Semantic Web Foundations

Part 2: Reasoning in Description Logic

Peter Radics
CS6204 - Usable Security
Fall 2009
Dr. Dennis Kafura
Virginia Tech

September 24, 2009

Goal

Goal of Presentation

- Demonstrate the power (or lack thereof) of reasoning (**what** can be reasoned about?)
- Introduce an algorithm for reasoning (**how** can the computer reason?)

Recap

Three main building blocks

- Concepts
- Relationships
- Individuals

Recap (cont'd)

Further building blocks

- Union
- Intersection
- Complement
- Existential quantification
- Universal quantification
- Number restriction

Attributive Languages

Introducing formality:

We will write:

A, B for *atomic concepts*
R for *atomic roles*
C, D for *concept descriptions* (concepts that are defined through combination of other concepts)

Attributive Languages (cont'd)

The basic description language $\mathcal{AL}$

Definition

$$
\begin{array}{rcl}
C, D \longrightarrow A & | & \text{(atomic concept)} \\
\top & | & \text{(universal concept)} \\
\bot & | & \text{(bottom concept)} \\
\neg A & | & \text{(atomic negation)} \\
C \sqcap D & | & \text{(intersection)} \\
\forall R.C & | & \text{(value restriction)} \\
\exists R.\top & | & \text{(limited existential quantification)}
\end{array}
$$

Attributive Language (cont'd)

Definition (Interpretations)

An interpretation $\mathcal{I}$ consists of a non-empty set $\Delta^{\mathcal{I}}$ (the domain of the interpretation) and an interpretation function $\cdot^{\mathcal{I}}$, which assigns to every atomic concept A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to every atomic role R a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. $\mathcal{I}$ furthermore maps every individual a to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.

Attributive Language (cont'd)

Therefore:

Definition

$$
\begin{array}{rcl}
\top^{\mathcal{I}} & = & \Delta^{\mathcal{I}}. \\
\bot^{\mathcal{I}} & = & \varnothing. \\
(\neg A) & = & \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}. \\
(C \sqcap D)^{\mathcal{I}} & = & C^{\mathcal{I}} \cap D^{\mathcal{I}}. \\
(\forall R.C)^{\mathcal{I}} & = & \left\{a \in \Delta^{\mathcal{I}} \mid \forall b.\, (a, b) \in R^{\mathcal{I}} \to b \in C^{\mathcal{I}}\right\}. \\
(\exists R.C)^{\mathcal{I}} & = & \left\{a \in \Delta^{\mathcal{I}} \mid \exists b.\, (a, b) \in R^{\mathcal{I}}\right\}.
\end{array}
$$

We say $C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$ for all interpretations $\mathcal{I}$.

Attributive Language (cont'd)

Extensions of $\mathcal{AL}$:

Definition $\mathcal{AL}[\mathcal{U}]$ (Union)

$$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}.$$

Definition $\mathcal{AL}[\mathcal{E}]$ (Full existential quantification)

$$(\exists R.C)^{\mathcal{I}} = a \in \Delta^{\mathcal{I}} \mid \exists b\,(a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}.$$

Definition $\mathcal{AL}[\mathcal{N}]$ (Number restrictions)

$$\begin{aligned}(\geq nR)^{\mathcal{I}} &= \left\{a \in \Delta^{\mathcal{I}} \mid \left|\left\{b \mid (a,b) \in R^{\mathcal{I}}\right\}\right| \geq n\right\}.\\ (\leq nR)^{\mathcal{I}} &= \left\{a \in \Delta^{\mathcal{I}} \mid \left|\left\{b \mid (a,b) \in R^{\mathcal{I}}\right\}\right| \leq n\right\}.\end{aligned}$$

Attributive Language (cont'd)

Definition

$C \sqsubseteq D$ and $R \sqsubseteq S$ are called *inclusions*.
$C \equiv D$ and $R \equiv S$ are called *equalities*.
$A \equiv C$ is called a *definition*.

Definition

$C(a)$ is called a *concept assertion*.
$R(a, b)$ is called a *role assertion*.

Definition

The interpretation $\mathcal{I}$ *satisfies* the concept assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and it *satisfies* the role assertion $R(a, b)$ if $\left(a^{\mathcal{I}}, b^{\mathcal{I}}\right) \in R^{\mathcal{I}}$.

Example scenario

Example

Atomic concepts:

- Store
- Issuer
- Credential
- GovernmentAgency

Atomic roles:

- HasCredential
- IssuedBy
- ControlledBy

Example scenario (cont'd)

Example

Definitions:

- $\text{UntrustedIssuer} \equiv \text{Issuer} \sqcap \neg\exists\text{ControlledBy.GovernmentAgency}$
- $\text{TrustedIssuer} \equiv \neg\ \text{UntrustedIssuer}$
- $\text{UntrustedCredential} \equiv \text{Credential} \sqcap \neg\exists\text{IssuedBy.TrustedIssuer}$
- $\text{TrustedCredential} \equiv \text{Credential} \sqcap \exists\text{IssuedBy.TrustedIssuer}$
- $\text{TrustedStore} \equiv \text{Store} \sqcap \exists\text{HasCredential.TrustedCredential}$

Example scenario (cont'd)

Example

Concept assertions:

- Store(amazon)
- Store(malroysShadyEmporium)
- Issuer(veriSign)
- Issuer(malroysShadyEmporium)
- GovernmentAgency(nsa)
- Credential(sslCertificate_amazon)
- Credential(sslCertificate_malroysShadyEmporium)

Example scenario (cont'd)

Example

Role assertions:

- HasCredential(amazon, sslCertificate_amazon)
- HasCredential(malroysShadyEmporium, sslCertificate_malroysShadyEmporium)
- IssuedBy(sslCertificate_amazon, veriSign)
- IssuedBy(sslCertificate_malroysShadyEmporium, malroysShadyEmporium)
- ControlledBy(veriSign, nsa)

Reasoning

There are four reasoning tasks for TBoxes:

- Satisfiability (Consistency)
- Subsumption
- Equivalence
- Disjointness

Reasoning

Definition (Satisfiability)

A Concept C is *satisfiable* with respect to a TBox $\mathcal{T}$ if a model $\mathcal{I}$ of $\mathcal{T}$ exists such that $C^{\mathcal{I}}$ is not empty. In this case, we say that $\mathcal{I}$ is a *model* of C.

Definition (Subsumption)

A concept C is *subsumed* by a concept D with respect to $\mathcal{T}$ if $C^{\mathcal{I}} \sqsubseteq D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$. In this case we write $C \sqsubseteq_{\mathcal{T}} D$ or $\mathcal{T} \models C \sqsubseteq D$.

Reasoning

Definition (Equivalence)

Two concepts C and D are *equivalent* with respect to $\mathcal{T}$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$. In this case we write $C \equiv_{\mathcal{T}} D$ or $T \models C \equiv D$.

Definition (Disjointness)

Two concepts C and D are disjoint with respect to $\mathcal{T}$ if $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \varnothing$ for every model $\mathcal{I}$ of $\mathcal{T}$.

Reasoning

Theorem

All reasoning questions for TBoxes can be reduced to satisfiability!

Corollary

1. C *is subsumed by* $D \Leftrightarrow C \sqcap \neg D$ *is unsatisfiable;*
2. C *and* D *are equivalent* $\Leftrightarrow$ *both* $(C \sqcap \neg D)$ *and* $(\neg C \sqcap D)$ *are unsatisfiable;*
3. C *and* D *are disjoint* $\Leftrightarrow C \sqcap D$ *is unsatisfiable.*

Reasoning

Reasoning for ABoxes:

- Satisfiability (Consistency)
- Instance Check (Entailment)

Definition

An ABox $\mathcal{A}$ is consistent with respect to a TBox $\mathcal{T}$, if there is an interpretation that is a model of both $\mathcal{A}$ and $\mathcal{T}$.

Definition (Entailment)

An assertion a is entailed by $\mathcal{A}$ and we write $\mathcal{A} \models a$ if every interpretation that satisfies $\mathcal{A}$, that is, every model of $\mathcal{A}$, also satisfies a.

Reasoning

Theorem

All reasoning questions for ABoxes can be reduced to consistency!

Corollary

$\mathcal{A} \models C(a)$ *iff* $\mathcal{A} \cup \{\neg C(a)\}$ *is inconsistent.*
C *is satisfiable iff* $\{C(a)\}$ *is consistent.*

Algorithm

Tableau Calculus

Definition

1. Formulate query
2. Expand query
3. Bring query into negative normal form
4. Start with ABox $\mathcal{A} = \{C_0(x_0)\}$
5. Iterate transformations on ABox (see next slide)
6. Check consistency on transformed ABoxes

Algorithm

The $\rightarrow \sqcap$-rule
Condition: $\mathcal{A}$ contains $(C_1 \sqcap C_2)(x)$, but it does not contain both $C_1(x)$ and $C_2(x)$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{C_1(x), C_2(x)\}$.
The $\rightarrow \sqcup$-rule
Condition: $\mathcal{A}$ contains $(C_1 \cup C_2)(x)$, but neither $C_1(x)$ nor $C_2(x)$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{C_1(x)\}, \mathcal{A}'' = \mathcal{A} \cup \{C_2(x)\}$.
The $\rightarrow \exists$-rule
Condition: $\mathcal{A}$ contains $(\exists R.C)(x)$, but there is no individual name z such that $C(z)$ and $R(x,z)$ are in $\mathcal{A}$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{C(y), R(x,y)\}$ where y is an individual name not occurring in $\mathcal{A}$.
The $\rightarrow \forall$-rule
Condition: $\mathcal{A}$ contains $(\forall R.C)(x)$ and $R(x,y)$, but it does not contain $C(y)$.
Action: $\mathcal{A}' = \mathcal{A} \cup C(y)$.

Algorithm

The $\rightarrow_{\geq}$-rule
Condition: $\mathcal{A}$ contains $(\geq nR)(x)$, and there are no individual names $z_1 \dots z_n$ such that $R(x, z_i)$ $(1 \leq i \leq n)$ and $z_i \neq z_j$ $(1 \leq i < j \leq n)$ are contained in $\mathcal{A}$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{R(x, y_i) \mid (1 \leq i \leq n)\} \cup \{y_i \neq y_j \mid 1 \leq i < j \leq n\}$, where $y_1 \dots y_n$ are distinct individual names not occurring in $\mathcal{A}$.
The $\rightarrow_{\leq}$-rule
Condition: $\mathcal{A}$ contains distinct individual names $y_1 \dots y_n + 1$ such that $(\leq nR)(x)$ and $R(x, y_1) \dots R(x, y_n + 1)$ are in $\mathcal{A}$, and $y_i \neq y_j$ is not in $\mathcal{A}$ for some $i \leq j$.
Action: For each pair y_i, y_j such that $i > j$ and $y_i \neq y_j$ is not in $\mathcal{A}$, the ABox $\mathcal{A}_{i,j} = [y_i/y_j]\mathcal{A}$ is obtained from $\mathcal{A}$ by replacing each occurrence of y_i by y_j.

Algorithm

Example

Help me try to reason, using the algorithm and the example defined earlier, whether malroysShadyEmporium is a TrustedStore!

Discussion

- What is the impact of reasoning on the usefulness of Description Logics?
- What uses do you see for reasoning in a Usable Security context?