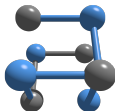


Learning OWL Class Expressions

betreut von Prof. Klaus-Peter Fährnich

Dissertations-Verteidigungsvortrag

Dipl.-Inf. Jens Lehmann



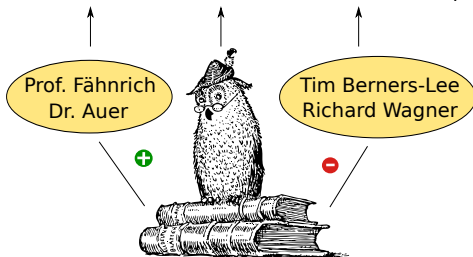
UNIVERSITÄT LEIPZIG

9. Juni 2010

Was bedeutet “Learning OWL Class Expressions”?

- gegeben:
 - **Hintergrundwissen** (speziell: OWL/DL Wissensbasis)
 - **positive** und **negative Beispiele** (speziell: Objekte in Wissensbasis)
- gesucht:
 - logische Formel (speziell: **OWL Class Expression**), die möglichst alle positiven und keines der negativen Beispiele abdeckt

Researcher and affiliation hasValue UniversitätLeipzig

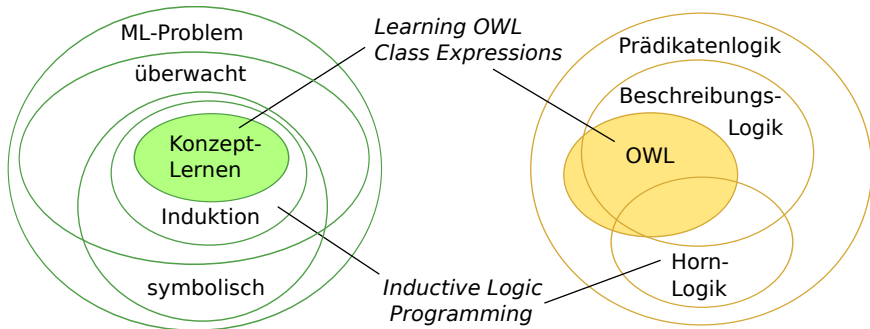


Hintergrundwissen

- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

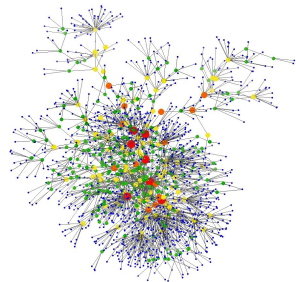
- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

Motivation - Forschungsperspektive



- seit Anfang 90er Jahre Induktive Logikprogrammierung
- nur wenige Ansätze basieren auf Beschreibungslogiken
- Web Ontology Language (OWL) wird 2004 W3C-Standard
- steigende Anzahl von RDF/OWL Wissensbasen, aber ILP noch stark auf Logikprogrammierung fixiert $\rightsquigarrow$ Forschungslücke

- **Ontology-Engineering** schwierig und ein Hauptproblem im Semantic Web
 - Domänenexperten oft keine Ontologie-Experten
 - halbautomatische Hilfsmethoden zur Konstruktion ausdrucksstarker Ontologien
- Verbesserung von Lösungen **existierender Lernprobleme**
- **direkte Nutzung des** Wissens im **Semantic Web** statt Konvertierung um ML-Werkzeuge anwenden zu können



Ontologienetzwerk



*Machine Learning
Probleme*

- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

- **Description Logics: Familie von Sprachen** zur Wissensrepräsentation
- Fragmente der Prädikatenlogik
- weniger ausdrucksstark als Prädikatenlogik, aber in der Regel **entscheidbare Inferenzprobleme**
- **intuitive variablenfreie Syntax**



- **Description Logics: Familie von Sprachen** zur Wissensrepräsentation
- Fragmente der Prädikatenlogik
- weniger ausdrucksstark als Prädikatenlogik, aber in der Regel **entscheidbare Inferenzprobleme**
- **intuitive variablenfreie Syntax**
- Basis der Ontologiesprache **OWL**
- **OWL** steht für **Web Ontology Language**
- **W3C recommendation** seit 2004 (OWL 2 seit 2009)



- Repräsentation von Wissen mit Rollen, Konzepten und Objekten

- Repräsentation von Wissen mit Rollen, Konzepten und Objekten
- **Objekte/Individuen**
 - entsprechen Konstanten
 - Beispiele: MARIA, LEIPZIG

- Repräsentation von Wissen mit Rollen, Konzepten und Objekten
- **Objekte/Individuen**
 - entsprechen Konstanten
 - Beispiele: MARIA, LEIPZIG
- **Konzepte/Klassen**
 - entsprechen einstelligen Prädikaten
 - Menge von Objekten
 - Beispiele: Student, Auto, Land

- Repräsentation von Wissen mit Rollen, Konzepten und Objekten
- **Objekte/Individuen**
 - entsprechen Konstanten
 - Beispiele: MARIA, LEIPZIG
- **Konzepte/Klassen**
 - entsprechen einstelligen Prädikaten
 - Menge von Objekten
 - Beispiele: Student, Auto, Land
- **Rollen/Properties**
 - entsprechen zweistelligen Prädikaten
 - beschreiben Verbindung zwischen Objekten
 - Beispiele: hatKind, istTeilVon

- Repräsentation von Wissen mit Rollen, Konzepten und Objekten
- **Objekte/Individuen**
 - entsprechen Konstanten
 - Beispiele: MARIA, LEIPZIG
- **Konzepte/Klassen**
 - entsprechen einstelligen Prädikaten
 - Menge von Objekten
 - Beispiele: Student, Auto, Land
- **Rollen/Properties**
 - entsprechen zweistelligen Prädikaten
 - beschreiben Verbindung zwischen Objekten
 - Beispiele: hatKind, istTeilVon
- können zu **(komplexen) Konzepten** kombiniert werden, z.B.:
 $\text{Kind} \sqcap \exists \text{hatElternteil}.\text{Professor}$
- komplexes Konzept $\hat{=}$ **OWL Class Expression**

Eine DL-Wissensbasis hat folgende Struktur:

Wissensbasis

TBox $\mathcal{T}$ (Terminologie/Schema)

ABox $\mathcal{A}$ ("assertions"/Instanzdaten)

Eine DL-Wissensbasis hat folgende Struktur:

Wissensbasis

TBox $\mathcal{T}$ (Terminologie/Schema), z.B.

$\text{Student} \equiv \text{Person} \sqcap \exists \text{studiertAn.Universität}$

$\text{Doktorand} \sqsubseteq \exists \text{arbeitetAn.}(\text{Universität} \sqcup \text{Hochschule})$

ABox $\mathcal{A}$ (“assertions”/Instanzdaten)

Eine DL-Wissensbasis hat folgende Struktur:

Wissensbasis

TBox $\mathcal{T}$ (Terminologie/Schema), z.B.

$\text{Student} \sqsubseteq \text{Person} \sqcap \exists \text{studiertAn.Universität}$

$\text{Doktorand} \sqsubseteq \exists \text{arbeitetAn.}(\text{Universität} \sqcup \text{Hochschule})$

ABox $\mathcal{A}$ (“assertions”/Instanzdaten), z.B.

$\text{Doktorand}(\text{SEBASTIAN})$

$\text{arbeitetAn}(\text{SEBASTIAN}, \text{LEIPZIG_UNIVERSITÄT})$

- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

Refinementoperator - Definition

- gegeben sei eine DL $\mathcal{L}$ und einen quasi-geordneten Raum $\langle \mathcal{C}(\mathcal{L}), \sqsubseteq_{\mathcal{T}} \rangle$ über Konzepten in $\mathcal{L}$
- $\rho : \mathcal{C}(\mathcal{L}) \rightarrow 2^{\mathcal{C}(\mathcal{L})}$ ist ein **downward $\mathcal{L}$ refinement operator** wenn für alle $C \in \mathcal{C}(\mathcal{L})$ gilt:

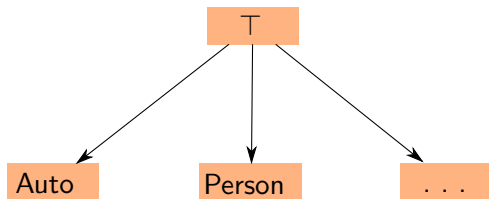
$$D \in \rho(C) \text{ impliziert } D \sqsubseteq_{\mathcal{T}} C$$

- Notation: oft $C \rightsquigarrow_{\rho} D$ statt $D \in \rho(C)$
- **Refinementkette (refinement chain):**

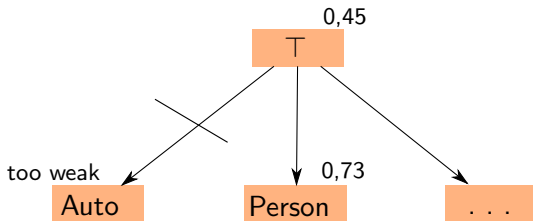
$$\top \rightsquigarrow_{\rho} \text{Person} \rightsquigarrow_{\rho} \text{Mann} \rightsquigarrow_{\rho} \text{Mann} \sqcap \exists \text{hatKind}.\top$$

T

- Start mit
allgemeinstem
Konzept (top down)

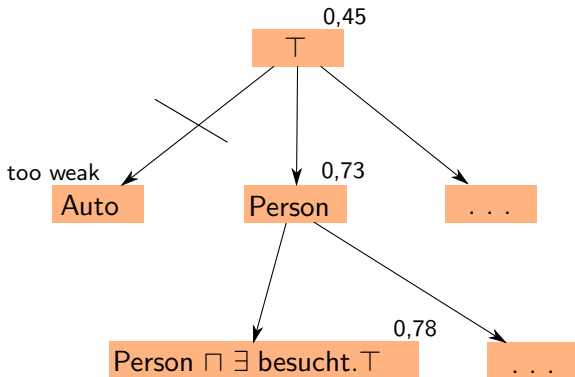


- Start mit allgemeinstem Konzept (top down)
- **Operator** spezialisiert

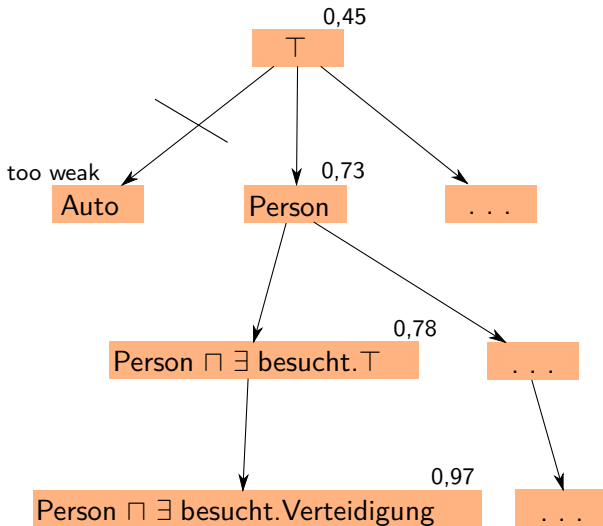


- Start mit allgemeinstem Konzept (top down)
- **Operator** spezialisiert
- **Heuristik** bewertet unter Nutzung von pos/neg Beispielen

Lernen mit Refinementoperatoren



- Start mit allgemeinstem Konzept (top down)
- **Operator** spezialisiert
- **Heuristik** bewertet unter Nutzung von pos/neg Beispielen
- Fortsetzung bis **Abbruchkriterium**

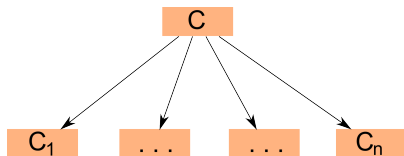


- Start mit allgemeinstem Konzept (top down)
- **Operator** spezialisiert
- **Heuristik** bewertet unter Nutzung von pos/neg Beispielen
- Fortsetzung bis **Abbruchkriterium**
=
Lernalgorithmus

Eigenschaften von Refinementoperatoren

Ein $\mathcal{L}$ downward refinement operator ρ ist

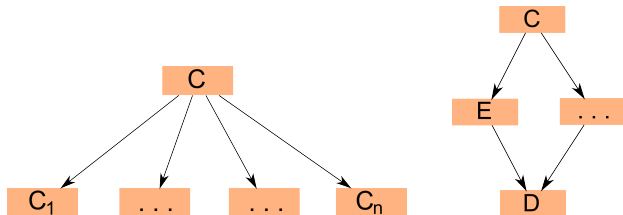
- **finite (endlich)** falls $\rho(C)$ für jedes $C \in \mathcal{C}(\mathcal{L})$ endlich ist



Eigenschaften von Refinementoperatoren

Ein $\mathcal{L}$ downward refinement operator ρ ist

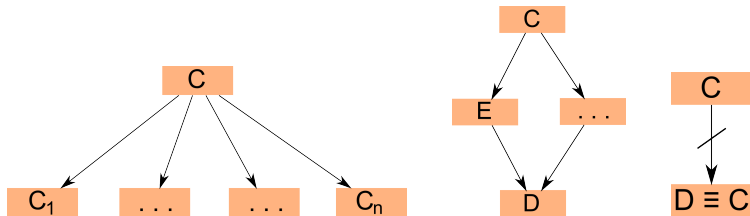
- **finite (endlich)** falls $\rho(C)$ für jedes $C \in \mathcal{C}(\mathcal{L})$ endlich ist
- **redundant** falls mehrere ρ -Refinementketten von einem Konzept C zu einem Konzept D existieren



Eigenschaften von Refinementoperatoren

Ein $\mathcal{L}$ downward refinement operator ρ ist

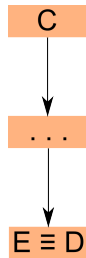
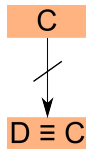
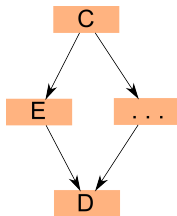
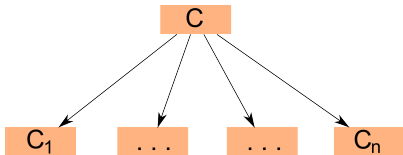
- **finite (endlich)** falls $\rho(C)$ für jedes $C \in \mathcal{C}(\mathcal{L})$ endlich ist
- **redundant** falls mehrere ρ -Refinementketten von einem Konzept C zu einem Konzept D existieren
- **proper (echt)** falls $C \rightsquigarrow_{\rho} D$ impliziert $C \not\equiv_{\mathcal{T}} D$



Eigenschaften von Refinementoperatoren

Ein $\mathcal{L}$ downward refinement operator ρ ist

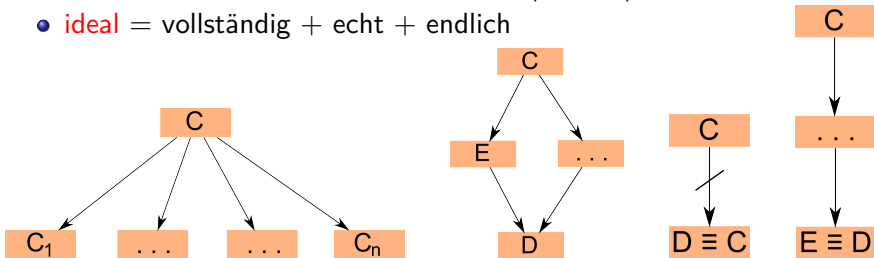
- **finite (endlich)** falls $\rho(C)$ für jedes $C \in \mathcal{C}(\mathcal{L})$ endlich ist
- **redundant** falls mehrere ρ -Refinementketten von einem Konzept C zu einem Konzept D existieren
- **proper (echt)** falls $C \rightsquigarrow_\rho D$ impliziert $C \not\sqsubseteq_{\mathcal{T}} D$
- **complete (vollständig)** falls es für alle $C, D \in \mathcal{C}(\mathcal{L})$ mit $D \sqsubseteq_{\mathcal{T}} C$ ein Konzept $E \equiv_{\mathcal{T}} D$ mit $C \rightsquigarrow_\rho \dots \rightsquigarrow_\rho E$ gibt
- **weakly complete (schwach vollständig)** falls für alle $C \in \mathcal{C}(\mathcal{L})$ mit $C \sqsubseteq_{\mathcal{T}} \top$ ein Konzept $D \equiv C$ mit $\top \rightsquigarrow_\rho \dots \rightsquigarrow_\rho D$ gibt



Eigenschaften von Refinementoperatoren

Ein $\mathcal{L}$ downward refinement operator ρ ist

- **finite (endlich)** falls $\rho(C)$ für jedes $C \in \mathcal{C}(\mathcal{L})$ endlich ist
- **redundant** falls mehrere ρ -Refinementketten von einem Konzept C zu einem Konzept D existieren
- **proper (echt)** falls $C \rightsquigarrow_\rho D$ impliziert $C \not\sqsubseteq_{\mathcal{T}} D$
- **complete (vollständig)** falls es für alle $C, D \in \mathcal{C}(\mathcal{L})$ mit $D \sqsubseteq_{\mathcal{T}} C$ ein Konzept $E \equiv_{\mathcal{T}} D$ mit $C \rightsquigarrow_\rho \dots \rightsquigarrow_\rho E$ gibt
- **weakly complete (schwach vollständig)** falls für alle $C \in \mathcal{C}(\mathcal{L})$ mit $C \sqsubseteq_{\mathcal{T}} \top$ ein Konzept $D \equiv C$ mit $\top \rightsquigarrow_\rho \dots \rightsquigarrow_\rho D$ gibt
- **ideal** = vollständig + echt + endlich



Eigenschaften von Refinementoperatoren

Eigenschaft	Effekt
unvollständig	Verpassen von Lösungen
redundant	doppelte Teilbäume, schlechtere Performance
unecht	schlechtere Performance
unendlich	Refinementberechnung terminiert nicht immer

Schlüsselfrage: Welche Eigenschaften können kombiniert werden?

Theorem zu Eigenschaften von $\mathcal{L}$ Refinementoperatoren

Theorem

Maximale kombinierbare Mengen von Eigenschaften von $\mathcal{L}$ Refinementoperatoren für $\mathcal{L} \in \{ALC, ALCN, SHOIN, SROIQ\}$ sind:

- ① {schwach vollständig, vollständig, endlich}
- ② {schwach vollständig, vollständig, echt}
- ③ {schwach vollständig, nicht-redundant, endlich}
- ④ {schwach vollständig, nicht-redundant, echt}
- ⑤ {nicht-redundant, endlich, echt}

 "Foundations of Refinement Operators for Description Logics",
J. Lehmann, P. Hitzler, ILP conference, 2008

 "Concept Learning in Description Logics Using Refinement Operators",
J. Lehmann, P. Hitzler, Machine Learning journal, 2010

Theorem zu Eigenschaften von $\mathcal{L}$ Refinementoperatoren

- keine idealen Operatoren in OWL und den meisten Beschreibungslogiken außer $\mathcal{EL}$ -Familie
- Theorem widerlegt teilweise vorher publizierte Resultate
- Indikator dafür dass Lernen in DLs schwierig ist
- Algorithmen sollten Maßnahmen ergreifen um Nachteile der Operatoren auszugleichen
- Ziel 1: Entwicklung eines OWL Refinementoperators nahe an theoretischen Beschränkungen
- Ziel 2: effizienter idealer Refinementoperator für leichtgewichtige $\mathcal{EL}$ Beschreibungslogik

Definition von ρ

$$\rho(C) = \begin{cases} \{\perp\} \cup \rho_{\top}(C) & \text{if } C = \top \\ \rho_{\top}(C) & \text{otherwise} \end{cases}$$

$$\rho_B(C) = \begin{cases} \emptyset & \text{if } C = \perp \\ \{C_1 \sqcup \dots \sqcup C_n \mid C_i \in M_B \ (1 \leq i \leq n)\} & \text{if } C = \top \\ \{A' \mid A' \in sh_{\downarrow}(A)\} & \text{if } C = A \ (A \in N_C) \\ \quad \cup \{A \sqcap D \mid D \in \rho_B(\top)\} & \\ \{\neg A' \mid A' \in sh_{\uparrow}(A)\} & \text{if } C = \neg A \ (A \in N_C) \\ \quad \cup \{\neg A \sqcap D \mid D \in \rho_B(\top)\} & \\ \{\exists r.E \mid A = ar(r), E \in \rho_A(D)\} & \text{if } C = \exists r.D \\ \quad \cup \{\exists r.D \sqcap E \mid E \in \rho_B(\top)\} & \\ \quad \cup \{\exists s.D \mid s \in sh_{\downarrow}(r)\} & \\ \{\forall r.E \mid A = ar(r), E \in \rho_A(D)\} & \text{if } C = \forall r.D \\ \quad \cup \{\forall r.D \sqcap E \mid E \in \rho_B(\top)\} & \\ \quad \cup \{\forall r.\perp \mid & \\ \quad \quad D = A \in N_C \text{ and } sh_{\downarrow}(A) = \emptyset\} & \\ \quad \cup \{\forall s.D \mid s \in sh_{\downarrow}(r)\} & \\ \{C_1 \sqcap \dots \sqcap C_{i-1} \sqcap D \sqcap C_{i+1} \sqcap \dots \sqcap C_n \mid & \text{if } C = C_1 \sqcap \dots \sqcap C_n \\ \quad D \in \rho_B(C_i), 1 \leq i \leq n\} & (n \geq 2) \\ \{C_1 \sqcup \dots \sqcup C_{i-1} \sqcup D \sqcup C_{i+1} \sqcup \dots \sqcup C_n \mid & \text{if } C = C_1 \sqcup \dots \sqcup C_n \\ \quad D \in \rho_B(C_i), 1 \leq i \leq n\} & (n \geq 2) \\ \quad \cup \{(C_1 \sqcup \dots \sqcup C_n) \sqcap D \mid & \\ \quad \quad D \in \rho_B(\top)\} & \end{cases}$$

Basisoperator (Auszug)

Definition von ρ

$$\rho(C) = \begin{cases} \{\perp\} \cup \rho_{\top}(C) & \text{if } C = \top \\ \rho_{\top}(C) & \text{otherwise} \end{cases}$$



$$\rho_B(C) = \begin{cases} \emptyset & \text{if } C = \perp \\ \{C_1 \sqcup \dots \sqcup C_n \mid C_i \in M_B \ (1 \leq i \leq n)\} & \text{if } C = \top \\ \{A' \mid A' \in sh_{\downarrow}(A)\} & \text{if } C = A \ (A \in N_C) \\ \quad \cup \{A \sqcap D \mid D \in \rho_B(\top)\} \\ \{\neg A' \mid A' \in sh_{\uparrow}(A)\} & \text{if } C = \neg A \ (A \in N_C) \\ \quad \cup \{\neg A \sqcap D \mid D \in \rho_B(\top)\} \\ \boxed{\{\exists r. E \mid A = ar(r), E \in \rho_A(D)\}} & \text{if } C = \exists r. D \\ \quad \cup \{\exists r. D \sqcap E \mid E \in \rho_B(\top)\} \\ \quad \cup \{\exists s. D \mid s \in sh_{\downarrow}(r)\} \\ \{\forall r. E \mid A = ar(r), E \in \rho_A(D)\} & \text{if } C = \forall r. D \\ \quad \cup \{\forall r. D \sqcap E \mid E \in \rho_B(\top)\} \\ \quad \cup \{\forall r. \perp \mid \\ \quad \quad D = A \in N_C \text{ and } sh_{\downarrow}(A) = \emptyset\} \\ \quad \cup \{\forall s. D \mid s \in sh_{\downarrow}(r)\} \\ \{C_1 \sqcap \dots \sqcap C_{i-1} \sqcap D \sqcap C_{i+1} \sqcap \dots \sqcap C_n \mid & \text{if } C = C_1 \sqcap \dots \sqcap C_n \\ \quad D \in \rho_B(C_i), 1 \leq i \leq n\} & (n \geq 2) \\ \{C_1 \sqcup \dots \sqcup C_{i-1} \sqcup D \sqcup C_{i+1} \sqcup \dots \sqcup C_n \mid & \text{if } C = C_1 \sqcup \dots \sqcup C_n \\ \quad D \in \rho_B(C_i), 1 \leq i \leq n\} & (n \geq 2) \\ \quad \cup \{(C_1 \sqcup \dots \sqcup C_n) \sqcap D \mid \\ \quad \quad D \in \rho_B(\top)\} \end{cases}$$

Basisoperator (Auszug)

Definition von ρ

$$\begin{aligned} & \{\exists r.E \mid A = ar(r), E \in \rho_A(D)\} && \text{if } C = \exists r.D \\ & \cup \{\exists r.D \sqcap E \mid E \in \rho_B(\top)\} \\ & \cup \{\exists s.D \mid s \in sh_{\downarrow}(r)\} \end{aligned}$$

Beispiele:

`∃nimmtTeilAn.SozialeZusammenkunft` $\rightsquigarrow$

`∃nimmtTeilAn.Meeting`

Definition von ρ

$$\begin{aligned} & \{\exists r.E \mid A = ar(r), E \in \rho_A(D)\} && \text{if } C = \exists r.D \\ & \cup \{\exists r.D \sqcap E \mid E \in \rho_B(\top)\} \\ & \cup \{\exists s.D \mid s \in sh_{\downarrow}(r)\} \end{aligned}$$

Beispiele:

$\exists \text{nimmtTeilAn.SozialeZusammenkunft} \rightsquigarrow$

$\exists \text{nimmtTeilAn.Meeting}$

$\text{Student} \sqcap \exists \text{nimmtTeilAn.SozialeZusammenkunft}$

Definition von ρ

$$\begin{aligned} & \{\exists r.E \mid A = ar(r), E \in \rho_A(D)\} && \text{if } C = \exists r.D \\ & \cup \{\exists r.D \sqcap E \mid E \in \rho_B(\top)\} \\ & \cup \{\exists s.D \mid s \in sh_{\downarrow}(r)\} \end{aligned}$$

Beispiele:

$$\begin{aligned} & \exists \text{nimmtTeilAn.SozialeZusammenkunft} \rightsquigarrow \\ & \exists \text{nimmtTeilAn.Meeting} \\ & \text{Student} \sqcap \exists \text{nimmtTeilAn.SozialeZusammenkunft} \\ & \exists \text{leitet.SozialeZusammenkunft} \end{aligned}$$

- $\rho_{\downarrow}$ ist **vollständig**
- $\rho_{\downarrow}$ ist **unendlich**, z.B. gibt es unendlich viele Refinementschritte der Form:

$$\top \rightsquigarrow_{\rho_{\downarrow}} C_1 \sqcup C_2 \sqcup C_3 \sqcup \dots$$

- $\rho_{\downarrow}$ ist **unecht**, aber kann erweitert werden zu einem **echten Operator** $\rho_{\downarrow}^{cl}$ (Refinements zeitaufwändiger zu berechnen)
- $\rho_{\downarrow}$ ist **redundant**: $\forall r_1.A_1 \sqcup \forall r_2.A_1 \rightsquigarrow_{\rho_{\downarrow}} \forall r_1.(A_1 \sqcap A_2) \sqcup \forall r_2.A_1$

$$\begin{array}{ccc} \downarrow & & \downarrow \\ \forall r_1.A_1 \sqcup \forall r_2.(A_1 \sqcap A_2) & \rightsquigarrow_{\rho_{\downarrow}} & \forall r_1.(A_1 \sqcap A_2) \sqcup \forall r_2.(A_1 \sqcap A_2) \end{array}$$

 “A Refinement Operator Based Learning Algorithm for the $\mathcal{ALC}$ Description Logic”, J. Lehmann, P. Hitzler, ILP conference, 2008

 “Concept Learning in Description Logics Using Refinement Operators”, J. Lehmann, P. Hitzler, Machine Learning journal, 2010

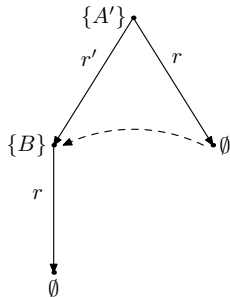
Konstruktor	Syntax
top	$\top$
Konjunktion	$C \sqcap D$
existenzielle Restriktion	$\exists r.C$

Tabelle: $\mathcal{EL}$ Konzeptkonstruktoren

- motiviert durch niedrige Reasoning-Komplexität von $\mathcal{EL}$ und **EL Profil** in OWL 2
- $\mathcal{EL}$ weit verbreitet z.B. SNOMED-CT, GEN-ONTOLOGIEN

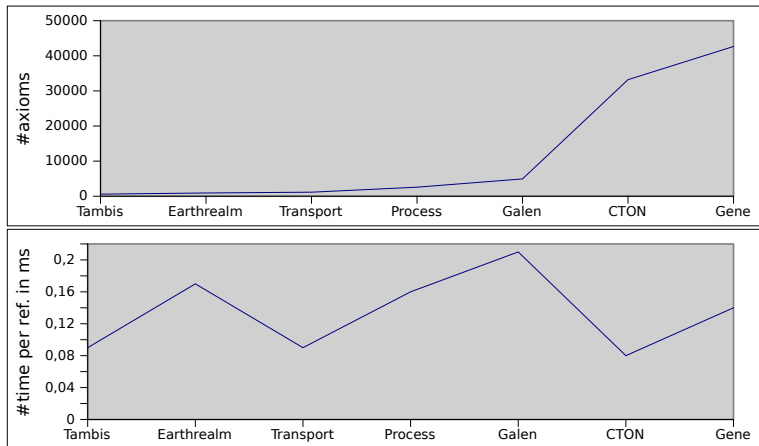
Konstruktor	Syntax
top	$\top$
Konjunktion	$C \sqcap D$
existenzielle Restriktion	$\exists r.C$

Tabelle: $\mathcal{EL}$ Konzeptkonstruktoren



- motiviert durch niedrige Reasoning-Komplexität von $\mathcal{EL}$ und **EL Profil** in OWL 2
- $\mathcal{EL}$ weit verbreitet z.B. SNOMED-CT, GEN-ONTOLOGIEN
- $\mathcal{EL}$ -Konzepte (class expressions) als **Bäume** repräsentieren
- **Reasoning durch sogenannte "Simulationen"** auf Bäumen
- Refinement und Reasoning sind stark miteinander verwoben

Operatoreffizienz auf verschiedenen Ontologien



< 1 ms pro Refinement (gemessen über "random refinement chains")



"Ideal Downward Refinement in the EL Description Logic",

J. Lehmann, C. Haase, Int. Conf. on Inductive Logic Programming, 2009

- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

- ① OCEL - OWL Class Expression Learner
- ② ELTL - $\mathcal{EL}$ Tree Learner
- ③ CELOE - Class Expression Learning for Ontology Engineering

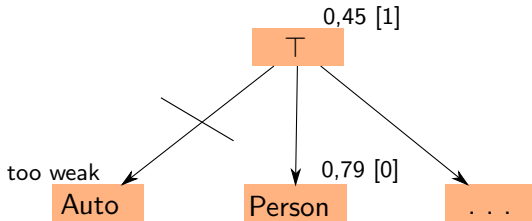
- verwendet ρ für top-down Suche
- OCEL ist vollständig - falls Lösung existiert, kann sie gefunden werden
- stark konfigurierbar, z.B. flexible Wahl von Ziel-/Hypothesensprache, Terminierungskriterium und Heuristik
- implementiert Redundanzeliminierung mit niedriger Komplexität bzgl. Größe des Suchbaums
- kann unendliche Operatoren verwenden durch schrittweise längenbeschränkte Knotenexpansion

0,47 [0]

T

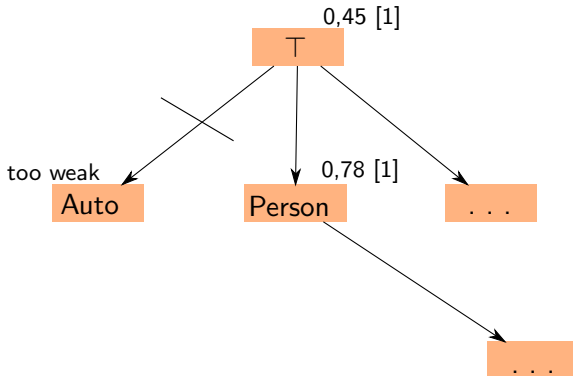
- Länge der Kinderkonzepte begrenzt durch horizontal expansion (he)

Schrittweise Längenbeschränkte Knotenerkennung



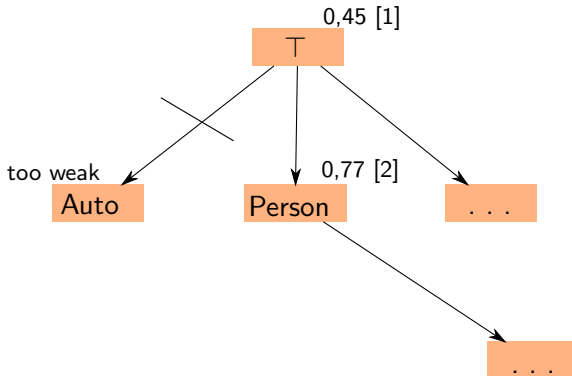
- Länge der Kinderkonzepte begrenzt durch **horizontal expansion** (he)
- dadurch ρ (infinite) anwendbar

Schrittweise Längenbeschränkte Knotenerxpansion



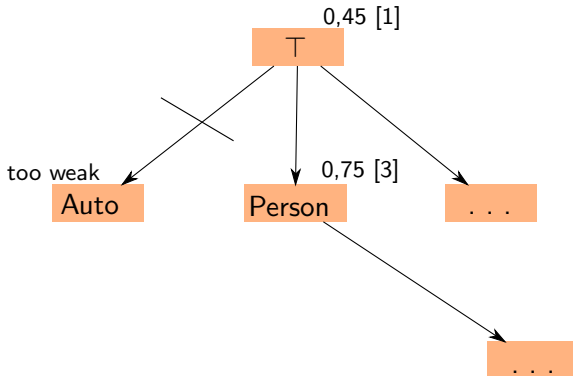
- Länge der Kinderkonzepte begrenzt durch **horizontal expansion** (he)
- dadurch ρ (infinite) anwendbar
- he **geht in Heuristik ein** (Bias zu kurzen Konzepten - Occam's Razor, größere Streuung)

Schrittweise Längenbeschränkte Knotenerxpansion



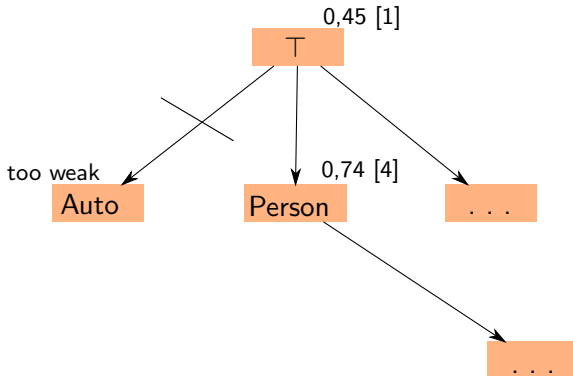
- Länge der Kinderkonzepte begrenzt durch **horizontal expansion** (he)
- dadurch ρ (infinite) anwendbar
- he **geht in Heuristik ein** (Bias zu kurzen Konzepten - Occam's Razor, größere Streuung)

Schrittweise Längenbeschränkte Knotenerpansion



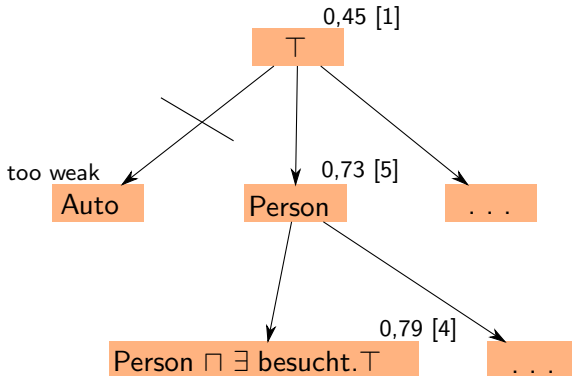
- Länge der Kinderkonzepte begrenzt durch **horizontal expansion** (he)
- dadurch ρ (infinite) anwendbar
- he **geht in Heuristik ein** (Bias zu kurzen Konzepten - Occam's Razor, größere Streuung)

Schrittweise Längenbeschränkte Knotenerpansion



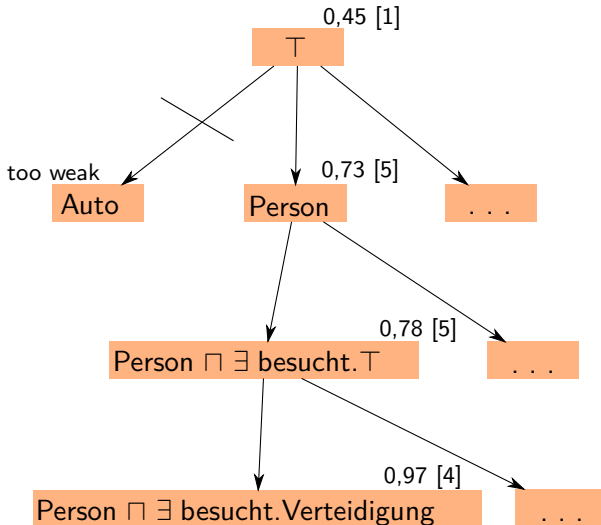
- Länge der Kinderkonzepte begrenzt durch **horizontal expansion** (he)
- dadurch ρ (infinite) anwendbar
- he **geht in Heuristik ein** (Bias zu kurzen Konzepten - Occam's Razor, größere Streuung)

Schrittweise Längenbeschränkte Knotenerxpansion



- Länge der Kinderkonzepte begrenzt durch **horizontal expansion** (he)
- dadurch ρ (infinite) anwendbar
- he **geht in Heuristik ein** (Bias zu kurzen Konzepten - Occam's Razor, größere Streuung)

Schrittweise Längenbeschränkte Knotenerxpansion



- Länge der Kinderkonzepte begrenzt durch **horizontal expansion** (he)
- dadurch ρ (infinite) anwendbar
- he **geht in Heuristik ein** (Bias zu kurzen Konzepten - Occam's Razor, größere Streuung)

$\mathcal{K} = \{\text{Männlich} \sqsubseteq \text{Person},$
 $\text{Männlich}(\text{MICHAEL}), \text{Männlich}(\text{JONAS}), \text{Männlich}(\text{PAUL}),$
 $\text{hatKind}(\text{MICHAEL}, \text{JONAS}), \text{hatKind}(\text{MICHAEL}, \text{PAUL})\}$

- positives Beispiel: MICHAEL
- $C = \text{Person} \sqcap \forall \text{hatKind. Männlich}$ scheinbar gute Lösung, aber Objekt “MICHAEL” keine Instanz von C unter OWA
- Idee: **materialisieren von $\mathcal{K}$** mit Standard (OWA) DL-Reasoner, aber **Instance Checks unter CWA** ausführen
- näher an ML-Intuition und führt zu **starker Performanceverbesserung**

Heuristiken benötigen oft viele Instance Checks oder Retrieval, z.B.:

$$\frac{1}{2} \cdot \left(\frac{|R(A) \cap R(C)|}{|R(A)|} + \sqrt{\frac{|R(A) \cap R(C)|}{|R(C)|}} \right)$$

Heuristiken benötigen oft viele Instance Checks oder Retrieval, z.B.:

$$\frac{1}{2} \cdot \left(\frac{a}{|R(A)|} + \sqrt{\frac{a}{b}} \right)$$

- $|R(A) \cap R(C)|$ und $|R(C)|$ durch abzuschätzende Variablen a und b ersetzen
- **Wald-Methode** zur Berechnung des 95% Konfidenz-Intervalls
- erst a abschätzen, danach gesamten Ausdruck (beweisbar **pessimistische Abschätzung**)
- Verfahren **anwendbar auf verschiedene Heuristiken**

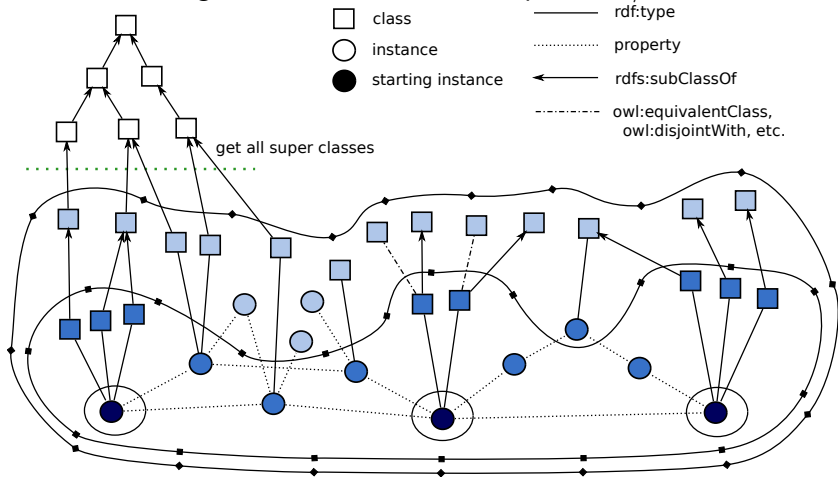
Heuristiken benötigen oft viele Instance Checks oder Retrieval, z.B.:

$$\frac{1}{2} \cdot \left(\frac{a}{|R(A)|} + \sqrt{\frac{a}{b}} \right)$$

- $|R(A) \cap R(C)|$ und $|R(C)|$ durch abzuschätzende Variablen a und b ersetzen
- **Wald-Methode** zur Berechnung des 95% Konfidenz-Intervalls
- erst a abschätzen, danach gesamten Ausdruck (beweisbar **pessimistische Abschätzung**)
- Verfahren **anwendbar auf verschiedene Heuristiken**
- in Tests auf realen Ontologien **bis zu 99% weniger Instance Checks** und dadurch Algorithmus **bis zu 30mal schneller**
- **geringer Einfluss auf Lernergebnisse** empirisch in 380 Lernproblemen auf 7 realen Ontologien nachgewiesen (Abweichung ca. $0,2\% \pm 0,4\%$)

Skalierbarkeit: Fragmentextraktion

Extraktion von Fragmenten aus SPARQL Endpunkten / Linked Data:

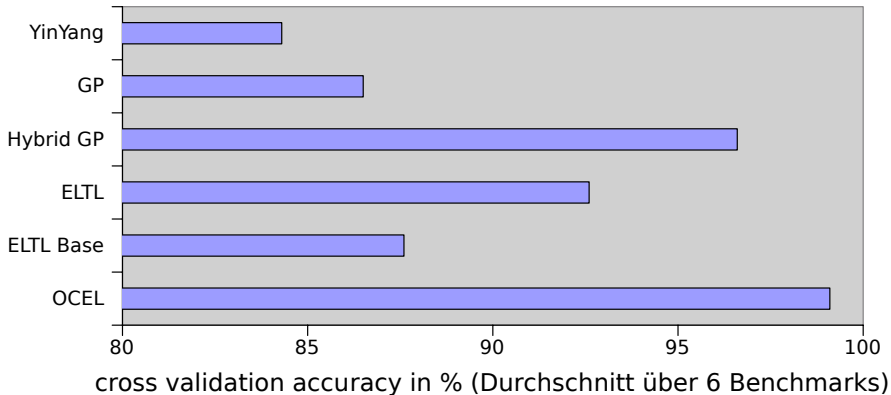


*"Learning of OWL Class Descriptions on Very Large Knowledge Bases",
Hellmann, Lehmann, Auer, Int. Journal Semantic Web Inf. Syst, 2009*

- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

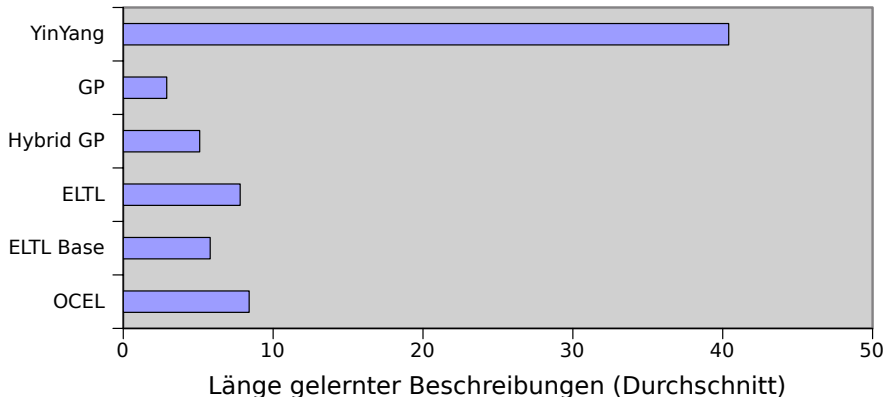
- Mangel an Evaluationsstandards für Lernen in OWL/DL
- Vorgehen: Konvertieren von existierenden Benchmarks nach OWL (zeitaufwändig, erfordert Domänenwissen)
- Messen von predictive accuracy mit ten fold cross validation
- Teil 1: Evaluation gegenüber OWL/DL Lernsystemen
- Teil 2: Evaluation gegenüber anderen ML-Systemen (Krebserkennungsproblem)
- Teil 3: Evaluation im Bereich Ontology Engineering

Evaluation: Genauigkeit/Accuracy



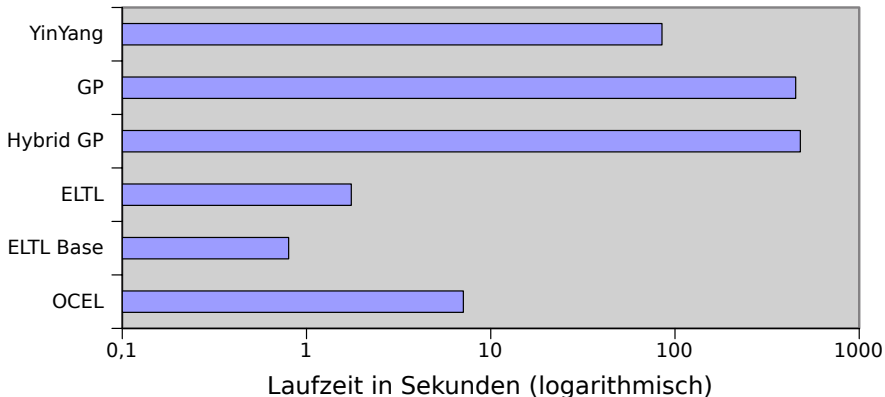
- Zusammenfassung von 6 Benchmarks
- OCEL häufig statistisch signifikant besser als andere Algorithmen für die meisten Benchmarks
- ELTL beschränkt durch Ausdrucksstärke bei einigen Benchmarks

Evaluation: Länge/Lesbarkeit



- YinYang erzeugt statistisch signifikant längere Lösungen

Evaluation: Laufzeit

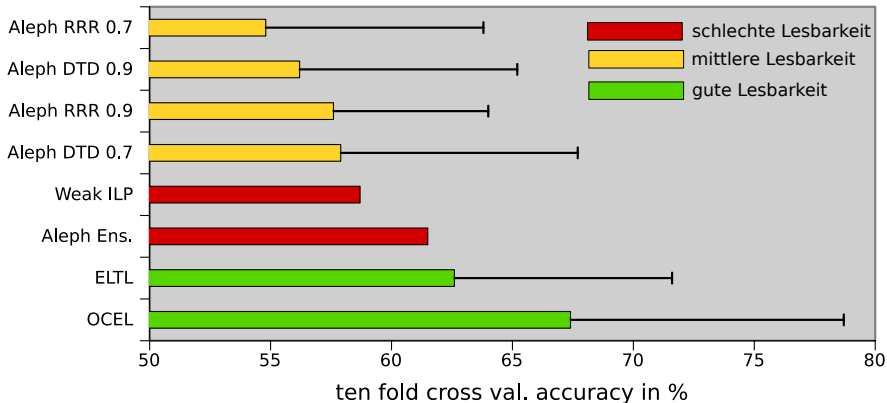


- Ziel: **vorhersagen** ob Substanzen **krebserregend** sind
- Begründung:
 - jedes Jahr 1000 **neue Substanzen**
 - Substanzen können nur teilweise durch **langwierige und kostspielige Experimente** mit Ratten/Mäusen auf Krebsrisiko untersucht werden
- Hintergrundwissen:
 - Datenbank des **US National Toxicology Program** (NTP)
 - Konvertiert von Prolog zu OWL



“Obtaining accurate structural alerts for the causes of chemical cancers is a problem of great scientific and humanitarian value.” (A. Srinivasan, R.D. King, S.H. Muggleton, M.J.E. Sternberg 1997)

Carcinogenesis



- sehr schwieriges Problem: niedrige Genauigkeit, hohe Standardabweichung
- OCEL stat. signifikant besser als die meisten anderen Ansätze

- initialer Test:
 - zwei Masterstudenten mit Einarbeitungszeit in verschiedene Domänen
 - Test auf 7 realen Ontologien
 - 383 Lernprobleme
 - für 60% der Probleme konnte CELOE akzeptierte Vorschläge zur Erweiterung der Ontologie machen
 - 4197 neue Instanzzuordnungen
 - 2 “versteckte Inkonsistenzen” aufgedeckt
- momentan: umfangreicherer Test mit Protégé Evaluationsplugin

- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

DL-Learner Projekt

- **DL-Learner** Open-Source-Projekt: <http://dl-learner.org>, <http://sf.net/projects/dl-learner>
- **leicht erweiterbare Plattform** für verschiedene Lernprobleme und Lernansätze (4 Komponententypen, 21 Komponenten)
- mehrere **Interfaces**: Kommandozeile, GUI, Web-Service
- unterstützt gängige OWL-Formate und SPARQL Endpunkte
- erlaubt verschiedene **Reasoner-Anbindungen** (OWL API, DIG)
- sourceforge.net (Open Source Portal): 3500 Downloads
- mloss.org (ML & Open Source Software): 1400 Downloads



"DL-Learner: Learning Concepts in Description Logics",

Jens Lehmann, Journal of Machine Learning Research (JMLR), 2009

- „klassische“ ML-Probleme
 - krebserregende Substanzen
 - Erbkrankheiten (Mutation)

- „klassische“ ML-Probleme
 - krebserregende Substanzen
 - Erbkrankheiten (Mutation)
- Ontology Engineering
 - Protégé Plugin

The screenshot shows the 'CustomerRequirement' window of the DL-Learner plugin. It features three tabs: 'Object restriction creator', 'Data restriction creator', and 'DL-Learner'. The 'DL-Learner' tab is active, showing a 'Class expression editor' and an 'Asserted class hierarchy'. The class expression editor contains a list of suggested equivalent class expressions with their accuracies. The asserted class hierarchy shows a tree structure with 'CustomerRequirement' at the top, followed by '(Requirement and (QualityRequirement or isCreatedBy some Customer))', and then '(QualityRequirement or isCreatedBy some Customer)'. A legend on the right explains the colors used in the hierarchy: yellow for 'CustomerRequirement', orange for '(Requirement and (QualityRequirement or isCreatedBy some Customer))', and green for '(QualityRequirement or isCreatedBy some Customer)'. Below the legend, there are three checkboxes: 'Individuals covered by (OK)', 'Individuals covered by (potential problem)', and 'Individuals covered by (potential problem)'. At the bottom, there are three sliders for 'noise in %', 'maximum execution time', and 'max. number of results'. The 'noise in %' slider is set to 10, 'maximum execution time' is set to 10, and 'max. number of results' is set to 10. There are 'OK' and 'Abbrechen' buttons at the bottom right.

CustomerRequirement

Object restriction creator Data restriction creator DL-Learner

Class expression editor

Asserted class hierarchy

suggest equivalent class expression See <http://dl-learner.org/wiki/ProtegePlugin> for an introduction.

isCreatedBy some Customer Accuracy: 100%
(Requirement and isCreatedBy some Customer) Accuracy: 100%
(QualityRequirement or isCreatedBy some Customer) Accuracy: 96%
(Requirement and (QualityRequirement or isCreatedBy some Customer)) Accuracy: 96%
isCreatedBy some (Customer or Government) Accuracy: 90%
(Requirement and isCreatedBy some (Customer or Government)) Accuracy: 90%

To view details about why a class expression was suggested, please click on it.

CustomerRequirement
(Requirement and (QualityRequirement or isCreatedBy some Customer))
Individuals covered by (OK)
Individuals covered by (potential problem)
Individuals covered by (potential problem)

Advanced Settings

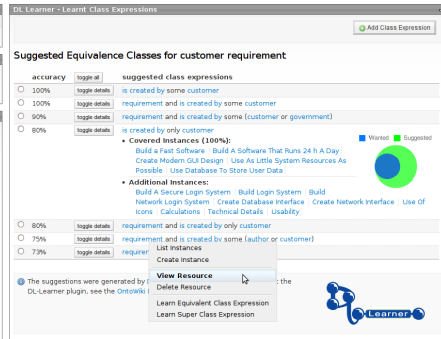
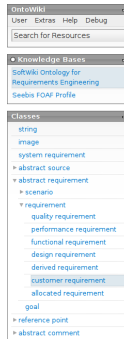
noise in % 0 10 20 30 40 50

maximum execution time: 0 10 20 30 40

max. number of results: 2 4 6 8 10 12 14 16 18 20

OK Abbrechen

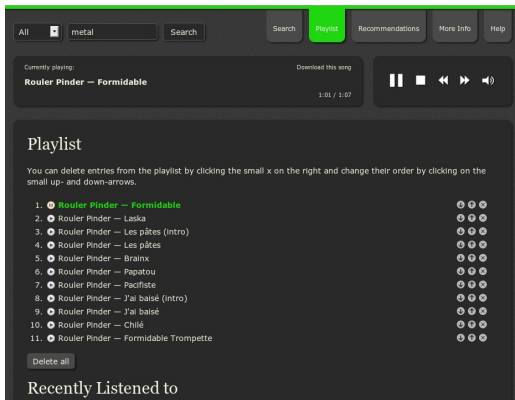
- „klassische“ ML-Probleme
 - krebserregende Substanzen
 - Erbkrankheiten (Mutation)
- Ontology Engineering
 - Protégé Plugin
 - OntoWiki Plugin



- „klassische“ ML-Probleme
 - krebserregende Substanzen
 - Erbkrankheiten (Mutation)
- Ontology Engineering
 - Protégé Plugin
 - OntoWiki Plugin
 - ORE

The screenshot shows the DL-learner Ontology Repair and Enrichment tool. The main window displays a list of axioms being processed, with columns for 'Axiom', 'ATV', 'Steps', and 'Score'. The interface is annotated with numbers 1 through 4, likely corresponding to the list items in the slide. The bottom section shows a detailed view of the ontology structure, including classes and individuals.

- „klassische“ ML-Probleme
 - krebserregende Substanzen
 - Erbkrankheiten (Mutation)
- Ontology Engineering
 - Protégé Plugin
 - OntoWiki Plugin
 - ORE
- Empfehlungen/Navigation
 - moosique.net



- „klassische“ ML-Probleme
 - krebserregende Substanzen
 - Erbkrankheiten (Mutation)
- Ontology Engineering
 - Protégé Plugin
 - OntoWiki Plugin
 - ORE
- Empfehlungen/Navigation
 - moosique.net
 - DBpedia Navigator



Navigation Suggestions

You could also be interested in articles matching these descriptions:

- an administrative district
- a city
- a district

Search DBpedia

New York

Autocomplete Search

Best Search Results

1. New York City
2. New York
3. New York Times
4. The New York Times
5. New York Yankees
6. New York Giants
7. New York Rangers
8. New York University
9. New York Metro
10. Buffalo, New York

DBpedia Navigator is powered by:

OL-Engine

DBpedia

OpenLink Virtuoso

... and implemented by Jens Lehmann and Sebastian Risse at the AODR research group (University of Leipzig)





New York City

Short Description

New York City (officially The City of New York) is the largest city in the United States, with its metropolitan area ranking among the largest urban areas in the world. Located on the country's east coast in New York State, it was founded as a commercial trading post by the Dutch in 1624, and has been the largest city in the United States since 1790. It also served as the capital of the United States from 1785 until 1790. Located on one of the world's natural harbors, New York is one of the world's major centers of commerce and finance. New York also exerts global influence in media, education, entertainment, arts, business and advertising. The city is also a major center for international affairs, housing the headquarters of the United Nations. New York City comprises five boroughs: The Bronx, Brooklyn, Manhattan, Queens, and Staten Island, each with the four counties of Bronx, Kings, New York, Queens, and Richmond respectively. With over 8.2 million residents within an area of , New York City is the most densely populated major city in the United States. Many of the city's neighborhoods and landmarks are known around the world. The Statue of Liberty greeted millions of immigrants as they came to America in the late 19th and early 20th centuries. Wall Street, in Lower Manhattan, has been a dominant global financial center since World War I and is home to the New York Stock Exchange. The city has been home to several of the tallest buildings in the world, including the Empire State Building and former the twin towers of the World Trade Center. New York is the birthplace of many cultural movements, including the Harlem Renaissance in literature and visual art, abstract expressionism (also known as the New York School) in painting, and hip hop, punk, salsa, and Tejano. New York is also the home of Broadway theater, which exists in an area often referred to as The Main Stem. The United Nations Office at the United Nations, in 2005, reports 170 languages were spoken in the city and 36% of its population was born outside the United States. With its 24-hour subway and constant building of traffic and people, New York is sometimes called "The City That Never Sleeps", "Gotham", and "The Big Apple".

View Wikipedia article, **View DBpedia resource description**, **View photo collection**

Same as

- <http://www.geonames.org/5128581/>
- http://api.dbpedia.org/resource/New_York_City

W650 Classes

- city in the United States → search instances → show class in hierarchy
- city in New York → search instances → show class in hierarchy
- city in the United States → search instances → show class in hierarchy
- capital → search instances → show class in hierarchy
- capital of the United States → search instances → show class in hierarchy
- largest metropolis established in 1625 → search instances → show class in hierarchy
- area of the United States → search instances → show class in hierarchy
- economy by city → search instances → show class in hierarchy
- village → search instances → show class in hierarchy

Map of the location



Server Call...

Search Results

Using New York City

Using New York City

Using New York City

Articles Last Viewed

Using New York City

Classes Last Viewed

city in Germany

- „klassische“ ML-Probleme
 - krebserregende Substanzen
 - Erbkrankheiten (Mutation)
- Ontology Engineering
 - Protégé Plugin
 - OntoWiki Plugin
 - ORE
- Empfehlungen/Navigation
 - moosique.net
 - DBpedia Navigator
- nicht betreut:
 - ISS (Gerken et al.)
 - Learning in Probabilistic DLs (Ochoa Luna et al.)
 - TIGER Corpus Navigator (Hellmann et al.)

The screenshot displays the TIGER Corpus Navigator web application. At the top, the title "TIGER Corpus Navigator" is followed by a link "see here for data license" and a "reset" button. The interface is divided into several sections:

- Search:** Contains two search boxes. The "Fulltext Search" box is empty, and the "Lemma Search" box contains the word "werden". Both boxes have a "search" button to their right.
- Search Results:** A section with a "show" button.
- Classified Instances:** Displays a list of search results. The first result is "Ich glaube kaum, daß mit seinem, naja, etwas undiplomatischen Stil im Weißen Haus dem Land ein Gefallen getan wäre." Below it, there are two more results, each with a "show" button.
- Learned Concept:** Shows a concept definition: "(Sentence and hasToken some VVPP and hasToken some (ADV and nextToken some VAFIN))". Below this, it states "Accuracy: 1.0" and a "Matching" button. A "comment" box contains the text: "These are finite auxiliary verbs, e.g. '[du] bist', '[wir] werden'."
- Learning Input:** Contains a "learn" button, a "save" button, and a "Positive Samples" section. The "Positive Samples" section lists two examples: "Nur 1,4 Milliarden Mark seien gestrichen worden ." and "Zuletzt war am 25. Dezember das Wohnmobil des Göttinger Oberstadtdirektors Hermann Schienwater in Brand gesteckt worden ." Each example has a close button (x).
- Negative Samples:** Lists one example: "Er wird zum direkten Partner, der umweglos alle und laufen erscheint ." with a close button (x).

- 1 *Motivation*
- 2 *Überblick zu Beschreibungslogiken und OWL*
- 3 *Refinementoperatoren in OWL/DLs*
 - Theoretische Erkenntnisse
 - Entwicklung von Refinementoperatoren
- 4 *Lernalgorithmen auf Basis von Refinement-Operatoren*
 - OCEL
 - Skalierbarkeit
- 5 *Evaluation*
- 6 *Implementierung und Anwendung*
- 7 *Zusammenfassung und weitere Arbeit*

- **theoretische Analyse** der Eigenschaften von DL-Refinementoperatoren
- Entwicklung von **zwei Operatoren** und Beweis ihrer Eigenschaften
- **drei Algorithmen** mit Erweiterungen gegenüber dem State of the Art
- signifikante **Skalierbarkeitsverbesserungen**
- Evaluation mit **mehreren Benchmarks**, die **in OWL konvertiert worden**
- Implementation im Open Source Projekt **DL-Learner**
- Entwicklung von mehreren **Anwendungen/Plugins**

- Lösen weiterer **schwieriger ML-Probleme** mit DL-Learner Framework
- Untersuchen von **bottom-up Ansätzen**
- Nutzung der Techniken für Matching von Instanzen in Ontologien
- Integration von NLP Techniken
- breitere Evaluation des **Ontology Engineering Use Cases**
- **Buch “Perspectives of Ontology Learning”** (mit Johanna Völker)
- allgemein: weiterhin **Anwendungsbereich von ML/ILP-Techniken auf das Semantic Web erweitern**

Danke für Ihre Aufmerksamkeit!



- verwendet **Konzept- und Rollenhierarchie** im Gegensatz zu anderen Operatoren
- **verwendet Domain und Range** von Rollen (ähnlich zu “mode declarations” in ILP-Programmen) → reduziert Suchraum
- endliche Anzahl von Reasoneranfragen = durch einen Cache wird der **Reasoner nur in der Aufwärmphase** vom Operator verwendet
- unterstützte Konstrukte: $\sqcap$, $\sqcup$, $\neg$, $\exists$, $\forall$, $\top$, $\perp$
- existierende Erweiterungen: Zahlenrestriktionen (min/max), hasValue-Konstruktor, Datentypen: boolean, double, string

- verwendet idealen Refinementoperator ψ
- (optionale) Verwendung eines “covering” Ansatzes d.h. schrittweise Erstellung einer Disjunktion
- vereinfachen von Lösungen (über OWL-Reasoner) um Lesbarkeit weiter zu erhöhen
- geeignet für einfache Lernaufgaben und Ontologien im EL Profil von OWL 2

- Adaptierung von OCEL für Ontology Engineering
- verwendet existierende Klassen als Eingabe und nutzt deren Instanzen für überwachtetes Lernen
- kann existierende Definitionen wiederverwenden (durch initiale upward refinement Phase)
- Heuristik optimiert für breite Suche (verpasst häufiger komplexe Lösungen, aber findet fast immer einfache Lösungen)

Carcinogenesis

