Johann Wolfgang Goethe-Universität Frankfurt am Main
Fachbereich 12 / Institut für Informatik
Studiengang Informatik (M.Sc.)

Sticky Policies in OWL2

Hausarbeit für das Seminar
Künstliche Intelligenz

Maximilian Althaus
7162860

Inhaltsverzeichnis

1 Einleitung

Die rasante Zunahme des Datenverkehrs und die steigende Bedeutung des Datenschutzes haben zu einem verstärkten Interesse an der Entwicklung effektiver Mechanismen zur Einhaltung von Datenschutzrichtlinien geführt. Insbesondere die European General Data Protection Regulation (GDPR) hat die Anforderungen an die Verarbeitung personenbezogener Daten erhöht und Unternehmen dazu verpflichtet, angemessene Maßnahmen zu ergreifen, um die Privatsphäre der Dateninhaber zu schützen. In diesem Zusammenhang gewinnen „sticky policies “zunehmend an Bedeutung.

Eine der derzeit am meisten behandelten Implementierungen konzentriert sich auf die Erweiterung einer PL (Policy Language) in OWL2, um die Unterstützung von Sticky Policies zu ermöglichen. Sticky Policies sind spezielle Lizenzen, die auf Datenübertragungen angewendet werden und genau festlegen, wie der Empfänger die übertragenen Daten nutzen darf. Im Gegensatz zu herkömmlichen Datenschutzrichtlinien können Sticky Policies rekursiv sein, was bedeutet, dass sie nicht nur für die erste Datenübertragung gelten, sondern auch für alle zukünftigen Übertragungen, die direkt oder indirekt durchgeführt werden. Dies ermöglicht eine umfassende Kontrolle über den Datenfluss und stellt sicher, dass die Datenschutzrichtlinien auch über verschiedene Übertragungen hinweg eingehalten werden.

Das Hauptziel dieser Arbeit besteht darin, die theoretischen Grundlagen und praktischen Aspekte der Modellierung und Überprüfung von Sticky Policies in OWL2 zu erläutern und vermitteln. Wir werden untersuchen, wie Sticky Policies effektiv in die OWL2-PL integriert werden können und wie ihre Einhaltung automatisch und in Echtzeit überprüft werden kann. Dabei werden wir uns mit verschiedenen Modellierungsansätzen und potenziellen Anwendungsfällen auseinandersetzen.

2 Grundlagen

2.1 Einführung in die OWL-Sprache

Die Web Ontology Language (OWL) ist eine formale Sprache zur Beschreibung von Ontologien und Wissensrepräsentation im Web. OWL wurde speziell entwickelt, um Informationen maschinenlesbar zu machen und eine effektive Zusammenarbeit und Integration von Wissen in verschiedenen Domänen zu ermöglichen. OWL basiert auf der Description Logic, einer logischen Formalismus, der die Modellierung von Konzepten, Eigenschaften und Beziehungen zwischen Entitäten ermöglicht.

OWL bietet verschiedene Sprachprofile mit unterschiedlichen Komplexitätsstufen, um den spezifischen Anforderungen verschiedener Anwendungsbereiche gerecht zu werden. Eines dieser Profile ist OWL2-PL (Policy Language), das speziell für die Modellierung von Datenschutzrichtlinien und -regelungen entwickelt wurde. OWL2-PL bietet eine einfache und leicht verständliche Sprachbasis für die Definition und Überprüfung von Datenschutzrichtlinien in einer maschinenlesbaren Form.

Auf die Frage, warum die Web Ontology Language OWL und nicht WLO heißt, schreibt Guus Schreiber „Why not be inconsistent in at least one aspect of a language which is all about consistency? “[1]. Gleichzeit wollte die Erkentnisse der „One World Language “von William A. Martin 1970 honorieren.

2.2 OWL2 Syntax & Semantik

In OWL gibt es unterschiedliche Darstellungsformen für Axiome, wobei das XML/RDF-Format am weitesten verbreitet ist. Die OWL 2-Spezifikation verwendet hingegen das funktionale Syntaxformat, während die Autoren von Protegé[1] ein kompakteres

[1]Protégé ist eine Softwarelösung zur Erstellung und Verwaltung von RDF / OWL Modellen. Es ermöglicht die Definition von Ontologien, die Modellierung von Klassen und Eigenschaften sowie die Anwendung von Inferenzregeln. https://protege.stanford.edu/

Manchester-Format entwickelt haben. Das prägnanteste Format ist jedoch Turtle. In wissenschaftlichen Veröffentlichungen wird in der Regel eine abstrakte Syntax verwendet.[2] [3]

OWL2 Functional Syntax

```
Ontology(<http://example.org/dog.owl>
  Declaration( Class( :Tea ) )
)
```

OWL2 XML Syntax

```
  <Ontology ontologyIRI="http://example.org/dog.owl" ...>
   <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
   <Declaration>
     <Class IRI="Dog"/>
   </Declaration>
 </Ontology>
```

Manchester Syntax

```
    Ontology: <http://example.org/dog.owl>
    Class: Dog
```

RDF/XML Syntax

```
    <rdf:RDF ...>
        <owl:Ontology rdf:about=""/>
        <owl:Class rdf:about="#Dog"/>
    </rdf:RDF>
```

RDF/Turtle

```
     <http://example.org/Dog.owl> rdf:type owl:Ontology .
     :Tea  rdf:type            owl:Class .
```

Abstrakte Syntax

OWL 2 bedient sich zur Definition der abstrakten Sprache bei den Description Logics. Diese werden in dem Buch „The Description Logic Handbook "[4] umfassend definiert.
Der logische Aufbau der Sprache besteht aus Individuen X, welche Klassen/Konzepten C zugeordnet werden. Konzepte können in einem Rollenverhältnis R stehen. Sei $\mathcal{I}$ eine Interpretation des Aufbaus. Dann bildet $\Delta^{\mathcal{I}}$ die Menge aller Konzepte in dieser Interpretation. Ein Konzept ist $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$. Eine Rolle $R^{\mathcal{I}}$ bildet das Verhältnis zwischen zwei Konzepten ab $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.
Folgende Operationen können mit Konzepten durchgeführt werden.

Name	Syntax	Semantics
transitive closure	R^+	$\bigcup_{i\geq 1} (R^i)^{\mathcal{I}} \quad (R \in \mathrm{N_R})$
top	$\top$	$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
bottom	$\bot$	$\bot^{\mathcal{I}} = \emptyset$
complement	$\neg C$	$\neg C^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
intersection	$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
union	$C \sqcup D$	$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
existential	$\exists R.C$	$\left\{d \in \Delta^{\mathcal{I}} \mid \exists (d,e) \in R^{\mathcal{I}} : e \in C^{\mathcal{I}}\right\}$
restriction universal	$\forall R.C$	$\left\{d \in \Delta^{\mathcal{I}} \mid \forall (d,e) \in R^{\mathcal{I}} : e \in C^{\mathcal{I}}\right\}$
restriction interval restrictions	$\exists f \cdot [\ell, u]$	$\left\{d \in \Delta^{\mathcal{I}} \mid \exists i \in [\ell, u] : (d,i) \in f^{\mathcal{I}}\right\}$

Tabelle 1: Syntax und Semantik der Sprache

Das Paper Bonatti 2022 [5] erweitert dabei die Definitionen aus „The Description Logic Handbook "[4] um die transitive closure und die restriction interval restrictions. Zweiteres sind im Folgenden jedoch weniger relevant, weswegen diese hier nicht weiter betrachten werden.

Beispiel zu Konzepten und Rollen

Das folgende Beispiel soll dabei helfen, die Definition einfacher verständlich zu machen.
Seien **Person** und **Female** Konzepte und **hasChild** eine Rolle. Dann können wir mit $\mathbf{Person} \sqcap \exists \mathbf{hasChild}.\top$ alle Personen mit einem Kind und $\mathbf{Person} \sqcap \exists \mathbf{hasChild}.\mathbf{Female}$ alle Frauen mit einem Kind finden.

Eine Policie zur Weitergabe der Email Daten mit Zustimmung des Nutzers könnte wie folgt aussehen.

$$\exists\ \mathbf{has\_data.Email}$$
$$\sqcap \exists\ \mathbf{has\_processing.Transfer}$$
$$\sqcap \exists\ \mathbf{has\_purpose.Advertising}$$
$$\sqcap\ \mathbf{has\_legal\_basis.Consent.}$$

2.3 OWL2 Dialekte und Profile

Das W3C hat die OWL-Spezifikation mit drei Varianten (Dialekten) eingeführt, die sich in ihrem Ausdrucksvermögen unterscheiden: OWL Lite, OWL DL und OWL Full. Das Ziel dieser Spzezifikationen ist es eine größere Brandbreite zwischen leichter Bedienbarkeit und maximaler Ausdrucksstärke zu geben.
Zusätzlich unterscheidet man die OWL 2 Profile in ihrer Komplexität: OWL 2 EL mit polynomieller Schlussfolgerungskomplexität, OWL 2 QL für den einfacherern Zugriff auf Datenbanken und OWL 2 RL als Profil von OWL 2 mit weniger Regeln. In dem Paper Bonatti 2022 [5] wird die Sprache OWL2-PL für Policy Language weiter untersucht. Diese Sprache wurde in den Projekten SPECIAL[1]. und TRAPEZE[2] entwickelt und verwendet und soll wie vorher bereits beschrieben die Einhaltung der Datenschutzrichtlinien gewährleisten.

[1] https://specialprivacy.ercim.eu/
[2] https://trapeze-project.eu/

2.4 Policies in der Datenübertragung

In dem Umgang mit gespeicherten Daten sind immer drei Parteien eingebunden. 1. Data Subjects, welche die Daten zur Verfügung stellen 2. Data Controllers, welche die Daten verwalten und speichern und 3. Data Protection Authorities, welche den korrekten Umgang mit den Daten überwachen und kontrollerien. Policies spielen eine zentrale Rolle beim Datenschutz und bei der Einhaltung von Vorschriften. Sie definieren Regeln und Bestimmungen, die den Umgang mit personenbezogenen Daten steuern. In der Datenübertragungsumgebung legen Policies fest, wie Daten zwischen verschiedenen Parteien ausgetauscht werden dürfen und welche Bedingungen erfüllt sein müssen, damit der Datentransfer stattfinden kann.

Eine Policy kann als eine Sammlung von Autorisierungsregeln betrachtet werden, die angeben, welche Aktionen erlaubt oder verboten sind. Diese Regeln können verschiedene Attribute und Bedingungen umfassen, wie z.B. die Art der Daten, den Zweck der Verarbeitung, die beteiligten Parteien, die Rechtsgrundlage usw. Die Definition und Überprüfung von Policies erfordert eine formale Repräsentation und eine Möglichkeit zur automatischen Verarbeitung.

2.5 Sticky Policies

Sticky Policies stellen eine besondere Art von Policies dar, die in der Datenübertragung angewendet werden können. Im Gegensatz zu herkömmlichen Policies, die nur für eine spezifische Datenübertragung gelten, haben Sticky Policies eine rekursive Natur. Das bedeutet, dass sie nicht nur für die erste Datenübertragung gelten, sondern auch für alle zukünftigen Übertragungen.

Die Idee hinter Sticky Policies ist, dass bestimmte Bedingungen und Einschränkungen, die bei der ersten Datenübertragung definiert wurden, auch für nachfolgende Übertragungen beibehalten werden. Dies ermöglicht eine konsistente Anwendung der Datenschutzrichtlinien über den gesamten Datenfluss hinweg und stellt sicher, dass die Daten gemäß den ursprünglichen Zustimmungs- und Verwendungsvorgaben verwendet werden.

Die Modellierung von Sticky Policies erfordert spezielle Erweiterungen der bestehenden Sprachprofile in OWL2-PL, um die rekursiven Aspekte und die kontinuierliche Anwendung der Richtlinien zu berücksichtigen. Darüber hinaus ist die

Überprüfung der Einhaltung von Sticky Policies eine anspruchsvolle Aufgabe, da die Überwachung und Verifizierung der Richtlinien über den gesamten Datenfluss hinweg erfolgen muss.

In den folgenden Abschnitten werden wir zwei Ansätze betrachten, mit denen Sticky Policies implementiert werden können. Dabei werden sich verschiedene Komplexitäten herausstellen.

3 Modellierung von Sticky Policies

3.1 Sticky Policies mit Fixpunkten

Bei der Modellierung von Sticky Policies im Zusammenhang mit greatest Fixpoints wird der mathematische Fixpunkt Operator verwendet. Dabei ist ein Fixpunkt einer Funktion auf sich selbst gerichtet. [6]Eine Funktion mit dem Wert x wird also immer den Wert x liefern. Es gilt $f(x) = x$ und daher auch

$$\hat{y}f = x$$
$$\hat{y}f = f(\hat{y}f)$$
$$\hat{y}f = f(\hat{y}\hat{y}....\hat{y}f)$$

Eine Sticky Policy wird definiert durch den Term $(\nu X.C)^{\mathcal{I}}$. Wobei X ein abzählbar unendlich großes Set an Individuen und C ein Konzept darstellt. Der Term wird betrachtet unter der Interpretation $\mathcal{I}$. In Worten sagt dieser Term, dass alle Individuen in X Teil des Konzepts C sein müssen.

Um die Modellierung von Sticy Policies für den Fall von Foxpunkten zu erläutern, betrachten wir ein Beispiel. Wir nehmen an, dass das Konzept C $\exists$**hasParent.**X sei. Die Sticky Policy lautet $(\nu X.\exists$**hasParent.**$X)^{\mathcal{I}}$. In Worten: Ein Individum e hat genau dann eine unendlich Lange Kette an **hasParent** Beziehungen, wenn für jedes Paar (e_i, e_{i+1}), mit $i > 0$ eine Beziehung **hasParent** besteht.

Um in der Praxis in einem Schritt der Übertragung verifizieren zu können, ob eine solche Kette vorliegt, benötigt man ein Verfahren welches hier als Subsumption Checking beschrieben wird. Das Subsumption Checking mit greatest Fixpoints ist in dieser Form der Policy Language coNP-schwer. Der Beweis dafür erfolgt durch eine Reduktion des Gültigkeitsproblems (validity problem) auf die Überprüfung der Subsumption.

Ein Algorithmus, der das Subsumption Checking durchführt, wäre also nicht effizient und würde bei entsprechender Datenmenge große Rechenzeiten benötigen. Daher wurden weitere Möglichkeiten untersucht.

3.2 Sticky Policies mit Transitive Role Closure

Die Transitive Closure ist eine mathematische Konstruktion, die dazu dient, transitive Relationen darzustellen. Sie wird speziell für Beziehungen/Relationen verwendet und ermöglicht eine effiziente und einfachere Reasoning-Struktur im Vergleich zur Verwendung von Fixpunkten. Dies liegt daran, dass sie weniger komplexe Ausdrücke erfordert.

Ein Beispiel für die Verwendung der Transitive Closure ist die Modellierung der Eigenschaft „Ein Mann, der nur Söhne hat“mit Hilfe der Rolle **hasChild** und des Konzepts **Men**. Hierbei wird die Transitive Closure angewendet, um die transitive Beziehung zwischen **Man** und **hasChild** zu erfassen.

Eine Sticky Policie wird mit der Form $(\nu X.\forall R.(C \sqcap X))^{\mathcal{I}}$ geschrieben. Das Beispiel der MOS „Men who Only has Sons“kann dabei mit dem Konzept C **Men** und die Rolle R **hasChild** definiert werden. Es gilt für ein Set an Individuen X, dass sie genau dann ein MOS sind, wenn $(X.\forall \mathbf{hasChild}.(\mathbf{Men} \sqcap X))^{\mathcal{I}}$ gilt.

Es wurde gezeigt, dass die Verwendung von Fixpunkten und der Transitive Closure zur Reduktion des NP-vollständigen „Exact Cover “(XC) Problems verwendet werden kann. Das XC-Problem hat genau dann eine Lösung, wenn die Reduktion ein unerfüllbares Konzept liefert, also kein Set an Individuen gefunden werden kann, welches die Bedingung erfüllt.

Ein weiterer Versuch, die Komplexität zu verringern, ist die Verwendung von Universal Restrictions und Existential Restrictions einzuschränken und sie nur in Verbindung mit der Transitive Role Closure zuzulassen. Somit ist der Ausdruck $\forall R.C$ (Alle Individuen mit Rolle R in C) nicht mehr erlaubt, der Ausdruck der Transitive Role Closure $\forall R^{+}.C$ (alle Individuen, die durch einen transitiven Schluss mit der Rolle identifizierbar und in C sind) jedoch immernoch. Es wurde gezeigt, dass in diesem Fall eine Reduktion des XC-Problems nicht mehr möglich ist und die Komplexität der Modellierung mit dieser Einschränkung NP-hard (statt vorher NP-vollständig) ist.

4 Anwendungsfälle

In verschiedenen Bereichen finden Sticky Policies Anwendung, darunter auch in der Creative Commons Lizenz und die Lizenzierung im Bereich der Finanzdatenmärkte. Insbesondere in Bezug auf den Support für Datenübertragungen und die Notwendigkeit einer umfassenden und zuverlässigen Compliance-Prüfung, um das Risiko von Sanktionen aufgrund von Lizenzverletzungen zu reduzieren.

In der Creative Commons Lizenz versucht man allgeines Gut zu schützen und allen zugänglich zu machen. Durch Sticky Policies kann man dieses effektiv umsetzen und mit einem hohen Sicherheitsgrad gegen Ausnutzung schützen.

In den letzten Monaten ist ein neuer Diskussionsbereich entstanden. Durch die vermehrte Generierung von Daten mithilfe Künstlicher Intelligenz gibt es Ansätze künstlich generierte Daten mit einem Label zu versehen, sodass für jeden Nutzer erkennbar ist, woher diese Daten stammen. Hierbei können die Erkenntnisse zu Sticky Policies große Hilfe leisten.

5 Fazit

OWL 2 in Verbindung mit Sticky Policies zeigt viel Potenzial für die Verwaltung von Richtlinien und die Durchsetzung von Compliance. Jedoch sind die Berechnungszeiten noch zu komplex und die Anwendung ist noch nicht weit genug verbreitet. Weitere Fortschritte sind erforderlich, um die Effizienz zu verbessern und die Akzeptanz in verschiedenen Branchen zu steigern.

A Literaturverzeichnis

[1] G. Schreiber, "W3c owl web ontology language overview." `https://www.w3.org/TR/owl-features`, 2004.

[2] M. Héon and G. Paquette, "G-owl: A complete visual syntax for owl 2 ontology modeling and communication," *Semantic Web*, 08 2020.

[3] P. Ciccarese and S. Peroni, "The collections ontology: Creating and handling collections in owl 2 dl frameworks," *Semantic Web*, vol. 5, pp. 515–529, 2014.

[4] F. Baader and W. Nutt, *Basic Description Logics*, p. 47–104. Cambridge University Press, 2 ed., 2007.

[5] P. A. Bonatti and L. Sauro, "Sticky Policies in OWL2: Extending PL with Fixpoints and Transitive Closure," in *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 73–80, 8 2022.

[6] C. Lutz, R. Piro, and F. Wolter, "Enriching el-concepts with greatest fixpoints," vol. 215, pp. 41–46, 08 2010.