ulm university universität ulm

Semantic Web Grundlagen

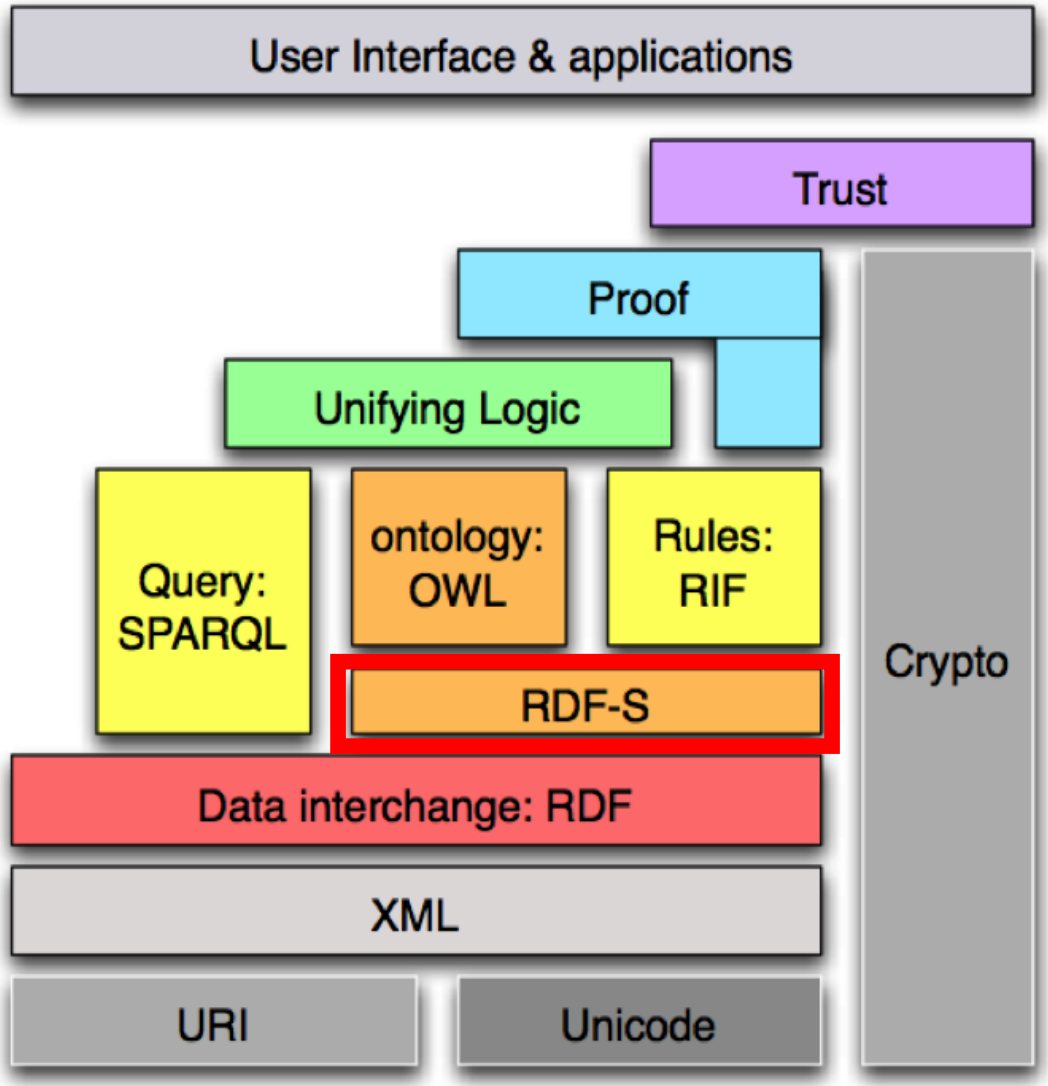
Semantik von RDF(S)

Birte Glimm
Institut für Künstliche Intelligenz | 7. Nov 2011

Organisatorisches: Inhalt

Einleitung und XML	17. Okt	SPARQL Syntax & Intuition	12. Dez
Einführung in RDF	20. Okt	Übung 4	15. Dez
RDF Schema	24. Okt	SPARQL Semantik	19. Dez
fällt aus	27. Okt	SPARQL 1.1	22. Dez
Logik – Grundlagen	31. Okt	Übung 5	9. Jan
Übung 1	3. Nov	SPARQL Entailment	12. Jan
Semantik von RDF(S)	7. Nov	SPARQL Implemetierung	16. Jan
RDF(S) & Datalog Regeln	10. Nov	Abfragen & RIF	19. Jan
OWL Syntax & Intuition	14. Nov	Übung 6	23. Jan
Übung 2	17. Nov	Ontology Editing	26. Jan
OWL & BLs	21. Nov	Ontology Engineering	30. Jan
OWL 2	24. Nov	Linked Data	2. Feb
Tableau	28. Nov	Übung 7	6. Feb
Übung 3	1. Dez	SemWeb Anwendungen	9. Feb
Blocking & Unravelling	5. Dez	Wiederholung	13. Feb
Hypertableau	8. Dez	Übung 8	16. Feb

RDF Schema


User Interface & applications
Trust
Proof
Unifying Logic
Query: SPARQL
ontology: OWL
Rules: RIF
RDF-S
Crypto
Data interchange: RDF
XML
URI
Unicode

Agenda

- Motivation
- Vorbetrachtungen
- Einfache Folgerung
- RDF-Folgerung
- RDFS-Folgerung
- Unzulänglichkeiten von RDF(S)

Warum formale Semantik?

- Nach Einführung von RDFS Kritik von Tool-Herstellern: verschiedene Tools – Inkompatibilitäten (trotz Spezifikation)
- Z.B. bei triple stores:
  - Gleiches RDF-Dokument
  - Gleiche SPARQL-Anfrage
  - Verschiedene Antworten
- Daher: modelltheoretische Semantik für RDF(S)

Agenda

- Motivation
- Vorbetrachtungen
- Einfache Folgerung
- RDF-Folgerung
- RDFS-Folgerung
- Unzulänglichkeiten von RDF(S)

Was ist die Syntax?

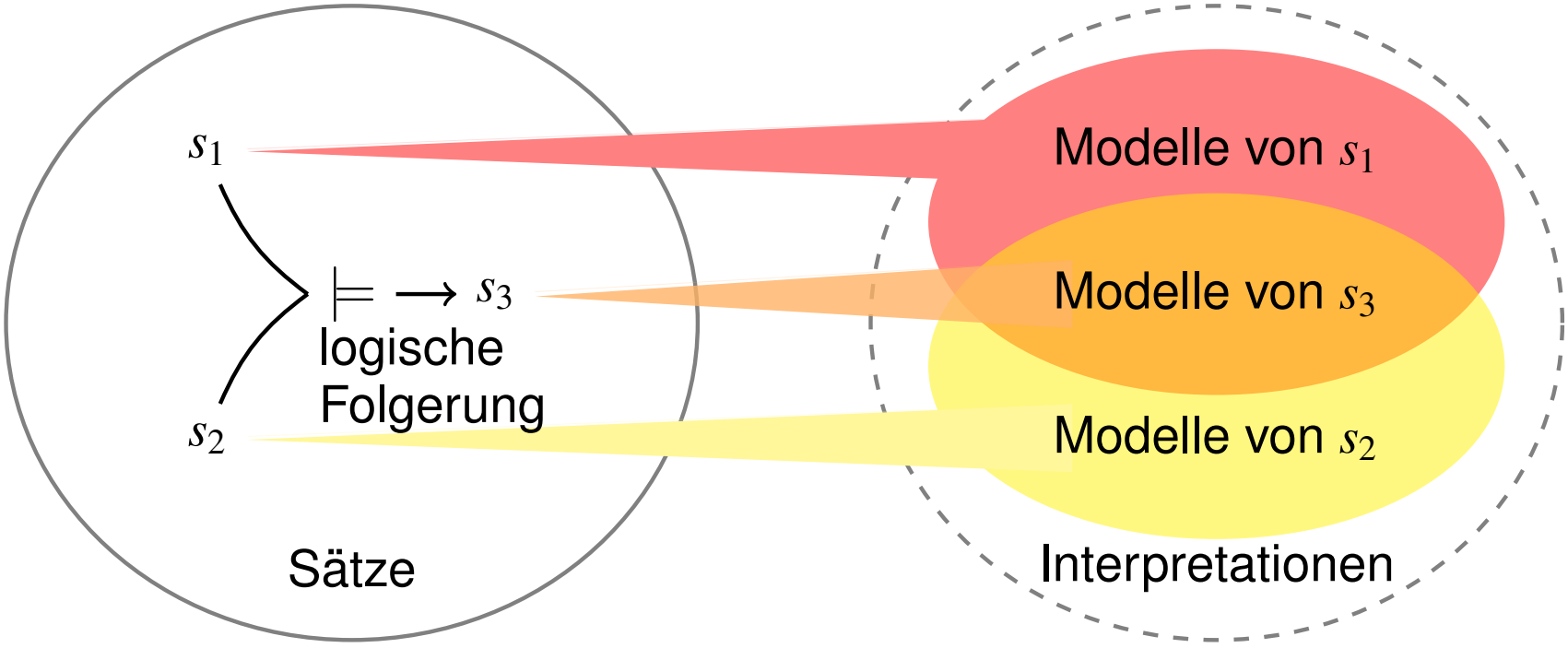
- Also: was sind die Sätze in RDF(S)?
  - Grundelemente (Vokabular V): IRIs, leere Knoten und Literale
    (sind selbst keine Sätze)
  - Jedes Tripel

$$(s, p, o) \in (\mathsf{IRI} \cup \mathsf{bnode}) \times \mathsf{IRI} \times (\mathsf{IRI} \cup \mathsf{bnode} \cup \mathsf{Literal})$$

    ist ein Satz
  - Jede endliche Menge von Tripeln (genannt Graph) ist ein Satz

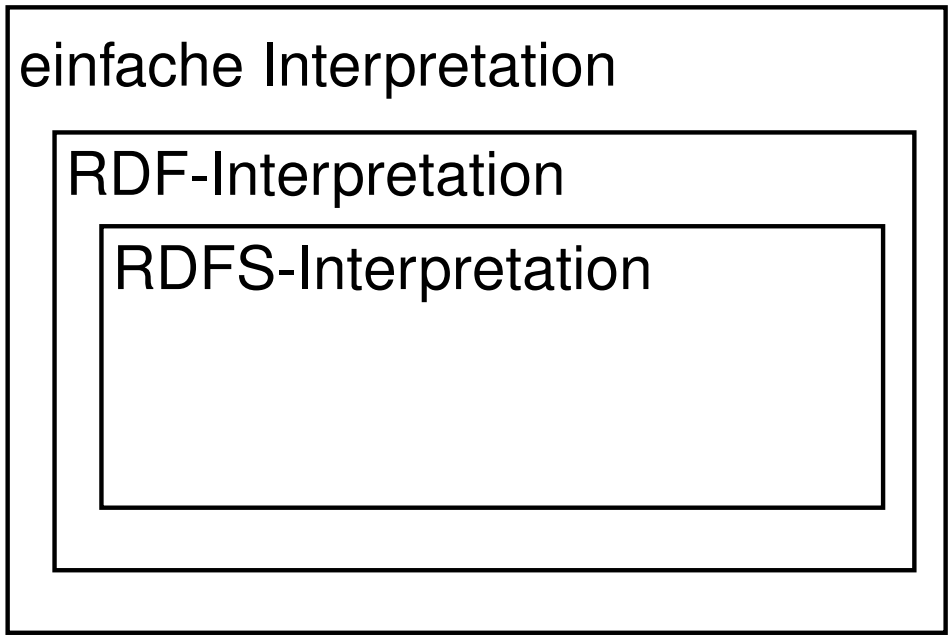
Was ist die Semantik?

- Konsequenzrelation, die sagt, wann ein RDF(S)-Graph G' aus einem RDF(S)-Graphen G folgt, d.h. $G \models G'$
- Modelltheoretische Semantik: wir definieren Menge von Interpretationen und legen fest, wann eine Interpretation Modell eines Graphen ist



Was ist die Semantik?

- Vorgehen schrittweise:



- Je eingeschränkter die Interpretationen umso stärker die Folgerungsrelation

Agenda

- Motivation
- Vorbetrachtungen
- Einfache Folgerung
- RDF-Folgerung
- RDFS-Folgerung
- Unzulänglichkeiten von RDF(S)

Semantik der einfachen Folgerung

Definition

Eine einfache Interpretation $\mathcal{I}$ für ein Vokabular V besteht aus

- IR, einer nichtleeren Menge von *Ressourcen*,
- IP, der Menge der *Propertys* von $\mathcal{I}$,
- I_S, einer Funktion, welche IRIs aus V in die Vereinigung der Mengen IR und IP abbildet, also $I_S : V \to IR \cup IP$,
- I_{EXT}, einer Funktion, welche jeder Property eine Menge von Paaren aus IR zuordnet, also $I_{EXT} : IP \to 2^{IR \times IR}$,
- I_L, einer Funktion von den getypten Literalen aus V in die Menge IR der Ressourcen und
- LV, einer speziellen Teilmenge von IR, genannt Menge der Literalwerte, die (mindestens) alle ungetypten Literale aus V enthält.

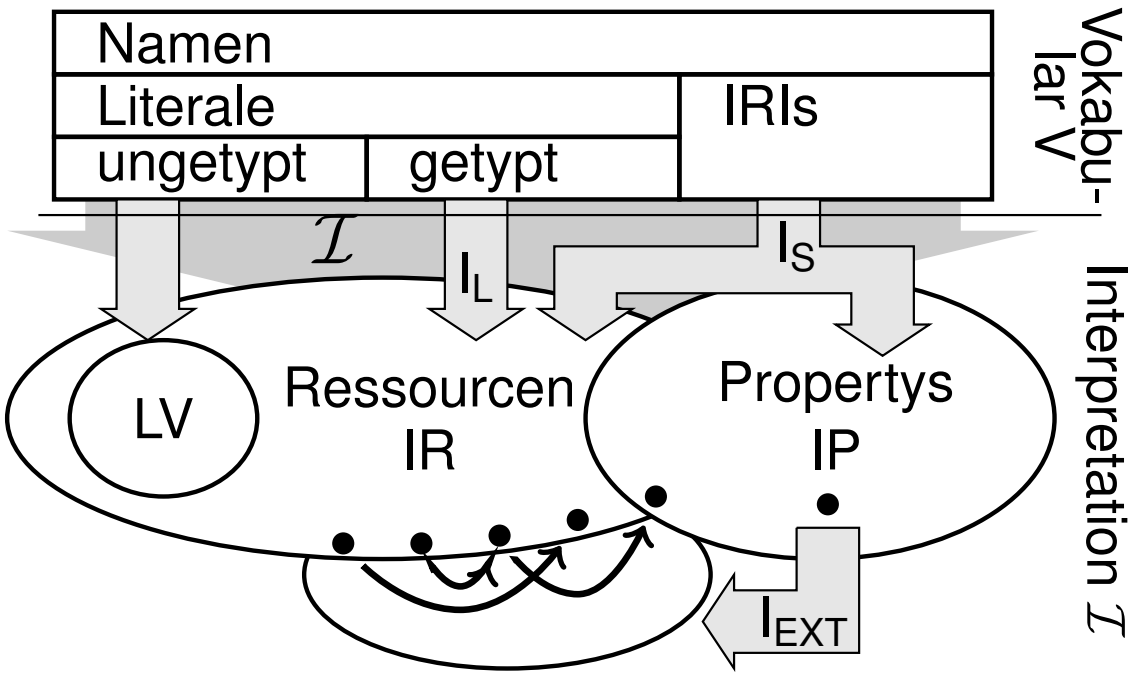
Semantik der einfachen Folgerung

- IR wird auch Domäne oder (Diskurs-)Universum von $\mathcal{I}$ genannt
- $\mathsf{I}_{\mathsf{EXT}}(p)$ wird auch die Extension der Property p genannt

Semantik der einfachen Folgerung

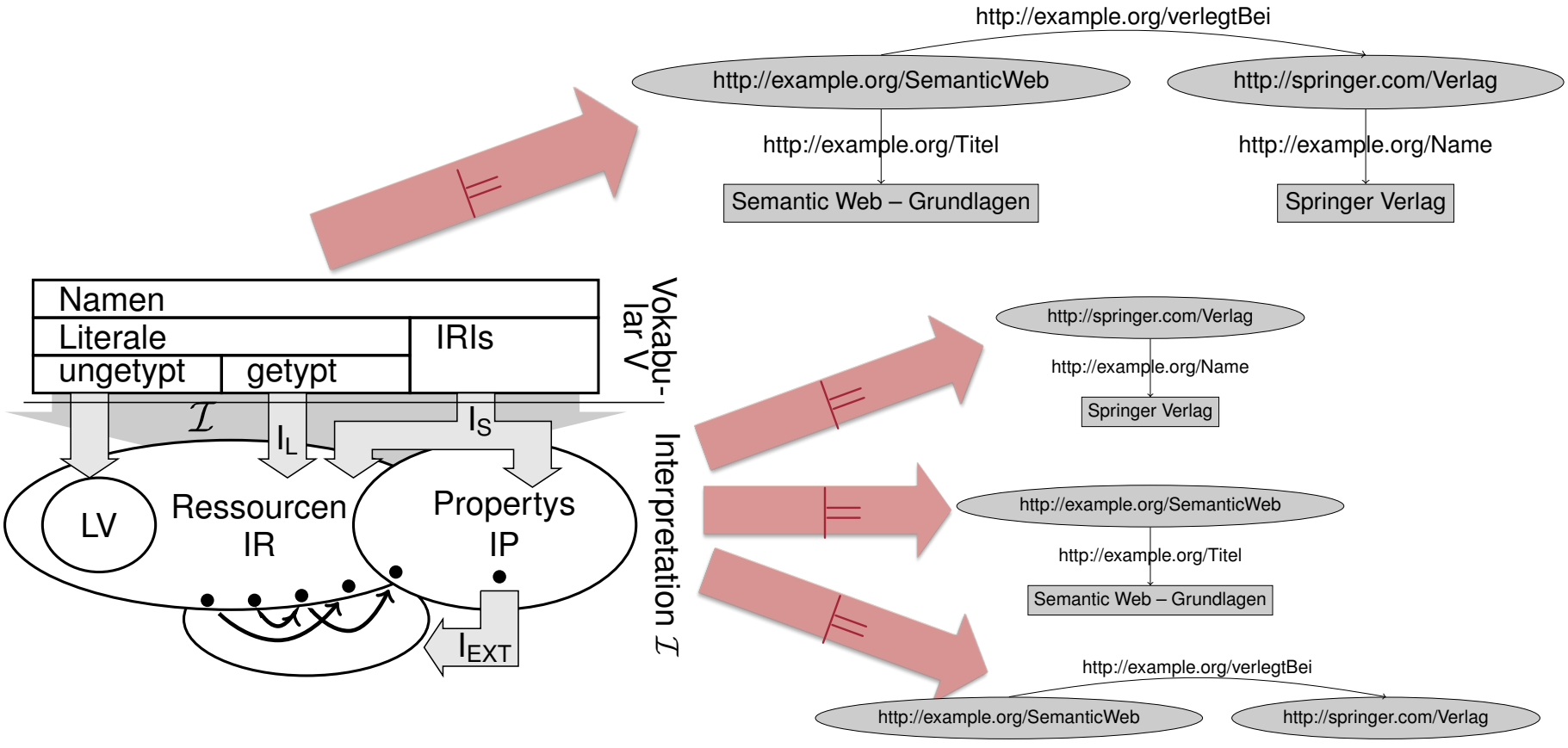
- Ein ungetyptes Literal "a" wird auf a abgebildet: $("a")^{\mathcal{I}} = a$
- Ein ungetypte Literal mit Sprachangabe "a"@t wird auf das Paar $\langle a, t\rangle$ abgebildet: $("a"@t)^{\mathcal{I}} = \langle a, t\rangle$,
- Ein getyptes Literal l wird auf $I_L(l)$ abgebildet: $l^{\mathcal{I}} = I_L(l)$ und
- Eine IRI i wird auf $I_S(i)$ abgebildet: $i^{\mathcal{I}} = I_S(i)$.

Interpretation (schematisch):



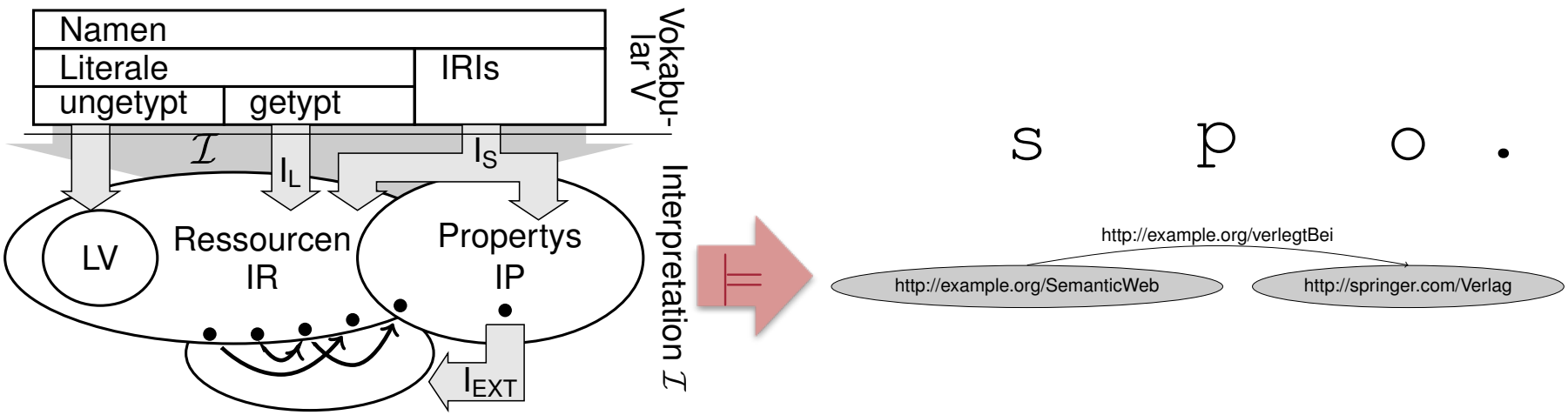
Semantik der einfachen Folgerung

- Frage: Wann ist eine gegebene Interpretation Modell eines Graphen?
- . . . wenn sie Modell jedes Triples des Graphen ist!



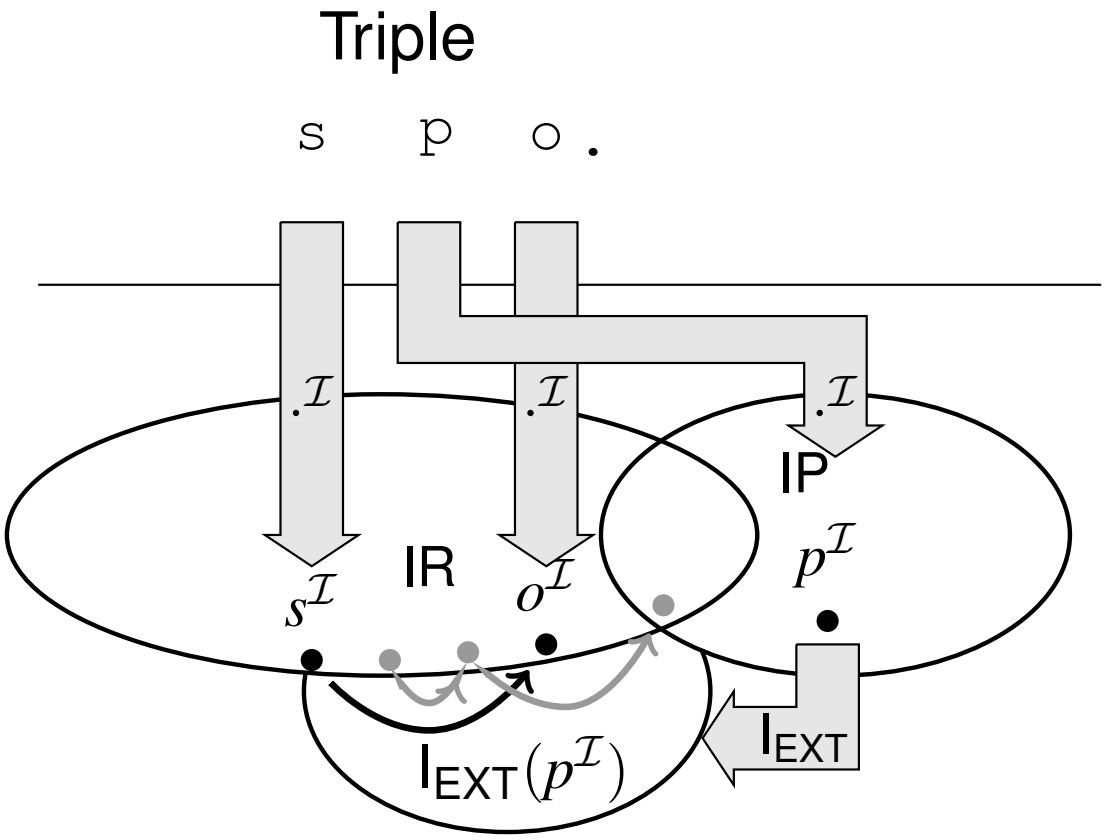
Semantik der einfachen Folgerung

- Frage: Wann ist eine gegebene Interpretation Modell eines Tripels?
- . . . wenn Subjekt, Prädikat und Objekt in V enthalten sind und außerdem $\langle s^{\mathcal{I}}, o^{\mathcal{I}} \rangle \in \mathsf{I}_{\mathsf{EXT}}(p^{\mathcal{I}})$



Semantik der einfachen Folgerung

Schematisch:

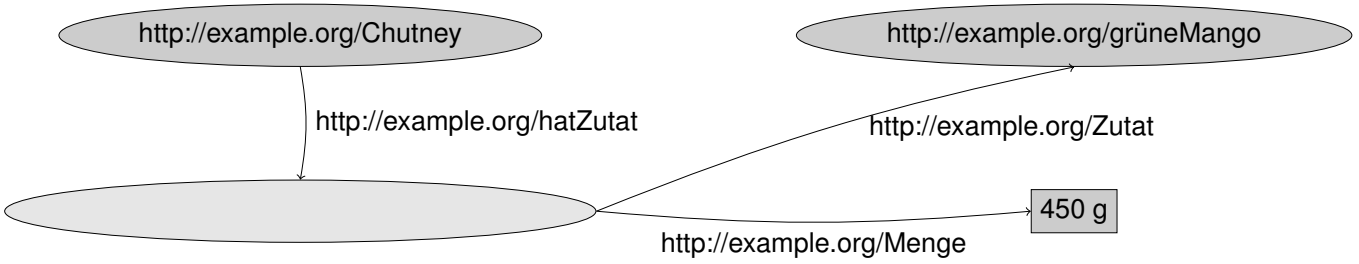


Semantik der einfachen Folgerung

- . . . Ups, wir haben die bnodes vergessen!
- Wird nachgeholt: sei A eine Funktion, die alle bnodes auf Elemente von IR abbildet
- Für eine Interpretation $\mathcal{I}$, sei $\mathcal{I} + A$ wie $\mathcal{I}$, wobei zusätzlich für jeden bnode `_:label` gilt $(\texttt{\_:label})^{\mathcal{I}+A} = A(\texttt{\_:label})$
- Eine Interpretation $\mathcal{I}$ ist nun Modell eines RDF-Graphen G, wenn es ein A gibt, so dass alle Tripel bezüglich $\mathcal{I} + A$ wahr werden

Einfache Interpretation: Beispiel

- Gegeben Graph G:



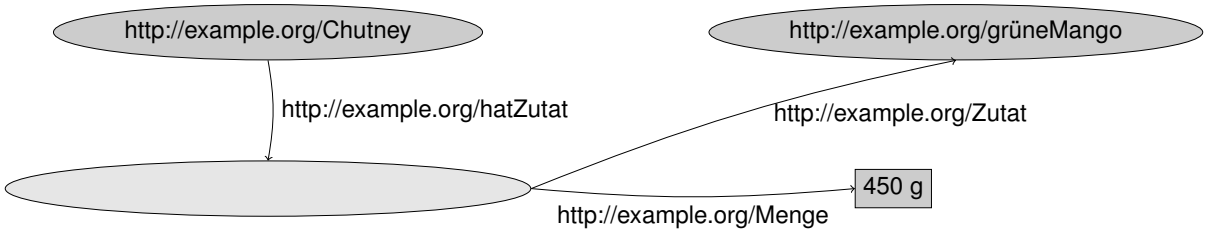
und Interpretation $\mathcal{I}$:

$$
\begin{aligned}
\mathsf{IR} &= \{\chi, \upsilon, \tau, \nu, \epsilon, \iota, 450g\} \\
\mathsf{IP} &= \{\tau, \nu, \iota\} \\
\mathsf{LV} &= \{450g\} \\
\mathsf{I_{EXT}} &= \tau \mapsto \{\langle \chi, \epsilon \rangle\} \\
&\quad\ \nu \mapsto \{\langle \epsilon, \upsilon \rangle\} \\
&\quad\ \iota \mapsto \{\langle \epsilon, 450g \rangle\}
\end{aligned}
\qquad
\begin{aligned}
\mathsf{I_S} &= \texttt{ex:Chutney} \mapsto \chi \\
&\quad\ \texttt{ex:grüneMango} \mapsto \upsilon \\
&\quad\ \texttt{ex:hatZutat} \mapsto \tau \\
&\quad\ \texttt{ex:Zutat} \mapsto \nu \\
&\quad\ \texttt{ex:Menge} \mapsto \iota
\end{aligned}
$$

$\mathsf{I_L}$ ist die “leere Funktion”, da es keine getypten Literale gibt

- . . . ist $\mathcal{I}$ ein Modell von G?

Einfache Interpretation: Beispiel



$$\begin{array}{rcl} \mathsf{IR} & = & \{\chi, \upsilon, \tau, \nu, \epsilon, \iota, 450\mathrm{g}\} \\ \mathsf{IP} & = & \{\tau, \nu, \iota\} \\ \mathsf{LV} & = & \{450\mathrm{g}\} \\ \mathsf{I_{EXT}} & = & \tau \mapsto \{\langle \chi, \epsilon \rangle\} \\ & & \nu \mapsto \{\langle \epsilon, \upsilon \rangle\} \\ & & \iota \mapsto \{\langle \epsilon, 450\mathrm{g} \rangle\} \end{array} \qquad \begin{array}{rcl} \mathsf{I_S} & = & \texttt{ex:Chutney} \mapsto \chi \\ & & \texttt{ex:grüneMango} \mapsto \upsilon \\ & & \texttt{ex:hatZutat} \mapsto \tau \\ & & \texttt{ex:Zutat} \mapsto \nu \\ & & \texttt{ex:Menge} \mapsto \iota \end{array}$$

$\mathsf{I_L}$ ist die “leere Funktion”, da es keine getypten Literale gibt

- Wählt man A: $\texttt{\_:id1} \mapsto \epsilon$, dann ergibt sich

$$\begin{array}{lll} \langle \texttt{ex:Chutney}^{\mathcal{I}+A}, \texttt{\_:id1}^{\mathcal{I}+A} \rangle & = \langle \chi, \epsilon \rangle & \in \mathsf{I_{EXT}}(\tau) = \mathsf{I_{EXT}}(\texttt{ex:hatZutat}^{\mathcal{I}+A}) \\ \langle \texttt{\_:id1}^{\mathcal{I}+A}, \texttt{ex:grüneMango}^{\mathcal{I}+A} \rangle & = \langle \epsilon, \upsilon \rangle & \in \mathsf{I_{EXT}}(\nu) = \mathsf{I_{EXT}}(\texttt{ex:Zutat}^{\mathcal{I}+A}) \\ \langle \texttt{\_:id1}^{\mathcal{I}+A}, \text{“}\texttt{450g}\text{”}^{\mathcal{I}+A} \rangle & = \langle \epsilon, 450\mathrm{g} \rangle & \in \mathsf{I_{EXT}}(\iota) = \mathsf{I_{EXT}}(\texttt{ex:Menge}^{\mathcal{I}+A}) \end{array}$$

- Also ist $\mathcal{I}$ Modell von G

Einfache Folgerung

- Definition der einfachen Interpretation legt (modelltheoretisch) einfache Folgerung für RDF-Graphen fest
- Frage: wie lässt sich diese (abstrakt definierte) Semantik im Sinne des automatischen Schlussfolgerns umsetzen
- Antwort: Ableitungsregeln

Einfache Folgerung

- Ableitungsregeln für einfache Folgerung:

$$\frac{\texttt{u a x .}}{\texttt{u a \_:n .}}\ \text{se1}$$

$$\frac{\texttt{u a x .}}{\texttt{\_:n a x .}}\ \text{se2}$$

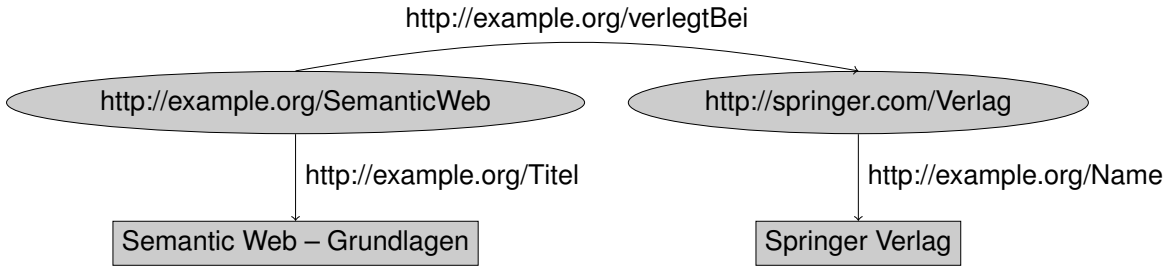
- Bedingung für Anwendung: leerer Knoten nicht bereits anderer IRI/anderem Literal zugeordnet

Einfache Folgerung

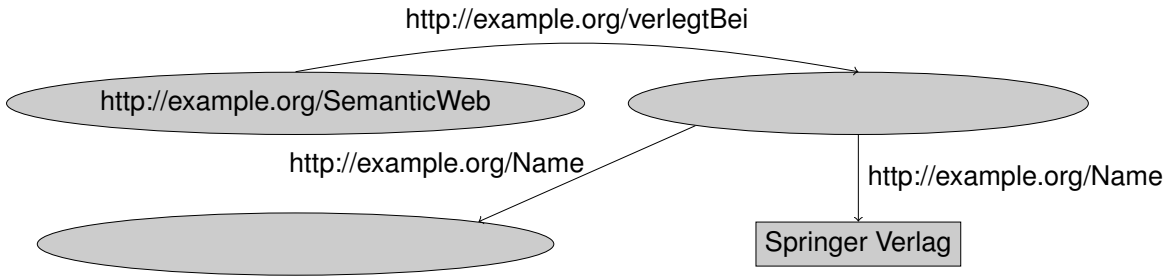
Satz

Ein Graph G_2 folgt einfach aus einem Graphen G_1, wenn G_1 mithilfe der Regeln se1 und se2 zu einem Graphen G_1' ergänzt werden kann, so dass G_2 in G_1' enthalten ist.

Bsp.: Aus



folgt einfach



Agenda

- Motivation
- Vorbetrachtungen
- Einfache Folgerung
- RDF-Folgerung
- RDFS-Folgerung
- Unzulänglichkeiten von RDF(S)

RDF-Interpretationen

- . . . RDF-Interpretationen sind spezielle einfache Interpretationen, wobei für die URIs des RDF-Vokabulars

  `rdf:type rdf:Property rdf:XMLLiteral rdf:nil rdf:List rdf:Statement rdf:subject rdf:predicate rdf:object rdf:first rdf:rest rdf:Seq rdf:Bag rdf:Alt rdf:_1 rdf:_2 ...`

  zusätzliche Forderungen gestellt werden, die die intendierte Semantik der RDF-Bezeichner realisieren:

RDF-Interpretationen

Eine RDF-Interpretation für ein Vokabular V ist eine einfache Interpretation für das Vokabular $V \cup V_{RDF}$, welche zusätzlich folgende Bedingungen erfüllt:

- $x \in \mathsf{IP}$ genau dann, wenn $\langle x, \texttt{rdf:PropertyI}^{\mathcal{I}} \rangle \in \mathsf{I_{EXT}}(\texttt{rdf:type}^{\mathcal{I}})$.
- Wenn "s"^^rdf:XMLLiteral in V enthalten und s ein wohlgeformtes XML-Literal ist, dann
  - $\mathsf{I_L}(\text{"s"}\texttt{\^\^rdf:XMLLiteral})$ ist der XML-Wert von s;
  - $\mathsf{I_L}(\text{"s"}\texttt{\^\^rdf:XMLLiteral}) \in \mathsf{LV}$;
  - $\langle \mathsf{I_L}(\text{"s"}\texttt{\^\^rdf:XMLLiteral}), \texttt{rdf:XMLLiteral}^{\mathcal{I}} \rangle \in \mathsf{I_{EXT}}(\texttt{rdf:type}^{\mathcal{I}})$
- Wenn "s"^^rdf:XMLLiteral in V enthalten und s ein *nicht* wohlgeformtes XML-Literal ist, dann
  - $\mathsf{I_L}(\text{"s"}\texttt{\^\^rdf:XMLLiteral}) \notin \mathsf{LV}$ und
  - $\langle \mathsf{I_L}(\text{"s"}\texttt{\^\^rdf:XMLLiteral}), \texttt{rdf:XMLLiteral}^{\mathcal{I}} \rangle \notin \mathsf{I_{EXT}}(\texttt{rdf:type}^{\mathcal{I}})$.

RDF-Interpretationen

- Merke: x ist eine Property genau dann, wenn es mit der durch `rdf:Property` bezeichneten Ressource über die `rdf:type`-Property verbunden ist (dies führt auch automatisch dazu, dass für jede RDF-Interpretation $IP \subseteq IR$ gilt).
- Der Wertebereich des `rdf:XMLLiteral` Datentyps enthält für jede wohlgeformte XML-Zeichenkette genau einen sogenannten XML-Wert. Die RDF-Spezifikation fordert lediglich, dass ein XML-Wert keine XML-Zeichenkette, kein Datenwert irgendeines XML-Schema-Datentyps und auch keine Unicode-Zeichenkette ist. Für den intuitiven Gebrauch kann man sich die XML-Werte einfach als XML-Zeichenketten vorstellt.

RDF-Interpretationen

- Zusätzliche Forderung: jede RDF-Interpretation muss Modell der folgenden, “axiomatischen” Tripel sein:

```
rdf:type        rdf:type  rdf:Property.
rdf:subject     rdf:type  rdf:Property.
rdf:predicate   rdf:type  rdf:Property.
rdf:object      rdf:type  rdf:Property.
rdf:first       rdf:type  rdf:Property.
rdf:rest        rdf:type  rdf:Property.
rdf:value       rdf:type  rdf:Property.
rdf:_1          rdf:type  rdf:Property.
rdf:_2          rdf:type  rdf:Property.
...             rdf:type  rdf:Property.
rdf:nil         rdf:type  rdf:Property.
```

RDF-Folgerungen

- Automatische Folgerungen werden wieder über Ableitungsregeln realisiert:

$$\frac{}{\texttt{u a x}}\ \text{rdfax}$$

Jedes axiomatische Tripel “u a x .” kann immer abgeleitet werden

$$\frac{\texttt{u a l}}{\texttt{u a \_:n}}\ \text{lg}$$

Literale dürfen durch nicht anderweitig gebundene bnodes ersetzt werden

$$\frac{\texttt{u a y}}{\texttt{a rdf:type rdf:Property}}\ \text{rdf1}$$

Für jedes Tripelprädikat kann abgeleitet werden, dass es eine Entität aus der Klasse der Properties ist

$$\frac{\texttt{u a l}}{\texttt{\_:n rdf:type rdf:XMLLiteral}}\ \text{rdf2}$$

Wenn `_:n` durch lg dem wohlgeformten XML-Literal l zugewiesen wurde

RDF-Folgerung

- Satz: Ein Graph G_2 RDF-folgt aus einem Graphen G_1, wenn es einen Graphen G_1' gibt, so dass
  - G_1' aus G_1 via lg, rdf1, rdf2 und rdfax hergeleitet werden kann und
  - G_2 aus G_1' einfach folgt.

- Beachte: zweistufiger Folgerungsprozess

Agenda

- Motivation
- Vorbetrachtungen
- Einfache Folgerung
- RDF-Folgerung
- RDFS-Folgerung
- Unzulänglichkeiten von RDF(S)

RDFS-Interpretationen

. . . RDFS-Interpretationen sind spezielle RDF-Interpretationen, wobei für die URIs des RDFS-Vokabulars

```
rdfs:domain       rdfs:range            rdfs:Resource
rdfs:Literal      rdfs:Datatype         rdfs:Class
rdfs:subClassOf   rdfs:subPropertyOf    rdfs:Container
rdfs:member       rdfs:ContainerMembershipProperty
rdfs:comment      rdfs:seeAlso          rdfs:isDefinedBy
rdfs:label
```

zusätzliche Forderungen gestellt werden, die die intendierte Semantik der RDF-Bezeichner realisieren:

RDFS-Interpretationen

- Aus Gründen der einfacheren Darstellung führen wir für eine gegebene RDF-Interpretation die Funktion $\mathsf{I}_{\mathsf{CEXT}}$ ein, die Ressourcen auf Mengen von Ressourcen abbildet (also: $\mathsf{I}_{\mathsf{CEXT}} : \mathsf{IR} \to 2^{\mathsf{IR}}$). Dabei enthalte $\mathsf{I}_{\mathsf{CEXT}}(y)$ genau die Elemente x, für die $\langle x, y\rangle$ in $\mathsf{I}_{\mathsf{EXT}}(I(\texttt{rdf:type}))$ enthalten ist. $\mathsf{I}_{\mathsf{CEXT}}(y)$ nennt man auch die *(Klassen-)Extension* von y.
- Weiterhin definieren wir IC als die Extension der speziellen IRI `rdfs:Class`, also: $\mathsf{IC} = \mathsf{I}_{\mathsf{CEXT}}(\texttt{rdfs:Class}^{\mathcal{I}})$.
- Merke: Sowohl $\mathsf{I}_{\mathsf{CEXT}}$ als auch IC sind durch $\cdot^{\mathcal{I}}$ sowie $\mathsf{I}_{\mathsf{EXT}}$ bereits eindeutig festgelegt.

RDFS-Interpretationen

Eine *RDFS-Interpretation* für ein Vokabular V ist eine RDF-Interpretation des Vokabulars $V \cup V_{RDFS}$, welche zusätzlich die folgenden Kriterien erfüllt:

- $IR = I_{CEXT}(\texttt{rdfs:Resource}^{\mathcal{I}})$
  Jede Ressource ist vom Typ `rdfs:Resource`.
- $LV = I_{CEXT}(\texttt{rdfs:Literal}^{\mathcal{I}})$
  Jedes ungetypte und jedes wohlgeformte getypte Literal ist vom Typ `rdfs:Literal`.
- Wenn $\langle x, y\rangle \in I_{EXT}(\texttt{rdfs:domain}^{\mathcal{I}})$ und $\langle u, v\rangle \in I_{EXT}(x)$, dann $u \in I_{CEXT}(y)$.
  Ist x mit y durch die Property `rdfs:domain` verbunden und verbindet die Property x die Ressourcen u und v, dann ist u vom Typ y.

RDFS-Interpretationen

- Wenn $\langle x, y\rangle \in \mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:range}^{\mathcal{I}})$ und $\langle u, v\rangle \in \mathsf{I}_{\mathsf{EXT}}(x)$, dann $v \in \mathsf{I}_{\mathsf{CEXT}}(y)$.
  Ist x mit y durch die Property `rdfs:range` verbunden und verbindet die Property x die Ressourcen u und v, dann ist v vom Typ y.
- $\mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:subPropertyOf}^{\mathcal{I}})$ ist reflexiv und transitiv auf IP.
  Die `rdfs:subPropertyOf`-Property verbindet jede Property mit sich selbst.
  Darüber hinaus gilt: Verbindet `rdfs:subPropertyOf` die Property x mit Property y und außerdem y mit der Property z, so verbindet `rdfs:subPropertyOf` auch x direkt mit z.

RDFS-Interpretationen

- Wenn $\langle x, y\rangle \in \mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:subPropertyOf}^{\mathcal{I}})$, dann $x, y \in \mathsf{IP}$ und $\mathsf{I}_{\mathsf{EXT}}(x) \subseteq \mathsf{I}_{\mathsf{EXT}}(y)$.
  Wird x mit y durch `rdfs:subPropertyOf` verbunden, dann sind sowohl x als auch y Propertys und jedes in der Extension von x enthaltene Ressourcenpaar ist auch in der Extension von y enthalten.
- Wenn $x \in \mathsf{IC}$, dann $\langle x, \texttt{rdfs:Resource}^{\mathcal{I}}\rangle \in \mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$.
  Bezeichnet x eine Klasse, dann muss es eine Unterklasse der Klasse aller Ressourcen sein, d.h., das Paar aus x und `rdfs:Resource` ist in der Extension von `rdfs:subClassOf`.

RDFS-Interpretationen

- Wenn $\langle x, y\rangle \in \mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$, dann $x, y \in \mathsf{IC}$ und $\mathsf{I}_{\mathsf{CEXT}}(x) \subseteq \mathsf{I}_{\mathsf{CEXT}}(y)$.
  Stehen x und y in der `rdfs:subClassOf`-Beziehung, sind sowohl x als auch y Klassen und die (Klassen-)Extension von x ist Teilmenge der (Klassen-)Extension von y.
- $\mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$ ist reflexiv und transitiv auf IC.
  Die `rdfs:subClassOf`-Property verbindet jede Klasse mit sich selbst.
  Darüber hinaus folgt, wann immer diese Property Klasse x mit Klasse y und Klasse y mit Klasse z verbindet, dass sie x auch direkt mit z verbindet.

RDFS-Interpretationen

- Wenn $x \in \mathsf{I}_{\mathsf{CEXT}}(\texttt{rdfs:ContainerMembershipProperty}^{\mathcal{I}})$, dann $\langle x, \texttt{rdfs:member}^{\mathcal{I}} \rangle \in \mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:subPropertyOf}^{\mathcal{I}})$.
  Ist x eine Property vom Typ `rdfs:ContainerMembershipProperty`, so steht sie in der `rdfs:subPropertyOf`-Beziehung zur `rdfs:member`-Property.
- Wenn $x \in \mathsf{I}_{\mathsf{CEXT}}(\texttt{rdfs:Datatype}^{\mathcal{I}})$, dann $\langle x, \texttt{rdfs:Literal}^{\mathcal{I}} \rangle \in \mathsf{I}_{\mathsf{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$.
  Ist ein x als Element der Klasse `rdfs:Datatype` “getypt”, dann muss dieses auch eine Unterklasse der Klasse aller Literalwerte (bezeichnet mit `rdfs:Literal`) sein.

- . . . dazu kommen dann noch jede Menge weitere axiomatische Tripel:

RDFS-Interpretationen

```
rdf:type             rdfs:domain      rdfs:Resource .
rdfs:domain          rdfs:domain      rdf:Property .
rdfs:range           rdfs:domain      rdf:Property .
rdfs:subPropertyOf   rdfs:domain      rdf:Property .
rdfs:subClassOf      rdfs:domain      rdfs:Class .
rdf:subject          rdfs:domain      rdf:Statement .
rdf:predicate        rdfs:domain      rdf:Statement .
rdf:object           rdfs:domain      rdf:Statement .
rdfs:member          rdfs:domain      rdfs:Resource .
rdf:first            rdfs:domain      rdf:List .
rdf:rest             rdfs:domain      rdf:List .
rdfs:seeAlso         rdfs:domain      rdfs:Resource .
rdfs:isDefinedBy     rdfs:domain      rdfs:Resource .
rdfs:comment         rdfs:domain      rdfs:Resource .
rdfs:label           rdfs:domain      rdfs:Resource .
rdf:value            rdfs:domain      rdfs:Resource .
```

RDFS-Interpretationen

```
rdf:type              rdfs:range    rdfs:Class .
rdfs:domain           rdfs:range    rdfs:Class .
rdfs:range            rdfs:range    rdfs:Class .
rdfs:subPropertyOf    rdfs:range    rdf:Property .
rdfs:subClassOf       rdfs:range    rdfs:Class .
rdf:subject           rdfs:range    rdfs:Resource .
rdf:predicate         rdfs:range    rdfs:Resource .
rdf:object            rdfs:range    rdfs:Resource .
rdfs:member           rdfs:range    rdfs:Resource .
rdf:first             rdfs:range    rdfs:Resource .
rdf:rest              rdfs:range    rdf:List .
rdfs:seeAlso          rdfs:range    rdfs:Resource .
rdfs:isDefinedBy      rdfs:range    rdfs:Resource .
rdfs:comment          rdfs:range    rdfs:Literal .
rdfs:label            rdfs:range    rdfs:Literal .
rdf:value             rdfs:range    rdfs:Resource .
```

RDFS-Interpretationen

```
rdfs:ContainerMembershipProperty
                        rdfs:subClassOf     rdf:Property .
rdf:Alt                 rdfs:subClassOf     rdfs:Container .
rdf:Bag                 rdfs:subClassOf     rdfs:Container .
rdf:Seq                 rdfs:subClassOf     rdfs:Container .

rdfs:isDefinedBy        rdfs:subPropertyOf  rdfs:seeAlso .

rdf:XMLLiteral          rdf:type            rdfs:Datatype .
rdf:XMLLiteral          rdfs:subClassOf     rdfs:Literal .
rdfs:Datatype           rdfs:subClassOf     rdfs:Class .

rdf:_1                  rdf:type
                           rdfs:ContainerMembershipProperty .
rdf:_1                  rdfs:domain         rdfs:Resource .
rdf:_1                  rdfs:range          rdfs:Resource .
rdf:_2                  rdf:type
                           rdfs:ContainerMembershipProperty .
...
```

RDFS-Folgerung

Automatische Folgerungen werden wieder über Ableitungsregeln realisiert:

$$\frac{}{\texttt{u a x .}}\ \text{rdfsax}$$

Jedes axiomatische Tripel “u a x .” kann immer abgeleitet werden

$$\frac{\texttt{u a \_:n .}}{\texttt{u a l .}}\ \text{gl}$$

Die Umkehrung von Regel lg: `_:n` wurde von der Regel lg für das ungetypte Literal `l` vergeben

$$\frac{\texttt{u a l .}}{\texttt{\_:n rdf:type rdfs:Literal}}\ \text{rdfs1}$$

`_:n` wurde von der Regel lg für das ungetypte Literal `l` vergeben

$$\frac{\texttt{a rdfs:domain x . u a y .}}{\texttt{u rdf:type x .}}\ \text{rdfs2}$$

Implementiert die Semantik von Domänen für Propertys

$$\frac{\texttt{a rdfs:range x . u a v .}}{\texttt{v rdf:type x .}}\ \text{rdfs3}$$

Implementiert die Semantik von Ranges für Propertys

`a, b` IRIs
`u, v` IRI oder blank node
`x, y` IRI, blank node oder Literal
`l` Literal
`_:n` blank nodes

RDFS-Folgerung

$$\frac{u\ a\ x\ .}{u\ \text{rdf:type}\ \text{rdfs:Resource}\ .}\ \text{rdfs4a}$$

Das Subjekt jedes Triples ist eine Ressource

$$\frac{u\ a\ v\ .}{v\ \text{rdf:type}\ \text{rdfs:Resource}\ .}\ \text{rdfs4b}$$

Objekte, die keine Literale sind, sind ebenfalls eine Ressourcen

$$\frac{u\ \text{rdfs:subPropertyOf}\ v\ .\ v\ \text{rdfs:subPropertyOf}\ x\ .}{u\ \text{rdfs:subPropertyOf}\ x\ .}\ \text{rdfs5}$$

Transitivität

$$\frac{u\ \text{rdf:type}\ \text{rdf:Property}\ .}{u\ \text{rdfs:subPropertyOf}\ u\ .}\ \text{rdfs6}$$

Reflexivität

$$\frac{a\ \text{rdfs:subPropertyOf}\ b\ .\ u\ a\ y\ .}{u\ b\ y\ .}\ \text{rdfs7}$$

Sub-Property Schlussfolgerungen für Instanzen

$$\frac{u\ \text{rdf:type}\ \text{rdfs:Class}\ .}{u\ \text{rdf:type}\ \text{rdfs:Resource}\ .}\ \text{rdfs8}$$

Klassen sind Ressourcen

RDFS-Folgerung

$$\frac{\texttt{u rdfs:subClassOf x . v rdf:type u .}}{\texttt{v rdf:type x .}}\ \text{rdfs9}$$ Sub-Klassen Schlussfolgerungen für Instanzen

$$\frac{\texttt{u rdf:type rdfs:Class .}}{\texttt{u rdfs:subClassOf u .}}\ \text{rdfs10}$$ Reflexivität

$$\frac{\texttt{u rdfs:subClassOf v . v rdfs:subClassOf x .}}{\texttt{u rdfs:subClassOf x .}}\ \text{rdfs11}$$ Transitivität

$$\frac{\texttt{u rdf:type rdfs:ContainerMembershipProperty .}}{\texttt{u rdfs:subPropertyOf rdfs:member .}}\ \text{rdfs12}$$

$$\frac{\texttt{u rdf:type rdfs:Datatype .}}{\texttt{u rdfs:subClassOf rdfs:Literal .}}\ \text{rdfs10}$$ Jeder Datentyp ist eine Unterklasse von `rdfs:Literal`

RDFS-Folgerung

- Wichtige Definition: XML-Clash

```
ex:hatSmiley          rdfs:range     rdf:Literal .
ex:böseBemerkung      ex:hatSmiley   ">:->"^^rdf:XMLLiteral .
```

- Tritt auf, wenn einem Knoten vom Typ `rdf:Literal` ein nicht-wohlgeformter Literalwert zugewiesen werden muss

RDFS-Folgerung

Satz:

Ein Graph RDFS-folgt aus G_1 genau dann, wenn es einen Graphen G_1' gibt, der durch Anwendung der Regeln lg, gl, rdfax, rdf1, rdf2, rdfs1 – rdfs13 und rdfsax aus G_1 folgt, so dass

- G_2 aus G_1' einfach folgt oder
- G_1' einen XML-Clash enthält.

Agenda

- Motivation
- Vorbetrachtungen
- einfache Folgerung
- RDF-Folgerung
- RDFS-Folgerung
- Unzulänglichkeiten von RDF(S)

Was kann RDF(S) nicht?

- Bestimmte (vernünftig) scheinende Folgerungen können nicht RDFS-gefolgert werden, z.B.

```
ex:sprichtMit   rdfs:domain       ex:Homo .
ex:Homo         rdfs:subClassOf   ex:Primates .
```

impliziert

```
ex:sprichtMit   rdfs:domain   ex:Primates .
```

- Mögliche Lösung: noch stärkere, “extensionale”, Semantik
- Keine Möglichkeit, Negation auszudrücken