

# Vorlesung Semantic Web



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Vorlesung im Wintersemester 2011/2012

Dr. Heiko Paulheim

Fachgebiet Knowledge Engineering

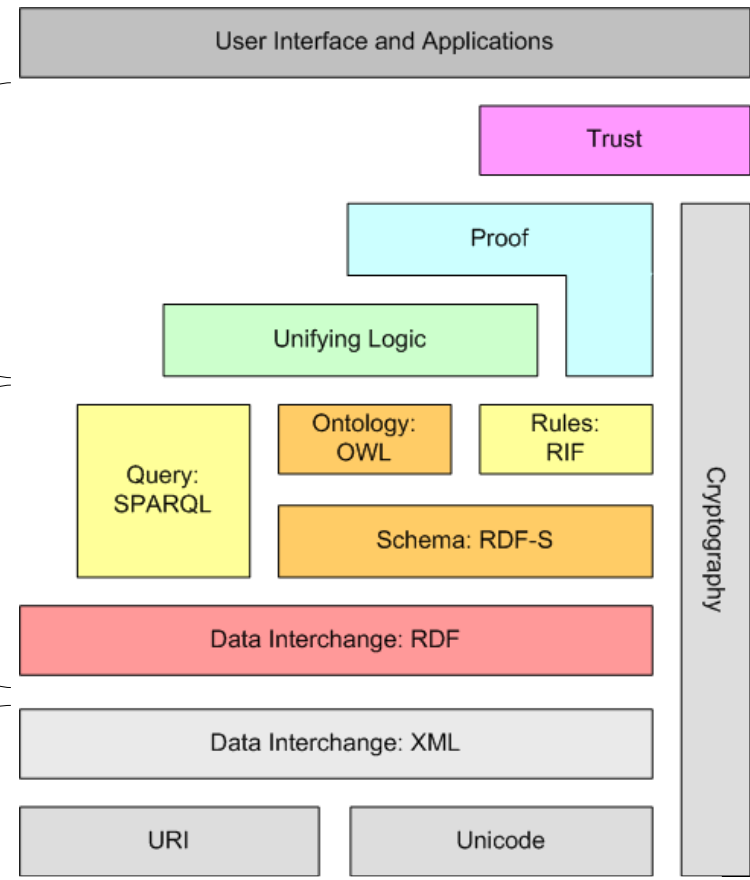
# Semantic Web – Aufbau



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/>

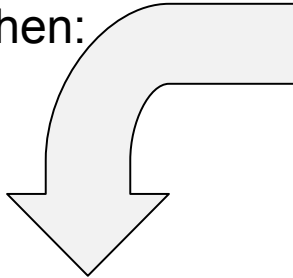
# Was bisher geschah

- Daten werden als Linked Data maschinenlesbar bereitgestellt
  - Daten mit RDF (XML/N3)
  - Schemata & Ontologien mit RDFS/OWL
- Man kann darauf Anwendungen programmieren
- ...und mit Reasoning komplexe Sachverhalte ableiten

# Warum das alles?



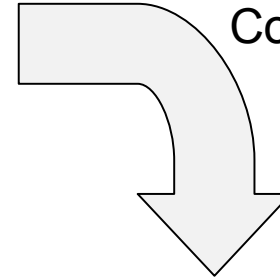
aus Sicht des  
Menschen:



```
<html>
...
<b>Dr. Mark Smith</b>
<i>Physician</i>
Main St. 14
Smalltown
Mon-Fri 9-11 am
Wed 3-6 pm
...
</html>
```

**Dr. Mark Smith**  
*Physician*  
Main St. 14  
Smalltown  
Mon-Fri 9-11 am  
Wed 3-6 pm

aus Sicht des  
Computers:



**Print in bold:** „hmf298hmmhudsa“  
**Print in italics:** „mj2i9ji0“  
**Print normal:** „fdsah  
02hfadsh0um2m0adsmf0ihm  
asdfjköfdsa298ndsfmij32mio  
lk2mjpoimjiofdpmsajiomjm“

# Warum das alles?

- Kann RDF mehr als XML?
- XML ist eine Auszeichnungssprache für Informationen
- In XML kann man beliebige Tags und Attribute definieren
- XML-Tagnamen haben für den Computer keine Bedeutung
- RDF ist eine Auszeichnungssprache für Informationen
- In RDF kann man beliebige Klassen / Relationen definieren
- RDF-Bezeichner haben für den Computer keine Bedeutung

# Warum das alles?

- Kann RDFS/OWL mehr als RDF?
- RDF ist eine Auszeichnungssprache für Informationen
- In RDF kann man beliebige Klassen / Relationen definieren
- RDF-Bezeichner haben für den Computer keine Bedeutung
- RDFS/OWL sind Auszeichnungssprachen für RDF-Daten
- In RDFS/OWL kann man beliebige Klassen / Relationen definieren
- RDFS/OWL-Bezeichner (also Klassen- und Propertynamen) haben für den Computer keine Bedeutung

# Warum das alles?

- Aufgabe:
  - Finde alle Ärzte, die "Mark Smith" heißen
  - Befrage verschiedene Linked Open Data Sets
- Problem:
  - jedes Linked Data Set kann sein eigenes Vokabular benutzen
  - Vokabulare *können*, aber *müssen* nicht wiederverwendet werden

```
:p a foaf:Person .  
:p foaf:name "Mark Smith" .  
:p bar:profession :Physician .  
...
```

*Linked Data Set 1*

```
:q a foo:Human .  
:q foo:called "Mark Smith" .  
:q foo:worksAs :MedDoctor .  
...
```

*Linked Data Set 2*

# Linked Open Data: Die vier Grundregeln

- Vier Grundregeln,  
vorgeschlagen von Tim Berners-Lee (mal wieder)

1. Identifiziere Dinge mit URIs
2. Verwende auflösbare HTTP-URIs
3. Hinterlege an diesen URIs nützliche Informationen,  
verwende dabei Standards
4. Füge Links zu anderen URIs hinzu

gemeint sind hier:  
technische Standards,  
z.B. RDF



# Linked Open Data: Die vier Grundregeln

- "Füge Links zu anderen URIs hinzu"
- das wird in der Regel auf Instanzebene gemacht

```
:p a foaf:Person .  
:p owl:sameAs foo:q .  
:p foaf:name "Mark Smith" .  
:p bar:profession :Physician .  
...
```

*Linked Data Set 1*

```
:q a foo:Human .  
:q owl:sameAs bar:p .  
:q foo:called "Mark Smith" .  
:q foo:worksAs :MedDoctor .  
...
```

*Linked Data Set 2*

# Was fehlt?

- "Füge Links zu anderen URIs hinzu"
  - beachte das auch für die Schemaebene
  - Schemata werden ja auch als Linked Data bereitgestellt
  - das sollte also eigentlich selbstverständlich sein...

```
:p a foaf:Person .  
:p owl:sameAs foo:q .  
:p foaf:name "Mark Smith" .  
:p bar:profession :Physician .  
...
```

*Linked Data Set 1*

```
:q a foo:Human .  
:q owl:sameAs bar:p .  
:q foo:called "Mark Smith" .  
:q foo:worksAs :MedDoctor .  
...
```

*Linked Data Set 2*

# Ontology Mappings

- Links auf Schemaebene heißen *Ontology Mapping*
  - auch: *Ontology Alignment*, *Ontology Matching*
  - Abbildungen einer Ontologie auf eine andere
  - Vereinfacht: ein Lexikon zwischen Ontologien
  - sowohl für Klassen als auch für Properties

FOAF	FOO
Person	Human
name	called
...	...

# Ausdrücken von Mappings



- Formal ist ein Mapping eine Menge von Tripeln  $\langle c_1, c_2, r \rangle$ 
  - wobei  $c_1$  ein Konzept aus Ontologie 1 ist
  - $c_2$  ein Konzept aus Ontologie 1 ist
  - $r$  eine Relation zwischen beiden ist
    - In der Regel eines von  $\equiv$ ,  $\sqsubseteq$  und  $\sqsupseteq$ , manchmal auch  $\neq$
- Mappings werden in der Regel nur zwischen gleichartigen Entitäten definiert (Konformität mit OWL Lite/DL!)
  - zwischen Klassen und Klassen
  - zwischen ObjectProperties und ObjectProperties
  - zwischen DatatypeProperties und DatatypeProperties
  - zwischen Instanzen und Instanzen

# Mappings im Semantic Web

- Asymmetrisch: eine Ontologie enthält Mappings zu einer anderen

```
ex:myOntology owl:imports <http://xmlns.com/foaf/0.1/>
ex:Human a owl:Class ;
    owl:equivalentClass foaf:Person .
ex:hasName a owl:DatatypeProperty ;
    owl:equivalentProperty foaf:name .
ex:mobilePhone a owl:ObjectProperty .
ex:mobilePhone rdfs:subPropertyOf foaf:Phone .
```

- Achtung: durch falsch gesetzte Mappings kann man schnell den Raum von OWL Lite/DL verlassen:

```
ex:mobilePhone a owl:DatatypeProperty .
ex:mobilePhone rdfs:subPropertyOf foaf:Phone .
aber: foaf:Phone a owl:ObjectProperty .
```

# Mappings im Semantic Web

- Symmetrisch:
  - wechselseitige Verweise (2x asymmetrisch)
  - eine Ontologie nur für Mappings ("Bridge-Ontologie")
- Sieht z.B. so aus:

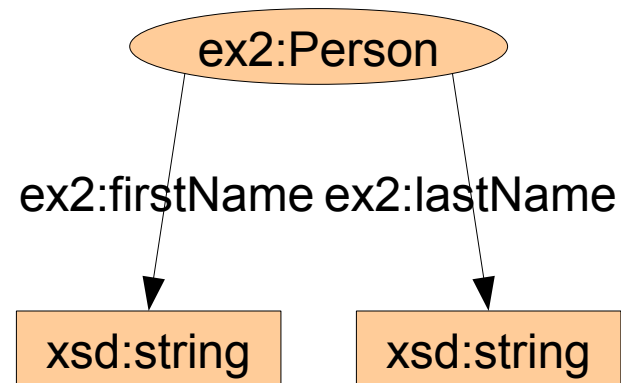
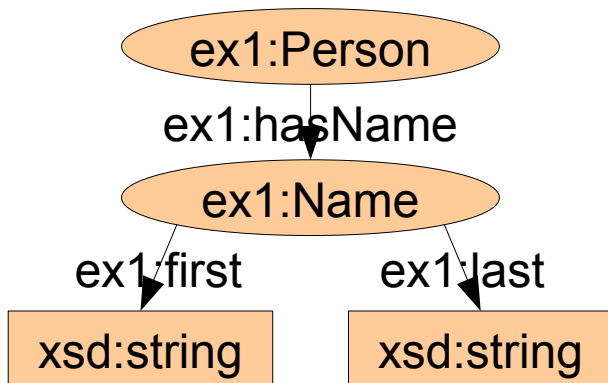
```
ex:bridge a owl:Ontology .  
ex:bridge owl:imports <http://example.org/onto1.owl>  
ex:bridge owl:imports <http://example.org/onto2.owl>  
ex1:Researcher owl:subClassOf ex2:Person .  
ex1:worksOn owl:equivalentProperty ex2:involvedIn .  
...
```

# Mappings im Semantic Web

- Linked Open Data:
  - Mappings müssen auffindbar sein
  - Daher müssen Mapping-Axiome immer auch in der Definition enthalten sein
  - kann auch durch Importieren einer Bridge-Ontologie geschehen
- Beschränkungen dieses Ansatzes:
  - Nur OWL-Axiome als Mappings möglich

# Mappings im Semantic Web

- Beschränkungen des OWL-basierten Ansatzes
  - keine komplexeren Ausdrücke, z.B. zum Matchen von  
ex1:temperatureInC a owl:DatatypeProperty .  
ex2:temperatureInF a owl:DatatypeProperty .
- oder von





# Mappings im Semantic Web



- Warum ist das problematisch?

- Was spricht denn gegen

- ```
ex1:first owl:equivalentProperty ex2:firstName .  
ex2:last owl:equivalentProperty ex2:lastName .
```

- Gegeben

- ```
ex1:first rdfs:domain ex1:Name .
```

- und

- ```
ex2:PeterSchmidt ex2:firstName "Peter" .
```

- Was folgt daraus?

- ```
ex2:PeterSchmidt a ex1:Name .
```

# Mappings im Semantic Web

- Aktuelle Forschung
  - verwenden komplexerer Repräsentationen
    - z.B. Regeln
  - Fuzzy Mappings
    - z.B. `ex1:Forscher`, `ex2:UniMitarbeiter`
    - nicht alle Forscher sind Uni-Mitarbeiter
    - nicht alle Uni-Mitarbeiter sind Forscher
    - aber trotzdem sind die beiden *zu einem bestimmten Grad ähnlich*

# Wie findet man Mappings?

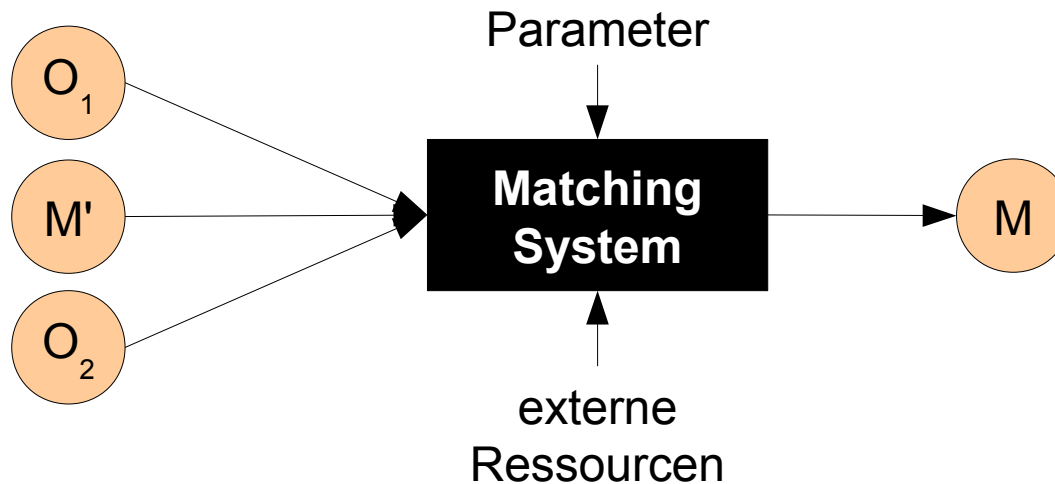


TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- z.B. durch Experten
- Nehmen wir mal zwei Ontologien, z.B.
  - YAGO (~100.000 Klassen)
  - OpenCyc (~150.000 Klassen)
- Dieser Ansatz skaliert offenbar nicht besonders gut...

# Ontology Matching

- Ontology Matching:
  - automatisches Finden von Mappings
  - Gegeben: zwei Ontologien

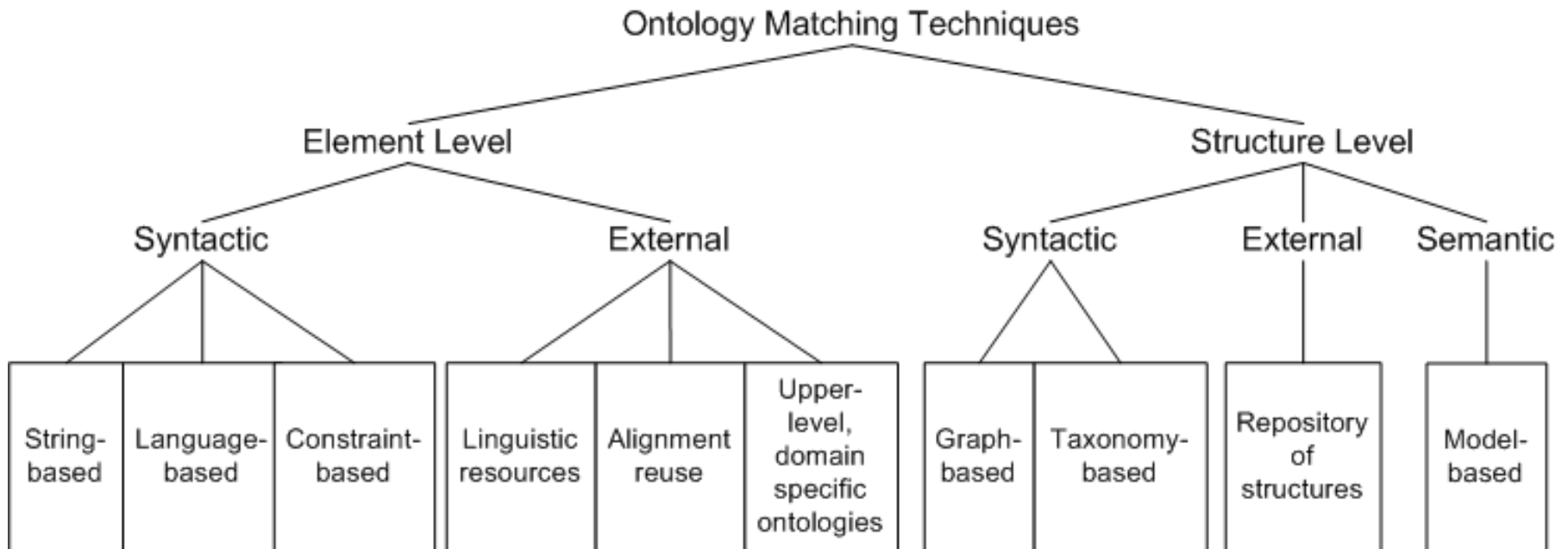


Euzenat & Shvaiko: Ontology Matching (2007)

# Ontology Matching

- Matcher erzeugen ein Mapping
  - üblicherweise simples Mapping
  - und liefern meist ein zusätzliches Konfidenzmaß
  - Also 4-Tupel der Form  $\langle e_1, e_2, r, c \rangle$
  - z.B.  $\langle \text{ex:name}, \text{foaf:name}, \equiv, 0.9 \rangle$

# Arten von Matchern



Euzenat & Shvaiko: Ontology Matching (2007)

# Arten von Matchern



- Element- vs. strukturbasiert:
  - elementbasierte Verfahren: vergleichen nur einzelne Elemente
  - strukturbasierte Verfahren: nutzen auch Struktur aus
    - z.B. Vererbungsbeziehungen
- Syntaktisch vs. extern vs. semantisch
  - syntaktische Verfahren nutzen nur die Ontologien selbst
  - externe Verfahren ziehen weitere Wissensquellen hinzu
  - semantische Verfahren nutzen die inhärente Semantik von Ontologien
    - z.B. mit Reasoning

# Simple elementbasierte Techniken

- Element-basierte Verfahren vergleichen in der Regel Bezeichner
  - z.B. URI-Fragmente
    - `ex1:name`
    - `ex2:hasName`
  - z.B. Labels
    - `ex1:name rdfs:label "A person's name"@en .`
    - `ex2:hasName rdfs:label "The name of a person"@en .`
  - z.B. comments
    - `ex1:name rdfs:comment "Usually the family name"@en .`
    - `ex2:name rdfs:comment "Usual order: family name, given name"@en .`



# String-basierte Verfahren



- Direkte Stringgleichheit
  - z.B. ex1:Person, ex2:Person
- Suche von gemeinsamen Präfixen
  - z.B. ex1:phone, ex2:phoneNumber
- Suche von gemeinsamen Postfixen
  - z.B. ex1:name, ex2:hasName
  
- Verfeinerung:
  - Längenrelation von gemeinsamem Präfix/Suffix und String als Konfidenzmaß
  - z.B. ex1:phone, ex2:phoneNumber  $\Rightarrow c = 5/11$
  - z.B. ex1:name, ex2:hasName  $\Rightarrow c = 3/7$

# String-basierte Verfahren

- Edit-Distanz
  - Minimale Anzahl von Editier-Schritten, um von String 1 zu String 2 zu kommen, geteilt durch Länge des längeren Strings
    - ein Zeichen einfügen
    - ein Zeichen löschen
    - ein Zeichen ändern
  - z.B. ex1:hasName, ex2:firstName
    - hasName → fhasName → fhastName → fiastName → firstName
    - Edit-Distanz =  $4/9$

# String-basierte Verfahren



- N-Gramm-Analyse
  - wie viele Buchstabengruppen der Länge  $n$  stimmen überein geteilt durch Anzahl  $n$ -Gramme in längerem String (= Länge -  $n$  + 1)
  - häufig verwendet:  $n=3$
  - z.B. ex1:hasName, ex2:firstName
    - übereinstimmende 3-Gramme: Nam, ame
    - gesamte 3-Gramme: fir, irs, rst, stN, tNa, Nam, ame
    - 3-Gramm-Distanz =  $2/7$

# Sprachbasierte Techniken

- Werden insbesondere zur Vorverarbeitung genutzt, um die Treffgenauigkeit zu verbessern
- Linguistische Verfahren
- Eliminierung von Stop-Words
  - ex1:locatedIn → ex1:located
- Lemmatisierung/Stemming:
  - ex1:located, ex2:location
  - beides wird zu locat-
- Zerlegen in Token (Tokenization)
  - ex1:graduated\_from\_university → {graduated,from,university}
  - ex2:isGraduateFromUniversity → {is,Graduate,from,University}
  - Token werden einzeln weiterverarbeitet

# Sprachbasierte Verfahren



- können Treffgenauigkeit verbessern
  - ex1:located, ex2:location
  - Stemming: ex1:locat-, ex2:locat-
  - Edit-Distance: 0 → hohe Ähnlichkeit
- Gegenbeispiel:
  - ex1:locationOf, ex2:locatedIn (Inverse Properties!)
  - Erster Schritt: Eliminierung von Stop-Words:  
ex1:location, ex2:located
  - Zweiter Schritt: Stemming: ex1:locat-, ex2:locat-
  - Edit-Distance: 0 → hohe Ähnlichkeit
- dagegen Edit-Distance ohne Vorverarbeitung: 0.5  
→ nicht so hoch

# Constraint-basierte Verfahren



- Zusätzlicher Abgleich der Eigenschaften von gemappten Entitäten
  - z.B. verwendete Multiplizitäten von Restriktionen
  - z.B. Datentypen bei DatatypeProperties
- Im letzten Beispiel:
  - `ex1:locationOf`
    - definiert als `InverseFunctionalProperty`
    - Kardinalität ist nicht eingeschränkt
  - `ex2:locatedIn`
    - definiert als `FunctionalProperty`
    - `exactCardinality=1`
- Damit können wir die Ähnlichkeit von `ex1:locationOf` und `ex2:locatedIn` reduzieren

# Matching mit linguistischen Ressourcen

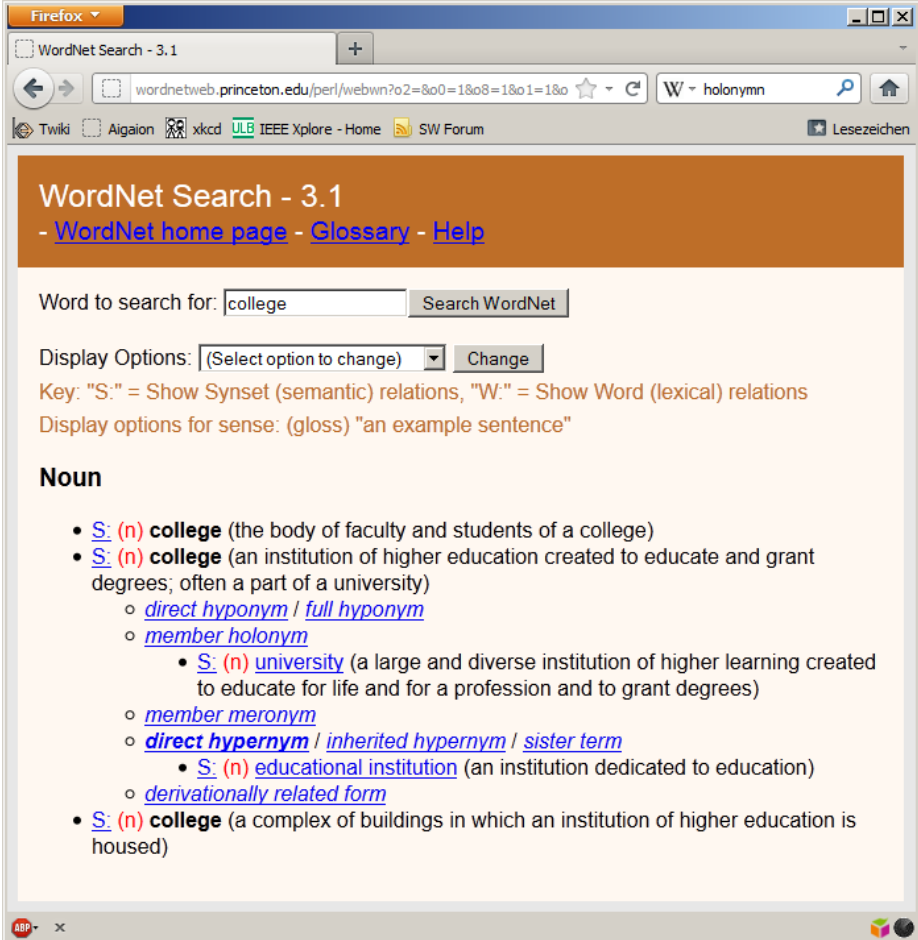


TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Externes Wissen
- Beispiel: Synonyme
  - ex1:Verfasser, ex2:Autor
  - Edit-Distance: 8/9 (sehr hoch!)
  - Synonymwörterbuch kann hier eine Lösung sein
- Beispiel: Ontologien in unterschiedlichen Sprachen
  - ex1:Stadt, ex2:city
  - Edit-Distance: 1
  - Lexikalische Ressource kann hier eine Lösung sein

# Matching mit linguistischen Ressourcen

- WordNet
  - strukturiert für englisch
  - Synonyme, Hyponyme, Hyperonyme
  - Holo- und Meronymie



Firefox

WordNet Search - 3.1

wordnetweb.princeton.edu/perl/webwn?o2=&o0=1&o8=1&o1=1&o

W - holonym

Twiki Aigaion xkcd ULB IEEE Xplore - Home SW Forum Lesezeichen

WordNet Search - 3.1  
- [WordNet home page](#) - [Glossary](#) - [Help](#)

Word to search for:

Display Options:

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations  
Display options for sense: (gloss) "an example sentence"

**Noun**

- [S:](#) (n) **college** (the body of faculty and students of a college)
- [S:](#) (n) **college** (an institution of higher education created to educate and grant degrees; often a part of a university)
  - [direct hyponym](#) / [full hyponym](#)
  - [member holonym](#)
    - [S:](#) (n) **university** (a large and diverse institution of higher learning created to educate for life and for a profession and to grant degrees)
  - [member meronym](#)
  - [direct hypernym](#) / [inherited hypernym](#) / [sister term](#)
    - [S:](#) (n) **educational institution** (an institution dedicated to education)
  - [derivationally related form](#)
- [S:](#) (n) **college** (a complex of buildings in which an institution of higher education is housed)



# Matching mit linguistischen Ressourcen

---



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Auch domänenspezifisch
  - ex1:ServiceOrientedArchitecture, ex2:SOA
  - Edit-Distance: 0.89 (sehr hoch)
  - Ein Domänenwörterbuch kann hier Abhilfe schaffen

# Wiederverwendung von Mappings



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Betrachten wir folgendes Beispiel:
  - `ex:hasCreator, foaf:maker`
  - Edit-Distance 0.7
- Für FOAF gibt es aber teilweise schon Mappings zu anderen Ontologien, z.B. Dublin Core
  - `foaf:maker owl:equivalentProperty dc:creator`
  - Wir können also auch `ex:hasCreator` mit `dc:creator` vergleichen
  - Edit-Distance: 0.4



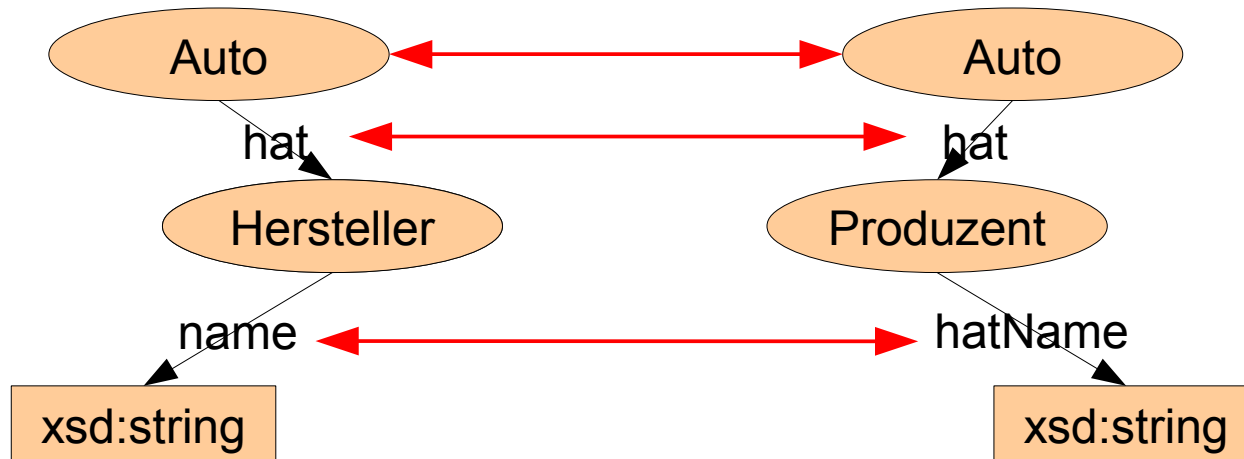
# Verwendung von Upper-Level-Ontologien



- Upper-Level-Ontologien werden wir noch kennen lernen
- Idee: eine allgemeine Ontologie wiederverwenden
  - `ex1:Researcher rdfs:subClassOf upper:Human .`
  - `ex2:Scientist rdfs:subClassOf upper:Human .`
- Eine gemeinsame Oberklasse kann zusätzliches Indiz für ein Mapping sein
- Aber nicht alleiniges Indiz!
  - `ex2:Baker rdfs:subClassOf upper:Human .`

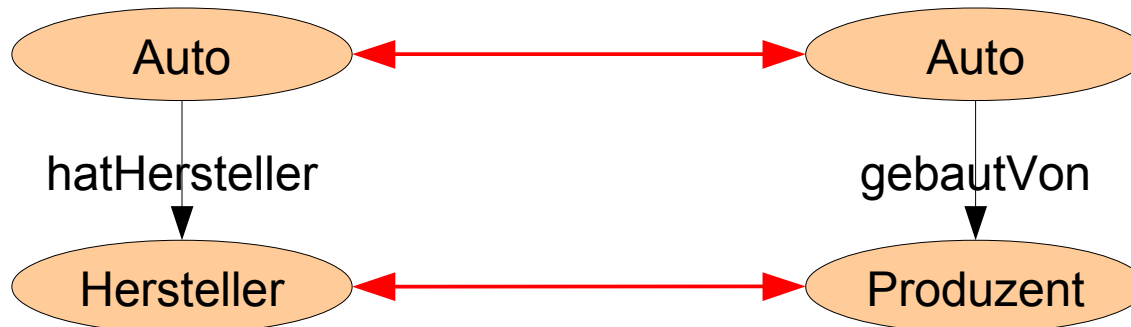
# Graphenbasierte Verfahren

- Ontologien bilden Graphen
- Idee: Ähnlichkeiten in den Graphen propagieren
- z.B.
  - #gemappte Nachbarknoten / #Nachbarknoten
  - #gemappte Kanten / #Kanten



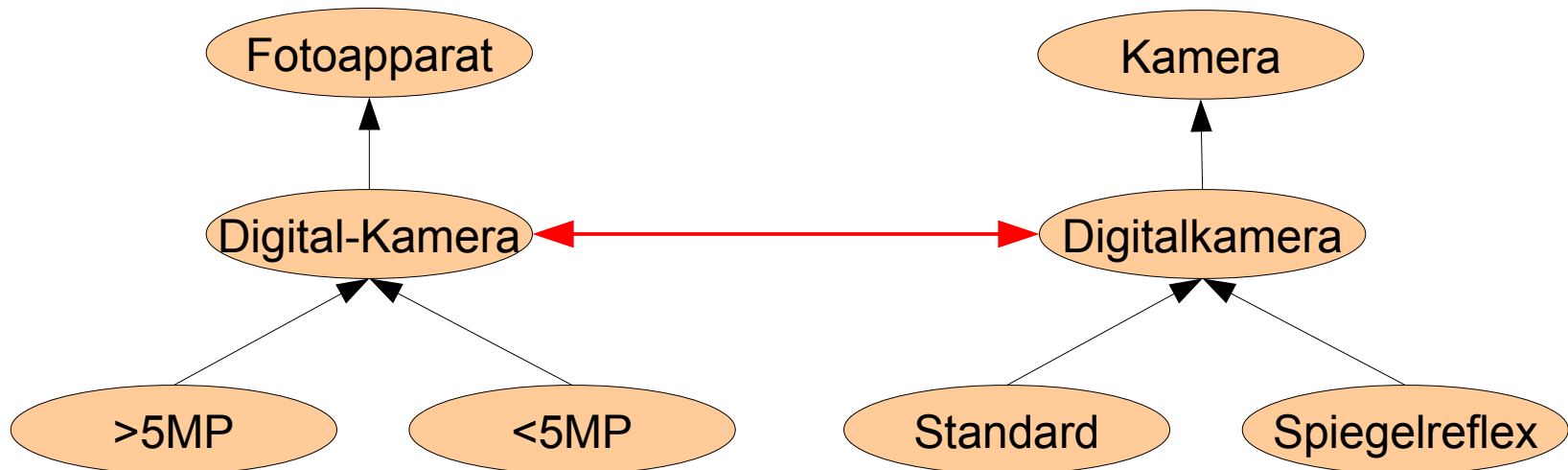
# Graphenbasierte Verfahren

- Sind auch geeignet, um Mappings für Properties zu bestimmen oder zu verfeinern:
  - wenn je zwei Klassen durch ein Property verbunden sind, dann kann dieses mit hoher Wahrscheinlichkeit gemappt werden



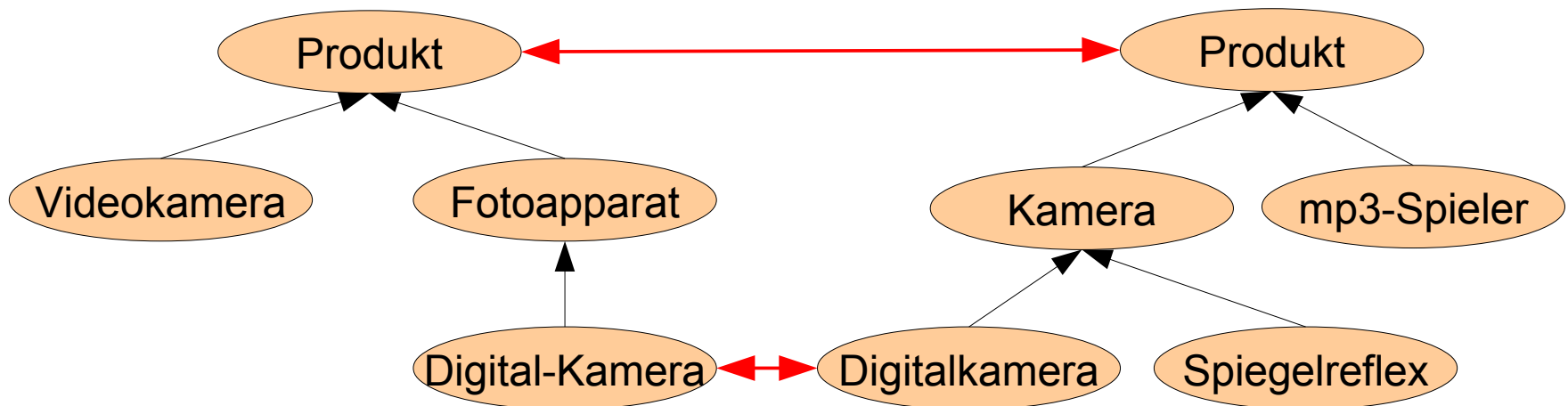
# Taxonomiebasierte Verfahren

- Klassenhierarchien sind essentiell in Ontologien
- Super/Subkonzeptregeln: Kinder und Eltern von gemappten Elementen sind Kandidaten



# Taxonomiebasierte Verfahren

- Klassenhierarchien sind essentiell in Ontologien
- Bounded Path Matching: Auf Pfaden zwischen gemappten Elementen liegen wahrscheinlich weitere Kandidaten



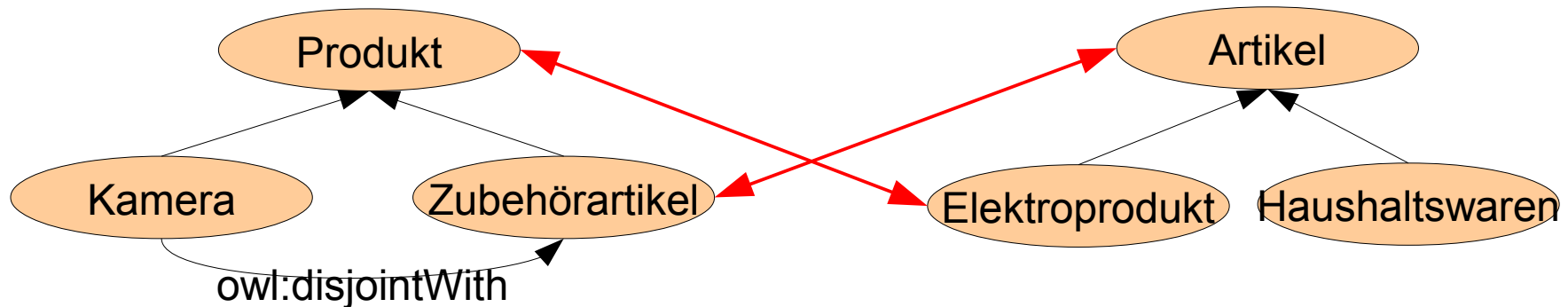
# Struktursammlungen

- Annahme: es gibt bestimmte häufig wiederkehrende Muster
  - z.B.: Person mit Vor- und Nachname
  - z.B.: Adresse mit Straße, PLZ und Ort
- Idee: Suche gezielt nach solchen Mustern
  - wenn in beiden Ontologien gefunden: untersuche diese im Detail

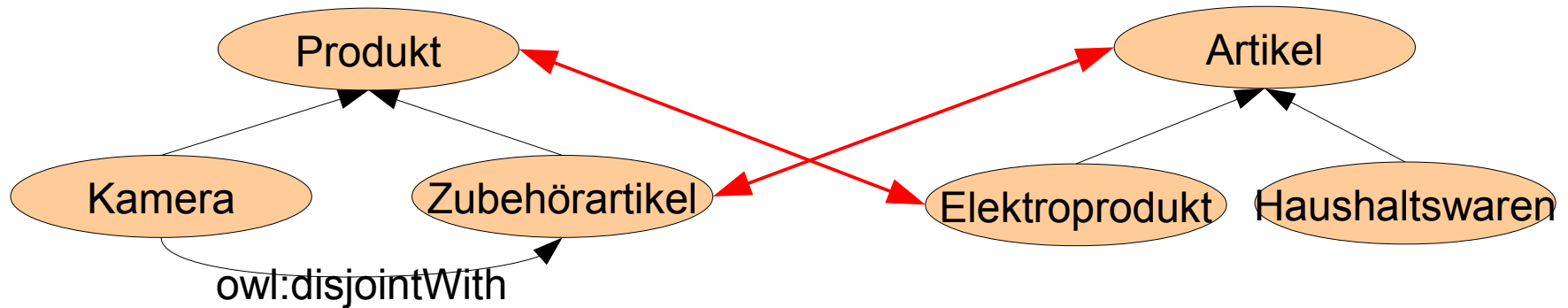


# Modellbasierte Verfahren

- Verwendung von einem Reasoner
- Verfahren:
  - beide Ontologien zu einer vereinigen
  - gefundene Mappings einfügen
  - mit Reasoner auf Widerspruchsfreiheit prüfen



# Modellbasierte Verfahren



```
:Kamera  
  rdfs:subClassOf :Produkt .  
:Zubehörartikel  
  rdfs:subClassOf :Produkt .  
:Kamera owl:disjointWith  
  :Zubehörartikel .
```

```
:Elektroprodukt  
  rdfs:subClassOf :Artikel .  
:Haushaltswaren  
  rdfs:subClassOf :Artikel .
```

```
ex1:Produkt owl:equivalentClass  
  ex2:Elektroprodukt .  
ex1:Zubehörartikel  
  owl:equivalentClass ex2:Artikel .
```

- Lassen wir das den Reasoner mal untersuchen

```
ex1:Kamera rdfs:subClassOf ex1:Produkt .  
+ ex1:Produkt owl:equivalentClass ex2:Elektroprodukt .  
→ ex1:Kamera rdfs:subClassOf ex2:Elektroprodukt .  
+ ex2:Elektroprodukt rdfs:subClassOf ex2:Artikel .  
→ ex1:Kamera rdfs:subClassOf ex2:Artikel .  
+ ex2:Artikel owl:equivalentClass ex1:Zubehörartikel .  
→ ex1:Kamera rdfs:subClassOf ex1:Zubehörartikel .
```

- Gleichzeitig soll gelten:

```
ex1:Kamera owl:disjointWith ex1:Zubehörartikel .
```

- Widerspruch gefunden!

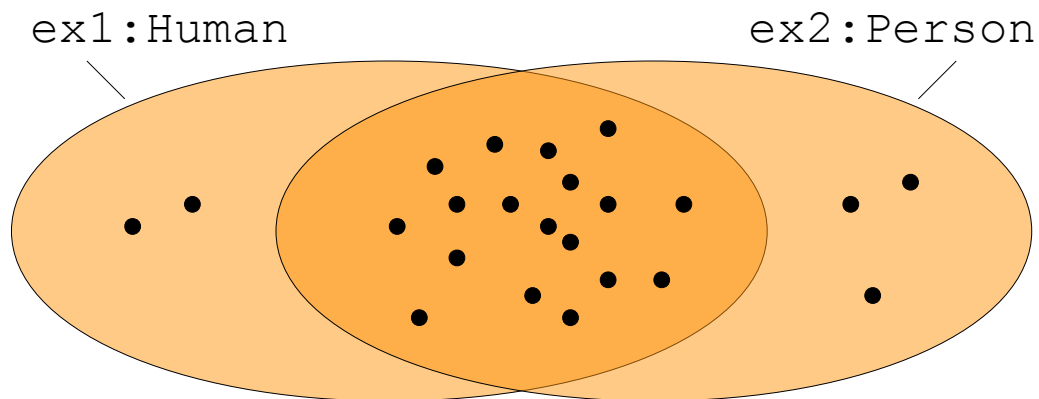
# Instanzbasierte Verfahren

- Bis jetzt haben uns Instanzen noch nicht gekümmert
  - nur schemabasierte Ansätze
  - das ist die Mehrzahl
- Linked Open Data bietet viele Instanzdaten
  - die können auch hilfreich sein:
    - ex1:Peter a ex1:Human .
    - ex1:Peter owl:sameAs ex2:Pete .
    - ex2:Pete a ex2:Person .
  - mit genügend solcher Instanzen kann man auf ein Mapping schließen

# Instanzbasierte Verfahren

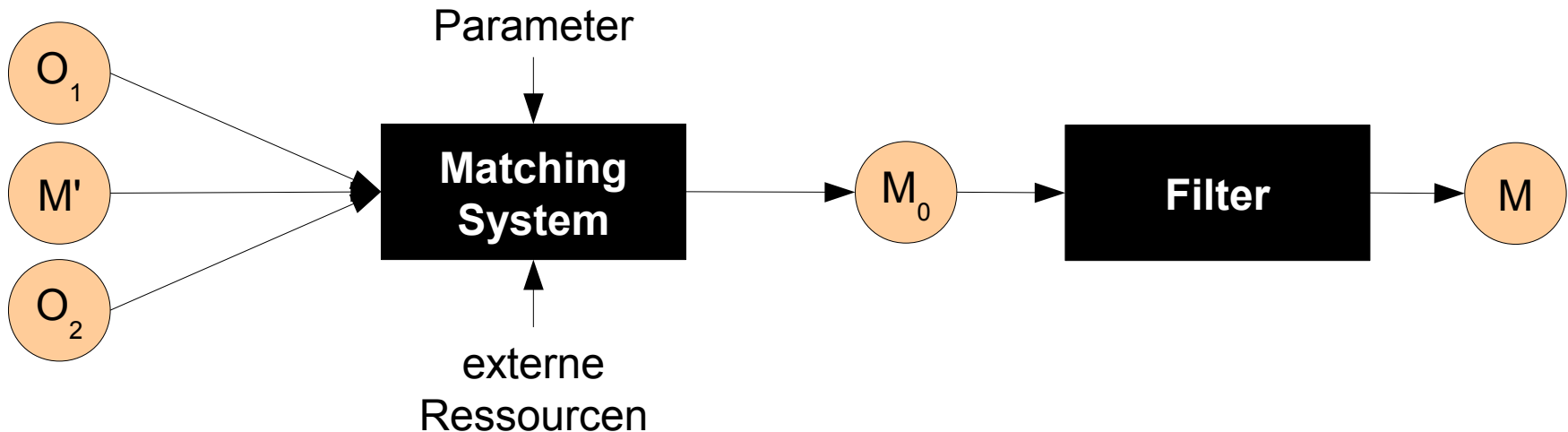


- Ansatz z.B. über Accuracy:
  - $\#(\text{ex1:Human} \cap \text{ex2:Person}) / \#(\text{ex1:Human} \cup \text{ex2:Person})$
  - im Beispiel:  $18/23 \rightarrow$  Mapping-Konfidenz von  $\sim 0.78$
- Gut geeignet, um nicht-triviale Mappings zu finden
  - z.B. `dbpedia:Park`  $\leftrightarrow$  `yago:ProtectedArea`



# Finalisierung der Mapping-Ergebnisse

- Ontology Matcher liefert
  - $\langle e_1, e_2, r, c \rangle$
  - Übliches Verfahren: Rückgabe aller Mappings, für die  $c \geq t$  gilt
  - Auch möglich: Filtern mit Plausibilitätsprüfung



# Kombination von Matchern



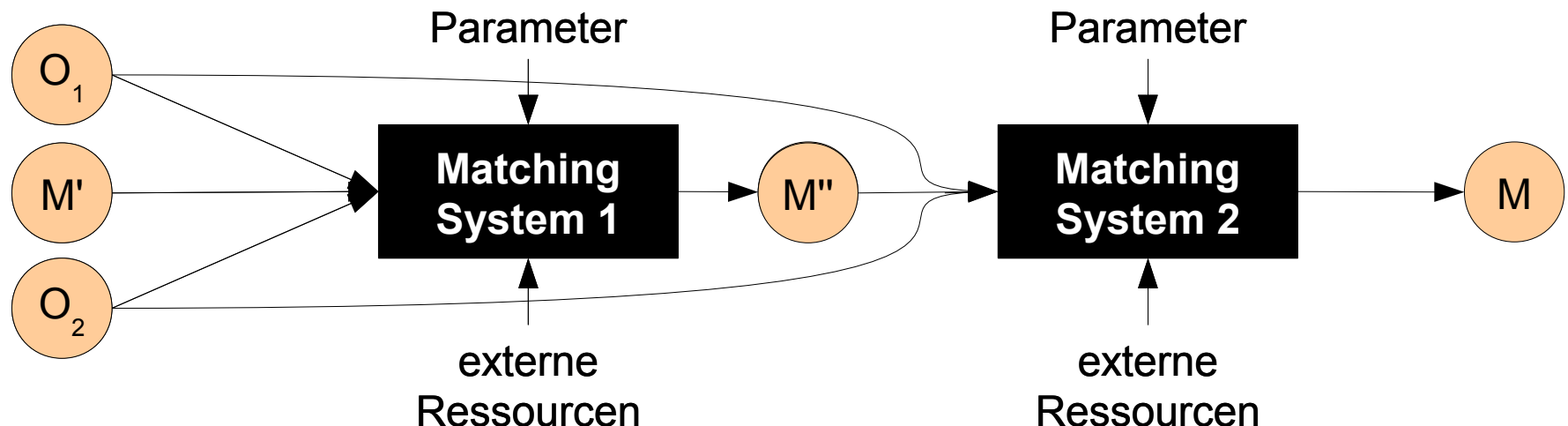
TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Was wir bis jetzt gesehen haben
  - eine Reihe von Strategien
  - jede hat Stärken und Schwächen
  - in der Regel werden mehrere Verfahren kombiniert



# Kombination von Matchern

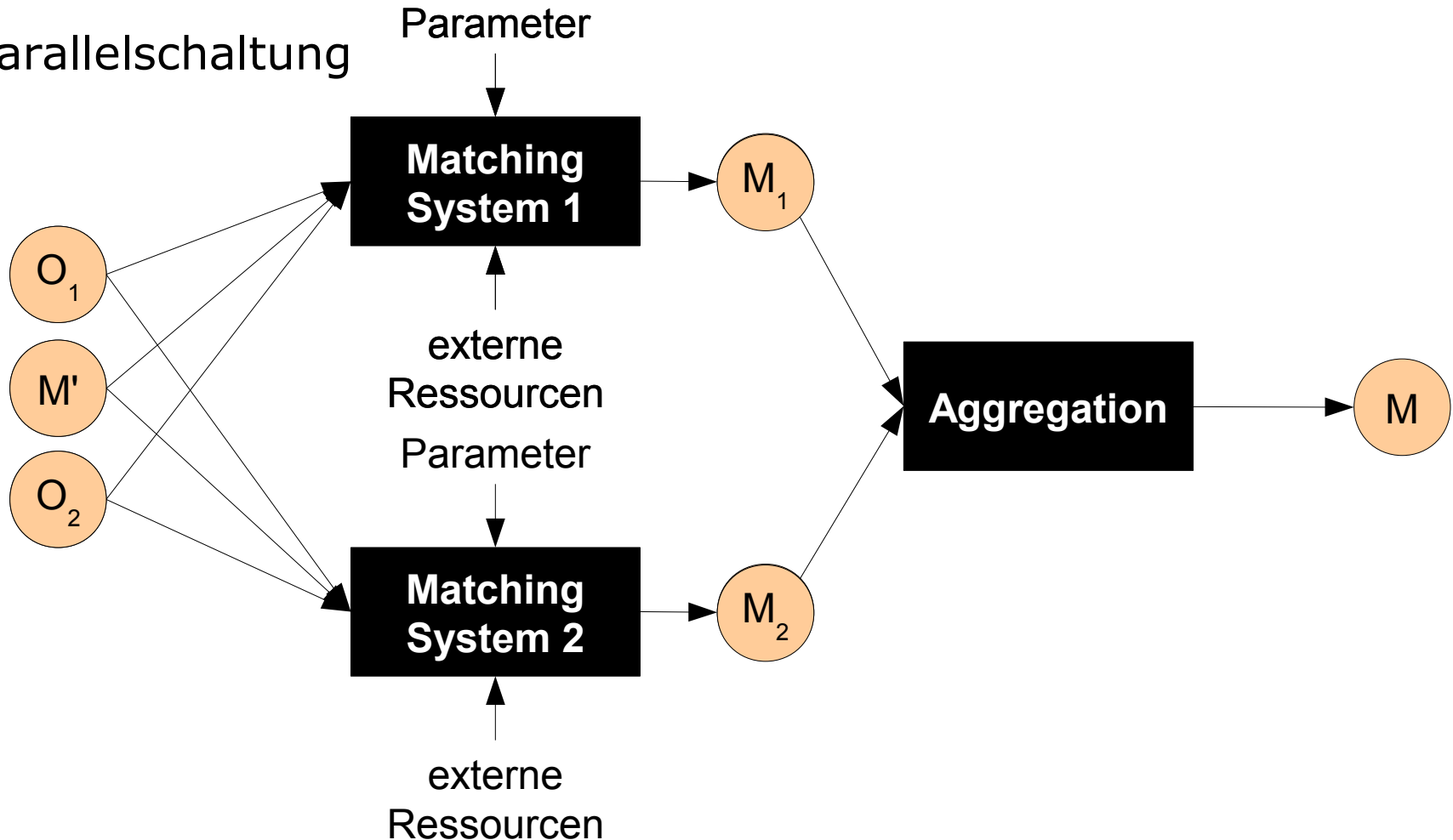
- Reihenschaltung
  - gut für Matcher, die Kandidaten benötigen:
    - Strukturbasierte Verfahren, z.B. Bounded Path
    - Modellbasierte Verfahren (Validierung von Kandidaten)





# Kombination von Matchern

- Parallelschaltung

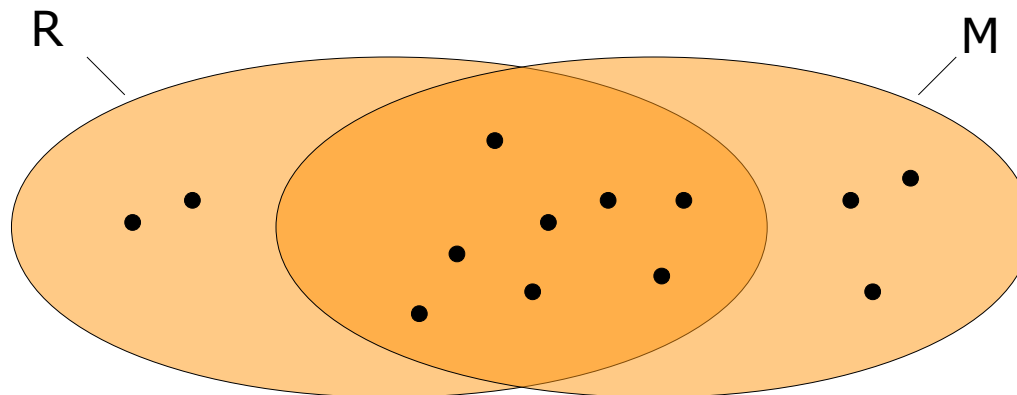


# Kombination von Matchern

- Parallelschaltung
- ermöglicht
  - z.B. verschiedene elementbasierte Techniken
  - unterschiedliche Techniken auf unterschiedlichen Ontologieteilen
- Aggregation der Ergebnisse
  - auf Basis der Konfidenz
  - z.B. gewichtetes Mittel, Minimum, Maximum

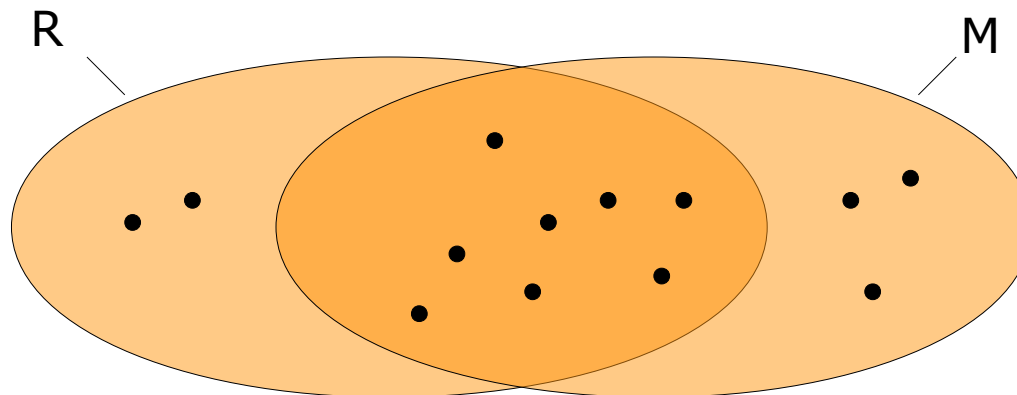
# Auswertung von Mappings

- Gegeben:
  - das vom Matcher gefundene Mapping (M)
  - das tatsächliche Referenz-Mapping (R)
- Recall: der Anteil der gefundenen Elemente des Referenz-Mappings
  - $rec = \#(R \cap M) / \#R$
  - im Beispiel:  $rec = 8/10 = 0.8$



# Auswertung von Mappings

- Gegeben:
  - das vom Matcher gefundene Mapping (M)
  - das tatsächliche Referenz-Mapping (R)
- Precision: der Anteil der korrekten Elemente unter den gefundenen
  - $rec = \#(R \cap M) / \#M$
  - im Beispiel:  $rec = 8/11 = 0.73$



# Auswertung von Mappings

- Recall und Precision sind allein leicht zu optimieren
  - Recall: gib einfach alle möglichen Mappings zurück (Kreuzprodukt)
    - $\rightarrow \text{rec} = 1$
  - Precision
    - gib nur Elemente zurück, die fast sicher stimmen
    - $\rightarrow \text{prec}$  nahe 1 sehr wahrscheinlich
- Zwischen beidem existiert offenbar ein Trade-off
- Daher in der Praxis oft genutzt: F-Measure
  - harmonisches Mittel von Recall und Precision
  - $2 * \text{prec} * \text{rec} / (\text{prec} + \text{rec})$

# Aktuelle Trends und Herausforderungen



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

- Performance verbessern
  - Jährlicher Wettbewerb: OAEI
  - Manche Probleme nur  $\sim 65\%$  F-Measure
- Externes Wissen automatisch finden
  - z.B. im Web, in Linked Open Data, ...
- Matcher automatisch kombinieren und konfigurieren
  - ...

# Beispiel für aktuelle Arbeit: Matcher konfigurieren



- Gegeben: zwei Ontologien, ein partielles Mapping  $M'$ , und ein Matcher mit Parametern
- Gesucht: optimales Mapping mit diesem Matcher
  
- Idee:
  - probiere verschiedene Konfigurationen aus
  - evaluiere auf  $M'$ 
    - Beobachtung:  $F(M')$  und  $F(M)$  sind stark korreliert.
  - erstelle mit der auf  $M'$  besten Konfiguration ein Mapping  $M$
  
- Resultat:
  - für manche Matcher starke Verbesserung möglich

# Zusammenfassung



- Ontology Mappings liefern fehlende Semantik über Datenset-Grenzen
  - Abbildung von Schemata
- Ontology Matching: automatisches Finden von Mappings
  - elementbasiert
  - strukturbasiert
  - mit und ohne externe Quellen
- ...now get your hands dirty!



# Vorlesung Semantic Web



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT

Vorlesung im Wintersemester 2011/2012

Dr. Heiko Paulheim

Fachgebiet Knowledge Engineering