

OWL 2 – Syntax und Semantik

Sebastian Rudolph

Institut AIFB · Universität Karlsruhe

Grundlagen Semantic Web (WS08/09)

Seminar für Computerlinguistik, Universität Heidelberg

<http://semantic-web-grundlagen.de>

Foliensatz von M. Krötzsch. Die nichtkommerzielle Vervielfältigung, Verbreitung und Bearbeitung dieser Folien ist zulässig (→ Lizenzbestimmungen CC-BY-NC).

- 1 Einleitung und Ausblick
- 2 XML und URIs
- 3 Einführung in RDF
- 4 RDF Schema
- 5 Logik – Grundlagen
- 6 Semantik von RDF(S)
- 7 OWL – Syntax und Intuition
- 8 OWL – Semantik und Reasoning
- 9 SPARQL – Syntax und Intuition
- 10 Semantik von SPARQL
- 11 Konjunktive Anfragen und Regelsprachen
- 12 **OWL 2 – Syntax und Semantik**

Literatur zu dieser Vorlesung online siehe → [Vorlesungswebseite](#)

OWL für viele Aufgaben noch zu schwach (siehe → letzte Vorlesung).

- OWL als Anfragesprache ungenügend
↪ Konjunktive Anfragen, SPARQL für OWL
- OWL als Ontologiesprache ungenügend
↪ Prädikatenlogische Regelerweiterungen, SWRL
- OWL als Programmiersprache ungenügend
↪ Logikprogrammierung im Semantic Web
(→ Vorlesung SWebT2)

Sollte auch der OWL-Standard selbst erweitert werden?

↪ OWL 1.1

OWL 2 als „nächste Version“ von OWL

Erweiterungen aufgrund von Praxiserfahrung mit OWL 1.0:

- zusätzliche Ausdrucksstärke durch neue ontologische Axiome
- nicht-logische Erweiterungen (Syntax, Metadaten, ...)
- Überarbeitung der OWL-Varianten (Lite/DL/Full)

Zielstellungen:

- weitestgehende Kompatibilität zum existierenden OWL-Standard
- Erhaltung der Entscheidbarkeit von OWL DL
- Behebung von Problemen im OWL-1.0-Standard

OWL DL basiert auf Beschreibungslogik *SHOIN*(*D*):

- Axiome:
 - Tbox: Subklassenbeziehungen $C \sqsubseteq D$
 - Rbox: Subrollenbeziehungen $R \sqsubseteq S$ (\mathcal{H}), Inverse Rollen R^- (\mathcal{I}), Transitivität
 - Abox: Fakten zu Klassen $C(a)$, Rollen $R(a, b)$, und Gleichheit $a \approx b$ bzw. $a \neq b$
- Klassenkonstruktoren:
 - Konjunktion $C \sqcap D$, Disjunktion $C \sqcup D$, Negation $\neg C$ von Klassen
 - Rollenrestriktionen: universell $\forall R.C$ und existenziell $\exists R.C$
 - Zahlenrestriktionen (\mathcal{N}): $\leq n R$ und $\geq n R$ (n nicht-negative Zahl)
 - Nominale (\mathcal{O}): $\{a\}$
- Datentypen (*D*)

Erweiterung in OWL 2 zu *SROIQ*(*D*)

SHOIN unterstützt verschiedene Abox-Fakten:

- Klassenzugehörigkeit $C(a)$ (C komplexe Klasse),
- Sonderfall: negierte Klassenzugehörigkeit $\neg C(a)$ (C komplexe Klasse),
- Gleichheit $a \approx b$,
- Ungleichheit $a \not\approx b$
- Rollenbeziehungen $R(a, b)$
- *negierte Rollenbeziehungen?*

\rightsquigarrow *SROIQ* erlaubt auch **negierte Rollen** in der Abox: $\neg R(a, b)$

SHOIN unterstützt nur einfache Zahlenrestriktionen (\mathcal{N}):

$\text{Person} \sqcap \geq 3 \text{ hatKind}$

„Klasse aller Personen mit 3 oder mehr Kindern.“

\rightsquigarrow *SROIQ* erlaubt auch **qualifizierte Zahlenrestriktionen** (\mathcal{Q}):

$\text{Person} \sqcap \geq 3 \text{ hatKind.}(\text{Frau} \sqcap \text{Professor})$

„Klasse aller Personen mit 3 oder mehr Töchtern, die Professoren sind.“

Modellierungsaufgabe: „Jeder Mensch kennt sich selbst.“

- *SHOIN*:

kennt(tom, tom) kennt(tina, tina) kennt(udo, udo) ...

↪ nicht allgemein anwendbar

- *SROIQ*: spezieller Ausdruck **Self**

Mensch $\sqsubseteq \exists$ kennt.Self

SROIQ bietet zusätzliche Aussagen über Rollen:

- $\text{Tra}(R)$: R ist **transitiv** (definiert wie in *SHOIN*)
Beispiel: $\text{Tra}(\text{liegtIn})$
- $\text{Sym}(R)$: R ist **symmetrisch** (definiert wie in *SHOIN*)
Beispiel: $\text{Sym}(\text{verwandtMit})$
- $\text{Ref}(R)$: R ist **reflexiv**, $(x, x) \in R^I$ für alle Domänenindividuen x
Beispiel: $\text{Ref}(\text{kennt})$
- $\text{Irr}(R)$: R ist **irreflexiv**, $(x, x) \notin R^I$ für alle Domänenindividuen x
Beispiel: $\text{Irr}(\text{hatKind})$
- $\text{Dis}(R, S)$: R und S sind **disjunkt**, $(x, y) \notin R^I \cap S^I$ für alle x, y
Beispiel: $\text{Dis}(\text{hatVater}, \text{hatSohn})$
- **Universelle Rolle** U : $(x, y) \in U^I$ für alle x, y
Beispiel: $\top \sqsubseteq \leq 7000000000 U.\text{Menschen}$ (nicht empfohlen!)
 $\rightsquigarrow U$ ist vor allem als Gegenstück zu \top sinnvoll, z.B. als Wurzel der Rollenhierarchie in grafischen Editoren

Allgemeine Rolleninklusion

„Die Freunde meiner Freunde sind auch meine Freunde.“

↪ Kann in *SHOIN* ausgedrückt werden: hatFreund ist transitiv.

„Die Feinde meiner Freunde sind auch meine Feinde.“

↪ Kann nicht in *SHOIN* ausgedrückt werden!

Rolleninklusion

- Rbox-Ausdrücke der Form $R_1 \circ R_2 \circ \dots \circ R_n \sqsubseteq S$,
Beispiel: hatFreund \circ hatFeind \sqsubseteq hatFeind
- Semantik: wenn $(x_0, x_1) \in R_1^I, (x_1, x_2) \in R_2^I, \dots, (x_{n-1}, x_n) \in R_n^I$,
dann gilt auch $(x_0, x_n) \in S^I$
Beispiel: wenn $(x, y) \in \text{hatFreund}^I$ und $(y, z) \in \text{hatFeind}^I$,
dann gilt auch $(x, z) \in \text{hatFeind}^I$

Weitere Beispiele:

teilVon \circ gehört \sqsubseteq gehört

hatBruder \circ hatKind \sqsubseteq istOnkelVon

Ausdrucksstärke der Rolleninklusion

Wie kompliziert ist Rolleninklusion?

Mit Rboxen kann man formale Sprachen kodieren: (Skizze!)

Grammatik für Sprache der Wörter $ab, aabb, aaabbb, \dots$:

$$\begin{array}{l} L ::= ab \\ L ::= aLb \end{array} \quad \text{wird zu Rbox} \quad \begin{array}{l} R_a \circ R_b \sqsubseteq L \\ R_a \circ L \circ R_b \sqsubseteq L \end{array}$$

- $\rightsquigarrow \exists L. T \neq \perp$ (“ $\exists L. T$ notwendig nicht-leer”) bedeutet*:
„Es gibt eine Kette aus R_a und R_b , die zur Sprache gehört.“
- $\rightsquigarrow \exists L_1. \exists L_2. L_1 \cap L_2 \neq \perp$ für zwei kodierte Sprachen L_1 und L_2 bedeutet:
„Es gibt ein Wort, das zu L_1 und zu L_2 gehört.“

*) bei entsprechender Tbox!

Leider gilt: Leerheit der Überschneidung kontextfreier Sprachen ist unentscheidbar.

\rightsquigarrow OWL mit Rolleninklusionen ist unentscheidbar

Kann man Rolleninklusion zwecks Entscheidbarkeit einschränken?

- Rboxen sind wie Grammatiken für kontextfreie formale Sprachen
- Überschneidungen von kontextfreien Sprachen problematisch

⇒ Einschränkung auf reguläre Sprachen!

Reguläre Rboxen

Rollenamen werden mit \prec geordnet (strenge totale Ordnung).
Jede Rbox-Inklusion muss eine der folgenden Formen haben:

- $R \circ R \sqsubseteq R$
- $R \circ S_1 \circ S_2 \circ \dots \circ S_n \sqsubseteq R$
- $R^- \sqsubseteq R$
- $S_1 \circ S_2 \circ \dots \circ S_n \circ R \sqsubseteq R$
- $S_1 \circ S_2 \circ \dots \circ S_n \sqsubseteq R$

Dabei gilt: $S_i \prec R$ für alle $i = 1, 2, \dots, n$.

Rbox ist regulär, wenn es so eine Ordnung \prec gibt.

Reguläre Rboxen – Beispiel

Beispiel:

$$R \circ S \sqsubseteq R \quad S \circ S \sqsubseteq S \quad R \circ S \circ R \sqsubseteq T$$

\rightsquigarrow ist regulär mit Ordnung $S \prec R \prec T$

Beispiel:

$$R \circ T \circ S \sqsubseteq T$$

\rightsquigarrow ist nicht regulär (unzulässige Inklusions-Form)

Beispiel:

$$R \circ S \sqsubseteq S \quad S \circ R \sqsubseteq R$$

\rightsquigarrow ist nicht regulär (keine gültige Ordnung möglich)

Beschränkung einfacher Rollen

- Einfache Rollen in *SHOIN* = Rollen ohne transitive Unterrollen
- In *SROIQ*: Beachtung der Rolleninklusionen nötig!

Einfache Rollen sind alle Rollen ...

- die nicht auf der rechten Seite einer Rolleninklusion vorkommen,
- die Inverse von anderen einfachen Rollen sind,
- die nur auf der rechten Seite von Rolleninklusionen vorkommen, bei denen links ausschließlich einfache Rollen stehen.

(Achtung: induktive Definition)

↔ nicht-einfach sind Rollen, die direkt oder indirekt von sich selbst abhängen (und deren Überrollen)

Warum ist das wichtig?

Ausdrücke $\leq n R.C$, $\geq n R.C$, $\text{Irr}(R)$, $\text{Dis}(R, S)$, $\exists R.\text{Self}$ $\neg R(a, b)$
nur für einfache Rollen R und S erlaubt!

(Grund: Sicherstellung von Entscheidbarkeit)

Klassenausdrücke

Klassenamen	A, B
Konjunktion	$C \sqcap D$
Disjunktion	$C \sqcup D$
Negation	$\neg C$
Exist. Rollenrestr.	$\exists R.C$
Univ. Rollenrestr.	$\forall R.C$
Self	$\exists S.\text{Self}$
Größer-als	$\geq n S.C$
Kleiner-als	$\leq n S.C$
Nominale	$\{a\}$

Rollen

Rollennamen	R, S, T
einfache Rollen	S, T
Inverse Rollen	R^{-}
Universelle Rolle	U

Tbox (Klassenaxiome)

Inklusion	$C \sqsubseteq D$
Äquivalenz	$C \equiv D$

Rbox (Rollenaxiome)

Inklusion	$R_1 \sqsubseteq R_2$
Allgemeine Inkl.	$R_1^{(-)} \circ \dots \circ R_n^{(-)} \sqsubseteq R$
Transitivität	$\text{Tra}(R)$
Symmetrie	$\text{Sym}(R)$
Reflexivität	$\text{Ref}(R)$
Irreflexivität	$\text{Irr}(S)$
Disjunktheit	$\text{Dis}(S, T)$

Abox (Fakten)

Klassenzugehörigkeit	$C(a)$
Rollenbeziehung	$R(a, b)$
Neg. Rollenbeziehung	$\neg S(a, b)$
Gleichheit	$a \approx b$
Ungleichheit	$a \not\approx b$

Wie kompliziert ist *SROIQ*?

Rückblick: *SHOIN* (OWL DL) ist sehr komplex (NEXPTIME)

Wie komplex ist *SROIQ*?

Beobachtung: einige Ausdrucksmittel sind nicht wirklich nötig!

- $\text{Tra}(R)$ durch $R \circ R \sqsubseteq R$ ausdrückbar
- $\text{Sym}(R)$ durch $R^- \sqsubseteq R$ ausdrückbar
- $\text{Irr}(S)$ durch $\top \sqsubseteq \neg \exists S.\text{Self}$ ausdrückbar
- Universelle Rolle durch transitive, reflexive Überrolle aller Rollen ersetzbar (hier nicht vertieft)
- Abox durch Nominale darstellbar, z.B. $R(a, b)$ durch $\{a\} \sqsubseteq \exists R.\{b\}$

Qualifizierte Zahlenrestriktionen kaum problematisch (bekannt und implementiert, siehe Vorlesung zu OWL)

↪ Hauptproblem Rollenaxiome (Rbox)

Wie geht man mit Rboxen um?

- Rbox-Regeln ähneln formalen Grammatiken
- jede Rolle R definiert eine reguläre Sprache:
die Sprache der Rollen-Ketten, aus denen R folgt
- reguläre Sprachen \equiv reguläre Ausdrücke \equiv endliche Automaten

\rightsquigarrow Ansatz: Tableauverfahren werden mit „Rbox-Automaten“ erweitert

Details siehe Literaturangaben zu SROIQ

Tableauverfahren von *SROIQ* zeigt:

SROIQ ist entscheidbar.

- Algorithmus hat gute Anpassungseigenschaften: ungenutzte Merkmale belasten die Abarbeitung kaum („pay as you go“)
- Tableau-Verfahren ungeeignet für enge Komplexitätsabschätzungen
↪ Komplexität von *SROIQ* bisher nicht bekannt (30.1.2008)

SROIQ ist „nur“ logische Grundlage von OWL DL 1.1

Weitere nicht-logische Aspekte:

- Syntax (Erweiterung nötig)
- Datentypdeklaration und Datentypfunktionen, neue Datentypen?
- Metamodellierung: „Punning“
- Kommentarfunktionen und ontologische Metadaten
- Invers-funktionale konkrete Rollen (DatatypeProperties): Keys?
- Mechanismen zu Ontologieimport?
- ...

⇒ viele laufende Änderungen

Hier: erste Übersicht für einige dieser Punkte

Metamodellierung

Spezifikation ontologischen Wissens *über* einzelne Elemente der Ontologie (einschließlich Klassen, Rollen, Axiome).

Beispiele:

- „Die Klasse *Person* wurde am 30.1.2008 von *MarkusK* angelegt.“
- „Für die Klasse *Stadt* wird die Property *Einwohnerzahl* empfohlen.“
- „Die Aussage ‚Dresden wurde 1206 gegründet‘ wurde maschinell ermittelt mit einer Sicherheit von 85%.“

(Vergleich auch Reifikation in RDF Schema)

Metamodellierung in ausdrucksstarken Logiken ist gefährlich und teuer!

OWL 2. unterstützt zurzeit einfachste Form von Metamodellierung:

Punning

- Bezeichner für Klassen, Rollen, Individuen müssen nicht disjunkt sein
- keine *logische* Beziehung zwischen Klasse, Individuum und Rolle gleichen Namens
- Beziehung nur relevant für pragmatische Interpretation

Beispiel:

Person(Sebastian) klasseErstelltVon(Person, Markus)

Punning unterstützt einfache Metadaten mit (schwacher) semantischer Bedeutung

Wie kann man rein syntaktische Kommentare zu einer Ontologie machen?

- Kommentare in XML-Dateien: `<!-- Kommentar -->`
↪ kein Bezug auf OWL-Axiome dieser Datei
- nicht-logische Annotationen in OWL:
`owl:AnnotationProperty`
↪ fest verknüpft mit (semantischem) ontologischem Element, kein syntaktischer Bezug

OWL 2 soll „echte“ syntaktische Kommentare unterstützen

Syntax für OWL 2 zum Teil noch unklar

Wahrscheinlich zwei grundlegende Varianten:

- Funktionale Syntax: ersetzt „Abstrakte Syntax“ von OWL 1.0
- RDF-Syntax: Erweiterung von OWL/RDF 1.0

↪ funktionale Syntax einfacher zu definieren
(keine RDF-Beschränkungen), kompakter

↪ RDF-Syntax für Abwärtskompatibilität wichtig

↪ verschiedene Ausdrucksmittel bisher ohne RDF-Syntax

OWL Lite als Fehlschlag:

- beinahe so komplex wie OWL DL
- komplizierte Syntax gibt keinen direkten Zugang zu wahrer Ausdrucksstärke
- Verwendung in Ontologien heute praktisch nur „zufällig“, nicht bewusst

Ursprüngliches Ziel:

einfach und effizient implementierbarer Teil von OWL

⇒ offene Diskussion für OWL 2

Beschreibungslogik \mathcal{EL}^{++}

- Konzepte nur mit Konjunktion $C \sqcap D$, Existenz $\exists R.C$, \top und \perp
- Nominale
- allgemeine Rolleninklusionen (Rbox), Transitivität

Vorteile:

- polynomielle Komplexität
- schnelle Implementierungen verfügbar
- einfache Definition
- unterstützt praktisch relevante Ontologien (SNOMED)

Beschreibungslogik DL Lite_R

- Konzipiert nur mit Konjunktion $C \sqcap D$, unqualifizierter Existenz $\exists R.T$, und \perp
- Inverse Rollen, einfache Rollenhierarchien
- Abox wie in *SROIQ*

Vorteile:

- sub-polynomielle Komplexität (verwandt mit relationalen Datenbanken)
- schnelle Implementierungen verfügbar
- relativ einfache Definition
- unterstützt RDFS

Was tun mit OWL Full?

Ziel von OWL 2 DL: viele OWL-Full-1.0-Ontologien als DL interpretierbar machen (siehe z.B. Punning)

Was soll aus OWL Full 1.1 werden?

- Erweiterung von OWL Full im Sinne von OWL 2 wird durch verschiedene Anwender unterstützt
- Definition zurzeit nicht ausgearbeitet \rightsquigarrow Spezifikation noch in Arbeit
- Erweiterung von OWL Full Teil des Mandates der OWL-Arbeitsgruppe

Offene Fragen zu OWL Full 1.1

Viele Fragen um OWL Full 1.1 zurzeit unklar

- Komplette Neuformulierung der Spezifikation? (Konsistenz der *Spezifikation* von OWL Full 1.0 konnte bis heute nicht nachgewiesen werden [30.1.2008])
- Probleme mit OWL-1.1-Erweiterungen (z.B. Punning)?
- Gewünschter semantischer Bezug zu OWL Full 1.0 und OWL DL 1.1?
- Sollte es „OWL Lite 1.1“ und „OWL Lite 1.1 Full“ geben? (Erweiterung syntaktischer Fragmente um OWL-Full-Semantik)
- ...

Relativ sicher ist:

- Wenn technisch möglich, so wird die OWL-Arbeits OWL Full 1.1 definieren.
- Es wird (weiterhin) kleinere(?) inkompatible Unterschiede der Semantik von OWL Full und OWL DL geben.

Aktueller Status:

- OWL-Arbeitsgruppe besteht seit September 2007 (erstes Treffen Anfang Dezember)
- erste Arbeitsentwürfe für Syntax, Semantik und RDF-Umsetzung am 8. Januar veröffentlicht
- Softwareunterstützung für Inferenz und Ontologieerstellung zum Teil bereits verfügbar

Offizieller Zeitplan:

- Dezember 2008: fertig ausgearbeitete Standardisierungsempfehlung (*Candidate Recommendation*)
- März 2009: Verabschiedung des neuen Standards

OWL 2 als erste Weiterentwicklung des OWL-Standards

- logische Erweiterung: Beschreibungslogik *SR_QIQ* als Grundlage
- neue Ausdrucksmittel vor allem Rollenaxiome, qualifizierte Zahlenrestriktionen
- nicht-logische Erweiterungen: Punning, Kommentare, Datentypen, u.a.
- Entwicklung von OWL Lite und OWL Full zurzeit noch offen
- Verabschiedung bis März 2009 geplant

Pascal Hitzler
Markus Krötzsch
Sebastian Rudolph
York Sure

Semantic Web Grundlagen

Springer 2008, 277 S., Softcover
ISBN: 978-3-540-33993-9
Aktuelle Literaturhinweise online

