \sqsubseteq C$

  Gives: $J_1 = \{B \sqsubseteq C, A \sqsubseteq B\}$
- To compute all justifications it is sufficient to consider all permutations of axioms
  - each justification comes at the end of some permutation

Computing All Justifications

- Notice that the justification returned by the previous algorithm depends on the order in which axioms are considered.
- Example: removal in the reversed order of axioms:

  $\{A \sqcap B \sqsubseteq \bot, A \sqsubseteq C, B \sqsubseteq C, A \sqsubseteq B\} \models A \sqsubseteq C$

  Gives: $J_1 = \{B \sqsubseteq C, A \sqsubseteq B\}$
- To compute all justifications it is sufficient to consider all permutations of axioms
  - each justification comes at the end of some permutation
- The number of permutations of n axioms is $n! \leq 2^{n^2}$
  $\Rightarrow$ algorithmically optimal, but not practical
  - previously computed justifications are ignored

Computing All Justifications

- Notice that the justification returned by the previous algorithm depends on the order in which axioms are considered.
- Example: removal in the reversed order of axioms:

  $\{A \sqcap B \sqsubseteq \bot, A \sqsubseteq C, B \sqsubseteq C, A \sqsubseteq B\} \models A \sqsubseteq C$

  Gives: $J_1 = \{B \sqsubseteq C, A \sqsubseteq B\}$
- To compute all justifications it is sufficient to consider all permutations of axioms
  - each justification comes at the end of some permutation
- The number of permutations of n axioms is $n! \leq 2^{n^2}$
  $\Rightarrow$ algorithmically optimal, but not practical
  - previously computed justifications are ignored
- Next we describe a more goal-directed algorithm

Outline

Description Logics

Tableau Procedures

Axiom Pinpointing

- Justifications
- The Reiter's Hitting Set Tree Algorithm
- Axiom Pinpointing using SAT Solvers

Conclusions

Computing a New Justification

- Suppose we have computed justifications $J_1, \ldots, J_n$ for $\mathcal{O} \models \alpha$
- How to find a new justification J?
  - J should miss at least one axiom β_i from each J_i $(1 \leq i \leq n)$

Computing a New Justification

- Suppose we have computed justifications $J_1, \ldots, J_n$ for $\mathcal{O} \models \alpha$
- How to find a new justification J?
  - J should miss at least one axiom β_i from each J_i $(1 \leq i \leq n)$
- Solution:
  1. iterate over tuples $\langle \beta_1, \ldots, \beta_n \rangle$ such that $\beta_i \in J_i$ $(1 \leq i \leq n)$
  2. check whether $\mathcal{O} \setminus \{\beta_1, \ldots, \beta_n\} \models \alpha$
  3. if so, extract a minimal $J \subseteq \mathcal{O} \setminus \{\beta_1, \ldots, \beta_n\}$ such that $J \models \alpha$

Computing a New Justification

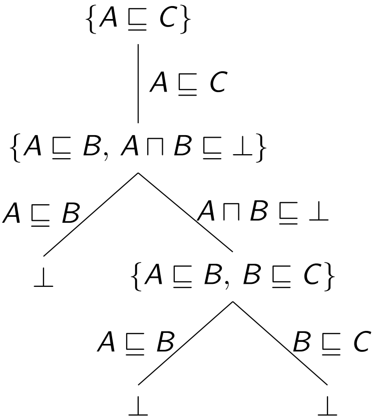
- Suppose we have computed justifications $J_1, \ldots, J_n$ for $\mathcal{O} \models \alpha$
- How to find a new justification J?
  - J should miss at least one axiom β_i from each J_i $(1 \leq i \leq n)$
- Solution:
  1. iterate over tuples $\langle\beta_1, \ldots, \beta_n\rangle$ such that $\beta_i \in J_i$ $(1 \leq i \leq n)$
  2. check whether $\mathcal{O} \setminus \{\beta_1, \ldots, \beta_n\} \models \alpha$
  3. if so, extract a minimal $J \subseteq \mathcal{O} \setminus \{\beta_1, \ldots, \beta_n\}$ such that $J \models \alpha$
- The Hitting Set Tree algorithm (short: HST-algorithm) explores such tuples $\langle\beta_1, \ldots, \beta_n\rangle$ in a systematic way

The Hitting Set Tree

The Hitting Set Tree (short: HS-tree) for $\mathcal{O} \models \alpha$ is a labeled tree such that:

1. Each non-leaf node is labeled by a justification for $\mathcal{O} \models \alpha$
2. Each edge is labeled by an axiom from the justification of the parent
3. Each justification misses all axioms on the path to the root
4. If there is no such a justification, the node is labeled by $\perp$ (leaf)

Example:

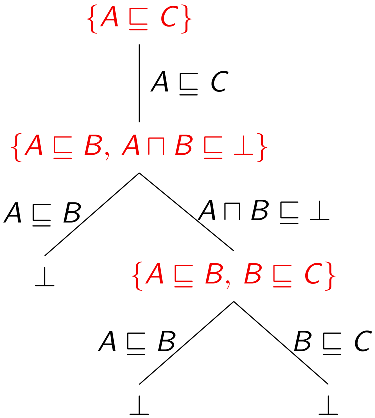


The Hitting Set Tree

The Hitting Set Tree (short: HS-tree) for $\mathcal{O} \models \alpha$ is a labeled tree such that:

1. Each non-leaf node is labeled by a justification for $\mathcal{O} \models \alpha$
2. Each edge is labeled by an axiom from the justification of the parent
3. Each justification misses all axioms on the path to the root
4. If there is no such a justification, the node is labeled by $\perp$ (leaf)

Example:

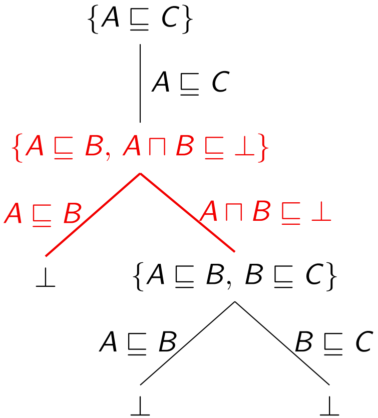


The Hitting Set Tree

The Hitting Set Tree (short: HS-tree) for $\mathcal{O} \models \alpha$ is a labeled tree such that:

1. Each non-leaf node is labeled by a justification for $\mathcal{O} \models \alpha$
2. Each edge is labeled by an axiom from the justification of the parent
3. Each justification misses all axioms on the path to the root
4. If there is no such a justification, the node is labeled by $\perp$ (leaf)

Example:

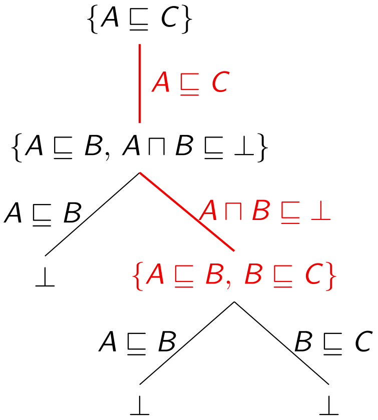


The Hitting Set Tree

The Hitting Set Tree (short: HS-tree) for $\mathcal{O} \models \alpha$ is a labeled tree such that:

1. Each non-leaf node is labeled by a justification for $\mathcal{O} \models \alpha$
2. Each edge is labeled by an axiom from the justification of the parent
3. Each justification misses all axioms on the path to the root
4. If there is no such a justification, the node is labeled by $\perp$ (leaf)

Example:

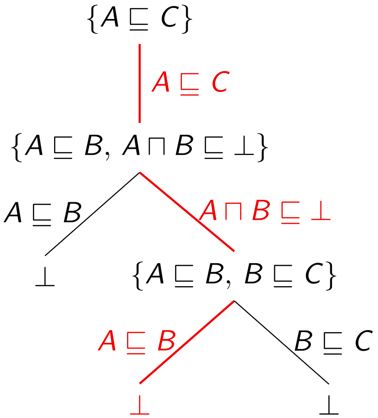


The Hitting Set Tree

The Hitting Set Tree (short: HS-tree) for $\mathcal{O} \models \alpha$ is a labeled tree such that:

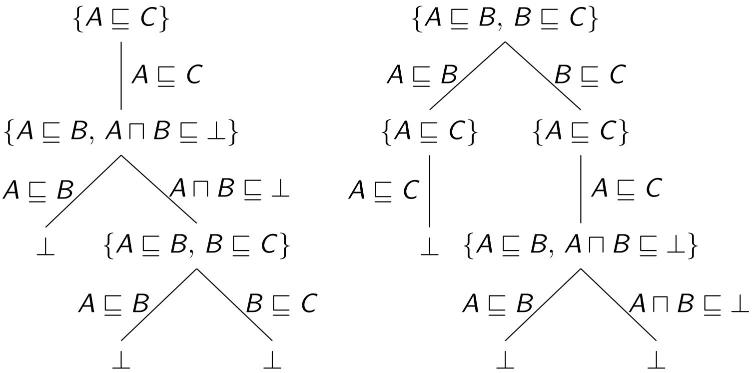
1. Each non-leaf node is labeled by a justification for $\mathcal{O} \models \alpha$
2. Each edge is labeled by an axiom from the justification of the parent
3. Each justification misses all axioms on the path to the root
4. If there is no such a justification, the node is labeled by $\bot$ (leaf)

Example:



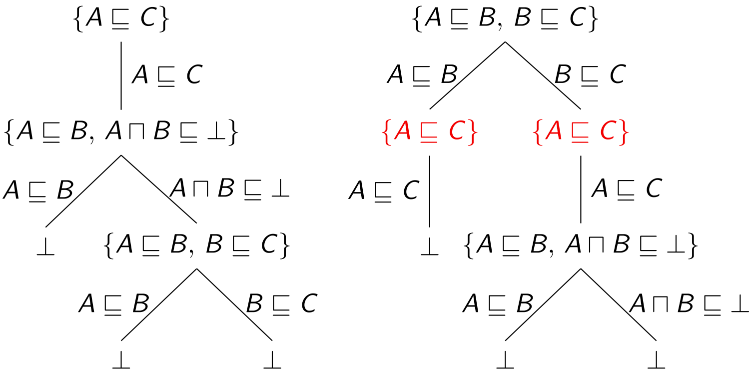
Properties of Hitting Set Trees

1. HS-tree for an entailment $\mathcal{O} \models \alpha$ is not unique:
   - Example: two different HS-trees for our entailment:



Properties of Hitting Set Trees

1. HS-tree for an entailment $\mathcal{O} \models \alpha$ is not unique:
   - Example: two different HS-trees for our entailment:



Note that a HS-tree may contain a justification multiple times!

Properties of Hitting Set Trees

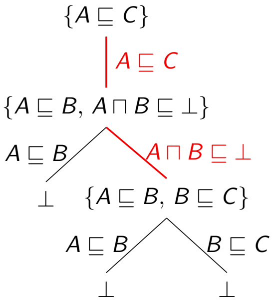
2. Each justification J appears in every HS-tree T at least once

Properties of Hitting Set Trees

2. Each justification J appears in every HS-tree T at least once

Proof Sketch.

- For a node v, let $H(v)$ be the set of the axioms on the path from v to the root of T

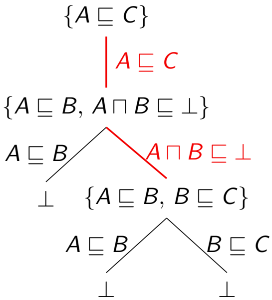


Properties of Hitting Set Trees

2. Each justification J appears in every HS-tree T at least once

Proof Sketch.

- For a node v, let $H(v)$ be the set of the axioms on the path from v to the root of T
- Let v be a node with a maximal $H(v)$ such that $H(v) \cap J = \emptyset$

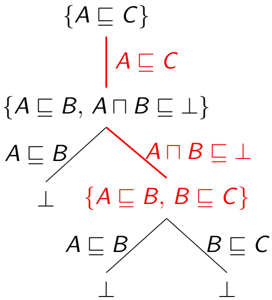


Properties of Hitting Set Trees

2. Each justification J appears in every HS-tree T at least once

Proof Sketch.

- For a node v, let $H(v)$ be the set of the axioms on the path from v to the root of T
- Let v be a node with a maximal $H(v)$ such that $H(v) \cap J = \emptyset$
- We claim tha v is labeled by J!

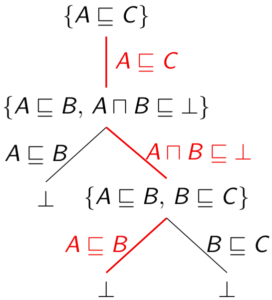


Properties of Hitting Set Trees

2. Each justification J appears in every HS-tree T at least once

Proof Sketch.

- For a node v, let $H(v)$ be the set of the axioms on the path from v to the root of T
- Let v be a node with a maximal $H(v)$ such that $H(v) \cap J = \emptyset$
- We claim tha v is labeled by J!



3. Each HS-tree contains at most exponentially-many nodes
   - every path is labeled by a unique sequence of different axioms

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\perp$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J,
     and create a successor for every $\beta \in J$.

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\perp$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \perp\} \models A \sqsubseteq C$

$\{A \sqsubseteq C\}$

$A \sqsubseteq C$

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

$\{A \sqsubseteq C\}$

$A \sqsubseteq C$

- $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

$\{A \sqsubseteq C\}$

$A \sqsubseteq C$

- $\{A \sqsubseteq B, B \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- $\{A \sqsubseteq B, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

$$\{A \sqsubseteq C\}$$

$$\Big|\ A \sqsubseteq C$$

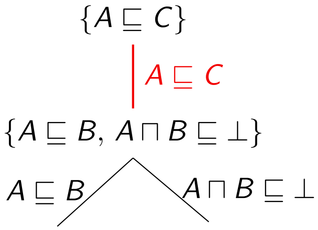
$$\{A \sqsubseteq B,\ A \sqcap B \sqsubseteq \bot\}$$

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$



Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

$\{A \sqsubseteq C\}$

$A \sqsubseteq C$

$\{A \sqsubseteq B,\ A \sqcap B \sqsubseteq \bot\}$

$A \sqsubseteq B$ $\qquad$ $A \sqcap B \sqsubseteq \bot$

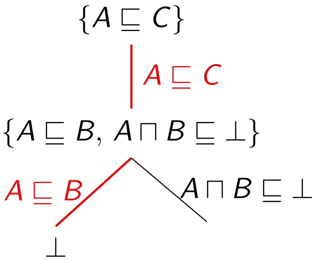
- $\{A \sqcap B \sqsubseteq \bot\} \not\models A \sqsubseteq C$

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

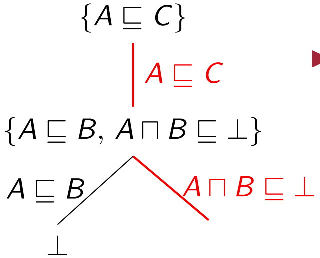


Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$



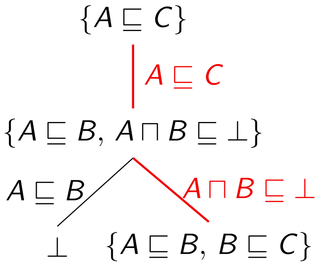
- $\{A \sqsubseteq B,\ B \sqsubseteq C\} \models A \sqsubseteq C$

Construction of a HS-Tree

A HS-Tree for $\mathcal{O} \models \alpha$ can be constructed as follows:

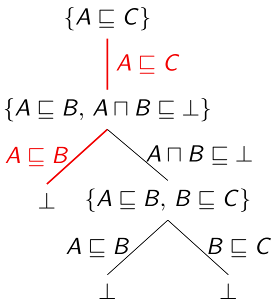
1. Create a root node v_0
2. Repeatedly assign a label to every node v:
   - If $\mathcal{O} \setminus H(v) \not\models \alpha$ then label v by $\bot$.
   - Else, extract a justification $J \subseteq \mathcal{O} \setminus H(v)$ by removing axioms, label v with J, and create a successor for every $\beta \in J$.

Example: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$



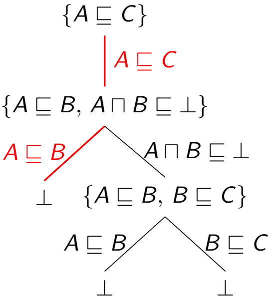
Repairs

- The sets $H(v)$ for leaf nodes are special:
  - $\mathcal{O} \setminus H(v) \not\models \alpha$



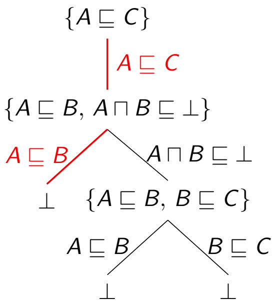
Repairs

- The sets $H(v)$ for leaf nodes are special:
  - $\mathcal{O} \setminus H(v) \not\models \alpha$
- A subset $R \subseteq \mathcal{O}$ is a repair for $\mathcal{O} \models \alpha$ if $\mathcal{O} \setminus R \not\models \alpha$



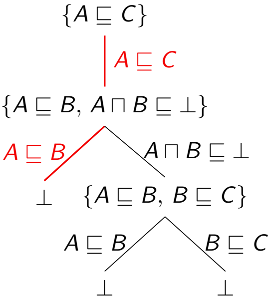
Repairs

- The sets $H(v)$ for leaf nodes are special:
  - $\mathcal{O} \setminus H(v) \not\models \alpha$
- A subset $R \subseteq \mathcal{O}$ is a repair for $\mathcal{O} \models \alpha$ if $\mathcal{O} \setminus R \not\models \alpha$
- The HST-algorithm, therefore, also computes repairs



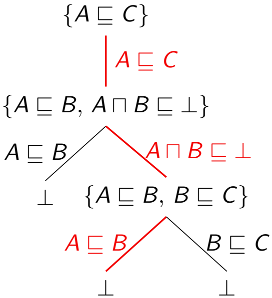
Repairs

- The sets $H(v)$ for leaf nodes are special:
  - $\mathcal{O} \setminus H(v) \not\models \alpha$
- A subset $R \subseteq \mathcal{O}$ is a repair for $\mathcal{O} \models \alpha$ if $\mathcal{O} \setminus R \not\models \alpha$
- The HST-algorithm, therefore, also computes repairs
- Note: not all computed repairs are minimal



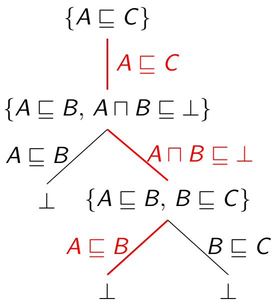
Repairs

- The sets $H(v)$ for leaf nodes are special:
  - $\mathcal{O} \setminus H(v) \not\models \alpha$
- A subset $R \subseteq \mathcal{O}$ is a repair for $\mathcal{O} \models \alpha$ if $\mathcal{O} \setminus R \not\models \alpha$
- The HST-algorithm, therefore, also computes repairs
- Note: not all computed repairs are minimal



Repairs

- The sets $H(v)$ for leaf nodes are special:
  - $\mathcal{O} \setminus H(v) \not\models \alpha$
- A subset $R \subseteq \mathcal{O}$ is a repair for $\mathcal{O} \models \alpha$ if $\mathcal{O} \setminus R \not\models \alpha$
- The HST-algorithm, therefore, also computes repairs
- Note: not all computed repairs are minimal
- However, each HS-tree contains all minimal repairs among $H(v)$



Optimizations

The HST-algorithm can be further optimized

- One branch at a time:
  - it is enough to store only the current branch in memory
  - can be implemented in polynomial space

Optimizations

The HST-algorithm can be further optimized

- One branch at a time:
  - it is enough to store only the current branch in memory
  - can be implemented in polynomial space
- Justification reuse:
  - check if $J \subseteq \mathcal{O} \setminus H(v)$ for the previously computed J
  - can reduce the number of entailment tests for finding J
  - but requires storing all computed justifications

Optimizations

The HST-algorithm can be further optimized

- One branch at a time:
  - it is enough to store only the current branch in memory
  - can be implemented in polynomial space
- Justification reuse:
  - check if $J \subseteq \mathcal{O} \setminus H(v)$ for the previously computed J
  - can reduce the number of entailment tests for finding J
  - but requires storing all computed justifications
- Early pruning:
  - check if $H(v) = H(w)$ for some processed node w
  - no need to expand below the node v (the subtree is identical)
  - however, requires storing all sets $H(v)$

Optimizations

The HST-algorithm can be further optimized

- One branch at a time:
  - it is enough to store only the current branch in memory
  - can be implemented in polynomial space
- Justification reuse:
  - check if $J \subseteq \mathcal{O} \setminus H(v)$ for the previously computed J
  - can reduce the number of entailment tests for finding J
  - but requires storing all computed justifications
- Early pruning:
  - check if $H(v) = H(w)$ for some processed node w
  - no need to expand below the node v (the subtree is identical)
  - however, requires storing all sets $H(v)$

$\Rightarrow$ Flexible trade-off: memory vs speed

Outline

The Hitting Set Duality

- Let J be a justification R a repair for $\mathcal{O} \models \alpha$
  - i.e., $J \models \alpha$ and $\mathcal{O} \setminus R \not\models \alpha$

The Hitting Set Duality

- Let J be a justification R a repair for $\mathcal{O} \models \alpha$
  - i.e., $J \models \alpha$ and $\mathcal{O} \setminus R \not\models \alpha$
- Notice that $J \cap R \neq \emptyset$
  - otherwise, if $R \cap J = \emptyset$ then $J \subseteq \mathcal{O} \setminus R \not\models \alpha$

The Hitting Set Duality

- Let J be a justification R a repair for $\mathcal{O} \models \alpha$
  - i.e., $J \models \alpha$ and $\mathcal{O} \setminus R \not\models \alpha$
- Notice that $J \cap R \neq \emptyset$
  - otherwise, if $R \cap J = \emptyset$ then $J \subseteq \mathcal{O} \setminus R \not\models \alpha$

$\Rightarrow$ The Hitting Set Duality (between justifications and repairs)

The Hitting Set Duality

- Let J be a justification R a repair for $\mathcal{O} \models \alpha$
  - i.e., $J \models \alpha$ and $\mathcal{O} \setminus R \not\models \alpha$
- Notice that $J \cap R \neq \emptyset$
  - otherwise, if $R \cap J = \emptyset$ then $J \subseteq \mathcal{O} \setminus R \not\models \alpha$

$\Rightarrow$ The Hitting Set Duality (between justifications and repairs)

Definition

- Let $P = \{S_1, S_2, \ldots, S_n\}$ be a collection of sets.
- A set H is a hitting set for P if $H \cap S_i \neq \emptyset$ for each i $(1 \leq i \leq n)$.
- A hitting set is minimal if every $H' \subsetneq H$ is not a hitting set for P.

The Hitting Set Duality

- Let J be a justification R a repair for $\mathcal{O} \models \alpha$
  - i.e., $J \models \alpha$ and $\mathcal{O} \setminus R \not\models \alpha$
- Notice that $J \cap R \neq \emptyset$
  - otherwise, if $R \cap J = \emptyset$ then $J \subseteq \mathcal{O} \setminus R \not\models \alpha$

$\Rightarrow$ The Hitting Set Duality (between justifications and repairs)

Definition

- Let $P = \{S_1, S_2, \ldots, S_n\}$ be a collection of sets.
- A set H is a hitting set for P if $H \cap S_i \neq \emptyset$ for each i $(1 \leq i \leq n)$.
- A hitting set is minimal if every $H' \subsetneq H$ is not a hitting set for P.

$\Rightarrow$ Each justification is a minimal hitting set of all repairs

$\Rightarrow$ Each (minimal) repair is a (minimal) hitting set of all justifications

- Gives the name of the HST-algorithm

Finding new Justifications and Repairs

- Assume we have computed some justifications $J_1, \ldots, J_n$ and repairs $R_1, \ldots, R_n$ for $\mathcal{O} \models \alpha$ and want to find more

Finding new Justifications and Repairs

- Assume we have computed some justifications $J_1, \ldots, J_n$ and repairs $R_1, \ldots, R_n$ for $\mathcal{O} \models \alpha$ and want to find more
- A new justification must:

1. miss at least one axiom from each J_i $(1 \leq i \leq n)$
2. contain at least one axiom from each R_j $(1 \leq j \leq m)$

Finding new Justifications and Repairs

- Assume we have computed some justifications $J_1, \ldots, J_n$ and repairs $R_1, \ldots, R_n$ for $\mathcal{O} \models \alpha$ and want to find more
- A new justification must:

1. miss at least one axiom from each J_i $(1 \leq i \leq n)$
2. contain at least one axiom from each R_j $(1 \leq j \leq m)$

- Suppose some $M \subseteq \mathcal{O}$ satisfies these two conditions

Finding new Justifications and Repairs

- Assume we have computed some justifications $J_1, \ldots, J_n$ and repairs $R_1, \ldots, R_n$ for $\mathcal{O} \models \alpha$ and want to find more
- A new justification must:

1. miss at least one axiom from each J_i $(1 \leq i \leq n)$
2. contain at least one axiom from each R_j $(1 \leq j \leq m)$

- Suppose some $M \subseteq \mathcal{O}$ satisfies these two conditions
- If $M \models \alpha$, we can remove axioms to find a justification $J \subseteq M$
  - J will still miss an axiom from each J_i $\Rightarrow$ J is new!

Finding new Justifications and Repairs

- Assume we have computed some justifications $J_1, \ldots, J_n$ and repairs $R_1, \ldots, R_n$ for $\mathcal{O} \models \alpha$ and want to find more
- A new justification must:

1. miss at least one axiom from each J_i $(1 \leq i \leq n)$
2. contain at least one axiom from each R_j $(1 \leq j \leq m)$

- Suppose some $M \subseteq \mathcal{O}$ satisfies these two conditions
- If $M \models \alpha$, we can remove axioms to find a justification $J \subseteq M$
  - J will still miss an axiom from each J_i $\Rightarrow$ J is new!
- If $M \not\models \alpha$ then $R = \mathcal{O} \setminus M$ is a repair!
  - $\mathcal{O} \setminus R = \mathcal{O} \setminus (\mathcal{O} \setminus M) = M \not\models \alpha$
  - R misses one axiom from each R_j (the one in M) $\Rightarrow$ R is new!

Finding new Justifications and Repairs

- Assume we have computed some justifications $J_1, \ldots, J_n$ and repairs $R_1, \ldots, R_n$ for $\mathcal{O} \models \alpha$ and want to find more
- A new justification must:

> 1. miss at least one axiom from each J_i $(1 \leq i \leq n)$
> 2. contain at least one axiom from each R_j $(1 \leq j \leq m)$

- Suppose some $M \subseteq \mathcal{O}$ satisfies these two conditions
- If $M \models \alpha$, we can remove axioms to find a justification $J \subseteq M$
  - J will still miss an axiom from each J_i $\Rightarrow$ J is new!
- If $M \not\models \alpha$ then $R = \mathcal{O} \setminus M$ is a repair!
  - $\mathcal{O} \setminus R = \mathcal{O} \setminus (\mathcal{O} \setminus M) = M \not\models \alpha$
  - R misses one axiom from each R_j (the one in M) $\Rightarrow$ R is new!
- Either way we find a new justification or a new repair

Finding new Justifications and Repairs

- Assume we have computed some justifications $J_1, \ldots, J_n$ and repairs $R_1, \ldots, R_n$ for $\mathcal{O} \models \alpha$ and want to find more
- A new justification must:

1. miss at least one axiom from each J_i $(1 \leq i \leq n)$
2. contain at least one axiom from each R_j $(1 \leq j \leq m)$

- Suppose some $M \subseteq \mathcal{O}$ satisfies these two conditions
- If $M \models \alpha$, we can remove axioms to find a justification $J \subseteq M$
  - J will still miss an axiom from each J_i $\Rightarrow$ J is new!
- If $M \not\models \alpha$ then $R = \mathcal{O} \setminus M$ is a repair!
  - $\mathcal{O} \setminus R = \mathcal{O} \setminus (\mathcal{O} \setminus M) = M \not\models \alpha$
  - R misses one axiom from each R_j (the one in M) $\Rightarrow$ R is new!
- Either way we find a new justification or a new repair
- Question: how to find M satisfying 1 and 2?

SAT Encoding

- The conditions on M can be expressed in Propositional Logic
  - and solved using existing satisfiability (SAT) solvers

SAT Encoding

- The conditions on M can be expressed in Propositional Logic
  - and solved using existing satisfiability (SAT) solvers
- For each axiom $\beta \in \mathcal{O}$ introduce a propositional variable p_β

SAT Encoding

- The conditions on M can be expressed in Propositional Logic
  - and solved using existing satisfiability (SAT) solvers
- For each axiom $\beta \in \mathcal{O}$ introduce a propositional variable p_β
- Goal: find a model $\mathcal{I}$ such that $p_\beta^\mathcal{I} = 1$ iff $\beta \in M$

SAT Encoding

- The conditions on M can be expressed in Propositional Logic
  - and solved using existing satisfiability (SAT) solvers
- For each axiom $\beta \in \mathcal{O}$ introduce a propositional variable p_β
- Goal: find a model $\mathcal{I}$ such that $p_\beta^\mathcal{I} = 1$ iff $\beta \in M$
- Then the conditions can be expressed by the formula:

$$F = \bigwedge_{i=1}^{n} \bigvee_{\beta \in J_i} \neg p_\beta \quad \wedge \quad \bigwedge_{j=1}^{m} \bigvee_{\beta \in R_j} p_\beta$$

SAT Encoding

- The conditions on M can be expressed in Propositional Logic
  - and solved using existing satisfiability (SAT) solvers
- For each axiom $\beta \in \mathcal{O}$ introduce a propositional variable p_β
- Goal: find a model $\mathcal{I}$ such that $p_\beta^\mathcal{I} = 1$ iff $\beta \in M$
- Then the conditions can be expressed by the formula:

$$F = \bigwedge_{i=1}^{n} \bigvee_{\beta \in J_i} \neg p_\beta \quad \wedge \quad \bigwedge_{j=1}^{m} \bigvee_{\beta \in R_j} p_\beta$$

1. M must miss some $\beta \in J_i$ for each i $(1 \leq i \leq n)$

SAT Encoding

- The conditions on M can be expressed in Propositional Logic
  - and solved using existing satisfiability (SAT) solvers
- For each axiom $\beta \in \mathcal{O}$ introduce a propositional variable p_β
- Goal: find a model $\mathcal{I}$ such that $p_\beta^\mathcal{I} = 1$ iff $\beta \in M$
- Then the conditions can be expressed by the formula:

$$F = \bigwedge_{i=1}^{n} \bigvee_{\beta \in J_i} \neg p_\beta \quad \wedge \quad \bigwedge_{j=1}^{m} \bigvee_{\beta \in R_j} p_\beta$$

1. M must miss some $\beta \in J_i$ for each i $(1 \leq i \leq n)$
2. M must contain some $\beta \in R_j$ for each j $(1 \leq j \leq n)$

Example

- Entailment: $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$

Example

- Entailment: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Propositional assignment:
  - $A \sqsubseteq B \rightsquigarrow p_1$,
  - $B \sqsubseteq C \rightsquigarrow p_2$,
  - $A \sqsubseteq C \rightsquigarrow p_3$,
  - $A \sqcap B \sqsubseteq \bot \rightsquigarrow p_4$.

Example

- Entailment: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Propositional assignment:
  - $A \sqsubseteq B \rightsquigarrow p_1$,
  - $B \sqsubseteq C \rightsquigarrow p_2$,
  - $A \sqsubseteq C \rightsquigarrow p_3$,
  - $A \sqcap B \sqsubseteq \bot \rightsquigarrow p_4$.
- Suppose that justifications and repairs found so far are:
  - $J_1 = \{A \sqsubseteq B,\ B \sqsubseteq C\}$, $J_2 = \{A \sqsubseteq C\}$,
  - $R_1 = \{A \sqsubseteq B,\ A \sqsubseteq C\}$

Example

- Entailment: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Propositional assignment:
  - $A \sqsubseteq B \quad \rightsquigarrow p_1$,
  - $B \sqsubseteq C \quad \rightsquigarrow p_2$,
  - $A \sqsubseteq C \quad \rightsquigarrow p_3$,
  - $A \sqcap B \sqsubseteq \bot \rightsquigarrow p_4$.
- Suppose that justifications and repairs found so far are:
  - $J_1 = \{A \sqsubseteq B,\ B \sqsubseteq C\}$, $\quad J_2 = \{A \sqsubseteq C\}$,
  - $R_1 = \{A \sqsubseteq B,\ A \sqsubseteq C\}$
- The resulting formula is: $F = (\neg p_1 \vee \neg p_2) \wedge (\neg p_3) \wedge (p_1 \vee p_3)$

Example

- Entailment: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Propositional assignment:
  - $A \sqsubseteq B \quad \rightsquigarrow p_1,$
  - $B \sqsubseteq C \quad \rightsquigarrow p_2,$
  - $A \sqsubseteq C \quad \rightsquigarrow p_3,$
  - $A \sqcap B \sqsubseteq \bot \rightsquigarrow p_4.$
- Suppose that justifications and repairs found so far are:
  - $J_1 = \{A \sqsubseteq B,\ B \sqsubseteq C\}, \quad J_2 = \{A \sqsubseteq C\},$
  - $R_1 = \{A \sqsubseteq B,\ A \sqsubseteq C\}$
- The resulting formula is: $F = (\neg p_1 \vee \neg p_2) \wedge (\neg p_3) \wedge (p_1 \vee p_3)$
- F has a model $\mathcal{I}$: $p_1^{\mathcal{I}} = 1$ and $p_2^{\mathcal{I}} = p_3^{\mathcal{I}} = p_4^{\mathcal{I}} = 0$

Example

- Entailment: $\{A \sqsubseteq B,\ B \sqsubseteq C,\ A \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Propositional assignment:
  - $A \sqsubseteq B \quad \rightsquigarrow p_1,$
  - $B \sqsubseteq C \quad \rightsquigarrow p_2,$
  - $A \sqsubseteq C \quad \rightsquigarrow p_3,$
  - $A \sqcap B \sqsubseteq \bot \rightsquigarrow p_4.$
- Suppose that justifications and repairs found so far are:
  - $J_1 = \{A \sqsubseteq B,\ B \sqsubseteq C\}, \quad J_2 = \{A \sqsubseteq C\},$
  - $R_1 = \{A \sqsubseteq B,\ A \sqsubseteq C\}$
- The resulting formula is: $F = (\neg p_1 \vee \neg p_2) \wedge (\neg p_3) \wedge (p_1 \vee p_3)$
- F has a model $\mathcal{I}$: $p_1^{\mathcal{I}} = 1$ and $p_2^{\mathcal{I}} = p_3^{\mathcal{I}} = p_4^{\mathcal{I}} = 0$
- $\mathcal{I}$ corresponds to $M = \{A \sqsubseteq B\} \not\models A \sqsubseteq C$

Example

- Entailment: $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\} \models A \sqsubseteq C$
- Propositional assignment:
  - $A \sqsubseteq B \quad \rightsquigarrow p_1$,
  - $B \sqsubseteq C \quad \rightsquigarrow p_2$,
  - $A \sqsubseteq C \quad \rightsquigarrow p_3$,
  - $A \sqcap B \sqsubseteq \bot \rightsquigarrow p_4$.
- Suppose that justifications and repairs found so far are:
  - $J_1 = \{A \sqsubseteq B, B \sqsubseteq C\}, \quad J_2 = \{A \sqsubseteq C\}$,
  - $R_1 = \{A \sqsubseteq B, A \sqsubseteq C\}$
- The resulting formula is: $F = (\neg p_1 \vee \neg p_2) \wedge (\neg p_3) \wedge (p_1 \vee p_3)$
- F has a model $\mathcal{I}$: $p_1^{\mathcal{I}} = 1$ and $p_2^{\mathcal{I}} = p_3^{\mathcal{I}} = p_4^{\mathcal{I}} = 0$
- $\mathcal{I}$ corresponds to $M = \{A \sqsubseteq B\} \not\models A \sqsubseteq C$
- $\mathcal{O} \setminus M = \{B \sqsubseteq C, A \sqsubseteq C, A \sqcap B \sqsubseteq \bot\}$ is a new repair
  - in fact, even a new minimal repair

The SAT-Based Algorithm

1. Set $F = \top$
2. While F is satisfiable do:
   - take any model $\mathcal{I}$ of F
   - define $M = \{\beta \mid p_\beta^\mathcal{I} = 1\}$
   - if $M \models \alpha$ then extract a justification $J \subseteq M$
   - otherwise set $R = \mathcal{O} \setminus M$ (and optionally minimize)
   - update F based on J or R
3. Return all computed justifications J (and / or repairs R)

Comparison of Axiom Pinpointing Methods

	HST	SAT
Repetition of justifications:	may repeat(*)	no repetition
Memory consumption:	polynomial	exponential

(*)There is an example in which justifications repeat exponentially-many times

Outline

Description Logics

Tableau Procedures

Axiom Pinpointing

Conclusions

Description Logics

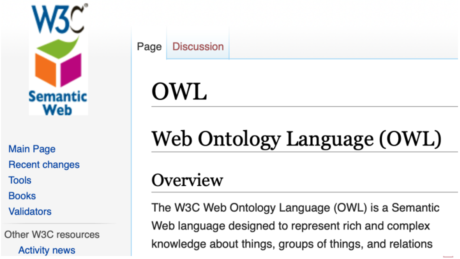
- A family of logic-based languages for knowledge representation
- Distinguished by the well-defined formal semantics

Description Logics

- A family of logic-based languages for knowledge representation
- Distinguished by the well-defined formal semantics
- Exceptional application support:

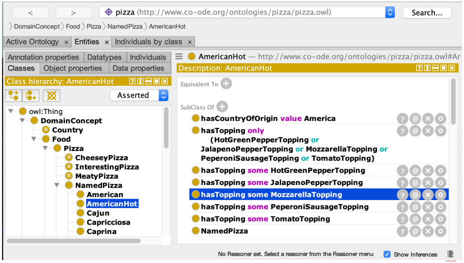
Description Logics

- A family of logic-based languages for knowledge representation
- Distinguished by the well-defined formal semantics
- Exceptional application support:
  - a W3C-standardized language OWL based on DLs



Description Logics

- A family of logic-based languages for knowledge representation
- Distinguished by the well-defined formal semantics
- Exceptional application support:
  - a W3C-standardized language OWL based on DLs
  - ontology editors



Description Logics

- A family of logic-based languages for knowledge representation
- Distinguished by the well-defined formal semantics
- Exceptional application support:
  - a W3C-standardized language OWL based on DLs
  - ontology editors
  - ontology reasoners

FaCT++	CB
HermiT	ConDOR
Konclude	ELK
MoRE	Sequoia
Pellet	
RACER	
TrOWL	

Description Logics

- A family of logic-based languages for knowledge representation
- Distinguished by the well-defined formal semantics
- Exceptional application support:
  - a W3C-standardized language OWL based on DLs
  - ontology editors
  - ontology reasoners
  - ontology repositories

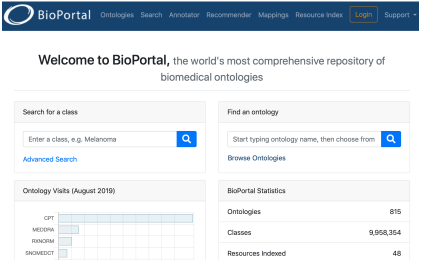


Tableau Procedures

- Main focus: expressivity and efficiency

Tableau Procedures

- Main focus: expressivity and efficiency
- Rely on a (generalized) tree model property

Tableau Procedures

- Main focus: expressivity and efficiency
- Rely on a (generalized) tree model property
- Development effort increases with expressivity:
  - termination requires extensions, such as blocking
  - correctness proofs become complicated
  - theoretical complexity vs. practical efficiency

Tableau Procedures

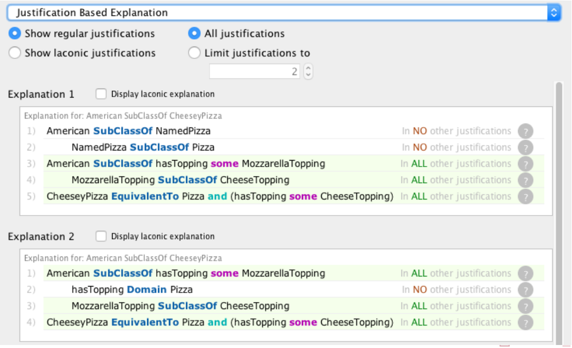
- Main focus: expressivity and efficiency
- Rely on a (generalized) tree model property
- Development effort increases with expressivity:
  - termination requires extensions, such as blocking
  - correctness proofs become complicated
  - theoretical complexity vs. practical efficiency
- Alternative reasoning procedures: consequence-based
  - work by deriving consequences, not building models

Explanations

- Main application: ontology debugging
  - other applications, e.g., inconsistency-tolerant reasoning

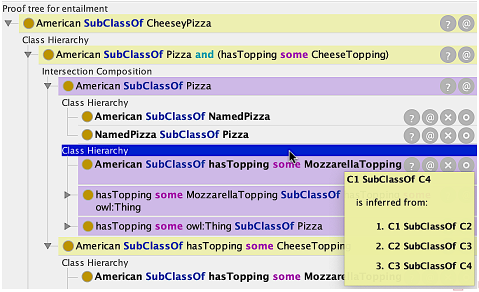
Explanations

- Main application: ontology debugging
  - other applications, e.g., inconsistency-tolerant reasoning
- Implemented in many tools
  - explanation workbench, EL+SAT, EL2MUS, SATPin,. . .



Explanations

- Main application: ontology debugging
  - other applications, e.g., inconsistency-tolerant reasoning
- Implemented in many tools
  - explanation workbench, EL+SAT, EL2MUS, SATPin,...
- Other explanation methods: proof-based explanations
  - show how consequences are derived from axioms



Algorithms

Lessons learned about algorithms in general:

1. Never neglect correctness!
2. Worst-case complexity may be misleading
3. Goal-directed behavior is the key to practical efficiency
4. Only empirical evaluation can give a complete picture