

# Vorlesung Semantic Web



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Vorlesung im Wintersemester 2011/2012

Dr. Heiko Paulheim

Fachgebiet Knowledge Engineering

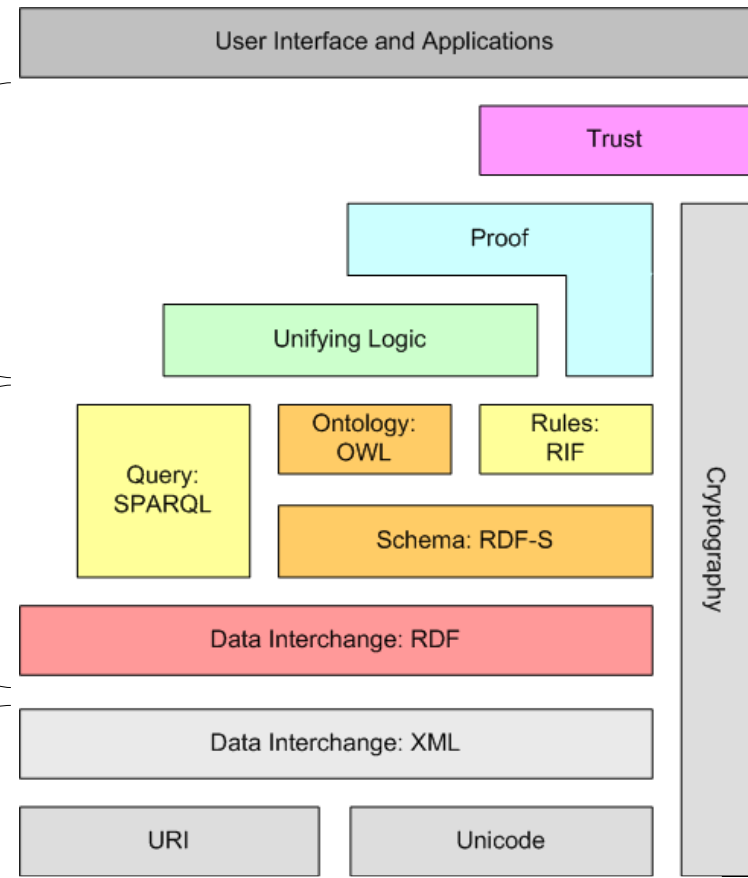
# Semantic Web – Aufbau



here be dragons...

Semantic-Web-  
Technologie  
(Fokus der Vorlesung)

Technische  
Grundlagen



Berners-Lee (2009): *Semantic Web and Linked Data*  
<http://www.w3.org/2009/Talks/0120-campus-party-tbl/>

# Was bisher geschah

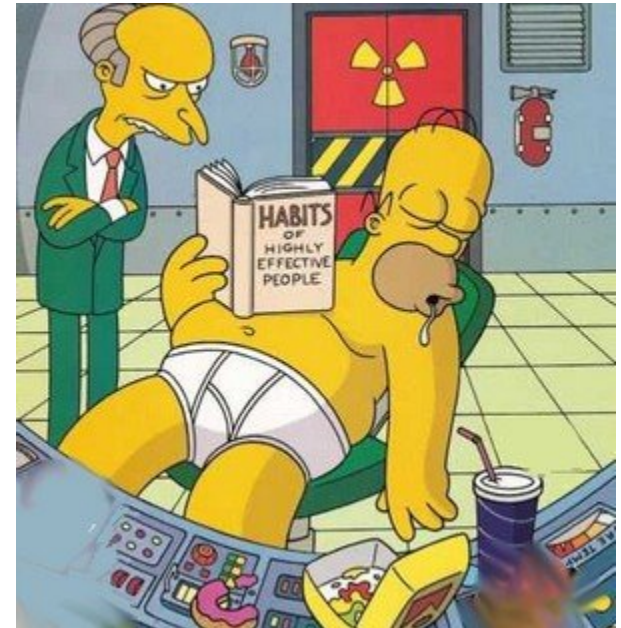
- Ontologien
  - liefern die Hintergrundinformation im Semantic Web
  - codieren Domänenwissen
  - ermöglichen Reasoning
- Ontology Engineering
  - wie baut man eine gute Ontologie?
  - Patterns & Anti-Patterns
- Ontologien bauen
  - ist aufwändig
  - besonders im großen Maßstab

# Was könnte man daran noch verbessern?



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Menschen sind ja von Natur aus faul
- Automatisierung
  - Code-Generatoren
  - MDA
  - ...
- Wie lässt sich das auch im Semantic Web erreichen?



<http://www.earthwave.com.au/blog/wp-content/uploads/2011/06/Homer-1.jpg>

# Ausflug ins Data Mining

- Was ist Data Mining?
  - "Data Mining is a non-trivial process of identifying
    - valid
    - novel
    - potentially useful
    - ultimately understandable patterns in data." (Fayyad et al. 1996)
- "Data Mining is torturing the data until it confesses."  
(oft zitiert, genaue Quelle unbekannt)

# Data Mining: Beispiel



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Anwendung: Optimierung von Supermärkten
- Ziel: häufig gemeinsam gekaufte Dinge gruppieren
- Datengrundlage:
  - Logfiles von Registrierkassen
- Häufig zitiertes Beispiel:
  - *Windeln und Bier*
  - wahrscheinlich ein Mythos...

# Data Mining: Beispiel

- Populäre Anwendung im Netz
  - Recommender-Systeme
  - Kunden, die A kauften, kauften auch B

## Wird oft zusammen gekauft



+



+



**Preis für alle drei: EUR 51,85**

 **Alle drei in den Einkaufswagen**

[Verfügbarkeit und Versanddetails anzeigen](#)

- ✓ **Dieser Artikel:** Semantic Web: Grundlagen (eXamen.press) von Pascal Hitzler Taschenbuch **EUR 24,95**
- ✓ **Semantic Web: Wege zur vernetzten Wissensgesellschaft (X.media.press)** von Tassilo Pellegrini Gebundene Ausgabe **EUR 9,95**
- ✓ **Ontologien: Konzepte, Technologien und Anwendungen (Informatik im Fokus)** von Heiner Stuckenschmidt Taschenbuch **EUR 16,95**

# Data Mining: Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Gegeben:
- Eine Menge von Einkäufen, z.B.
  - Nudeln, Tomaten, Basilikum, Tageszeitung
  - Brötchen, Tageszeitung
  - Nudeln, Tomaten, Hackfleisch, Basilikum, Zigaretten
  - ...
- Gesucht:
- Häufige Muster in Form von Regeln, z.B.
  - Nudeln → Tomaten
  - Hackfleisch, Basilikum → Nudeln, Tomaten
  - Brötchen → Tageszeitung
  - ...



# Data Mining: Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Assoziationsregeln beschreiben häufige Muster
  - nicht symmetrisch
  - warum?
- Populäre Ausreißer
  - z.B.: Verkaufsschlager
    - "Semantic Web" → "Harry Potter"
    - ist wahrscheinlicher als Rückrichtung

# Data Mining: Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Wie findet man Assoziationsregeln?
- Association Rule Mining
  - APRIORI-Algorithmus
  - Lernalgorithmus, der Assoziationsregeln lernt
- Folgende Folien teilweise übernommen von
  - J. Fürnkranz: Maschinelles Lernen – Symbolische Ansätze

# Der APRIORI-Algorithmus

- Entwickelt in den frühen 90ern bei IBM von Agrawal & Srikant
- Motivation
  - Steigende Verbreitung von Bar-Code-Kassen



# Der APRIORI-Algorithmus



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Qualitätsmaße für Assoziationsregeln
- Support
  - Anzahl der Beispiele, die eine Regel insgesamt abdeckt
  - Relevanz der Regel

$$\text{support}(A \rightarrow B) = \text{support}(A \cup B) = \frac{n(A \cup B)}{n}$$

- Confidence
  - Verhältnis von Beispielen, die die Implikation erfüllen, zu Beispielen, die die Bedingung erfüllen
  - Stärke der Implikation

$$\text{confidence}(A \rightarrow B) = \frac{\text{support}(A \cup B)}{\text{support}(A)} = \frac{n(A \cup B)}{n(A)}$$

# Der APRIORI-Algorithmus



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Beispiel-Datenset
  - Nudeln, Tomaten, Basilikum, Tageszeitung
  - Brötchen, Tageszeitung
  - Nudeln, Tomaten, Hackfleisch, Basilikum, Zigaretten
- Vorgeschlagene Regel:
  - Nudeln → Tomaten, Hackfleisch, Basilikum
- Support:  $1/3$
- Confidence:  $1/2$



# Der APRIORI-Algorithmus



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Gegeben:
  - eine untere Schranke für Support ( $s_{\min}$ )
  - eine untere Schranke für Confidence ( $c_{\min}$ )
- Gesucht:
  - alle Assoziationsregeln, die diesen Schranken gehorchen
- APRIORI läuft in zwei Schritten
  - 1.: finde alle *frequent itemsets*
    - d.h., alle Produkte, die häufig gemeinsam auftreten
    - beachte dabei  $s_{\min}$
  - 2.: erzeuge Regeln aus diesen Item sets
    - beachte dabei  $c_{\min}$

# Der APRIORI-Algorithmus



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Erster Schritt: finde frequent itemsets
  - beachte dabei  $s_{\min}$
- Beispiel-Datenset
  - Nudeln, Tomaten, Basilikum, Tageszeitung
  - Brötchen, Tageszeitung
  - Nudeln, Tomaten, Hackfleisch, Basilikum, Zigaretten
- Gegeben: minimaler Support  $s_{\min} = 0.5$ 
  - Frequent Itemsets:
    - $\{\text{Nudeln}\}$  (0.66),  $\{\text{Tomaten}\}$  (0.66),  $\{\text{Basilikum}\}$  (0.66),  $\{\text{Tageszeitung}\}$  (0.66)
    - $\{\text{Nudeln, Tomaten}\}$  (0.66),  $\{\text{Nudeln, Basilikum}\}$  (0.66)

# Der APRIORI-Algorithmus

- Beobachtungen:
  - Wenn ein Itemset größer wird, dann wird der Support nicht größer:

$$C \subseteq D \Rightarrow \text{support}(C) \geq \text{support}(D)$$

- Grund: Definition von Support

$$\text{support}(A \rightarrow B) = \text{support}(A \cup B) = \frac{n(A \cup B)}{n}$$

- Das ermöglicht eine effiziente Suche:
  - beginne mit ein-Elementigen Itemsets
  - erzeuge im k-ten Durchgang k-elementige Itemsets
    - als Vereinigung von bereits gefundenen



# Der APRIORI-Algorithmus



## ▪ Erster Schritt: finde frequent itemsets

1.  $k = 1$
2.  $C_1 = I$  (all items)
3. while  $C_k > \emptyset$ 
  - (a)  $S_k = C_k \setminus$  all infrequent itemsets in  $C_k$  ← d.h.,  $s \leq s_{\min}$
  - (b)  $C_{k+1} =$  all sets with  $k+1$  elements that can be formed by uniting of two itemsets in  $S_k$
  - (c)  $C_{k+1} = C_{k+1} \setminus$  itemsets that do not have all subsets of size  $k$  in  $S_k$
  - (d)  $S = S \cup S_k$
  - (e)  $k++$
4. return  $S$

# Der APRIORI-Algorithmus



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Zweiter Schritt: Erzeuge Regeln aus frequent itemsets
  - beachte dabei  $c_{\min}$
- Beispiel-Datenset
  - Nudeln, Tomaten, Basilikum, Tageszeitung
  - Brötchen, Tageszeitung
  - Nudeln, Tomaten, Hackfleisch, Basilikum, Zigaretten
- Gefundene frequent itemsets ( $n \geq 2$ ):
  - $\{\text{Nudeln, Tomaten}\}$  (0.66),  $\{\text{Nudeln, Basilikum}\}$  (0.66)
- Gegeben  $c_{\min} = 0.5$ :
  - $\text{Nudeln} \rightarrow \text{Tomaten}$  ( $s=0.66, c=1.0$ ),  $\text{Tomaten} \rightarrow \text{Nudeln}$  ( $s=0.66, c=1.0$ )
  - $\text{Nudeln} \rightarrow \text{Basilikum}$  ( $s=0.66, c=1.0$ ),  $\text{Basilikum} \rightarrow \text{Nudeln}$  ( $s=0.66, c=1.0$ )

# Der APRIORI-Algorithmus

- Beobachtungen
  - Für jedes frequent itemset der Größe  $n$  gibt es  $n!$  mögliche Regeln
    - $\{A, B, C\}$ :  $A \rightarrow BC$ ,  $B \rightarrow AC$ ,  $C \rightarrow AB$ ,  $AB \rightarrow C$ ,  $BC \rightarrow A$ ,  $CA \rightarrow B$
    - Problem: Skalierbarkeit
  - Verschieben von Elementen aus Wenn-Teil in Dann-Teil erhöht die Konfidenz nicht:

$$confidence(A \rightarrow B, C) \leq confidence(A, B \rightarrow C)$$

- Grund: Definition von Konfidenz

$$confidence(A \rightarrow B) = \frac{support(A \cup B)}{support(A)} = \frac{n(A \cup B)}{n(A)}$$

# Der APRIORI-Algorithmus



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Das ermöglicht einen effizienteren Algorithmus
  - Beginne bei Regeln mit 1-elementigem Dann-Teil
  - Verschiebe jeweils ein Element vom Wenn- in den Dann-Teil
    - solange Konfidenz hoch genug ist

# Der APRIORI-Algorithmus

- Effizientes Auffinden von Assoziationsregeln
  - mit Mindest-Support und Mindest-Konfidenz
- Mehr Informationen:
  - Vorlesung "Maschinelles Lernen – Symbolische Ansätze"

# Was hat das jetzt mit Semantic Web zu tun?



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Betrachten wir folgende Aussagenmenge

```
:Julia a :Woman, :Person.  
:Stephen a :Man, :Person.  
:Marc a :Man, :Person.  
:Anna a :Woman, :Person.  
:Ann a :Woman.  
:Tim a :Person.
```

- Nehmen wir statt Warenkörbe die Klassen einer Instanz:

```
Julia: {Woman, Person}  
Stephen: {Man, Person}  
Marc: {Man, Person}  
Anna: {Woman, Person}  
Ann: {Woman}  
Tim: {Person}
```

# Ontologien lernen durch Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Mögliche Assoziationsregeln:
  - Woman  $\rightarrow$  Person ( $s=0.4$ ,  $c=0.66$ )
  - Man  $\rightarrow$  Person ( $s=0.4$ ,  $c=1.0$ )
  - Person  $\rightarrow$  Woman ( $s=0.4$ ,  $c=0.4$ )
  - Person  $\rightarrow$  Man ( $s=0.4$ ,  $c=0.4$ )
- Regeln können auch als Subklassenbeziehungen aufgefasst werden
- Mit einem geeigneten Satz Parameter können wir so eine Klasenhierarchie lernen
  - z.B.  $s_{\min}=0.25$ ,  $c_{\min}=0.5$

# Ontologien lernen durch Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Linked Open Data
  - oft nur schwache Ontologien als Schemata
    - können durch Lernen angereichert werden
  - viel Instanzinformation
    - das ist gut zum Lernen!
- Komplement zum Reasoning
- Reasoning: deduktives Schließen
  - Durch Fakten und Regeln zu neuen Fakten
- Ontology Learning: induktives Schließen
  - Durch Fakten zu Regeln



# Ontologien lernen durch Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Betrachten wir noch einmal das Beispiel:

```
:Julia a :Woman, :Person.  
:Stephen a :Man, :Person.  
:Marc a :Man, :Person.  
:Anna a :Woman, :Person.  
:Ann a :Woman.  
:Tim a :Person.
```

- Gelernte Ontologie:

```
:Woman rdfs:subClassOf :Person .  
:Man rdfs:subClassOf :Person .
```

- Reasoning mit dieser Ontologie liefert zusätzlich:

```
:Ann a :Person .
```

# Ontologien lernen durch Assoziationsregeln

---



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Bis jetzt haben wir nur die Klassenhierarchie gelernt
- Was kann man noch mit Assoziationsregeln lernen?
- Z.B. Domain/Range von Relationen

# Ontologien lernen durch Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Verwenden neuer Features
  - `rel_in`: es gibt eingehende Relationen vom Typ `rel`
  - `rel_out`: es gibt ausgehende Relationen vom Typ `rel`
- Was man daraus schließen kann:
  - Gelernte Regel: `rel_out → C`
  - d.h.:  $\text{rel}(X, Y) \rightarrow C(X)$
  - das ist gleichbedeutend mit `rel rdfs:domain C`
  - Gelernte Regel: `rel_in → C`
  - d.h.:  $\text{rel}(X, Y) \rightarrow C(Y)$
  - das ist gleichbedeutend mit `rel rdfs:range C`

# Ontologien lernen durch Assoziationsregeln

- Erweitern wir unser Beispiel:

```
:Julia a :Woman, :Person ; :knows :Stephen, :Marc .  
:Stephen a :Man, :Person ; :fatherOf :Anna.  
:Marc a :Man, :Person. ; :knows :Ann ; :fatherOf :Julia .  
:Anna a :Woman, :Person ; :knows :Tim ; :motherOf :Julia .  
:Ann a :Woman ; :motherOf :Stephen .  
:Tim a :Person ; :knows :Marc, :Anna, :Ann .
```

- Unsere "Warenkörbe" enthalten jetzt nicht nur Klassen
  - sondern auch Informationen über eingehende/ausgehende Relationen

# Ontologien lernen durch Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Neuer "Warenkorb":

Julia: {Woman, Person, knows\_out, fatherOf\_in, motherOf\_in}

Stephen: {Man, Person, fatherOf\_out, knows\_in, motherOf\_in}

Marc: {Man, Person, knows\_out, fatherOf\_out, knows\_in}

Anna: {Woman, Person, knows\_out, motherOf\_out, fatherOf\_in}

Ann: {Woman, motherOf\_out, knows\_in}

Tim: {Person, knows\_out, knows\_in}

- Neue mögliche Regeln für domain/range von *knows*:

knows\_out → Person (s=0.66, c=1.0)

knows\_out → Man (s=0.33, c=0.25)

knows\_out → Woman (s=0.33, c=0.5)

knows\_in → Person (s=0.5, c=0.75)

knows\_in → Woman (s=0.16, c=0.25)

knows\_in → Man (s=0.33, c=0.5)

hier sind die  
besten Werte!

# Ontologien lernen durch Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Neue mögliche Regeln für domain/range von *fatherOf*:
  - fatherOf\_out → Person (s=0.33, c=1.0)
  - fatherOf\_out → Man (s=0.33, c=1.0)
  - fatherOf\_in → Person (s=0.33, c=1.0)
  - fatherOf\_in → Woman (s=0.33, c=1.0)
- Das ist allein nach support/confidence unentscheidbar
  - beides ist mit gleicher Wahrscheinlichkeit möglich
- Konfliktlösung nötig
  - z.B. allgemeinstes Konzept nehmen (Person)
    - das ist einmal korrekt (range von fatherOf)
    - und einmal zumindest nicht falsch (domain von fatherOf)
  - z.B. weitere Gütemaße definieren und berechnen

# Ontologien lernen durch Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Was wir jetzt gesehen haben
  - Ontologien kann man automatisiert lernen
  - z.B. aus Instanzmengen (Linked Open Data)
  - Lernen einfacher RDF-Schemata
- Grenzen des Ansatzes
  - Konflikte bei Bestimmung von domain/range
    - korrekte Lösung aber meist möglich
    - aber nicht immer genaueste
  - Man kann nur lernen, was man in Beispielen sieht
    - z.B. Man braucht `:Tom a :Human, :Mammal .`  
um zu lernen: `:Human rdfs:subClassOf :Mammal .`

# Ontology Matching mit Assoziationsregeln

- In Linked Open Data werden oft mehrere Ontologien parallel genutzt
- Beispiel:  

```
dbpedia:Nine_Inch_Nails  
a dbpedia:Band, dbpedia:Organization,  
yago:IndustrialRockMusicalGroups,  
yago:MusicalGroupsEstablishedIn1988, ...
```
- Was passiert, wenn wir hierauf Assoziationsregeln lernen?



# Ontology Matching mit Assoziationsregeln

- Beispiel für gelernte Regel:
  - `yago:IndustrialRockMusicalGroups` → `dbpedia:Band`
  - entspricht:  

```
yago:IndustrialRockMusicalGroups  
  rdfs:subClassOf dbpedia:Band .
```
- Merke:
  - Wir haben hier ein Mapping gelernt!
  - Ontology Learning mit mehreren Ontologien ist Ontology Matching!
  - und zwar aus der Klasse der instanzbasierten Verfahren

# Ontology Matching mit Assoziationsregeln

- Bei 1:1-Mappings zwischen zwei Klassen lernt man ein symmetrisches Regelpaar:

dbpedia:ProtectedArea → yago:Park  
yago:Park → dbpedia:ProtectedArea

- Daraus folgt:

```
dbpedia:ProtectedArea rdfs:subClassof yago:Park .  
yago:Park rdfs:subClassOf dbpedia:ProtectedArea .
```

- und damit

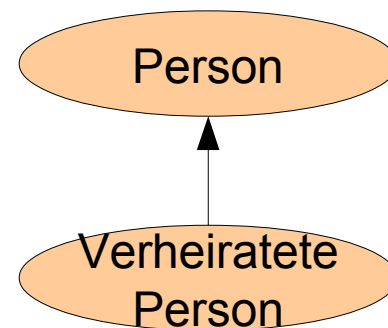
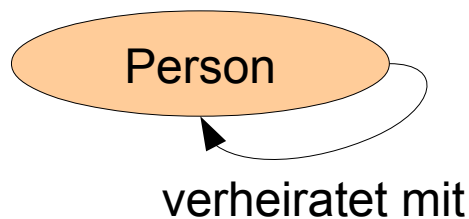
```
dbpedia:ProtectedArea owl:equivalentClass yago:Park .
```

- Merke:

- das funktioniert auch bei syntaktisch unähnlichen Klassennamen!

# Komplexe Mappings mit Assoziationsregeln

- Rückblick Ontology Matching:
  - die meisten Verfahren suchen simple Mappings
  - komplexe Mappings werden in der Regel nicht gefunden
- Assoziationsregeln können hier mehr...
- Betrachten wir ein Beispiel:



# Komplexe Mappings mit Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Gelernte Assoziationsregel:

o1:Person, o1:marriedTo\_out  $\rightarrow$  o2:MarriedPerson

- Das heißt in OWL:

```
o2:MarriedPerson owl:subClassOf
owl:intersectionOf (
  o1:Person
  [ a owl:Restriction ;
    owl:onProperty o1:marriedTo ;
    owl:minCardinality 1^^xsd:integer ] ) .
```

- Und das ist ein ziemlich präzises Mapping!

# Ontology Matching mit Assoziationsregeln

- Valider Ansatz für Linked Open Data
  - wenn mehrere Ontologien verwendet werden
  - auch in zwei Datensets, mit `owl:sameAs` auf Instanzebene verknüpft
  - instanzbasiertes Matching
  - nicht-triviale und komplexe Mappings möglich
- Restriktionen ähnlich wie Ontology Learning
  - man kann nur Mappings finden, wenn die Elemente verwendet werden
  - manche Mehrdeutigkeiten lassen sich nicht trivial auflösen

# Ontology Learning und Matching mit Assoziationsregeln



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Ausgangspunkt:
  - viele Instanzdaten
  - schwache Ontologien
  - fehlende Mappings auf Klassenebene
- Was wir gewinnen können
  - stärkere Ontologien
  - Mappings (auch komplexe und nicht-triviale)

- Recap:
  - Ontologien sind formalisierte Beschreibungen einer Domäne
  - solche liegen oft in textueller Form vor
- Beispiel: Übungsblatt 2, Aufgabe 1:
  - *Eine Bibliothek besitzt Bücher. Bibliotheken haben einen Namen, eine Adresse und eine Telefonnummer. Bücher haben einen Titel, einen oder mehrere Autoren, und eine ISBN-Nummer. Personen haben einen Namen, eine Adresse, eine Telefonnummer und eine E-Mailadresse. Bücher können von einer Person entliehen sein.*

# Ontologien aus Text lernen



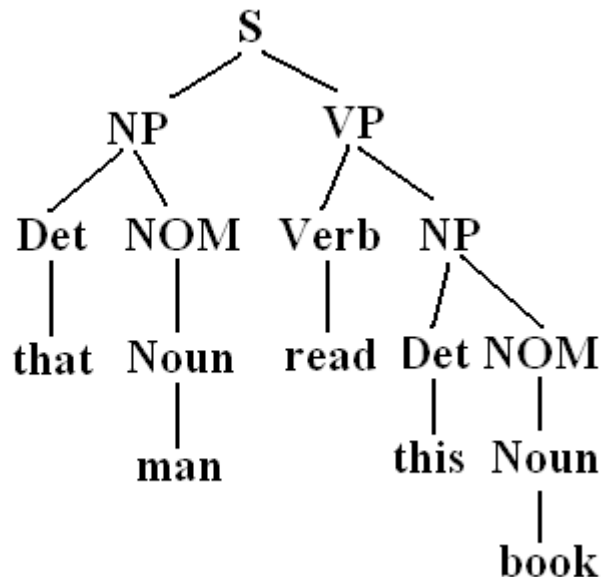
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Kann man das nicht (teil-)automatisieren?
- Mögliche Tasks:
  - Konzepte finden
  - Synonyme finden
  - Domain/Range festlegen



# Kleiner Exkurs: Part of Speech Tagging

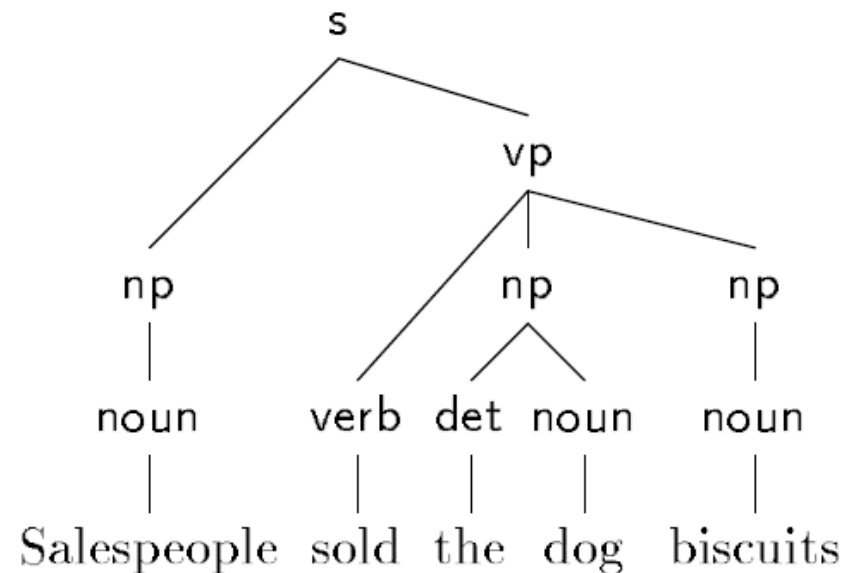
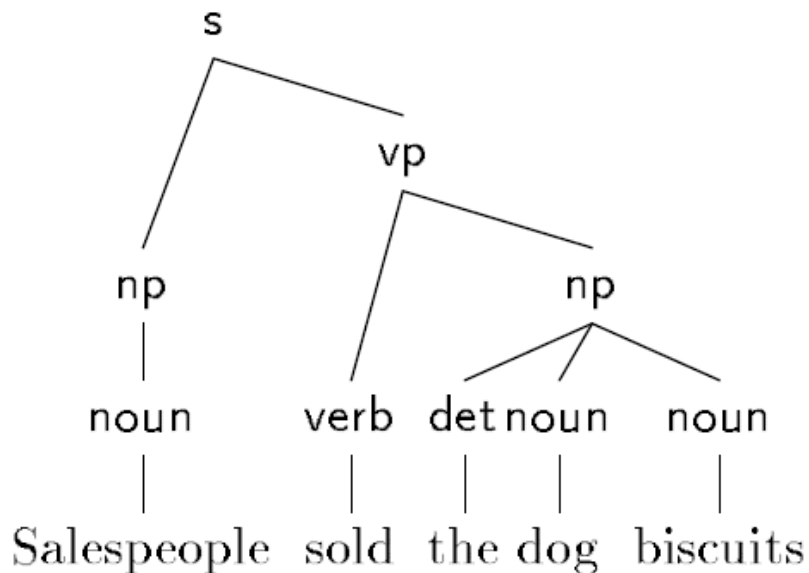
- Automatische Erkennung von
  - Wortarten
  - syntaktischen Funktionen



<http://cs.oberlin.edu/~jdonalds/333/lecture12.html>

# Kleiner Exkurs: Part of Speech Tagging

- Manchmal sind mehrere Taggings möglich
  - das deutet auf einen mehrdeutigen Satz hin



Charniak: Statistical techniques for natural language parsing (1997)

# Kleiner Exkurs: Part of Speech Tagging



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Verfahren
  - Annotiertes Korpus verwenden
  - Menge von Sätzen, die bereits POS Tags besitzen
- Naiver Algorithmus von Charniak (1997)
  - Verwende für jedes Wort das häufigste Tag
  - Alle unbekannten Wörter werden als *Nomen* deklariert
  - Bei einem Korpus von 300.000 Wörten: 90% Accuracy!

Charniak: Statistical techniques for natural language parsing (1997)



# Kleiner Exkurs: Part of Speech Tagging

- Verbesserung: Übergangswahrscheinlichkeiten berücksichtigen

The	can	will	rust
<b>det</b>	modal-verb	<b>modal-verb</b>	noun
	<b>noun</b>	noun	<b>verb</b>
	verb	verb	

- Damit sind 96-97% Genauigkeit möglich
- Obere Grenze: ca. 98%

Charniak: Statistical techniques for natural language parsing (1997)

# Ontologien lernen mit Part of Speech Tagging

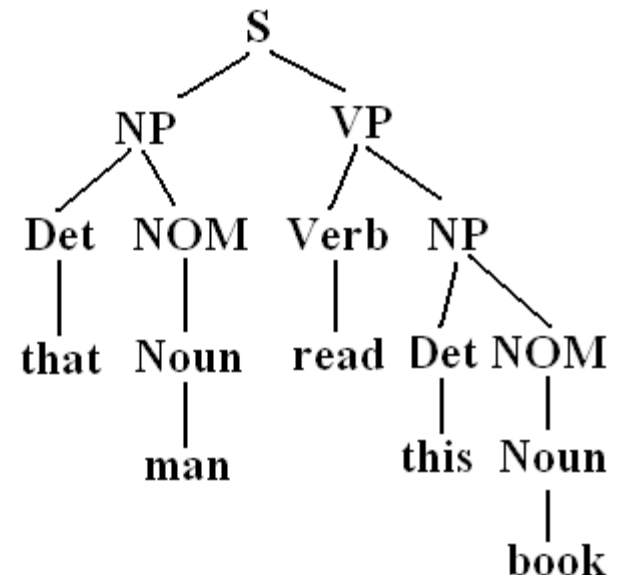


TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Grundidee:
  - Nomen stehen für Konzepte
  - Verben stehen für Relationen

- Erstes Ergebnis:

```
:Man a owl:Class .  
:Book a owl:Class .  
:read a owl:ObjectProperty .
```



# Ontologien lernen mit Part of Speech Tagging



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Verfeinerungen
  - Stemming
    - *Books* → *Book*, *Bücher* → *Buch*
  - Mindesthäufigkeit (Support)
    - wenig häufige Konzepte ausfiltern

# Ontologien lernen mit Part of Speech Tagging

- Synonyme erkennen
  - Wörter, die im ähnlichen Kontext verwendet werden
  - z.B.: als Objekt welcher Wörter?

	<i>book</i>	<i>rent</i>	<i>drive</i>	<i>ride</i>	<i>join</i>
Hotel	X				
Apartment	X	X			
Car	X	X	X		
Bike	X	X	X	X	
Excursion	X				X
Trip	X				X

Cimiano et al.: Ontology Learning. In: Handbook on Ontologies (2009)

# Ontologien lernen mit Part of Speech Tagging

- Analyse
  - z.B. Jaccard-Koeffizient:  $|A \cap B| / |A \cup B|$
  - Ergebnis: Ähnlichkeitsmatrix

	<i>Hotel</i>	<i>Apartment</i>	<i>Car</i>	<i>Bike</i>	<i>Excursion</i>	<i>Trip</i>
Hotel	1.0	0.5	0.33	0.25	0.5	0.5
Apartment		1.0	0.66	0.5	0.33	0.33
Car			1.0	0.75	0.25	0.25
Bike				1.0	0.2	0.2
Excursion					1.0	1.0
Trip						1.0

Cimiano et al.: Ontology Learning. In: Handbook on Ontologies (2009)



# Klassenhierarchien lernen



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Bis jetzt haben wir
  - Mengen von Klassen
  - Synonyme
    - d.h., `owl:equivalentClass`
- Viel häufiger ist aber `rdfs:subClassOf`
  - wie kommen wir da heran?

# Klassenhierarchien lernen durch Clusterbildung

---

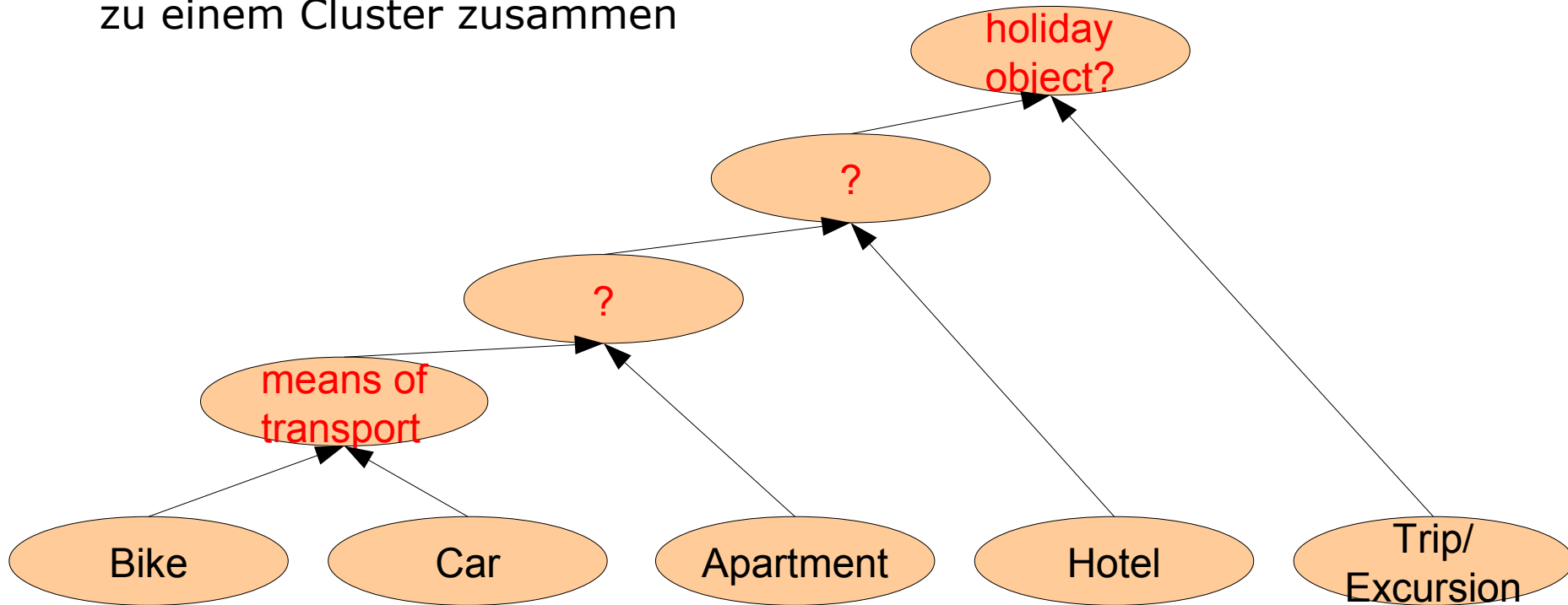


TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Idee: semi-automatisches Verfahren
- Annahme: ähnliche Klassen  
haben eine gemeinsame Superklasse
- Bilde Superklassen
  - lasse diese vom Nutzer benennen

# Klassenhierarchien lernen durch Clusterbildung

- Bottom-Up-Verfahren:
  - fasse jeweils die ähnlichsten Begriffe zu einem Cluster zusammen



# Klassenhierarchien lernen mit Textmustern



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Marti A. Hearst (1992):
  - bestimmte Wendungen deuten auf Hyponym-/Hyperonym-Beziehung hin
- Beispiel:
  - *Säugetiere, wie zum Beispiel Hunde oder Katzen, bringen ihre Jungen lebend zur Welt.*
- Abgeleitete Beziehungen:
  - `:Katze rdfs:subClassOf :Säugetier .`
  - `:Hund rdfs:subClassOf :Säugetier .`

# Klassenhierarchien lernen mit Textmustern



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Beispiel:
  - *Säugetiere, wie zum Beispiel Hunde oder Katzen, bringen ihre Jungen lebend zur Welt.*
- Verallgemeinertes Muster:
  - NP0, wie zum Beispiel NP1, NP2 (und|oder) Npn
  - Daraus folgt:

```
Concept (NP1)  rdfs:subClassOf  Concept (NP0)
...
Concept (NPn)  rdfs:subClassOf  Concept (NP0)
```

# Klassenhierarchien lernen mit Textmustern



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Beispiel-Muster für englische Texte:
  - $NP_{hyper}$  such as  $\{NP_{hypo},\}^* \{(and|or)\} NP_{hypo}$
  - such  $NP_{hyper}$  as  $\{NP_{hypo},\}^* \{(and|or)\} NP_{hypo}$
  - $NP_{hypo} \{,NP_{hypo}\}^*, (and|or)$  other  $NP_{hyper}$
  - $NP_{hyper}$  including  $\{NP_{hypo},\}^*, (and|or) \{other\} NP_{hypo}$
  - $NP_{hyper}$  especially  $\{NP_{hypo},\}^*, (and|or) NP_{hypo}$

Cimiano et al.: Ontology Learning. In: Handbook on Ontologies (2009)

# Klassenhierarchien lernen mit Textmustern



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Geläufiges Problem:
  - Instanzen und Klassen unterscheiden
- Vergleiche:
  - *Mammals, such as cats, dogs, and cows*
  - *Writers, such as Shakespeare, Goethe, and Schiller*
- Besonders bei unbekannten Domänen ein nicht-triviales Problem

# Instanzen und Klassen unterscheiden



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Verschiedene Ansätze:
  - Syntaktische Eigenschaften
    - Nomen mit Artikeln sind Klassen
      - *Der Stör ist ein Fisch. Paul ist ein Mensch.*
      - aber: *Der Irak ist ein Land.*
    - Pluralnomen sind Klassen
      - *Elefanten und Giraffen sind Säugetiere.*
      - aber: *Die Ärzte sind eine Band.*
    - Großgeschriebene Nomen sind Instanzen (englisch, außer Satzanfang)
  - Lookup-Lösungen
    - Named Entity Recognition → Instanzen
    - funktionieren nicht für "exotische" Domänen



# Domain/Range von Relationen aus Sätzen lernen



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Betrachten wir diesen Satz:
  - *Darmstadt liegt in Hessen*
- Angenommen, wir wissen schon

```
:liegtIn a ObjectProperty .
:Darmstadt a :Stadt .
:Hessen a :Land .
```
- Dann können wir bei hinreichend vielen solchen Sätzen folgern:

```
:liegtIn rdfs:domain :Stadt .
:liegtIn rdfs:range :Land .
```

- Ontologien bauen ist aufwändig
- Verschiedene Verfahren zum (semi-)automatischen Ontologiebau existieren
  - Ontology Learning
- Aus Instanzmengen
  - z.B. mit Assoziationsregeln
- Aus Text
  - Part of Speech Tagging
  - Kolokationsanalyse
  - Textmuster



- Lernen von mächtigeren Konstrukten
  - Transitive, symmetrische, funktionale Properties
  - Restriktionen
  - Disjunkte Klassen
  - ...
-

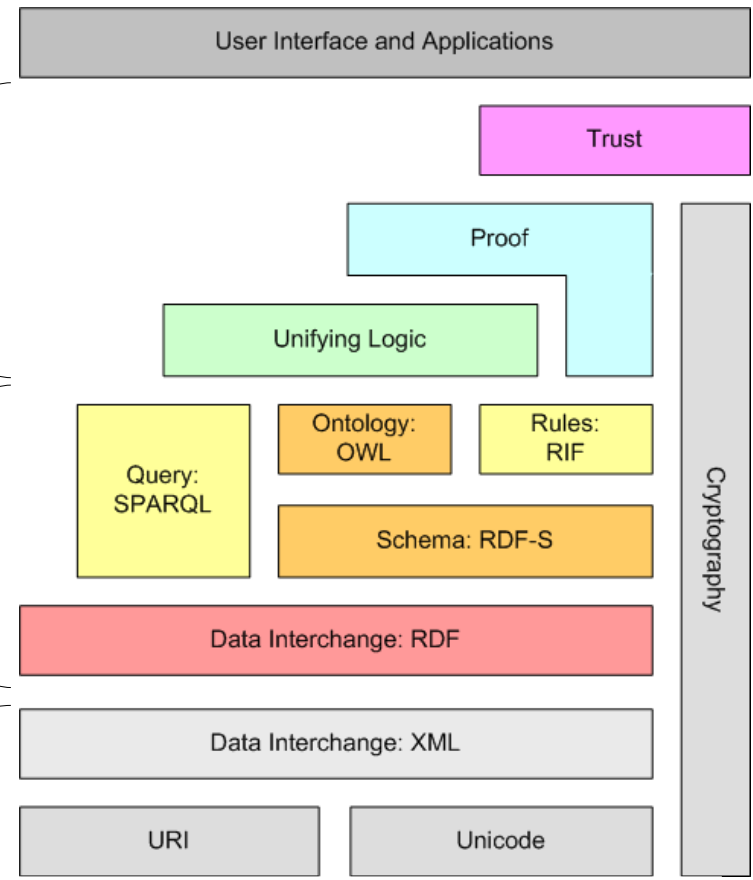
# Semantic Web – Aufbau



here be dragons...

Semantic-Web-  
Technologie  
(Fokus der Vorlesung)

Technische  
Grundlagen



Berners-Lee (2009): *Semantic Web and Linked Data*  
<http://www.w3.org/2009/Talks/0120-campus-party-tbl/>

# Vorlesung Semantic Web



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Vorlesung im Wintersemester 2011/2012

Dr. Heiko Paulheim

Fachgebiet Knowledge Engineering