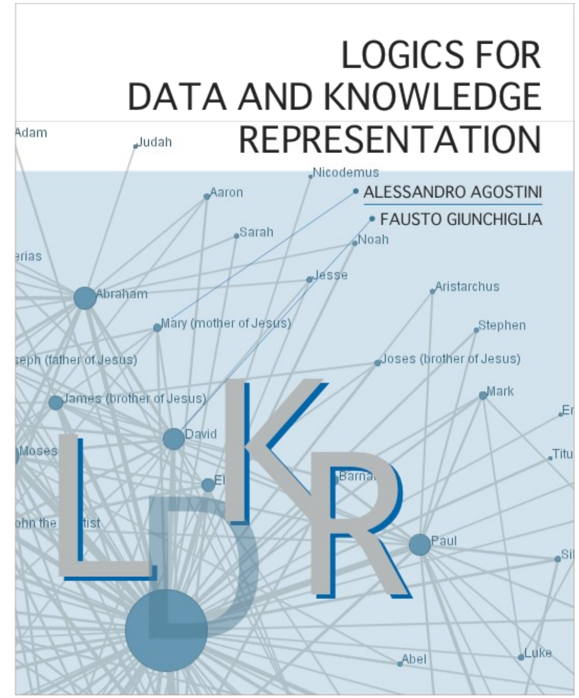


Logics for Data and Knowledge Representation

ClassL (part 2): TBOX and ABOX

Outline

- Terminology (TBox)
- Normalization of a TBox
- Reasoning with the TBox

- Some definitions
  - Primitive and defined concepts
  - Use and directly use
  - Cyclic and acyclic terminologies
  - Expansion of a TBox

- Eliminating the TBox: Reducing to DPLL reasoning

Terminology (TBox)

- A terminology (or TBox) is a set of definitions and specializations
- Terminological axioms express constraints on the concepts of the language, i.e. they limit the possible models
- The TBox is the set of all the constraints on the possible models

Equivalence

TBOX

Equality axiom
Definition

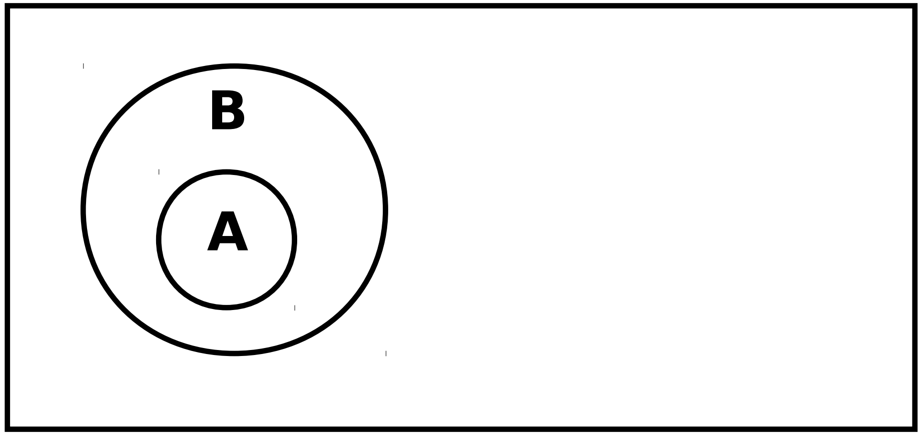
$$
\begin{array}{ll}
\text{Woman} & \equiv \text{Person} \sqcap \text{Female} \\
\text{Man} & \equiv \text{Person} \sqcap \neg\text{Woman} \\
\text{Student} & \sqsubseteq \text{Person} \sqcap \text{Study} \\
\text{Bachelor} & \equiv \text{Student} \sqcap \text{Undergraduate} \\
\text{PhD} & \sqsubseteq \text{Student} \sqcap \text{Lecturer}
\end{array}
$$

Inclusion axiom
Specialization

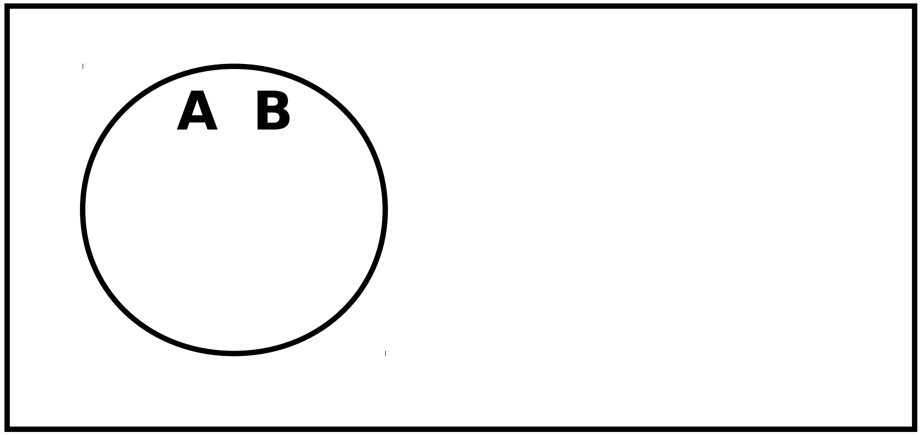
Subsumption

Semantics: Venn diagrams to represent axioms

- $\sigma(A \sqsubseteq B)$



- $\sigma(A \equiv B)$



Normalization of a TBox

- It is always possible to transform a specialization into a definition by introducing an auxiliary symbol as follows:

Woman $\sqsubseteq$ Person (the specialization)

Woman $\equiv$ Person $\sqcap$ Female (the normalized specialization)

- If from a TBox we transform all specializations into definitions we say we have normalized the TBox

- A TBox with definitions only is called regular.

Reasoning with a TBox T

- Given two class-propositions P and Q, we want to reason about:
  - Satisfiability w.r.t. T $T \models P$?

    A concept P is satisfiable w.r.t. a terminology T, <u>if there exists an interpretation</u> I with $I \models \theta$ for all $\theta \in T$, and such that $I \models P$, $I(P) \neq \emptyset$
  - Subsumption $T \models P \sqsubseteq Q$? $T \models Q \sqsubseteq P$?

    A concept P is subsumed by a concept Q w.r.t. T if $I(P) \subseteq I(Q)$ for every model I of T
  - Equivalence $T \models P \sqsubseteq Q$ and $T \models Q \sqsubseteq P$?

    Two concepts P and Q are equivalent w.r.t. T if $I(P) = I(Q)$ for every model I of T
  - Disjointness $T \models P \sqcap Q \sqsubseteq \bot$?

    Two concepts P and Q are disjoint with respect to T if their intersection is empty, $I(P) \cap I(Q) = \emptyset$, for every model I of T

TBox: primitive and defined concepts

- In a TBox there are two kinds of concepts (symbols):
  - Primitive concepts (or base symbols), which occur only on the right hand of axioms
  - Defined concepts (or name symbols) which occur on the left hand of axioms

$A \sqsubseteq B \sqcap (C \sqcup D)$

B, C and D are primitive concepts. A is a defined concept

Use and direct use

Let A and B be atomic concepts in a terminology T.

- We say that A directly uses B in T if B appears in the right hand of the defintion of A.

$A \sqsubseteq B \sqcap (C \sqcup D)$

A directly uses B, C, D

- We say that A uses B in T if B appears in the right hand after the definition of A has been "unfolded" so that there are only primitive concepts in the left hand side of the definition of A.

$A \sqsubseteq B \sqcap (C \sqcup D)$ ---> $A \sqsubseteq (C \sqcup E) \sqcap (C \sqcup D)$

$B \sqsubseteq C \sqcup E$

A directly uses B; A uses E (because B is defined in terms of E)

Cyclic and acyclic terminologies

- A terminology T contains a cycle (is cyclic) if it contains a concept which uses itself.

$\text{Father} \equiv \text{Male} \sqcap \text{hasChild}$

$\text{hasChild} \equiv \text{Father} \sqcup \text{Mother}$

Is cyclic

- A terminilogy is acyclic otherwise

$\text{Parent} \equiv \text{Father} \sqcup \text{Mother}$

$\text{Father} \sqsubseteq \text{Male}$

$\text{Mother} \sqsubseteq \text{Female}$

$\text{Male} \equiv \text{Person} \sqcap \neg \text{Female}$

Is acyclic

Expansion and equivalent terminologies

- The expansion T’ of an acyclic terminology T is a terminology obtained from T by unfolding all definitions until all concepts occurring on the right hand side of definitions are primitive (direct use only)

T	T’
$A \sqsubseteq B \sqcap (C \sqcup D)$	$A \sqsubseteq (C \sqcup E) \sqcap (C \sqcup D)$
$B \sqsubseteq (C \sqcup E)$	$B \sqsubseteq (C \sqcup E)$

- T and T’ are equivalent when they have the same expansion.
- Reasoning with T’ will yield the same results as reasoning with T.
- If T’ is the expansion of T then they are equivalent.

NOTE: it is possible to expand also a cyclic TBox.

In some cases some models exist even if the TBox is cyclic. These models are called fixpoints and there are some methods to find them and break the recursion (we will not see them).

Expansion requires normalization

- To expand a terminology we should first normalize it (not strictly necessary). Otherwise, if we use a specialization to expand a definition, definitions reduce to specializations, as below:

T	T'
Parent ≡ Father ⊔ Mother	Parent ⊑ (Person ⊓ ¬ Female) ⊔ Female
Father ⊑ Male	Father ⊑ Person ⊓ ¬ Female
Mother ⊑ Female	Mother ⊑ Female
Male ≡ Person ⊓ ¬ Female	Male ≡ Person ⊓ ¬ Female

- From now on we deal with regular terminologies only

  (see next slide for the regular version of the terminology T above)

Concept expansion

- For each concept C we define the expansion of C with respect to T as the concept C' that is obtained from C by replacing each occurrence of a name symbol A in C by the concept D, where A≡D is the definition of A in T', the expansion of T

T
Parent ≡ Mother ⊔ Father
Father ≡ Male ⊓ hasChild
Mother ≡ Female ⊓ hasChild
Male ≡ Person ⊓ ¬ Female

The expansion of Parent w.r.t. T is:
(Female ⊓ hasChild) ⊔ (Person ⊓ ¬ Female ⊓ hasChild)

NOTE: The expansion of T to T' or C to C' can be costly: In the worst case T' is exponential in the size of T, and this propagates to single concepts.

PL and ClassL: table of the symbols

PL and ClassL are notational variants

	PL	**ClassL**
Syntax	$\wedge$	$\sqcap$
	$\vee$	$\sqcup$
	$\neg$	$\neg$
	$\top$	$\top$
	$\bot$	$\bot$
	$\rightarrow$	$\sqsubseteq$
	$\leftrightarrow$	$\equiv$
	P, Q…	P, Q…
Semantics	Δ={true, false}	Δ={o, …} (compare models)

- RECALL: A proposition P is true iff it is satisfiable

Reduction to subsumption and unsatisfiability

- Reduction to subsumption. Given two concepts C and D,
  - C is unsatisfiable $\Leftrightarrow C \sqsubseteq \bot$
  - $C \equiv D \Leftrightarrow C \sqsubseteq D$ and $D \sqsubseteq C$
  - $C \perp D \Leftrightarrow C \sqcap D \sqsubseteq \bot$

- Reduction to unsatisfiability. Given two concepts C and D,
  - $C \sqsubseteq D \Leftrightarrow C \sqcap \neg D$ is unsatisfiable
  - $C \equiv D \Leftrightarrow$ both $(C \sqcap \neg D)$ and $(\neg C \sqcap D)$ are unsatisfiable
  - $C \perp D \Leftrightarrow C \sqcap D$ is unsatisfiable

Eliminating the TBox using expansion

Assume C' expansion of C w.r.t. T.

For all σ satisfying all the axioms in T we have:

- T ⊨ C iff σ ⊨ C' (Satisfiability)
- T ⊨ C ⊑ D iff σ ⊨ C' ⊑ D' (Subsumption, Equivalence)
- T ⊨ C ⊓ D ⊑ ⊥ iff σ ⊨ C' ⊓ D' ⊑ ⊥ (Disjointness)

T
Person ≡ Male ⊓ Female
Male ≡ Person ⊓ ¬ Female

Is Person satisfiable? NO!

The expansion of Person w.r.t. T is: (Person ⊓ ¬ Female) ⊓ Female
which is equivalent to ⊥ and therefore unsatisfiable

Eliminating the TBox: the algorithm

- With acyclic TBoxes T it is always possible to reduce reasoning problems w.r.t. T to problems without T. See for instance the algorithm for subsumption (all the others can be reduced to it).
- **Input**: a TBox T, the two concepts C and D
- **Output**: true if C $\sqsubseteq$ D holds or false otherwise

boolean function IsSubsumedBy(T, C, D) {

T’ = Normalize(T);

C’ = Expand(C, T’);

D’ = Expand(D, T’);

C’ = RewriteInPL(C’);

D’ = RewriteInPL(D’);

return DPLL(C’ → D’);

}

Normalization

Expansion, TBox elimination

Conversion in PL

DPLL Reasoning

Outline

- World descriptions, assertions (ABox)
- Reasoning with the ABox
- Eliminating the ABox: Reducing to DPLL reasoning

ABox, syntax

- The second component of the knowledge base is the world description, the ABox.
- In an ABox one introduces individuals, by giving them names, and one *asserts* properties about them.
- We denote individual names as a, b, c,…
- An assertion with concept C is called concept assertion (or simply assertion) in the form:

C(a), C(b), C(c), …

Student(paul)
Professor(fausto)

To be read:
paul belongs to (is in) Student
fausto belongs to (is in) Professor

ABox, semantics

- We give semantics to ABoxes by extending interpretations to individual names
- An interpretation I: L $\rightarrow$ pow(Δ^I) not only maps atomic concepts to sets, but in addition it maps each individual name a to an element $a^I \in \Delta^I$, namely

$$I(a) = a^I \in \Delta^I$$

$$I(C(a)) = a^I \in C^I$$

- Unique name assumption (UNA). We assume that distinct individual names denote distinct objects in the domain

NOTE: Δ^I denotes the domain of interpretation, a denotes the symbol used for the individual (the name), while a^I is the actual individual of the domain.

Reasoning Services

- Given an ABox A, we can reason (w.r.t. a TBox T) about the following:
  - Satisfiability/Consistency: An ABox A is consistent with respect to T if there is an interpretation I which is a model of both A and T.
  - Instance checking: checking whether an assertion C(*a*) is entailed by an ABox, i.e. checking whether *a* belongs to C.

    A ⊨ C(*a*) if every interpretation that satisfies A also satisfies C(*a*).
  - Instance retrieval: given a concept C, retrieve all the instances *a* which satisfy C.
  - Concept realization: given *a* set of concepts and an individual *a* find the most specific concept(s) C (w.r.t. subsumption ordering) such that A ⊨ C(*a*).

Eliminating the ABox

- RECALL: ABoxes contain assertions of the form C(a).
- To eliminate the ABox we need to create a corresponding concept for each assertion, e.g. of the form C-a and a new axiom C-a ⊑ C.
- This causes an exponential blow up.

A = {Master(Chen), Master(Paul), PhD(Enzo), PhD(Ronald), Assistant(Rui)}

New concepts:
Master-Chen, Master-Paul, PhD-Enzo, PhD-Ronald, Assistant-Rui
Their interpretation is the singleton set containing the individual.

T is extended with:
{Master-Chen ⊑ Master, PhD-Enzo ⊑ PhD, Assistant-Rui ⊑ Assistant}

Eliminating the ABox: the algorithm

- It is always possible to reduce reasoning problems w.r.t. an acyclic TBox T and an ABox A to problems without them. See for instance the algorithm for subsumption (all the others can be reduced to it).

- **Input**: a TBox T, an ABox A, the two concepts C and D
- **Output**: true if C ⊑ D holds or false otherwise

```
boolean function IsSubsumedBy(T, A, C, D) {
   A’ = Expand(A, T);
   T’ = ConvertAssertions(T, A’);
   return IsSubsumedBy(T’, C, D) ;
}
```

ABox expansion

ABox elimination

DPLL Reasoning by eliminating T’. (see previous lesson)