Query-Answering from Linked Data

Umberto Straccia

CNR-ISTI, Pisa, Italy

umberto.straccia@isti.cnr.it
www.umbertostraccia.it/cs

IA2 Autumn School on Artificial Intelligence:
Managing Imperfect and Heterogeneous Information and Data

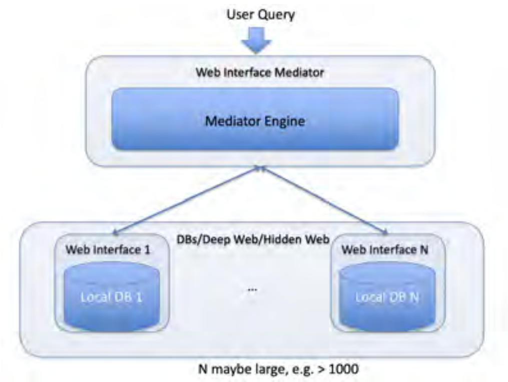
Funded by the European Union

November 2023

Overview

- General Scenario & Problem Description
- Proposed Solutions (sketch)

General Scenario & Problem Description



- Input: conjunctive query over a global mediator schema
- Problem: query the N resources
  - if N large, quering all N resources is unrealistic

Proposed Solution [Calì and Straccia, 2015, Calì and Straccia, 2017, Straccia and Troncy, 2006]

To integrate Distributed Information Resources (DIRs), one may adopt the Global As View (GAV) [Lenzerini, 2002] approach:

- Queries are posed on a Global Mediator Schema
  - It contains relational structures (relations), where each relation is associated with a query on the underlying Local Information Resources (LIRs)
  - A query over the mediator schema is processed by evaluating suitable queries over the LIRs

The Case of Small Size Mediators

- For query q over global schema $\mathcal{G}$
  1. Rewrite the query q into a set $\{q_i\}$ of queries over of the local schemas $\mathcal{S}$
  2. Submit the queries to the LIRs accessed through wrappers
  3. Merge all the ranked lists and provide the result back to the user

The Case of Large Size Mediators

- Quering all LIRs is unfeasible
- Borrow ideas from textual Distributed Information Retrieval [Shokouhi and Si, 2011, Thomas, 2012]. That is,
  - Sample each LIR
  - Use the samples to determine the top-*s* most relevant LIRs to query
  - Once queries are submitted, merge the ranked list of answers

- Sample the LIRs before hand
- For query q over global schema $\mathcal{G}$
  1. Rewrite the query q into a set $\{q_i\}$ of queries over of the local schemas $\mathcal{S}$
  2. Use the samples to determine which of the q_i are the top-s most relevant queries
  3. Submit the queries to the LIRs accessed through wrappers
  4. Merge all the ranked lists and provide the result back to the user

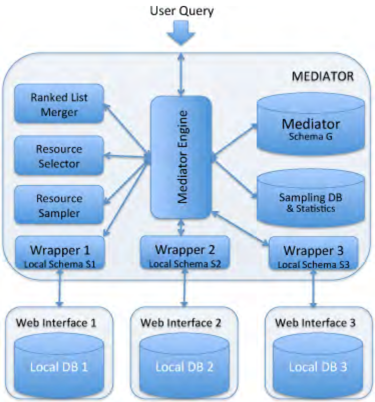


Figure: Architecture of a Mediator.

Data integration Model (General)

Data integration setting: Global As View (GAV)

- $\mathcal{G} = \{R_1, \ldots, R_n\}$
  - Relational entities of the Mediator's *Global Schema*
- $\mathcal{D} = \{D_1, \ldots, D_m\}$
  - LIRs, distributed local databases D_i
- $\mathcal{S} = \{S_1, ..., S_m\}$
  - Local relational entities used to access $D_i \in \mathcal{D}$
  - There is exactly one relation S_i through which we access D_i
- For $S_i \in \mathcal{S}$, $D_i \in \mathcal{D}$, vector of variables and constants $\mathbf{z}$, the local answer set of local query $S_i(\mathbf{z})$ over database D_i, is

$$ans(S_i(\mathbf{z}), D_i) = \{\mathbf{t} \mid D_i \models S_i(\mathbf{t}) \text{ s.t. } \mathbf{t} \text{ agrees with } \mathbf{z} \text{ on the constants in } \mathbf{z}\}.$$

- We assume tuples in $ans(S_i(\mathbf{z}), D_i)$ are ordered
- $ans_k(S_i(\mathbf{z}), D_i)$, top-$k$ retrieved tuples

Data Integration Model (General) cont.

- More than one database may be used to instantiate a global relation $R \in \mathcal{G}$
- E.g., $R(\texttt{carModel}, \texttt{year}, \texttt{partNumber}) \in \mathcal{G}$, there may be more than one information resources through which the Mediator may find car spare parts
- We model this using so-called mapping rules, i.e.
  - For each global $R \in \mathcal{G}$, let $\mathcal{M}_R$ be the set of k_R *mapping rules* w.r.t. R

$$
\begin{array}{rcl}
R(\mathbf{x}) & \leftarrow & S_{1_R}(\mathbf{x}) \\
 & \vdots & \\
R(\mathbf{x}) & \leftarrow & S_{k_R}(\mathbf{x}) ,
\end{array}
$$

- $\mathcal{M}$, set of all *mapping rules* w.r.t. $R \in \mathcal{G}$, i.e. $\mathcal{M} = \bigcup_{R \in \mathcal{G}} \mathcal{M}_R$.
- We assume that each $S \in \mathcal{S}$ is *typed*, in the sense that each attribute of a relation in $\mathcal{S}$ has a type (e.g. string, integer etc.).

- Conjunctive Query (CQ) $q(\mathbf{x})$ over global schema $\mathcal{G}$ is a rule r

$$q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$$

  where

  - $\varphi(\mathbf{x}, \mathbf{y})$ is a conjunction of relations $R \in \mathcal{G}$
  - $q(\mathbf{x})$ is called *head*
  - $\exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$ is called *body*
  - $\mathbf{x}$ are called *distinguished variables*
  - $\mathbf{y}$ are called *non-distinguished variables*
  - The existential $\exists \mathbf{y}$ may be omitted

- A Disjunctive Query (DQ) $q(\mathbf{x})$ is a set $\{r_1, ..., r_n\}$ of CQs r_i with head $q(\mathbf{x})$
- The answer set of CQ q is

$$ans(q, \mathcal{D}, \mathcal{M}) = \{\mathbf{t} \mid \mathcal{D} \cup \mathcal{M} \cup \{q\} \models q(\mathbf{t})\}$$

  i.e. the query body of some rule r_i evaluates to true

- As for LIRs, $ans(q, \mathcal{D}, \mathcal{M})$ is an ordered set
- $ans_k(q, \mathcal{D}, \mathcal{M})$ are the top-k retrieved tuples in $ans(q, \mathcal{D}, \mathcal{M})$

- Complex Conjunctive Query (CCQ) $q(\mathbf{x})$ over global schema $\mathcal{G}$ is a rule r (see [Straccia, 2014, Zimmermann et al., 2012]):

$$q(\mathbf{x}, s) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y}), \text{GroupedBy}(\mathbf{w}), s = @[f(\mathbf{z})] \ ,$$

  where additionally
  - $@ \in \{\text{SUM}, \text{AVG}, \text{MAX}, \text{MIN}, \text{COUNT}\}$ is an *aggregation operator*
  - $s = @[f(\mathbf{z})]$ is *scoring atom* and s is *scoring variable*
  - grouping, aggregation and scoring are optional
- A Complex Disjunctive Query (CDQ) $q(\mathbf{x})$ is a set $\{r_1, ..., r_n\}$ of CCQs r_i with head $q(\mathbf{x})$
- The answer set of a CCQ q is

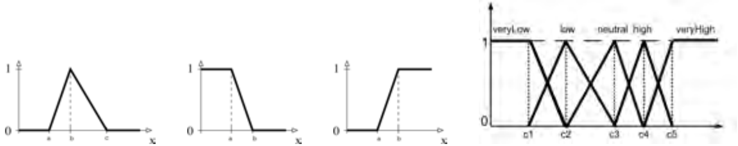
$$ans(q, \mathcal{D}, \mathcal{M}) = \{\langle \mathbf{t}, s \rangle \mid \mathcal{D} \cup \mathcal{M} \cup \{q\} \models q(\mathbf{t}, s)\} \ ,$$

  where each tuple has an unique score
- As for LIRs, $ans(q, \mathcal{D}, \mathcal{M})$ is an ordered set
- $ans_k(q, \mathcal{D}, \mathcal{M})$ are the top-k retrieved tuples in $ans(q, \mathcal{D}, \mathcal{M})$
  - possibly without computing the whole answer set

    (e.g. [Straccia, 2012, Straccia, 2014, Straccia and Madrid, 2012])

Scoring Function Examples

- Fuzzy set membership functions



(a) (b) (c) (d)

Figure: (a) triangular function $tri(a, b, c)$, (b) left shoulder function $ls(a, b)$, (c) right shoulder function $rs(a, b)$, and (d) fuzzy sets over centroids

- Conjunction $x \wedge y$: $\min(x, y), x \cdot y, \ldots$
- Disjunction $x \vee y$: $\max(x, y), x + y - x \cdot y, ,\ldots$
- Linear combination: $a \cdot x + (1 - a) \cdot y \, , \ldots$
- $\vdots$

Example (Soft Shopping Agent)

- User query:

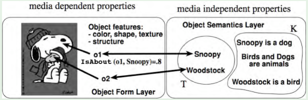
$$
\begin{aligned}
q(x,p,k,s) \quad \leftarrow \quad & HasPrice(x,p), s_p = ls(10000, 14000)(p), \\
& HasKM(x,k), s_k = ls(13000, 17000)(k), \\
& s = 0.7 \cdot s_p + 0.3 \cdot s_k
\end{aligned}
$$

ID	MODEL	PRICE	KM
455	MAZDA 3	12500	10000
34	ALFA 156	12000	15000
1812	FORD FOCUS	11000	16000
⋮	⋮	⋮	⋮

- Problem: All tuples of the database have a score
  - We cannot compute the score of all tuples, then rank them. Brute force approach not feasible for very large databases
- Top-k problem: Determine efficiently just the top-k ranked tuples, without evaluating the score of all tuples. E.g. top-3 tuples

ID	PRICE	SCORE
1812	11000	0.6
455	12500	0.56
34	12000	0.50

Example (Multimedia Information Retrieval [Meghini et al., 2001])



isAbout		
region	**obj**	**degr**
o1	snoopy	0.8
o2	woodstock	0.9

ISA		
obj1	**obj2**	**deg**
snoopy	Dog	1.0
woodstock	Bird	1.0
Dog	SmallAnimal	0.4
Bird	SmallAnimal	0.7
SmallAnimal	Animal	1.0
snoopy	SmallAnimal	0.4
woodstock	SmallAnimal	0.7
snoopy	Animal	0.4
woodstock	Animal	0.7

(ISA transitively closed w.r.t.$\cdot$)

- Query: “Find image regions about animals”

$$q(x, s) \quad \leftarrow \quad isAbout(x, y, s1), ISA(y, Animal, s2), s = s_1 \cdot s_2$$

- $ans_k(q, \mathcal{D}, \mathcal{M}) = [\langle o2, 0.63\rangle, \langle o1, 0.32\rangle, \ldots]$
- Top-k retrieval problem: $|ans(q, \mathcal{D}, \mathcal{M})|$ may be quite large

Example (TripAdvisor User Judgements about Hotels)

- Query: "Find hotels' that are good and cheap"
  - Definition of *good* and *cheap* may be subjective and context sensisitve

Example (Air quality in the province of Lucca)

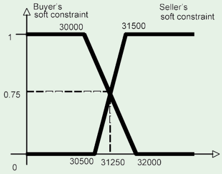
Sintesi dei dati rilevati dalle ore 0 alle ore 24 del giorno domenica 14/02/2010

Stazione		Tipo stazione	SO_2 µg/m³ (media su 24h)	NO_2 µg/m³ (max oraria)	CO mg/m³ (max oraria)	O_3 µg/m³ (max oraria)	PM_{10} µg/m³ (media su 24h)	Giudizio di qualità dell'aria
Lucca	P.za San Micheletto (RETE REGIONALE **)	urbana - traffico	1	75	---	---	56	Scadente
Lucca	V.le Carducci	urbana - traffico	2	---	2	---	75	Pessimo
Lucca	Carignano (RETE REGIONALE **)	rurale - fondo	---	---	---	87 (h.18*)	---	Buona
Viareggio	Largo Risorgimento	urbana - traffico	---	---	1,7	---	n.d.	Buona
Viareggio	Via Maroncelli (RETE REGIONALE **)	urbana - fondo	1	121	---	60 (h.17*)	45	Accettabile
Capannori	V. di Piaggia (RETE REGIONALE **)	urbana - fondo	---	79	2	---	59	Scadente
Porcari	V. Carrara (RETE REGIONALE **)	periferica - fondo	2	72	---	82 (h.16*)	63	Scadente

Giudizio di qualità	SO_2 µg/m³ (media su 24h)	NO_2 µg/m³ (max oraria)	CO mg/m³ (max oraria)	O_3 µg/m³ (max oraria)	PM_{10} µg/m³ (media su 24h)
Buona	0-50	0-50	0-2,5	0-120	0-25
Accettabile	51-125	51-200	2,6-15	121-180	26-50
Scadente	126-250	201-400	15,1-30	181-240	51-74
Pessima	>250	>400	>30	>240	>74

- Query: “Find locations with Bad air quality”
  - Problem 1: defined via theresholds
  - Problem 2: worst case among criteria adopted

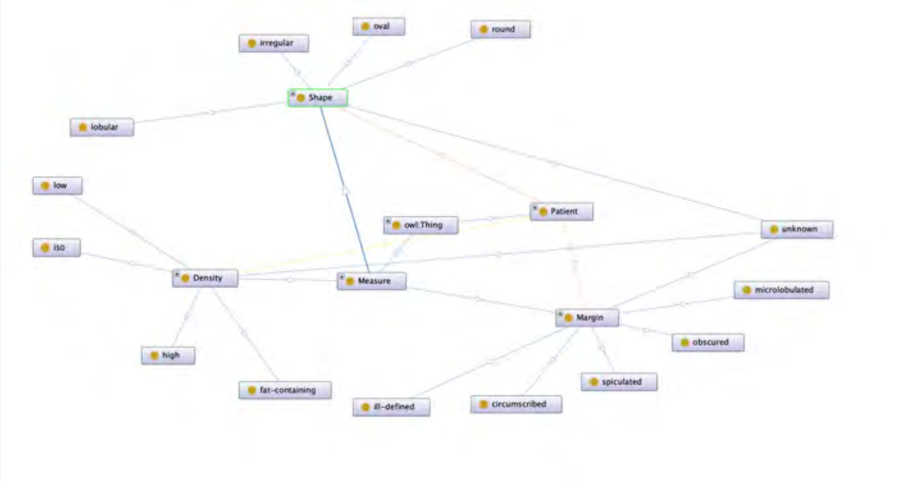
Example (Matchmaking [Ragone et al., 2009])



- A car seller sells an Audi TT for 31500 €, as from the catalog price
- A buyer is looking for a sports-car, but wants to to pay not more than around 30000 €
- Problem: with strict conditions there is no match
- More fine grained approach: to consider prices as soft constraints (fuzzy sets) (as usual in negotiation)
  - Seller prefers to sell above 31500 €, but can go down to 30500 €
  - Buyer prefers to spend less than 30000 €, but can go up to 32000 €
  - (Pareto optimal solution: Highest degree of matching is 0.75
  - The car may be sold at 31250 €.

Example (Ontology-based Machine Learning [Cardillo and Straccia, 2022, Cardillo et al., 2023])

Excerpt of a mammography ontology and data.



Example (Ontology-based Machine Learning [Cardillo and Straccia, 2022, Cardillo et al., 2023])

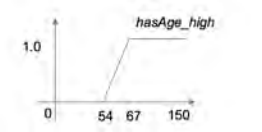
Excerpt of a mammography ontology and data.

Patient	hasDensity	hasShape	hasMargin	hasBiRads	hasAge	...
p0	low	lobular	spiculated	5	67	...
p10	high	irregular	spiculated	5	76	...
p102	–	irregular	ill-defined	4	58	...
p108	low	round	circumscribed	4	57	...
p109	–	irregular	ill-defined	5	33	...
p110	low	irregular	ill-defined	4	45	...
p111	low	irregular	ill-defined	5	71	...
...	...	...	...	...	...	...

p0, p10, p109, p111   positive examples
p102, p108, p110   negative examples

- What characterizes the patients with cancer ?

$$\begin{aligned} Cacer(x, s) \quad \leftarrow \quad & hasMargin(x, y1), ill-defined(y1, s1), \\ & hasShape(x, y2), irregular(y2, s2), \\ & hasAge(x, y3), hasAge\_high(y3, s3), s = f(s1, s2, s3) \end{aligned}$$



$rs(54, 67))$

CQ Answering over Distributed LIRs

Immediate solution: for a query

$$q(\mathbf{x}) \leftarrow R'_1(\mathbf{z}_1), \ldots, R'_l(\mathbf{z}_l)$$

- Determine the set $r(q, \mathcal{M})$ of rewritings of q

$$q(\mathbf{x}) \leftarrow S'_1(\mathbf{z}_1), \ldots, S'_l(\mathbf{z}_l)$$

  - where each $R'_i \in \mathcal{G}$ has been replaced with some $S'_j \in \mathcal{S}$
    - ★ $R_i(\mathbf{x}) \leftarrow S_j(\mathbf{x}) \in \mathcal{M}$
- Notice that there may be as many as

$$\prod_{R'_i} k_{R'_i}$$

  rewritings for q ($k_{R'_i}$ is the number of mapping rules w.r.t. R'_i)
- Note: number of rewritings can exponential viz. $O((\frac{|\mathcal{M}|}{k})^{|q|})$, with $k > 1$
- Response time may be exceedingly long in practice

CQ Answering over Distributed LIRs via Sampling

Goal: *Given a query q, how to select the top-s best rewritings $q' \in r(q, \mathcal{M})$ (with $s \ll |r(q, \mathcal{M})|$, e.g. $s = 10$ and $|r(q, \mathcal{M})| = 100$) such that a suitable, objective criteria is met.*

Using sampling: Recap,

1. Compute automatically a meaningful sample for each $D_i \in \mathcal{D}$ and store the data into the sampling database (Resource Sampling [Callan and Connell, 2001, Caverlee et al., 2006])
2. Using the sample database, determine which are the top-s best query rewritings $q' \in r(q, \mathcal{M})$ according to some criteria (Resource Selection [Si and Callan, 2004, Thomas and Hawking, 2009])
3. Submit the selected queries to the LIRs and merge the results (Ranked List Merging [Markov et al., 2013, Renda and Straccia, 2003, Shokouhi and Zobel, 2009, Yu et al., 2002])

Resource Sampling

To sample a local LIR: use Query-Based Sampling (QBS)

1. We start with a random query;
2. We submit the query to the LIR and store the retrieved tuples in the sample DB
3. We build a new query from the sample data
4. We iterate steps 2 and 3 until a stopping condition holds; such a condition expresses the fact that the sample changed less than a certain amount in the last iteration

We use *information entropy* on the sample data to

- Construct new queries
- Define the stopping criteria

Resource Sampling (cont.)

- Let $D \in \mathcal{D}$ be the database for which we want to build a sample
- Let $S \in \mathcal{S}$ be the related relational entity of arity p through which we query D
- We assume that the attributes' type of S are either 'number' or 'string' (bag of words)

Resource Sampling (cont.)

1. Initialise query dictionaries Q_i ($1 \leq i \leq p$) for each attribute of S
   - Q_i is a *multiset* of constants t that will be used to query the database D
   - If the i-th attribute of S is of type 'number' then Q_i will be a set of numbers
   - If the i-th attribute of S is of type 'string' then Q_i will be a set of words
2. Initialise the sample S_D of D as $S_D = \emptyset$
3. Choose an attribute (index) $1 \leq i \leq p$
4. Build a one-constant query $S(\mathbf{x}_1, t, \mathbf{x}_2)$ from $t \in Q_i$, where t has not already been selected
   - If such t does not exists, select t randomly from an external vocabulary
5. Submit the query $S(\mathbf{x}_1, t, \mathbf{x}_2)$ to the database D
6. Retrieve the top-k tuples from D in response to $S(\mathbf{x}_1, t, \mathbf{x}_2)$
   - i.e. determine $ans_k(S(\mathbf{x}_1, t, \mathbf{x}_2), D)$
7. Update the sample S_D with the retrieved tuples: i.e.

$$S_D := S_D \cup \; ans_k(S(\mathbf{x}_1, t, \mathbf{x}_2), D) \; .$$

8. Update the query dictionary Q_i ($1 \leq i \leq p$) with the relative constants in the retrieved tuples, i.e.

$$Q_i := Q_i \cup \bigcup_{\langle t_1, \ldots, t_i, \ldots, t_p \rangle \in ans_k(S(\mathbf{x}_1, t, \mathbf{x}_2), D)} \{t_i\}$$

   - Note: If t_i is a 'string' (bag of words) then we add all words in t_i, that are not stop words to Q_i
9. Goto Step 3, until a stopping condition is met

Resource Sampling (cont.)

Critical factors

- Step 1. Choice of the query dictionary Q_i
- Step 3 and 4. The query selection algorithm
- Step 8. The stopping condition

Resource Sampling (cont.)

Step 1. t can randomly be selected, or taken from a controlled vocabulary, or extracted from the web page through which we access to D

Steps 3 and 4

- One option is to chose i and $t \in Q_i$ random
- Use *entropy* [Hankerson et al., 1998] instead:
  - Let x_i be an attribute of S (that is, x_i is our random variable)
  - The values x_i can take are the values $t \in Q_i$
  - Let $p_i(t)$ be the probability that t occurs in the i-th column of tuples in S_D
  - The *entropy* of x_i is defined as

$$H(x_i) = -\sum_{t \in Q_i} p_i(t) \log_2 p_i(t) \qquad (1)$$

  - For Step 3., choose attribute index i with maximal entropy, i.e.

$$i = \arg \max_{1 \leq i \leq p} H(x_i) \; .$$

  - For Step 4, select then $t \in Q_i$ for which $p_i(t) \log_2 p_i(t)$ is minimal, i.e.

$$t = \arg \min_{t \in Q_i} p_i(t) \log_2 p_i(t)$$

  - Rationale: hope to reduce the entropy of x_i and t at the next round

Resource Sampling (cont.)

Step 8.

- Stopping criteria: based on joint entropy
- The joint entropy is

$$H(x_1, \ldots, x_p) = -\sum_{t_1 \in Q_1} \cdots \sum_{t_p \in Q_p} p(t_1, \ldots, t_p) \log_2 p(t_1, \ldots, t_p) \ ,$$

  where $p(t_1, \ldots, t_p)$ is the probability that the tuple $\langle t_1, \ldots, t_p \rangle$ occurs in $Q_1 \times \ldots \times Q_p$
- Note that in general

$$\max(H(x_1), \ldots, H(x_p)) \leq H(x_1, \ldots, x_p) \leq \sum_{1 \leq i \leq p} H(x_i) \ .$$

- Use estimate of joint entropy under probabilistic independence: i.e. $p(t_1, \ldots, t_n) = \prod_i p_i(t_i)$. So,

$$H(x_1, \ldots, x_p) = \sum_{1 \leq i \leq p} H(x_i) \qquad (2)$$

- Stopping criteria 1: stop when for $\epsilon \in [0, \sum_{1 \leq i \leq p} \log_2 |Q_i|]$

$$\sum_{1 \leq i \leq p} H(x_i) \leq \epsilon$$

- Stopping criteria 2: stop if joint entropy does not change too much, i.e. stop if

$$| \ H_{j+1}(x_1, \ldots, x_p) - H_j(x_1, \ldots, x_p) \ | \leq \delta \qquad (3)$$

  where $H_j(x_1, \ldots, x_p)$ joint entropy after the j-th loop of Steps 3. - 8. and $\delta \geq 0$.

Resource Selection

To select a query rewriting:

1. Let q be the CQ over the global schema
2. Let q_i be a rewriting over local DBs
3. Using the sample DBs, determine a score $s(q_i \mid q)$
   - The score estimates the 'goodness' of query q_i w.r.t. q in retrieving answers to q
4. Rank the rewritings in decreasing order of the score $s(q_i \mid q)$ and select only the top-s (with $s \ll |r(q, \mathcal{M})|$, e.g. $s = 10$) among them to be submitted to the real databases in $\mathcal{D}$

Score $s(q_i \mid q)$: adaption of the *ReDDE.top* method [Arguello et al., 2009]

- ReDDE.top is among the most effective for textual DIR (resembles somewhat kNN-classifiers)
- The score score $s(q_i \mid q)$ is defined as

$$s(q_i \mid q) = \frac{C^{q_i}}{C_{\max} \cdot R^{q_i}} \cdot \sum_{\langle \mathbf{t}, s\rangle \in ans_h(q, \mathcal{D}_S, \mathcal{M})} s \cdot I(\langle \mathbf{t}, s\rangle, q_i)$$

- R^{q_i} is the sample DB size of rewriting q_i
- C^{q_i} is the estimated DB size of rewriting q_i
- $C_{\max}$ is the maximum among all the C^{q_i}
- D_S sample database of S
- $I(\langle \mathbf{t}, s\rangle, q_i) = 1$ if $\langle \mathbf{t}, s\rangle \in ans(q_i, \mathcal{D}_S, \mathcal{M})$, else 0

Resource Selection (cont.). Estimating DB Size

- Estimatating the size of a source database $D \in \mathcal{D}$
  - *Sampling-resempling* method (e.g.see [Shokouhi and Si, 2011])
  - Also known as *mark-recapture* methods in the context in ecology to estimate the population size of particular species of animal in a region [Sutherland, 2006]
  - Standard mark-recapture technique: given number of animals is captured, marked, and released. After a suitable time has elapsed, a second set is captured. By inspecting the intersection of the two sets, the population size can be estimated
  - We adapt here the so-called the *Schumacher-Eschmeyer Method* (oldest methods used in ecology for estimating population size)

Resource Selection. Estimating DB Size (cont.)

- Let T be the number of applications of the sampling algorithm to a database $D \in \mathcal{D}$
- After applying the sampling algorithm T times from scratch we obtain T samples $S_D^1, \ldots, S_D^{|T|}$ of the database D
- Let K_i be the total number of tuples in the i-th sample of D, i.e. $K_i = |S_D^i|$
- Let R_i be the number of tuples in S_D^i that have been found in a previous run, i.e. $R_1 = 0$ and for $2 \leq i \leq T$

$$R_i = |\{\mathbf{t} \mid \mathbf{t} \in S_D^i \cap (\bigcup_{1 \leq j \leq i-1} S_D^j)\}|$$

  Note: R_i is the number of recaptured tuples
- Let M_i be is the number of tuples gathered so far, prior to the most recent sample, i.e. $M_1 = 0$ and for $2 \leq i \leq T$

$$M_i = |\bigcup_{1 \leq j \leq i-1} S_D^j| = \sum_{1 \leq j \leq i-1} (K_j - R_j)$$

- An estimate $\hat{N}_D$ of the number N_D of tuples in D is determined by

$$\hat{N}_D = \frac{\sum_{i=1}^T K_i M_i^2}{\sum_{i=1}^T R_i M_i}$$

- Note: T may not be known a priori, but a possible way to stop the iterations is when the estimate $\hat{N}_D^T$ does not significantly change w.r.t. the estimate $\hat{N}_D^{T+1}$

Resource Selection (cont). Parameters

- Given a CQ q of the form $q(\mathbf{x}) \leftarrow R'_1(\mathbf{z}_1), \ldots, R'_k(\mathbf{z}_k)$, where $R'_i \in \mathcal{G}$
- A rewriting $q_i \in r(q, \mathcal{M})$ of the form $q(\mathbf{x}) \leftarrow S'_1(\mathbf{z}_1), \ldots, S'_k(\mathbf{z}_k)$, where $S'_i \in \mathcal{S}$
- Let $\{D_1, \ldots, D_r\}$ the databases the relations $S'_1, \ldots, S'_k$ access to $(1 \leq r \leq k)$
- Let S_{D_j} be the sample database of D_j
- Let $\hat{N}_{D_j}$ is the estimated size of database D_j
- Then R^{q_i} is defined as

$$R^{q_i} = \sum_{j=1}^{r} |S_{D_j}|$$

- Then C^{q_i} is defined as

$$C^{q_i} = \sum_{j=1}^{r_q} \hat{N}_{D_j}$$

Remark. Dealing with Numerical Attributes

- There is a problem if a query involves some constraints on numerical attributes, such as 'the price is 28 euro'
- Hence, a sample database S_D may *not* contain '28' and, thus, $ans_h(q, \mathcal{D}_S, \mathcal{M})$ may be empty
  - As a consequence, the score $s(q_i \mid q)$ turns out to be 0
- Of course, the fact that that value does not occur in the sample database does not mean that the value does not occur in the real local database from which the sample has been drawn
- To mitigate such an effect, one may rely on *soft constraints*, inspired by *fuzzy set theory* [Klir and Yuan, 1995, Zadeh, 1965]
- Intuitively, in place of a hard constraints such as 'the price is 28 euro', we relax this constraint to a soft constraint of the form 'the price is about 28 euro', where '*about 28*' is a fuzzy set with a triangular membership function centered in 28, e.g. $tri(24, 28, 32)$
- Formally, for hard constraints $(x \geq n)$, $(x \leq n)$ and $(x = n)$, occurring in a CQ ($\alpha > 0$)

  Case $(x \geq n)$ replace it with scoring atom $s := rs(n - \alpha, n)(x)$
  Case $(x \leq n)$ replace it with scoring atom $s := ls(n, n + \alpha)(x)$
  Case $(x = n)$ replace it with scoring atom $s := tri(n - \alpha, n, n + \alpha)(x)$
- In case of multiple hard constraints $c_1, \ldots, c_k$ occur in a CQ, each of which is replaced with the function $f_i(x_i)$, as indicated above, then all of them may be replaced with the scoring atom

$$s := f(f_1(x_1), \ldots, f_k(x_k)) \tag{4}$$

where f is a suitable scoring function

Example (Find a house)

As illustrative example, consider the case in which the query is

'Find a house whose price is less than or equal to 150000 euro that is 80 square meters large at minimum.'

We may encode such a request as the CQ

$$q(x, x_1, x_2) \leftarrow \texttt{House}(x), \texttt{hasPrice}(x, x_1), \texttt{hasSqm}(x, x_2), (x_1 \leq 150000), (x_2 \geq 80)$$

Then the CQ above may be relaxed, according to our transformation, to the form ($\alpha_i > 0$)

$$q(x, x_1, x_2, s) \leftarrow \texttt{House}(x), \texttt{hasPrice}(x, x_1), \texttt{hasSqm}(x, x_2), s := ls(150000, 150000 + \alpha_1)(x_1) \cdot rs(80 - \alpha_2, 80)(x_2)$$

Note that the tuples of the answer set are now scored in decreasing order of satisfaction of the original hard constraints. The score decreases the 'more' the hard constraints are violated.

Remark. Dealing with String Valued Attributes

- Like for numerical hard constraints, in case textual hard constraints occur in a CQ, we may not find a match in the sample database
- A simple way to address the problem is to replace a textual hard constraint with a soft constraint by means of a text similarity-based scoring atom
- Specifically, given a query

$$q(\mathbf{x}) \leftarrow \exists \varphi(\mathbf{z}_1, t, \mathbf{z}_2)$$

  where t is a textual hard constraint on some attribute
- Relax the query with

$$q(\mathbf{x}, s) \leftarrow \exists \varphi(\mathbf{z}_1, y, \mathbf{z}_2), s := sim(y, t)$$

  where $sim(y, t)$ computes the degree of similarity between the text t and the textual value y occurring in a tuple in the sample database
- The case of multiple hard constraints is addressed as for the numerical case

Ranked List Merging

To merge ranked lists:

1. Given the selected top-s queries $q_1, \ldots, q_s$
2. Assume each query returns a ranked list $\ell_1, \ldots, \ell_s$ of tuples
   - $\ell_i = \{\langle \mathbf{t}^i_1, s^i_1 \rangle, \ldots, \langle \mathbf{t}^i_{|l_i|}, s^i_{|l_i|} \rangle\}$
   - If no score is provided, score is determined by the rank of the tuple in some way
   - E.g. $s = (r_{\max} - r + 1)/r_{\max}$, where $r_{\max}$ is the number of returned tuples in a ranked list and r is the rank of tuple $\mathbf{t}$ in that list
3. Merge them by build a unique list from which we select the top-k tuples only

Ranked List Merging (cont.)

Merging continued:

- We use the so-called minmax score normalisation method [Markov et al., 2013, Renda and Straccia, 2002]
- Let $V = \{v_1, \ldots, v_t\}$ be a set of t score values
- Let $v_{\min}$ ($v_{\max}$) be the minimum (maximum) among the $v_i \in V$
- The minimax normalisation of a score $v \in V$ w.r.t. V, denoted $v^V_{\min\max} \in [0, 1]$, is defined as

$$v^V_{\min\max} = \frac{v - v_{\min}}{v_{\max} - v_{\min}}$$

- Now, consider top-s query rewritings $Q = \{q_1, \ldots, q_s\}$ of query q
- Let $R = \{s(q_1 \mid q), \ldots, s(q_s \mid q)\}$ be the score values of $q_i \in Q$
- For query $q_i \in Q$, consider
  - Its ranked list of answers $\ell_i = \{\langle \mathbf{t}^i_1, s^i_1\rangle, \ldots, \langle \mathbf{t}^i_{|l_i|}, s^i_{|l_i|}\rangle\}$
  - The set of score values $S = \{s^i_1, \ldots, s^i_{|l_i|}\}$
  - For $\langle \mathbf{t}, s\rangle \in \ell_i$, we normalise s as follows: the normalised score $s_{\mathrm{norm}} \in [0, 1]$ of $\mathbf{t}$ is

$$s_{\mathrm{norm}} = s^R_{\min\max} \cdot s^S_{\min\max}$$

- Finally, we take the union of the ranked list in which all tuple scores' have been normalised

$$\ell_{\mathrm{norm}} = \{\langle \mathbf{t}, s_{\mathrm{norm}}\rangle \mid \langle \mathbf{t}, s\rangle \in \bigcup_{i=1}^{s} \ell_i\}$$

  (if a tuple **t** occurs in more than one ranked list, take **t**'s highest normalised score)

- Return the top-k tuples in ℓ_{norm} (order ℓ_{norm} and then select the top-k)

Ranked List Merging (cont.)

Return the top-k tuples in $\ell_{\mathtt{norm}}$ continued:

- Size of $\ell_{\mathtt{norm}}$ may still become large: but, we may avoid to order whole $\ell_{\mathtt{norm}}$
- Improvement: use the Disjunctive Threshold Algorithm (DTA) [Straccia, 2006]
  - Consider top-s query rewritings $Q = \{q_1, \ldots, q_s\}$ of query q
  - For query $q_i \in Q$, consider its ranked list of answers $\ell_i = \{\langle \mathbf{t}^i_1, s^i_1 \rangle, \ldots, \langle \mathbf{t}^i_{|l_i|}, s^i_{|l_i|} \rangle\}$
  - Now process each list ℓ_i in alternating fashion, and top-down w.r.t. score values
    - For each $\langle \mathbf{t}, s \rangle$ seen, normalise the score s
    - If s is one of the k highest we have seen, then add $\langle \mathbf{t}, s \rangle$ to $\ell_{\mathtt{norm}}$ (ties are broken arbitrarily)
    - For each ℓ_i , let v_i be the score value of the last tuple seen in this set
    - Define the threshold

$$\theta = \max(v_1, \ldots, v_s)$$

    - As soon as at least k tuples have been seen whose score is at least equal to θ, then halt
    - Indeed, any successive retrieved tuple will have score $\leq \theta$
    - Return the top-k tuples in $\ell_{\mathtt{norm}}$

Example (DTA)

Suppose we are interested in retrieving the top-3 answers of

$$\begin{array}{lcl} \ell_1 & = & [\langle a, 1.0\rangle, \langle d, 0.7\rangle, \langle e, 0.6\rangle] \\ \ell_2 & = & [\langle b, 0.9\rangle, \langle c, 0.8\rangle, \langle f, 0.5\rangle] \end{array}$$

- We process alternatively ℓ_1 then ℓ_2 in decreasing order of the score
- The table below summaries the execution of the DTA algorithm

Step	tuple	s_1	s_2	θ	ranked list ℓ_{norm}
1	$\langle a, 1.0\rangle$	1.0	-	1.0	$\langle a, 1.0\rangle$
2	$\langle b, 0.9\rangle$	1.0	0.9	1.0	$\langle a, 1.0\rangle, \langle b, 0.9\rangle$
3	$\langle d, 0.7\rangle$	0.7	0.9	0.9	$\langle a, 1.0\rangle, \langle b, 0.9\rangle, \langle d, 0.7\rangle$
4	$\langle c, 0.8\rangle$	0.8	0.7	0.8	$\langle a, 1.0\rangle, \langle b, 0.9\rangle, \langle d, 0.7\rangle, \langle c, 0.8\rangle$

- At Step 4. we stop as the ranked list already contains three tuples above the threshold $\theta = 0.8$
- So, the final output is

$$\text{top-}k(\ell_{\mathrm{norm}}) = [\langle a, 1.0\rangle, \langle b, 0.9\rangle, \langle c, 0.8\rangle]$$

- Note that not all tuples have been processed

Ontology Based Data Access (OBDA)

(The Case of Structured Mediator Schema)

Ontology Based Data Access (OBDA)

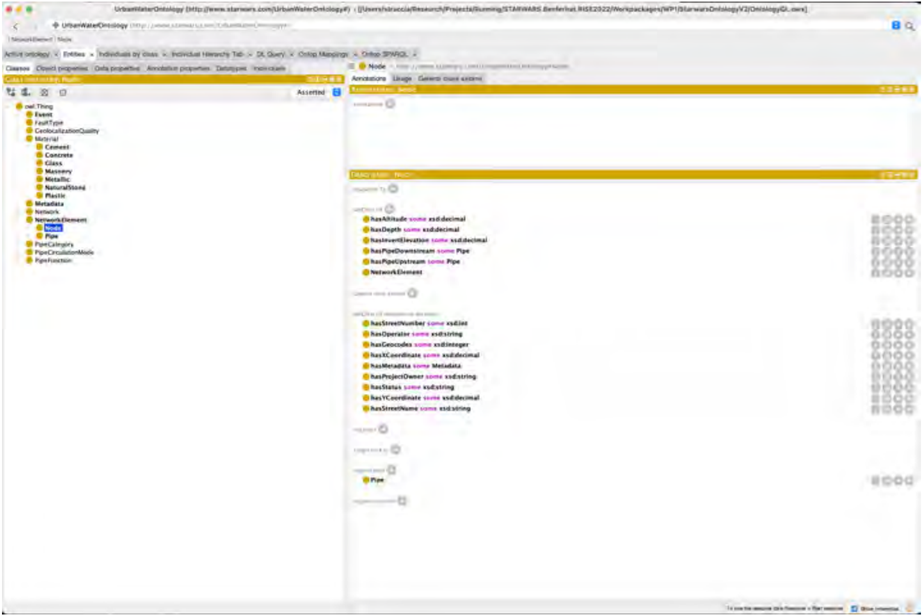
- So far, the mediator schema is based on schema mappings only, i.e. rules of the form

$$R(\mathbf{x}) \leftarrow S(\mathbf{x})$$

- They tell how to materilise the global relation R by acccessing the local relation S
- For instance, `hasPipeMaterial`(x,y) $\leftarrow$ `(SELECT idcana, materiau_l FROM starwars.wastewatercanalisation)`(x,y)
  - `hasPipeMaterial` is a global relation
  - `wastewatercanalisation` is a relational table the local database `starwars`

Ontology Based Data Access (OBDA)

- The global mediator schema is an ontology
- An ontology is a description of a set of conceptual entities of a domain that shows their properties and the relations between them
- It ensures a common understanding of information and makes explicit domain assumptions
  - Allows organizations to make better sense of their data
- Ontologies do not only represent sharable and reusable knowledge, but can also used to infer new knowledge about a domain
- To enable such a representation, we need to formally specify components such as individuals, classes, attributes and relations as well as restrictions, rules and axioms



owl:Thing
Event
FaultType
GeolocalizationQuality
Material
Cement
Concrete
Glass
Masonry
Metallic
NaturalStone
Plastic
Metadata
Network
NetworkElement
Node
Pipe
PipeCategory
PipeCirculationMode
PipeFunction
Node
Annotations
Usage
hasAltitude some xsd:decimal
hasDepth some xsd:decimal
hasInvertElevation some xsd:decimal
hasPipeDownstream some Pipe
hasPipeUpstream some Pipe
NetworkElement
hasStreetNumber some xsd:int
hasOperator some xsd:string
hasGeocodes some xsd:integer
hasXCoordinate some xsd:decimal
hasMetadata some Metadata
hasProjectOwner some xsd:string
hasStatus some xsd:string
hasYCoordinate some xsd:decimal
hasStreetName some xsd:string
Pipe

The Semantic Web Family of Languages

- Semantic Web family of languages widely used to specify ontologies
- Wide variety of languages
  - RDFS: *Triple language*, -*Resource Description Framework*
    - The logical counterpart is ρdf
  - RIF: *Rule language*, -*Rule Interchange Format*,
    - Relate to the Logic Programming (LP) paradigm
  - OWL 2: *Conceptual language*, -*Ontology Web Language*
    - Relate to Description Logics (DLs)

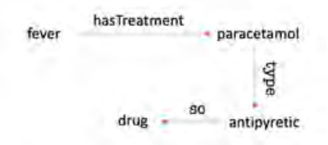
The case of RDFS

Resource Description Framework Schema (RDFS)

- RDFS: W3C standard and popular logic for KR
- Statements
  - Triples of the form (s, p, o)
  - Informally, binary predicate $p(s, o)$

(fever, hasTreatment, paracetamol)

  - Special predicates: typing and specialisations, etc.
    (paracetamol, type, antipyretic)
    (antipyretic, sc, drug)



- *Knowledge Graphs* may be seen as a special case

ρdf

- The logic ρdf
  - A minimal, but significant RDFS fragment
  - Covers all essential features of RDFS
- ρdf: defined on subset of the RDFS vocabulary:

$$\rho\text{df} = \{\text{sp}, \text{sc}, \text{type}, \text{dom}, \text{range}\}$$

Informally,

- (p, sp, q)
  - p is a sub property of property q
- (c, sc, d)
  - c is a sub class of class d
- (a, type, b)
  - a is of type b
- (p, dom, c)
  - domain of property p is c
- $(p, \textit{range}, c)$
  - range of property p is c

ρdf Syntax

- Alphabets:
  - **U** (*RDF URI references*)
  - **B** (*Blank nodes*)
  - **L** (*Literals*)
- Terms: **UBL** $(a, b, \ldots, w)$
- Variables: **B** (x, y, z)
- Triple:

$$(s, p, o) \in \mathbf{UBL} \times \mathbf{U} \times \mathbf{UBL}$$

  - $s, o \notin \rho$df
  - s subject, p predicate, o object
- Note: e.g. (type, sp, p) not allowed

- Graph/Knowledge Base G: set of triples τ
- Ground graph: no blank nodes, i.e. variables
- Map (or *variable assignment*):
  - μ : **UBL** $\to$ **UBL**, $\mu(t) = t$, for all $t \in$ **UL**

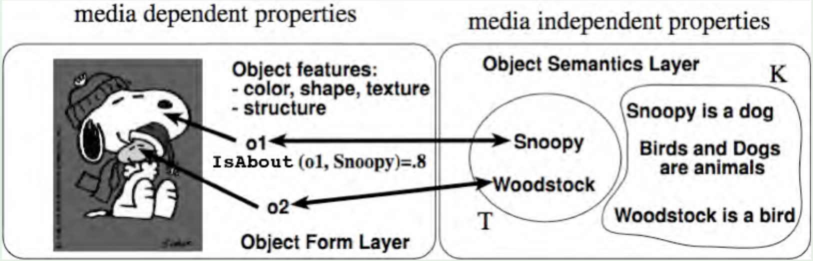
$$\mu(G) = \{(\mu(s), \mu(p), \mu(o)) \mid (s, p, o) \in G\}$$

  - Map μ from G_1 to G_2, and write $\mu : G_1 \to G_2$
    - if μ is such that $\mu(G_1) \subseteq G_2$

Example

$$
\begin{aligned}
G = \{&(\texttt{paracetamol}, \textsf{type}, \texttt{antipyretic}),\\
&(\texttt{antipyretic}, \textsf{sc}, \texttt{drugTreatment}),\\
&(\texttt{morphine}, \textsf{type}, \texttt{opioid}), (\texttt{opioid}, \textsf{sc}, \texttt{drugTreatment}),\\
&(\texttt{drugTreatment}, \textsf{sc}, \texttt{treatment}),\\
&(\texttt{brainTumour}, \textsf{type}, \texttt{tumour}),\\
&(\texttt{hasDrugTreatment}, \textsf{sp}, \texttt{hasTreatment}),\\
&(\texttt{hasTreatment}, \textsf{dom}, \texttt{illness}),\\
&(\texttt{hasTreatment}, \textsf{range}, \texttt{treatment}),\\
&(\texttt{hasDrugTreatment}, \textsf{range}, \texttt{drugTreatment}),\\
&(\texttt{fever}, \texttt{hasDrugTreatment}, \texttt{paracetamol})\\
&(\texttt{brainTumour}, \texttt{hasDrugTreatment}, \texttt{morphine})\ \}
\end{aligned}
$$

Example



$$G = \left\{ \begin{array}{ll} (\mathtt{o1}, \mathtt{IsAbout}, \mathtt{snoopy}) & (\mathtt{o2}, \mathtt{IsAbout}, \mathtt{woodstock}) \\ (\mathtt{snoopy}, \mathsf{type}, \mathtt{dog}) & (\mathtt{woodstock}, \mathsf{type}, \mathtt{bird}) \\ (\mathtt{dog}, \mathsf{sc}, \mathtt{animal}) & (\mathtt{bird}, \mathsf{sc}, \mathtt{animal}) \end{array} \right\}$$

ρdf (Intentional) Semantics

ρdf interpretation:

$$\mathcal{I} = \langle \Delta_\mathsf{R}, \Delta_\mathsf{P}, \Delta_\mathsf{C}, \Delta_\mathsf{L}, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^\mathcal{I} \rangle ,$$

1. Δ_R are the resources
2. Δ_P are property names
3. $\Delta_\mathsf{C} \subseteq \Delta_\mathsf{R}$ are the classes
4. $\Delta_\mathsf{L} \subseteq \Delta_\mathsf{R}$ are the literal values and contains all the literals in $\mathbf{L} \cap V$
5. $P[\![\cdot]\!]$ is a function $P[\![\cdot]\!]\colon \Delta_\mathsf{P} \to 2^{\Delta_\mathsf{R} \times \Delta_\mathsf{R}}$
6. $C[\![\cdot]\!]$ is a function $C[\![\cdot]\!]\colon \Delta_\mathsf{C} \to 2^{\Delta_\mathsf{R}}$
7. $\cdot^\mathcal{I}$ maps each $t \in \mathbf{UL} \cap V$ into a value $t^\mathcal{I} \in \Delta_\mathsf{R} \cup \Delta_\mathsf{P}$, where $\cdot^\mathcal{I}$ is the identity for literals; and
8. $\cdot^\mathcal{I}$ maps each variable $x \in \mathbf{B}$ into a value $x^\mathcal{I} \in \Delta_\mathsf{R}$

ρdf model/entailment

$\mathcal{I} \models G$ if and only if $\mathcal{I}$ satisfies conditions

Simple:

1. for each $(s, p, o) \in G$, $p^{\mathcal{I}} \in \Delta_{\mathrm{P}}$ and $(s^{\mathcal{I}}, o^{\mathcal{I}}) \in P[\![p^{\mathcal{I}}]\!]$

Subproperty:

1. $P[\![\mathrm{sp}^{\mathcal{I}}]\!]$ is transitive over Δ_{P}
2. if $(p, q) \in P[\![\mathrm{sp}^{\mathcal{I}}]\!]$ then $p, q \in \Delta_{\mathrm{P}}$ and $P[\![p]\!] \subseteq P[\![q]\!]$

Subclass:

1. $P[\![\mathrm{sc}^{\mathcal{I}}]\!]$ is transitive over Δ_{C}
2. if $(c, d) \in P[\![\mathrm{sc}^{\mathcal{I}}]\!]$ then $c, d \in \Delta_{\mathrm{C}}$ and $C[\![c]\!] \subseteq C[\![d]\!]$

Typing I:

1. $x \in C[\![c]\!]$ if and only if $(x, c) \in P[\![\mathrm{type}^{\mathcal{I}}]\!]$;
2. if $(p, c) \in P[\![\mathrm{dom}^{\mathcal{I}}]\!]$ and $(x, y) \in P[\![p]\!]$ then $x \in C[\![c]\!]$
3. if $(p, c) \in P[\![\mathrm{range}^{\mathcal{I}}]\!]$ and $(x, y) \in P[\![p]\!]$ then $y \in C[\![c]\!]$

Typing II:

1. for each $\mathrm{e} \in \rho\mathrm{df}$, $\mathrm{e}^{\mathcal{I}} \in \Delta_{\mathrm{P}}$;
2. if $(p, c) \in P[\![\mathrm{dom}^{\mathcal{I}}]\!]$ then $p \in \Delta_{\mathrm{P}}$ and $c \in \Delta_{\mathrm{C}}$
3. if $(p, c) \in P[\![\mathrm{range}^{\mathcal{I}}]\!]$ then $p \in \Delta_{\mathrm{P}}$ and $c \in \Delta_{\mathrm{C}}$
4. if $(x, c) \in P[\![\mathrm{type}^{\mathcal{I}}]\!]$ then $c \in \Delta_{\mathrm{C}}$.

$G \models H$ if and only if every model of G is also a model of H

Deductive System for ρdf

$G \vdash H$

1. Simple:

   $(a) \quad \frac{G}{G'}$ for a map $\mu : G' \to G \qquad (b) \quad \frac{G}{G'}$ for $G' \subseteq G$

2. Subproperty:

   $(a) \quad \frac{(A,\mathrm{sp},B),(B,\mathrm{sp},C)}{(A,\mathrm{sp},C)} \qquad (b) \quad \frac{(D,\mathrm{sp},E),(X,D,Y)}{(X,E,Y)}$

3. Subclass:

   $(a) \quad \frac{(A,\mathrm{sc},B),(B,\mathrm{sc},C)}{(A,\mathrm{sc},C)} \qquad (b) \quad \frac{(A,\mathrm{sc},B),(X,\mathrm{type},A)}{(X,\mathrm{type},B)}$

4. Typing:

   $(a) \quad \frac{(D,\mathrm{dom},B),(X,D,Y)}{(X,\mathrm{type},B)} \qquad (b) \quad \frac{(D,\mathrm{range},B),(X,D,Y)}{(Y,\mathrm{type},B)}$

5. Implicit Typing:

   $(a) \quad \frac{(A,\mathrm{dom},B),(D,\mathrm{sp},A),(X,D,Y)}{(X,\mathrm{type},B)}$

   $(b) \quad \frac{(A,\mathrm{range},B),(D,\mathrm{sp},A),(X,D,Y)}{(Y,\mathrm{type},B)}$

Closure of G:

$$\mathrm{Cl}(G) = \{\tau \mid G \vdash^* \tau\}$$

where $\vdash^*$ is as $\vdash$ except rule $(1a)$ is excluded

Some ρdf Properties

1. Every ρdf-graph is satisfiable (i.e. has canonical model)
   - RDFS is paraconsistent
2. $G \vdash H$ if and only if $G \models H$
3. The closure of G is unique and $|\mathrm{Cl}(G)| \in \Theta(|G|^2)$
4. Deciding $G \models H$ is an NP-complete problem
5. If H is ground, then $G \models H$ if and only if $H \subseteq \mathrm{Cl}(G)$
6. There is no triple τ such that $\emptyset \models \tau$
7. RDFS can represent only positive statements, e.g. “Paracetamol is a treatment for fever”
   - RDFS with negative statements, see [Straccia and Casini, 2022]

     “Opioids and antipyretics are disjoint classes"
     “Radio therapies are non drug treatments"
     “Ebola has no treatment"

   - Note: “Paracetamol is not a treatment for Ebola"
     - Can not be inferred (under OWA)
     - Can be under CWA, but CWA is not acceptable for RDFS

RDFS CQ Answering

- Conjunctive query: is a Datalog-like rule of the form

$$q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\tau_1, \ldots, \tau_n$$

where $\tau_1, \ldots, \tau_n$ are triples in which variables in **x** and **y** may occur (we may omit $\exists \mathbf{y}$)

- The answer set of CQ q is

$$ans(q, G) = \{\mathbf{t} \mid G \cup \{q\} \models q(\mathbf{t})\}$$

- Example:

$$q(x, y) \leftarrow (x, \texttt{creates}, y), (x, \text{type}, \texttt{Flemish}), (x, \texttt{paints}, y), (y, \texttt{exhibited}, \texttt{Uffizi})$$

"retrieve all the artifacts x created by Flemish artists y, being exhibited at Uffizi Gallery"

Standalone RDFS CQ Answering

- A simple query answering procedure for a (local) RDFS graph is the following:
  - Compute the closure of a graph off-line
  - Store the RDF triples into a Relational database
  - Translate the query into a SQL statement
  - Execute the SQL statement over the relational database
- In practice, some care should be in place due to the large size of data: $\geq 10^9$ triples
- To date, several systems exists

RDFS CQ Answering over Distributed LIRs

- An RDFS Global Schema $\mathcal{G}$ is made of RDFS triples of the form

$$(p, \mathsf{sp}, q), (c, \mathsf{sc}, d), (p, \mathsf{dom}, c), (p, range, c)$$

- The set $\mathcal{M}$ of RDFS Mapping Rules contains mappings of the form

$$(x, p, y) \leftarrow S(x, p, y)$$

  where $S(x, p, y)$ is a relation over a LIR and $p \notin \{\mathsf{sp}, \mathsf{sc}, \mathsf{range}, \mathsf{dom}\}$

- Conjunctive query: is a Datalog-like rule of the form

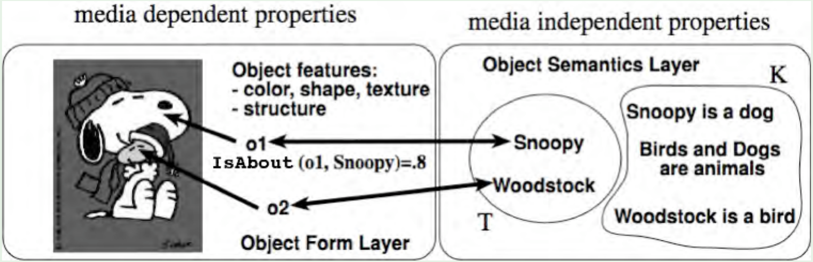
$$q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\tau_1, \ldots, \tau_n$$

  where $\tau_1, \ldots, \tau_n$ are triples in which variables in $\mathbf{x}$ and $\mathbf{y}$ may occur (we may omit $\exists \mathbf{y}$)

- The answer set of a CQ q is

$$ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M}) = \{\mathbf{t} \mid \mathcal{D} \cup \mathcal{G} \cup \mathcal{M} \cup \{q\} \models q(\mathbf{t})\}$$

Example



Global Schema $\mathcal{G}$: $\{(Dog, \text{sc}, Animal), (Bird, \text{sc}, Animal)\}$

Mapping Rules $\mathcal{M}$: $(r, isAbout, o) \leftarrow$ (`SELECT region, obj FROM imageClass`)(r, o)
$(i, \text{type}, c) \leftarrow$ (`SELECT obj, class FROM instances`)(i, c)

LIRs:

imageClass		
region	obj	degr
o1	snoopy	0.8
o2	woodstock	0.9

instances	
obj	class
snoopy	Dog
woodstock	Bird

Query: $q(x) \leftarrow (x, IsAbout, y), (y, \text{type}, Animal)$

$answer(q, \mathcal{D}, \mathcal{G}, \mathcal{M}) = \{o1, o2\}$

RDFS CQ Answering over Distributed LIRs (cont.)

- What if the results from the LIRs have a score?
- Mapping rules are of the form

$$\langle (x, p, y), s \rangle \leftarrow \langle S(x, p, y), s \rangle$$

  where now s is the score assigned to the triple (x, p, y) and $p \not\in \{\mathrm{sp}, \mathrm{sc}, \mathrm{range}, \mathrm{dom}\}$. If s omitted, then 1.0 is assumed
- Conjunctive query: extends previous RDFS query and is of the form

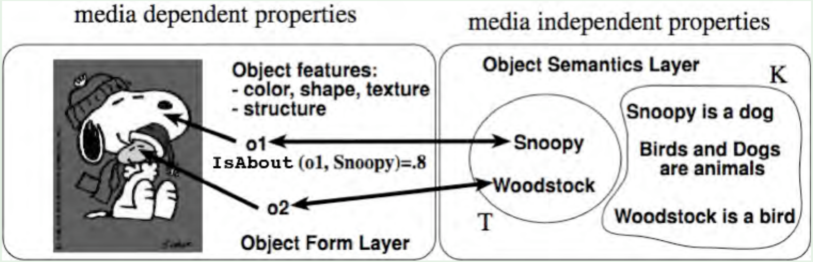
$$\langle q(\mathbf{x}), s \rangle \quad \leftarrow \quad \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y}), s = f(\mathbf{z})$$

  where
  - $\varphi(\mathbf{x}, \mathbf{y})$ is conjunction of $\langle \tau_i, s_i \rangle$
  - τ_i are triples involving literals and variables in $\mathbf{x}, \mathbf{y}$
  - $\mathbf{z}$ is a tuple of literals, or variables in $\mathbf{x}, \mathbf{y}$ or scores s_i
  - s_i is the score assigned to τ_i
  - the scoring variables s and s_i are distinct from those in $\mathbf{x}$ and $\mathbf{y}$ and s is distinct from each s_i
- The answer set of a CQ q is

$$ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M}) = \{\langle \mathbf{t}, s \rangle \mid \mathcal{D} \cup \mathcal{G} \cup \mathcal{M} \cup \{q\} \models q(\mathbf{t}, s)\} ,$$

  where each tuple has an unique score
- As for LIRs, $ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$ is an ordered set
- $ans_k(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$ are the top-k retrieved tuples in $ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$

Example



Global Schema $\mathcal{G}$: $\{(Dog, \text{sc}, Animal), (Bird, \text{sc}, Animal)\}$

Mapping Rules $\mathcal{M}$: $\langle(r, isAbout, o), d\rangle \leftarrow$ (`SELECT region, obj, degr FROM imageClass`)(r, o, d)
$(s, \text{type}, c) \leftarrow$ (`SELECT obj, class FROM instances`)(s, c)

LIRs:

imageClass		
region	obj	degr
o1	snoopy	0.8
o2	woodstock	0.9

instances	
obj	class
snoopy	Dog
woodstock	Bird

Query: $\langle q(x), s\rangle \leftarrow \langle(x, IsAbout, y), s1\rangle, (y, \text{type}, Animal), s = s1$

$answer(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$: $\{\langle o2, 0.9\rangle, \langle o1, 0.8\rangle\}$

RDFS CQ Answering over Distributed LIRs (cont.)

- How do we answer queries over a Global Schema?
- Apply Query Refomulation Algorithm
- By considering $\mathcal{G}$ only, the query q is *reformulated* into a set of conjunctive queries $r(q, \mathcal{G})$
- Informally, the basic idea is that the reformulation procedure closely resembles a top-down resolution procedure for logic programming, where, e.g. "schema triple" (c, sc, d) is seen as a logic programming rule of the form $d(x) \leftarrow c(x)$
- So, query

$$q(x, s) \leftarrow \langle q(x), s \rangle \leftarrow \langle (x, \mathit{IsAbout}, y), s1 \rangle, (y, \mathrm{type}, \mathit{Animal}), s = s1$$

  is rewritten as (the DCQ)

$$\begin{array}{lll} q(x, s) & \leftarrow & \langle q(x), s \rangle \leftarrow \langle (x, \mathit{IsAbout}, y), s1 \rangle, (y, \mathrm{type}, \mathit{Animal}), s = s1 \\ q(x, s) & \leftarrow & \langle q(x), s \rangle \leftarrow \langle (x, \mathit{IsAbout}, y), s1 \rangle, (y, \mathrm{type}, \mathit{Dog}), s = s1 \\ q(x, s) & \leftarrow & \langle q(x), s \rangle \leftarrow \langle (x, \mathit{IsAbout}, y), s1 \rangle, (y, \mathrm{type}, \mathit{Bird}), s = s1 \end{array}$$

- Exactly as it happens for top-down resolution methods in logic programming

RDFS CQ Answering over Distributed LIRs (cont.)

- Consider global schema $\mathcal{G}$, mapping rules $\mathcal{M}$ and CQ query q
- It suffices to provide a translation to *Logic Prgramming* (LP) and use a top-down algorithm for LPs
- We use a predicate `triple` to encode triples

$$
\begin{array}{rcl}
(x, p, y) & \mapsto & \texttt{triple}(x, p, y, 1.0) \\
\langle (x, p, y), s\rangle & \mapsto & \texttt{triple}(x, p, y, s)
\end{array}
$$

  where s is the score
- We need also to encode the semantics of the RDFS operators (see deduction rules for RDFS)
- For a suitable t-norm (function to interpret conjunction, e.g. minimum, product), let $RDFS_{rules}$ be

$$
\begin{array}{rcl}
\texttt{triple}(a, \text{sp}, c, s) & \leftarrow & \texttt{triple}(a, \text{sp}, b, s1), \texttt{triple}(b, \text{sp}, c, s2), s = s1 \otimes s2 \\
\texttt{triple}(x, e, y, s) & \leftarrow & \texttt{triple}(d, \text{sp}, e, s1), \texttt{triple}(x, d, y, s2), s = s1 \otimes s2 \\
\texttt{triple}(a, \text{sc}, c, s) & \leftarrow & \texttt{triple}(a, \text{sc}, b, s1), \texttt{triple}(b, \text{sc}, c, s2), s = s1 \otimes s2 \\
\texttt{triple}(x, \text{type}, b, s) & \leftarrow & \texttt{triple}(a, \text{sc}, b, s1), \texttt{triple}(x, \text{type}, a, s2), s = s1 \otimes s2 \\
\texttt{triple}(x, \text{type}, b, s) & \leftarrow & \texttt{triple}(d, \text{dom}, b, s1), \texttt{triple}(x, d, y, s2), s = s1 \otimes s2 \\
\texttt{triple}(y, \text{type}, b, s) & \leftarrow & \texttt{triple}(d, \text{range}, b, s1), \texttt{triple}(x, d, y, s2), s = s1 \otimes s2 \\
\texttt{triple}(x, \text{type}, b, s) & \leftarrow & \texttt{triple}(a, \text{dom}, b, s1), \texttt{triple}(d, \text{sp}, a, s2), \texttt{triple}(x, d, y, s3), \\
 & & s = s1 \otimes s2 \otimes s3 \\
\texttt{triple}(y, \text{type}, b, s) & \leftarrow & \texttt{triple}(a, \text{range}, b, s1), \texttt{triple}(d, \text{sp}, a, s2), \texttt{triple}(x, d, y, s3), \\
 & & s = s1 \otimes s2 \otimes s3
\end{array}
$$

RDFS CQ Answering over Distributed LIRs (cont.)

- A mapping rule

$$\langle (x, p, y), s\rangle \leftarrow \langle S(x, p, y), s\rangle$$

  is transformed into

$$\texttt{triple}(x, p, y, s) \leftarrow S(x, p, y, s)$$

- For instance,

$$\texttt{triple}(r, \mathit{isAbout}, o, d) \leftarrow (\texttt{SELECT region, obj, degr FROM imageClass})(r, o, d)$$

- A CQ

$$\langle q(\mathbf{x}), s\rangle \quad \leftarrow \quad \exists \mathbf{y}.\langle \tau_1, s_1\rangle, \ldots, \langle \tau_n, s_n\rangle, s = f(\mathbf{z})$$

  is transformed then in the obvious way into

$$q(\mathbf{x}, s) \quad \leftarrow \quad \exists \mathbf{y}.\texttt{triple}(\tau_1, s_1), \ldots, \texttt{triple}(\tau_1, s_n), s = f(\mathbf{z})$$

  where $\texttt{triple}(\tau_1, s_i)$ is the transformation of $\langle \tau_1, s_i\rangle$

Example (Multimedia Information Retrieval)

Global Schema $\mathcal{G}$: $\{\texttt{triple}(Dog, \text{sc}, Animal, 1.0), \texttt{triple}(Bird, \text{sc}, Animal, 1.0)\}$

Mapping Rules $\mathcal{M}$: $\texttt{triple}(r, isAbout, o, d) \leftarrow (\texttt{SELECT region, obj, degr FROM imageClass})(r, o, d)$
$\texttt{triple}(s, \text{type}, c, 1.0) \leftarrow (\texttt{SELECT obj, class FROM instances})(s, c)$

LIRs:

imageClass			instances	
region	**obj**	**degr**	**obj**	**class**
o1	snoopy	0.8	snoopy	Dog
o2	woodstock	0.9	woodstock	Bird

Query: $q(x, s) \leftarrow \texttt{triple}(x, IsAbout, y, s1), \texttt{triple}(y, \text{type}, Animal, 1.0), s = s1$

$r(q, \mathcal{G})$:

$$
\begin{array}{lcl}
q(x, s) & \leftarrow & \texttt{triple}(x, IsAbout, y, s1), \texttt{triple}(y, \text{type}, Animal, 1.0), s = s1 \\
q(x, s) & \leftarrow & \texttt{triple}(x, IsAbout, y, s1), \texttt{triple}(y, \text{type}, Dog, 1.0), s = s1 \\
q(x, s) & \leftarrow & \texttt{triple}(x, IsAbout, y, s1), \texttt{triple}(y, \text{type}, Bird, 1.0), s = s1
\end{array}
$$

$answer(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$: $\{\langle o2, 0.9\rangle, \langle o1, 0.8\rangle\}$

The case of OWL 2

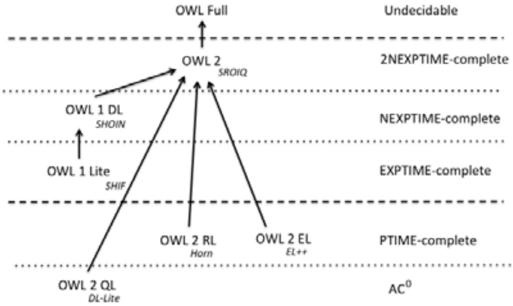
The Web Ontology Language OWL 2

- OWL 2 is a family of the object oriented languages

```
class      Person partial Human
           restriction (hasName someValuesFrom String)
           restriction (hasBirthPlace someValuesFrom Geoplace)
```

"The class Person *is a subclass of class* Human *and has two attributes:* hasName *having a string as value, and* hasBirthPlace *whose value is an instance of the class* Geoplace*".*

- Description Logics are the logics that stand behind OWL 2
- OWL languages differntiate in syntax and computational complexity of reasoning problems



OWL 2 Profiles

OWL 2 EL
- Useful for large size of properties and/or classes
- The EL acronym refers to the $\mathcal{EL}$ family of DLs

OWL 2 QL
- Useful for very large volumes of instance data
- Conjunctive query answering via query rewriting and SQL
- OWL 2 QL relates to the DL family *DL-Lite*

OWL 2 RL
- Useful for scalable reasoning without sacrificing too much expressive power
- OWL 2 RL maps to Datalog (an LP language)
- Computational complexity: same as for Datalog, polynomial in size of the data, EXPTIME w.r.t. size of knowledge base

Description Logics (DLs)

- Concept/Class: are unary predicates
- Role or attribute: binary predicates
- Taxonomy: Concept and role hierarchies can be expressed
- Individual: constants
- Operators: to build complex classes out from class names

- Basic ingredients:
  - $a{:}C$, meaning that individual a is an instance of concept/class C

$$\text{a:Person} \sqcap \exists\text{hasChild.Femal}$$

  - $(a, b){:}R$, meaning that the pair of individuals $\langle a, b\rangle$ is an instance of the property/role R

$$(\text{tom}, \text{mary})\text{:hasChild}$$

  - $C \sqsubseteq D$, meaning that the class C is a subclass of class D

$$\text{Father} \sqsubseteq \mathit{Male} \sqcap \exists\text{hasChild.Person}$$

The DL Family

- A given DL is defined by set of concept and role forming operators
- Basic language: $\mathcal{ALC}$ ($\mathcal{A}$ttributive $\mathcal{L}$anguage with $\mathcal{C}$omplement)

Syntax				Semantics	Example
C, D	$\rightarrow$	$\top$	|	$\top(x)$	
		$\bot$	|	$\bot(x)$	
		A	|	$A(x)$	*Human*
		$C \sqcap D$	|	$C(x) \wedge D(x)$	$Human \sqcap Male$
		$C \sqcup D$	|	$C(x) \vee D(x)$	$Nice \sqcup Rich$
		$\neg C$	|	$\neg C(x)$	$\neg Meat$
		$\exists R.C$	|	$\exists y.R(x,y) \wedge C(y)$	$\exists has\_child.Blond$
		$\forall R.C$		$\forall y.R(x,y) \Rightarrow C(y)$	$\forall has\_child.Human$
$C \sqsubseteq D$				$\forall x.C(x) \Rightarrow D(x)$	$Happy\_Father \sqsubseteq Man \sqcap \exists has\_child.Female$
$a{:}C$				$C(a)$	$John{:}Happy\_Father$

Note on DL Naming

$\mathcal{AL}$: $C, D \longrightarrow \top \mid \bot \mid A \mid C \sqcap D \mid \neg A \mid \exists R.\top \mid \forall R.C$

$\mathcal{C}$: Concept negation, $\neg C$. Thus, $\mathcal{ALC} = \mathcal{AL} + \mathcal{C}$

$\mathcal{S}$: Used for $\mathcal{ALC}$ with transitive roles $\mathcal{R}_+$

$\mathcal{U}$: Concept disjunction, $C_1 \sqcup C_2$

$\mathcal{E}$: Existential quantification, $\exists R.C$

$\mathcal{H}$: Role inclusion axioms, $R_1 \sqsubseteq R_2$, e.g. $is\_component\_of \sqsubseteq is\_part\_of$

$\mathcal{N}$: Number restrictions, $(\geq n\ R)$ and $(\leq n\ R)$, e.g. $(\geq 3\ has\_Child)$ (has at least 3 children)

$\mathcal{Q}$: Qualified number restrictions, $(\geq n\ R.C)$ and $(\leq n\ R.C)$, e.g. $(\leq 2\ has\_Child.Adult)$ (has at most 2 adult children)

$\mathcal{O}$: Nominals (singleton class), $\{a\}$, e.g. $\exists has\_child.\{mary\}$.
Note: $a{:}C$ equiv to $\{a\} \sqsubseteq C$ and $(a, b){:}R$ equiv to $\{a\} \sqsubseteq \exists R.\{b\}$

$\mathcal{I}$: Inverse role, R^-, e.g. $isPartOf = hasPart^-$

$\mathcal{F}$: Functional role, f, e.g. $functional(hasAge)$

$\mathcal{R}_+$: transitive role, e.g. $transitive(isPartOf)$

For instance,

$\mathcal{SHIF}$	=	$\mathcal{S} + \mathcal{H} + \mathcal{I} + \mathcal{F} = \mathcal{ALCR}_+\mathcal{HIF}$	OWL-Lite
$\mathcal{SHOIN}$	=	$\mathcal{S} + \mathcal{H} + \mathcal{O} + \mathcal{I} + \mathcal{N} = \mathcal{ALCR}_+\mathcal{HOIN}$	OWL-DL
$\mathcal{SROIQ}$	=	$\mathcal{S} + \mathcal{R} + \mathcal{O} + \mathcal{I} + \mathcal{Q} = \mathcal{ALCR}_+\mathcal{ROIN}$	OWL 2

Basics on Concrete Domains

- Concrete domains: reals, integers, strings, ...
  $(tim, 14)$*:hasAge*
  $(sf, \text{"}SoftComputing\text{"})$*:hasAcronym*
  $(source1, \text{"}ComputerScience\text{"})$*:isAbout*
  $(service2, \text{"}InformationRetrievalTool''\text{"})$*:Matches*
  $Minor = Person \sqcap \exists hasAge. \leq_{18}$
- Notation: (D). E.g., $\mathcal{ALC}(D)$ is $\mathcal{ALC}$ + concrete domains

Syntax and semantics of the DL $\mathcal{SROIQ}(\mathbf{D})$ (OWL 2)

Concepts	**Syntax (C)**	**FOL Reading of $C(x)$**
(C1)	A	$A(x)$
(C2)	$\top$	1
(C3)	$\bot$	0
(C4)	$C \sqcap D$	$C(x) \land D(x)$
(C5)	$C \sqcup D$	$C(x) \lor D(x)$
(C6)	$\neg C$	$\neg C(x)$
(C7)	$\forall R.C$	$\forall y.R(x,y) \to C(y)$
(C8)	$\exists R.C$	$\exists y.R(x,y) \land C(y)$
(C9)	$\forall T.\mathbf{d}$	$\forall v.T(x,v) \to \mathbf{d}(v)$
(C10)	$\exists T.\mathbf{d}$	$\exists v.T(x,v) \land \mathbf{d}(v)$
(C11)	$\{a\}$	$x = a$
(C12)	$(\geq m\ S.C)$	$\exists y_1 \ldots \exists y_m. \bigwedge_{i=1}^{m}(S(x,y_i) \land C(y_i)) \land \bigwedge_{1 \leq j < k \leq m} y_j \neq y_k$
(C13)	$(\leq m\ S.C)$	$\forall y_1 \ldots \forall y_{m+1}. \bigwedge_{i=1}^{m}(S(x,y_i) \land C(y_i)) \to \bigvee_{1 \leq j < k \leq m} y_j = y_k$
(C14)	$(\geq m\ T.\mathbf{d})$	$\exists v_1 \ldots \exists v_m. \bigwedge_{i=1}^{m}(T(x,v_i) \land \mathbf{d}(v_i)) \land \bigwedge_{1 \leq j < k \leq m} v_j \neq v_k$
(C15)	$(\leq m\ T.\mathbf{d})$	$\forall v_1 \ldots \forall v_{m+1}. \bigwedge_{i=1}^{m}(T(x,v_i) \land \mathbf{d}(v_i)) \to \bigvee_{1 \leq j < k \leq m} v_j = v_k$
(C16)	$\exists S.\mathrm{Self}$	$S(x,x)$
Roles	**Syntax (R)**	**Semantics of $R(x,y)$**
(R1)	R	$R(x,y)$
(R2)	R^-	$R(y,x)$
(R3)	U	1

Axiom	Syntax (E)	Semantics ($\mathcal{I}$ satisfies E if ...)
(A1)	$a{:}C$	$C(a)$
(A2)	$(a, b){:}R$	$R(a, b)$
(A3)	$(a, b){:}\neg R$	$\neg R(a, b)$
(A4)	$(a, v){:}T$	$T(a, v)$
(A5)	$(a, v){:}\neg T$	$\neg T(a, v)$
(A6)	$C \sqsubseteq D$	$\forall x.C(x) \to D(x)$
(A7)	$R_1 \ldots R_n \sqsubseteq R$	$\forall x_1 \forall x_{n+1} \exists x_2 \ldots$ $\exists x_n.(R_1(x_1, x_2) \land \ldots \land R_n(x_n, x_{n+1})) \to R(x_1, x_{n+1})$
(A8)	$T_1 \sqsubseteq T_2$	$\forall x \forall v.T_1(x, v) \to T_2(x, v)$
(A9)	trans(R)	$\forall x \forall y \forall z.R(x, z) \land R(z, y) \to R(x, y)$
(A10)	disj(S_1, S_2)	$\forall x \forall y.S_1(x, y) \land S_2(x, y) = 0$
(A11)	disj(T_1, T_2)	$\forall x \forall v.T_1(x, v) \land T_2(x, v) = 0$
(A12)	ref(R)	$\forall x.R(x, x)$
(A13)	irr(S)	$\forall x.\neg S(x, x)$
(A14)	sym(R)	$\forall x \forall y.R(x, y) = R(y, x)$
(A15)	asy(S)	$\forall x \forall y, S(x, y) \to \neg S(y, x)$

OWL 2 as Description Logic (excerpt)

Concept/Class constructors:

Abstract Syntax	DL Syntax	Example
Descriptions (C)		
A (URI reference)	A	`Conference`
`owl:Thing`	$\top$	
`owl:Nothing`	$\bot$	
`intersectionOf`(C_1 C_2 ...)	$C_1 \sqcap C_2$	`Reference` $\sqcap$ `Journal`
`unionOf`(C_1 C_2 ...)	$C_1 \sqcup C_2$	`Organization` $\sqcup$ `Institution`
`complementOf`(C)	$\neg C$	$\neg$ `MasterThesis`
`oneOf`(o_1 ...)	$\{o_1, \ldots\}$	`{"WISE","ISWC",...}`
`restriction`(R `someValuesFrom`(C))	$\exists R.C$	$\exists$`parts.InCollection`
`restriction`(R `allValuesFrom`(C))	$\forall R.C$	$\forall$`date.Date`
`restriction`(R `hasValue`(o))	$\exists R.\{o\}$	$\exists$`date.{2005}`
`restriction`(R `minCardinality`(n))	$(\geq n\ R)$	$(\geqslant 1$ `location`)
`restriction`(R `maxCardinality`(n))	$(\leq n\ R)$	$(\leqslant 1$ `publisher`)
`restriction`(U `someValuesFrom`(D))	$\exists U.D$	$\exists$`issue.integer`
`restriction`(U `allValuesFrom`(D))	$\forall U.D$	$\forall$`name.string`
`restriction`(U `hasValue`(v))	$\exists U. =_v\}$	$\exists$`series.`$=_{\text{"LNCS"}}$
`restriction`(U `minCardinality`(n))	$(\geq n\ U)$	$(\geqslant 1$ `title`)
`restriction`(U `maxCardinality`(n))	$(\leq n\ U)$	$(\leqslant 1$ `author`)

Note: R is an abstract role, while U is a concrete property of arity two.

Axioms:

Abstract Syntax	DL Syntax	Example
Axioms		
`Class`(A `partial` $C_1 \ldots C_n$)	$A \sqsubseteq C_1 \sqcap \ldots \sqcap C_n$	$Human \sqsubseteq Animal \sqcap Biped$
`Class`(A `complete` $C_1 \ldots C_n$)	$A = C_1 \sqcap \ldots \sqcap C_n$	$Man = Human \sqcap Male$
`EnumeratedClass`($A\ o_1 \ldots o_n$)	$A = \{o_1\} \sqcup \ldots \sqcup \{o_n\}$	$RGB = \{r\} \sqcup \{g\} \sqcup \{b\}$
`SubClassOf`($C_1\ C_2$)	$C_1 \sqsubseteq C_2$	
`EquivalentClasses`($C_1 \ldots C_n$)	$C_1 = \ldots = C_n$	
`DisjointClasses`($C_1 \ldots C_n$)	$C_i \sqcap C_j = \perp, i \neq j$	$Male \sqcap Female \sqsubseteq \perp$
`ObjectProperty`(R `super` (R_1)... `super` (R_n)	$R \sqsubseteq R_i$	$HasDaughter \sqsubseteq hasChild$
`domain`(C_1)...`domain`(C_n)	$(\geq 1\ R) \sqsubseteq C_i$	$(\geq 1\ hasChild) \sqsubseteq Human$
`range`(C_1)...`range`(C_n)	$\top \sqsubseteq \forall R.C_i$	$\top \sqsubseteq \forall hasChild.Human$
[`inverseof`(P)]	$R = P^-$	$hasChild = hasParent^-$
[`symmetric`]	$R \sqsubseteq R^-$	$similar = similar^-$
[`functional`]	$\top \sqsubseteq (\leq 1\ R)$	$\top \sqsubseteq (\leq 1\ hasMother)$
[`Inversefunctional`]	$\top \sqsubseteq (\leq 1\ R^-)$	
[`Transitive`])	$Tr(R)$	$Tr(ancestor)$
`SubPropertyOf`($R_1\ R_2$)	$R_1 \sqsubseteq R_2$	
`EquivalentProperties`($R_1 \ldots R_n$)	$R_1 = \ldots = R_n$	$cost = price$
`AnnotationProperty`(S)		

Abstract Syntax	DL Syntax	Example
DatatypeProperty(U super (U_1)... super (U_n)	$U \sqsubseteq U_i$	
domain(C_1)...domain(C_n)	$(\geq 1\ U) \sqsubseteq C_i$	$(\geq 1\ hasAge) \sqsubseteq Human$
range(D_1)...range(D_n)	$\top \sqsubseteq \forall U.D_i$	$\top \sqsubseteq \forall hasAge.posInteger$
[functional])	$\top \sqsubseteq (\leq 1\ U)$	$\top \sqsubseteq (\leq 1\ hasAge)$
SubPropertyOf($U_1 U_2$)	$U_1 \sqsubseteq U_2$	$hasName \sqsubseteq hasFirstName$
EquivalentProperties($U_1 \ldots U_n$)	$U_1 = \ldots = U_n$	
Individuals		
Individual(o type (C_1)... type (C_n))	$o{:}C_i$	$tim{:}Human$
value($R_1 o_1$)...value($R_n o_n$)	$(o, o_i){:}R_i$	$(tim, mary){:}hasChild$
value($U_1 v_1$)...value($U_n v_n$)	$(o, v_1){:}U_i$	$(tim, 14){:}hasAge$
SameIndividual($o_1 \ldots o_n$)	$o_1 = \ldots = o_n$	$president\_Bush = G.W.Bush$
DifferentIndividuals($o_1 \ldots o_n$)	$o_i \neq o_j, i \neq j$	$john \neq peter$
Symbols		
Object Property R (URI reference)	R	hasChild
Datatype Property U (URI reference)	U	hasAge
Individual o (URI reference)	U	tim
Data Value v (RDF literal)	U	“International Conference on Semantic Web”

DL Knowledge Base

- A DL Knowledge Base is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where
  - $\mathcal{T}$ is a TBox
    - ★ containing general inclusion axioms of the form $C \sqsubseteq D$,
    - ★ concept definitions of the form $A = C$
    - ★ primitive concept definitions of the form $A \sqsubseteq C$
    - ★ role inclusions of the form $R \sqsubseteq P$
    - ★ role equivalence of the form $R = P$
  - $\mathcal{A}$ is a ABox
    - ★ containing assertions of the form $a{:}C$
    - ★ containing assertions of the form $(a, b){:}R$
- An interpretation $\mathcal{I}$ is a model of $\mathcal{K}$, written $\mathcal{I} \models \mathcal{K}$ iff $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$, where
  - $\mathcal{I} \models \mathcal{T}$ ($\mathcal{I}$ is a model of $\mathcal{T}$) iff $\mathcal{I}$ is a model of each element in $\mathcal{T}$
  - $\mathcal{I} \models \mathcal{A}$ ($\mathcal{I}$ is a model of $\mathcal{A}$) iff $\mathcal{I}$ is a model of each element in $\mathcal{A}$

Basic Inference Problems (Formally)

Consistency: Check if knowledge is meaningful

- Is $\mathcal{K}$ satisfiability? $\mapsto$ Is there some model $\mathcal{I}$ of $\mathcal{K}$?
- Is C satisfiability? $\mapsto$ $C^{\mathcal{I}} \neq \emptyset$ for some some model $\mathcal{I}$ of $\mathcal{K}$?

Subsumption: structure knowledge, compute taxonomy

- $\mathcal{K} \models C \sqsubseteq D$? $\mapsto$ Is it true that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{K}$?

Equivalence: check if two classes denote same set of instances

- $\mathcal{K} \models C = D$? $\mapsto$ Is it true that $C^{\mathcal{I}} = D^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{K}$?

Instantiation: check if individual a instance of class C

- $\mathcal{K} \models a{:}C$? $\mapsto$ Is it true that $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{K}$?

Retrieval: retrieve set of individuals that instantiate C

- Compute the set $\{a \mid \mathcal{K} \models a{:}C\}$

Reduction to Satisfiability

Problems are all reducible to KB satisfiability

Subsumption: $\mathcal{K} \models C \sqsubseteq D$ iff $\langle \mathcal{T}, \mathcal{A} \cup \{a{:}C \sqcap \neg D\}\rangle$ not satisfiable, where a is a new individual

Equivalence: $\mathcal{K} \models C = D$ iff $\mathcal{K} \models C \sqsubseteq D$ and $\mathcal{K} \models D \sqsubseteq C$

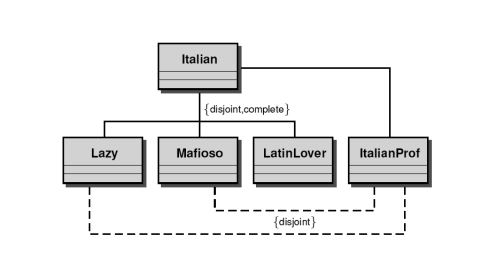
Instantiation: $\mathcal{K} \models a{:}C$ iff $\langle \mathcal{T}, \mathcal{A} \cup \{a{:}\neg C\}\rangle$ not satisfiable

Retrieval: The computation of the set $\{a \mid \mathcal{K} \models a{:}C\}$ is reducible to the instance checking problem

Exercise

Example (Latin lover)

Ontology: Consider the following conceptual UML like schema



$$\begin{array}{l} Lazy \sqsubseteq Italian \ , \ \ Mafioso \sqsubseteq Italian \ , \ \ LatinLover \sqsubseteq Italian \\ Italian \sqsubseteq (Lazy \sqcup Mafioso \sqcup LatinLover) \\ ItalianProf \sqsubseteq Italian \ , \ \ Lazy \sqsubseteq \neg Mafioso \\ Lazy \sqsubseteq \neg LatinLover \ , \ \ Mafioso \sqsubseteq \neg LatinLover \\ Mafioso \sqsubseteq \neg ItalianProf \ , \ \ Lazy \sqsubseteq \neg ItalianProf \end{array}$$

Consequence: $\mathcal{K} \models ItalianProf \sqsubseteq LatinLover$

Reasoning in DLs

- OWL 2: tableaux based algorithms
- OWL 2 EL: structural based algorithms
- OWL 2 QL: query rewriting based algorithms
- OWL 2 RL: query rewriting based algorithms

CQ Answering over OWL QL or OWL RL Global Schemas

- OWL 2 QL is related to the DL-Lite DL family [Artale et al., 2009]
- DL-Lite$_{core}$, the core language for the whole family (A atomic concept, P atomic role, and P^- is its inverse):

$$\begin{array}{lcl} B & \longrightarrow & A \mid \exists R \\ C & \longrightarrow & B \mid \neg B \\ \\ R & \longrightarrow & P \mid P^- \\ E & \longrightarrow & R \mid \neg R\,. \end{array}$$

- Inclusion axioms that are of the form $B \sqsubseteq C$
- DL-Lite$_{\mathcal{R}}$ from DL-Lite$_{core}$ allowing $R \sqsubseteq E$
- DL-Lite$_{\sqcap}$ is obtained from DL-Lite$_{core}$ allowing $B_1 \sqcap \ldots \sqcap B_n \sqsubseteq C$
- DL-Lite$_{\mathcal{F}}$ is obtained by extending DL-Lite$_{core}$ with global functional roles

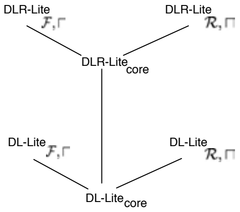


Figure: Excerpt of the DL-Lite family.

- OWL 2 RL is related to the Horn-DL family [Grosof et al., 2003, ter Horst, 2005] (A atomic concept, $m \in \{0, 1\}$, l is a value of the concrete domain, R is an object property, a individual, T is a datatype property):

$$
\begin{array}{lll}
B & \longrightarrow & A \mid \{a\} \mid B_1 \sqcap B_2 \mid B_1 \sqcup B_2 \mid \exists R.B \mid \exists T.\mathbf{d} \\
C & \longrightarrow & A \mid C_1 \sqcap C_2 \mid \neg B \mid \forall R.C \mid \exists R.\{a\} \mid \forall T.\mathbf{d} \mid \\
 & & (\leq m\ R.B) \mid (\leq m\ R) \mid (\leq m\ T.\mathbf{d}) \\
D & \longrightarrow & \exists R.\{a\} \mid \exists T.=_l \mid D_1 \sqcap D_2 \\
R & \longrightarrow & P \mid P^-
\end{array}
$$

- Inclusion axioms have the form

$$
\begin{array}{lll}
B & \sqsubseteq & C \\
A & = & D \\
 & & \\
R_1 & \sqsubseteq & R_2 \\
R_1 & = & R_2
\end{array}
$$

- There are others, such as disj(B_1, B_2), dom(R, C), ran(R, C), dom(T, C), fun(R), irr(R), sym(R), asy(R), *trans*$(()R)$, disj(R_1, R_2)

CQ Answering over OWL QL or OWL RL Global Schemas

- An OWL QL Global Schema is a set of OWL QL inclusion axioms
- An OWL RL Global Schema is a set of OWL RL inclusion axioms
- Mapping rules are of the form

$$\begin{array}{rcl}\langle A(x), s\rangle & \leftarrow & \langle S(x), s\rangle \\ \langle R(x,y), s\rangle & \leftarrow & \langle S'(x,y), s\rangle\end{array}$$

  where $S(x)$ and $S'(x, y)$ are relations over a LIR , and s is the score assigned to $A(x)$ and $R(x, y)$, respecively. If s omitted, then 1.0 is assumed
- A conjunctive query is a rule-like expression of the form (see also complex queries)

$$q(\mathbf{x}, s) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y}), s = f(\mathbf{z})$$

  where
  - the rule body $\varphi(\mathbf{x}, \mathbf{y})$ is a conjunction of $\langle P_i(\mathbf{z}_i), s_i\rangle$
  - P_i is either an atomic concept A or an atomic role R
  - $\mathbf{z}_i$ is a tuple of literals, or variables in $\mathbf{x}, \mathbf{y}$
  - $\mathbf{z}$ is a tuple of literals, or variables in $\mathbf{x}$, $\mathbf{y}$ or scores s_i
  - s_i is the score assigned to $P_i(\mathbf{z}_i)$
  - if P_i is an atomic concept (resp., a role) then $\mathbf{z}_i$ is unary (resp., binary) tuple
  - the scoring variables s and s_i are distinct from those in $\mathbf{x}$ and $\mathbf{y}$ and s is distinct from each s_i
- The answer set of a CQ q is

$$ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M}) = \{\langle \mathbf{t}, s\rangle \mid \mathcal{D} \cup \mathcal{G} \cup \mathcal{M} \cup \{q\} \models q(\mathbf{t}, s)\} \ ,$$

  where each tuple has an unique score
- As for LIRs, $ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$ is an ordered set
- $ans_k(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$ are the top-k retrieved tuples in $ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$

OWL QL or OQL RL CQ Answering over Distributed LIRs

- How do we answer queries over a Global Schema?
- As for RDFS, we apply Query Refomulation Algorithm [Madrid and Straccia, 2013, Straccia and Madrid, 2012, Straccia, 2012, Straccia, 2014]
- Ad-hoc solution exits [Straccia, 2012, Straccia, 2014]
- But, again, it suffices to provide a translation to *Logic Prgramming* (LP) and use a top-down algorithm for LPs [Madrid and Straccia, 2013, Straccia and Madrid, 2012, Straccia, 2014]
- For ease of presentation, we provide a translation for a simple, but useful, Horn-DL fragment only:
  - Note: transformation can be extended to whole OWL RL and OWL QL

$$
\begin{array}{lcl}
B & \longrightarrow & A \mid B_1 \sqcap B_2 \mid \exists R.B \\
C & \longrightarrow & A \\
R & \longrightarrow & P \mid P^-
\end{array}
$$

where inclusion axioms have the form

$$
\begin{array}{lcl}
B & \sqsubseteq & C \\
R_1 & \sqsubseteq & R_2
\end{array}
$$

OWL QL or OWL RL CQ Answering over Distributed LIRs (cont.)

- Consider glabal schema $\mathcal{G}$, mapping rules $\mathcal{M}$ and CQ query q
- We first extend the arity of predicates to accomodate scores

$$
\begin{array}{rcl}
\langle A(x), s\rangle & \mapsto & A(x, s) \\
\langle R(x, y), s\rangle & \mapsto & R(x, s)
\end{array}
$$

  where s is the score
- Now, Mapping rules

$$
\begin{array}{rcl}
\langle A(x), s\rangle & \leftarrow & \langle S(x), s\rangle \\
\langle R(x, y), s\rangle & \leftarrow & \langle S'(x, y), s\rangle
\end{array}
$$

  are transformed into

$$
\begin{array}{rcl}
A(x, s) & \leftarrow & S(x, s) \\
R(x, y, s) & \leftarrow & S'(x, y, s)
\end{array}
$$

OWL QL or OQL RL CQ Answering over Distributed LIRs (cont.)

- Next, we transform the inclusion axioms in the global schema $\mathcal{G}$
- To do so, we define a recursive mapping function σ which takes inclusions axioms and maps them into the following rules (s, s_i are scores and again $\otimes$ is a suitable t-norm to interpret conjunction):

$$
\begin{array}{rcl}
\sigma(R_1 \sqsubseteq R_2, s) & \mapsto & \sigma_{role}(R_2, x, y, s) \leftarrow \sigma_{role}(R_1, x, y, s) \\
\\
\sigma(B \sqsubseteq C, s) & \mapsto & \sigma_h(C, x, s) \leftarrow \sigma_b(B, x, s) \\
\\
\sigma_b(B_1 \sqcap B_2, x, s) & \mapsto & \sigma_b(B_1, x, s1), \sigma_b(B_2, x, s2), s = s1 \otimes s2 \\
\sigma_b(\exists R.B, x, s) & \mapsto & \sigma_{role}(R, x, y, s1), \sigma_b(B, y, s2), s = s1 \otimes s2 \\
\\
\sigma_h(A, x, s) & \mapsto & A(x, s) \\
\sigma_b(A, x, s) & \mapsto & A(x, s) \\
\\
\sigma_{role}(R, x, y, s) & \mapsto & R(x, y, s) \\
\sigma_{role}(R^-, x, y, s) & \mapsto & R(y, x, s)
\end{array}
$$

where x, y new variables

Example (Multimedia Information Retrieval)

Global Schema $\mathcal{G}$: $\{Dog \sqsubseteq Animal, Bird \sqsubseteq Animal, SmallDog \sqsubseteq Dog, SmallBird \sqsubseteq Bird\}$

Mapping Rules $\mathcal{M}$:
$\langle Dog(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{`}dog')(o, s)$
$\langle Bird(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{`}bird')(o, s)$
$\langle SmallDog(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{`}sd')(o, s)$
$\langle SmallBird(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{`}sb')(o, s)$
$\langle isAbout(r, o), s\rangle \leftarrow (SELECT\ region, obj, degr\ FROM\ imageClass)(r, o, s)$

LIRs:

imageClass			instances		
region	**obj**	**degr**	**obj**	**class**	**val**
o1	snoopy	0.8	snoopy	sd	0.4
o2	woodstock	0.9	woodstock	sb	0.7
o3	pluto	0.6	pluto	dog	1.0

Query: $q(x, y, s) \leftarrow \langle isAbout(x, y), s1\rangle, \langle Animal(y), s2\rangle, s = s1 \cdot s2$

Example (Multimedia Information Retrieval)

Global Schema $\mathcal{G}$: $\{Animal(x) \leftarrow Dog(x), Animal(x) \leftarrow Bird(x), Dog(x) \leftarrow SmallDog(x), Bird(x) \leftarrow SmallBird(x)\}$

Mapping Rules $\mathcal{M}$:
$\langle Dog(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'dog'})(o, s)$
$\langle Bird(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'bird'})(o, s)$
$\langle SmallDog(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'sd'})(o, s)$
$\langle SmallBird(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'sb'})(o, s)$
$\langle isAbout(r, o), s\rangle \leftarrow (SELECT\ region, obj, degr\ FROM\ imageClass)(r, o, s)$

LIRs:

imageClass		
region	**obj**	**degr**
o1	snoopy	0.8
o2	woodstock	0.9
o3	pluto	0.6

instances		
obj	**class**	**val**
snoopy	sd	0.4
woodstock	sb	0.7
pluto	dog	1.0

Query: $q(x, y, s) \leftarrow \langle isAbout(x, y), s1\rangle, \langle Animal(y), s2\rangle, s = s1 \cdot s2$

$r(q, \mathcal{G})$:

$$
\begin{array}{lll}
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle Animal(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle Dog(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle Bird(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle SmallDog(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle SmallBird(y), s2\rangle, s = s1 \cdot s2
\end{array}
$$

$answer(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$: $\{\langle o2, woodstock, 0.63\rangle, \langle o3, pluto, 0.6\rangle, \langle o1, snoopy, 0.32\rangle\}$

The case of Logic Programs

CQ Answering over LP Global Schemas [Straccia and Madrid, 2012, Straccia, 2013]

- So far, we have shown that CQ reformulation w.r.t. RDFS, OWL QL, OWL RL global schema can be transformted into CQ reformulation within LPs
- We address now the case the global schema is expressed via LP rules

LPs Basics (for ease, Datalog)

- Predicates are n-ary
- Terms are variables or constants
- Facts ground atoms
  For instance,
  $$has\_parent(mary, jo)$$
- Rules are of the form
  $$P(\mathbf{x}) \leftarrow \varphi(\mathbf{x}, \mathbf{y})$$
  where
  - $\varphi(\mathbf{x}, \mathbf{y})$ is a formula built from atoms of the form $B(\mathbf{z})$ and connectors $\wedge, \vee, 0, 1$
  - $\mathbf{z}_i$ is a tuple of literals, or variables in $\mathbf{x}$, $\mathbf{y}$
- For instance,
  $$has\_father(x, y) \quad \leftarrow \quad has\_parent(x, y) \wedge Male(y)$$

Remark

Note that

$$has\_father(x, y) \quad \leftarrow \quad has\_parent(x, y), Male(y)$$

is the same as repplacing "," with $\wedge$

- Extensional database (EDB): set of facts
- Intentional database (IDB): set of rules
- Logic Program $\mathcal{P}$:
  - $\mathcal{P} = \mathit{EDB} \cup \mathit{IDB}$
  - No predicate symbol in *EDB* occurs in the head of a rule in *IDB*
    - The principle is that we do not allow that *IDB* may redefine the extension of predicates in *EDB*
- *EDB* is usually, stored into a relational database

LPs Semantics: FOL semantics

- $\mathcal{P}^*$ is constructed as follows:
  1. set $\mathcal{P}^*$ to the set of all ground instantiations of rules in $\mathcal{P}$
  2. replace a fact $p(\mathbf{c})$ in $\mathcal{P}^*$ with the rule $p(\mathbf{c}) \leftarrow 1$
  3. if atom A is not head of any rule in $\mathcal{P}^*$, then add $A \leftarrow 0$ to $\mathcal{P}^*$
  4. replace several rules in $\mathcal{P}^*$ having same head

$$\left.\begin{array}{c} A \leftarrow \varphi_1 \\ A \leftarrow \varphi_2 \\ \vdots \\ A \leftarrow \varphi_n \end{array}\right\} \text{with } A \leftarrow \varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n$$

- Note: in $\mathcal{P}^*$ each atom $A \in B_{\mathcal{P}}$ is head of exactly one rule
- Herbrand Base of $\mathcal{P}$ is the set $B_{\mathcal{P}}$ of ground atoms
- Interpretation is a function $I : B_{\mathcal{P}} \to \{0, 1\}$
- Model $I \models \mathcal{P}$ iff for all $r \in \mathcal{P}^*$ $I \models r$, where $I \models A \leftarrow \varphi$ iff $I(\varphi) \leq I(A)$

- Entailment: for a ground atom $p(\mathbf{c})$

$$\mathcal{P} \models p(\mathbf{c}) \text{ iff all models of } \mathcal{P} \text{ satisfy } p(\mathbf{c})$$

- Least model $M_{\mathcal{P}}$ of $\mathcal{P}$ exists and is least fixed-point of

$$T_{\mathcal{P}}(I)(A) = I(\varphi), \text{ for all } A \leftarrow \varphi \in \mathcal{P}^*$$

- M can be computed as the limit of

$$\begin{array}{lcl} \mathbf{I}_0 & = & \mathbf{0} \\ \mathbf{I}_{i+1} & = & T_{\mathcal{P}}(\mathbf{I}_i) \ . \end{array}$$

Example

$$\mathcal{P} = \left\{ \begin{array}{lll} Q(x) & \leftarrow & B(x) \\ Q(x) & \leftarrow & C(x) \\ B(a) & & \\ C(b) & & \end{array} \right. \quad \mathcal{P}^* = \left\{ \begin{array}{lll} Q(a) & \leftarrow & B(a) \vee C(a) \\ Q(b) & \leftarrow & B(b) \vee C(b) \\ B(a) & \leftarrow & 1 \\ C(b) & \leftarrow & 1 \end{array} \right.$$

$\mathbf{I}_i$	$Q(a)$	$Q(b)$	$B(a)$	$B(b)$	$C(a)$	$C(b)$
$\mathbf{I}_0$	0	0	0	0	0	0
$\mathbf{I}_1$	0	0	1	0	0	1
$\mathbf{I}_2$	1	1	1	0	0	1
$\mathbf{I}_3$	1	1	1	0	0	1

- $\mathbf{I}_2 = \mathbf{I}_3$, i.e. $T_{\mathcal{P}}(\mathbf{I}_2) = \mathbf{I}_3 = \mathbf{I}_2$
- $\mathbf{I}_2$ is least fixed-point and, thus, minimal model $M_{\mathcal{P}} = \{Q(a), Q(b), B(a), C(b)\}$

CQ Answering over LP-based Global Schemas

- An LP Global Schema is a set of LP rules of the form $P(\mathbf{x}) \leftarrow \varphi(\mathbf{x}, \mathbf{y})$
- Mapping rules are of the form

$$\langle R(\mathbf{x}), s \rangle \quad \leftarrow \quad \langle S(\mathbf{x}), s \rangle$$

  where $S(\mathbf{x})$is a relation over a LIR , and s is the score assigned to $R(\mathbf{x})$. If s omitted, then 1.0 is assumed
- A conjunctive query is a rule-like expression of the form

$$q(\mathbf{x}, s) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y}), s = f(\mathbf{z})$$

  where
  - $\varphi(\mathbf{x}, \mathbf{y})$ is a conjunction of $\langle P_i(\mathbf{z}_i), s_i \rangle$
  - $\mathbf{z}_i$ is a tuple of literals, or variables in $\mathbf{x}, \mathbf{y}$
  - $\mathbf{z}$ is a tuple of literals, or variables in $\mathbf{x}, \mathbf{y}$ or scores s_i
  - s_i is the score assigned to $P_i(\mathbf{z}_i)$
  - the scoring variables s and s_i are distinct from those in $\mathbf{x}$ and $\mathbf{y}$ and s is distinct from each s_i
  - f is scoring function into $[0, 1]$
- The answer set of a CQ q is

$$ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M}) = \{\langle \mathbf{t}, s \rangle \mid \mathcal{D} \cup \mathcal{G} \cup \mathcal{M} \cup \{q\} \models q(\mathbf{t}, s)\} \ ,$$

  where each tuple has an unique score. If not, take the maximum.
- As for LIRs, $ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$ is an ordered set
- $ans_k(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$ are the top-k retrieved tuples in $ans(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$

- An example of CQ query is the following:

$$\begin{aligned}\langle GoodHotel(x), s\rangle \quad \leftarrow \quad & Hotel(x, price), hasDistanceToVenue(x, d), \\ & \langle Comfortable(x), s_3\rangle, s := 0.3 \cdot cheap(price) + 0.5 \cdot close(d) + 0.2 \cdot s_3\end{aligned}$$

The intended meaning is to retrieve good hotels, where the degree of goodness is a function of the degree of being cheap, close to the venue, and comfortable

Remark

We may also write an LP rule

$$p(\mathbf{x}) \leftarrow p_1(\mathbf{z}_1), \ldots, p_n(\mathbf{z}_n)$$

as

$$\langle p(\mathbf{x}), 1\rangle \leftarrow \langle p_1(\mathbf{z}_1), 1\rangle, \ldots, \langle p_n(\mathbf{z}_n), 1\rangle$$

Furthermore, a CQ

$$\langle p(\mathbf{x}), s\rangle \leftarrow \exists \mathbf{y}. \langle p_1(\mathbf{z}_1), s_1\rangle, \ldots, \langle p_n(\mathbf{z}_n), s_n\rangle, s := f(\mathbf{s})$$

may also be represented succinctly as

$$p(\mathbf{x}) \leftarrow f(p_1(\mathbf{z}_1), \ldots, p_n(\mathbf{z}_n))$$

For instance, we may write

$$\begin{aligned}GoodHotel(x) \quad \leftarrow \quad & \min(Hotel(x, price), hasDistanceToVenue(x, d), \\ & 0.3 \cdot cheap(price) + 0.5 \cdot close(d) + 0.2 \cdot Comfortable(x))\end{aligned}$$

We may also write $p(\mathbf{z}, s)$ *in place of* $\langle p(\mathbf{z}), s\rangle$

Query Reformulation w.r.t. LP Global Schema

- Consider an LP global schema $\mathcal{G}$ and a CQ q
- We present now a query rewriting procedure that resembles a top-town SLD-Resolution based procedure and presented in [Straccia, 2013], which is inspired to [Damásio et al., 2004, Kifer and Subrahmanian, 1992, Vojtás, 2001]
- The basic principle is as follows
- Assume we have propositional rules in which we assume that all scoring variables in the second rule have been renamed in order not to share any variable with the first one

$$\begin{array}{ll}
\text{From} & \langle A, s\rangle \leftarrow \langle A_1, s_1\rangle, \ldots, \langle A_j, s_j\rangle, \ldots, \langle A_k, s_k\rangle, s := f(\mathbf{s}) \\
\text{and} & \langle B, s'\rangle \leftarrow \langle B_1, s'_1\rangle, \ldots, \langle B_m, s'_m\rangle, s' := g(\mathbf{s}') \\
\text{and} & B = A_j \\
\hline
\text{infer} & \langle A, s\rangle \leftarrow \langle A_1, s_1\rangle, \ldots, \langle A_{j-1}, s_{j-1}\rangle, \\
 & \qquad \langle B_1, s'_1\rangle, \ldots, \langle B_m, s'_m\rangle, \\
 & \qquad \langle A_{j+1}, s_{j+1}\rangle \ldots, \langle A_k, s_k\rangle, \\
 & \qquad s := f(s_1, \ldots, s_{j-1}, g(\mathbf{s}'), s_{j+1}, \ldots, s_k)
\end{array}$$

  The propositional atom B is called the *selected* atom
- Essentially, we replace the fuzzy atom $\langle A_j, s_j\rangle$ with the fuzzy atoms $\langle B_1, s'_1\rangle, \ldots, \langle B_m, s'_m\rangle$ and accordingly replace the scoring variable s_j occurring in the scoring function f with $g(\mathbf{s}')$.

Query Reformulation w.r.t. LP Global Schema (cont.)

- The general case is essentially the same as the propositional case, except that now we have to take unification into account (as usual, we assume a variable renaming of the second input in the inference rules):

$$
\begin{array}{ll}
\text{From} & \langle A, s\rangle \leftarrow \langle A_1, s_1\rangle, \ldots, \langle A_j, s_j\rangle, \ldots, \langle A_k, s_k\rangle, s{:=}f(\mathbf{z}, \mathbf{s}) \\
\text{and} & \langle B, s'\rangle \leftarrow \langle B_1, s'_1\rangle, \ldots, \langle B_m, s'_m\rangle, s'{:=}g(\mathbf{z}', \mathbf{s}') \\
\text{and} & \theta \text{ as a mgu of } \{B, A_j\} \\
\hline
\text{infer} & \langle A\theta, s\rangle \leftarrow \langle A_1\theta, s_1\rangle, \ldots, \langle A_{j-1}\theta, s_{j-1}\rangle, \\
& \quad \langle B_1\theta, s'_1\rangle, \ldots, \langle B_m\theta, s'_m\rangle, \\
& \quad \langle A_{j+1}\theta, s_{j+1}\rangle \ldots, \langle A_k\theta, s_k\rangle, \\
& \quad s{:=}f(\mathbf{z}, s_1, \ldots, s_{j-1}, g(\mathbf{z}', \mathbf{s}'), s_{j+1}, \ldots, s_k)\theta \ .
\end{array}
$$

- where the notion of mgu (most general unifier) is defined as follows:
  - A *substitution* θ is of the form $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$, where each x_i is variable, each t_i is either a variable or constant distinct from x_i, and the variables $x_1, \ldots, x_n$ are distinct
  - Given atom A and substitution $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$, $A\theta$ denotes the atom obtained from A by replacing simultaneously all variables x_i with t_i
  - Given $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ and $\sigma = \{y_1/s_1, \ldots, y_m/s_m\}$, then the *composition* $\theta\sigma$ of θ and σ is the substitution obtained from the set $\{x_1/t_1\sigma, \ldots, x_n/t_n\sigma, y_1/s_1, \ldots, y_m/s_m\}$ by deleting the bindings $x_i/t_i\sigma$ for which $x_i = t_i\sigma$ and deleting any binding y_j/s_j for which $y_j \in \{x_1, \ldots, x_n\}$
  - Let $S = \{A_1, \ldots, A_n\}$ be a set of atoms A_i, we say that a substitution θ is an *unifier* for S iff $S\theta = \{A_1\theta, \ldots, A_n\theta\}$ is a singleton set
  - An unifier of S is called most general unifier (mgu) for S if, for each unifier σ of S there exists a non-empty substitution γ such that $\sigma = \theta\gamma$.

- Now, the set of rewritings of a query q w.r.t. $\mathcal{G}$, is the set $r(q, \mathcal{G}) = \{r_1, \ldots, r_n\}$, where each of which has q as head, r_1 is the query rule, and each rule r_{i+1} is inferred from r_i via the reformulation step above

- Termination guaranteed if $\mathcal{G}$ *acyclic*, i.e. non-recursive (no relation is defined directly or indirectly in terms of itself)

Example (Multimedia Information Retrieval)

Global Schema $\mathcal{G}$: $\{Animal(x) \leftarrow Dog(x), Animal(x) \leftarrow Bird(x), Dog(x) \leftarrow SmallDog(x), Bird(x) \leftarrow SmallBird(x)\}$

Mapping Rules $\mathcal{M}$: $\langle Dog(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'}dog'\text{'})(o, s)$
$\langle Bird(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'}bird'\text{'})(o, s)$
$\langle SmallDog(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'}sd'\text{'})(o, s)$
$\langle SmallBird(o), s\rangle \leftarrow (SELECT\ obj, val\ FROM\ instances\ WHERE\ class = \text{'}sb'\text{'})(o, s)$
$\langle isAbout(r, o), s\rangle \leftarrow (SELECT\ region, obj, degr\ FROM\ imageClass)(r, o, s)$

LIRs:

imageClass			instances		
region	**obj**	**degr**	**obj**	**class**	**val**
o1	snoopy	0.8	snoopy	sd	0.4
o2	woodstock	0.9	woodstock	sb	0.7
o3	pluto	0.6	pluto	dog	1.0

Query: $q(x, y, s) \leftarrow \langle isAbout(x, y), s1\rangle, \langle Animal(y), s2\rangle, s = s1 \cdot s2$

$r(q, \mathcal{G})$:

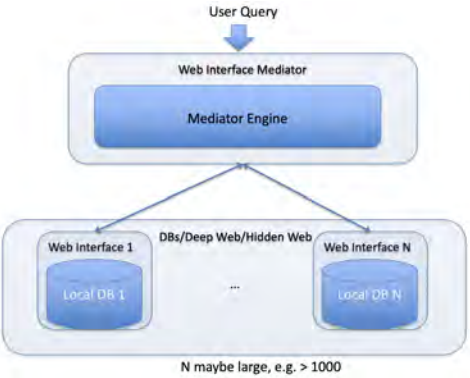
$$
\begin{array}{lcl}
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle Animal(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle Dog(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle Bird(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle SmallDog(y), s2\rangle, s = s1 \cdot s2 \\
q(x, y, s) & \leftarrow & \langle isAbout(x, y), s1\rangle, \langle SmallBird(y), s2\rangle, s = s1 \cdot s2
\end{array}
$$

$answer(q, \mathcal{D}, \mathcal{G}, \mathcal{M})$: $\{\langle o2, woodstock, 0.63\rangle, \langle o3, pluto, 0.6\rangle, \langle o1, snoopy, 0.32\rangle\}$

Not yet . . . one moment please . . .

Recap

Scenario



- Input: conjunctive query over a global mediator schema
- Problem: query the N resources
  - if N large, quering all N resources is unrealistic

In case of Small size mediators

- Given global schema $\mathcal{G}$, which may be unstructured, based on RDFS, OWL QL, OWL RL or LPs
- Given a query q over $\mathcal{G}$
  1. Rewrite the query q into a set $\{q_i\}$ of queries over of the local schemas $\mathcal{S}$, using mapping rules $\mathcal{M}$
  2. Submit the queries to the LIRs accessed through wrappers
  3. Merge all the ranked lists, using score normalisation and the DTA, and provide the result back to the user

In case of Large size mediators

- Given global schema $\mathcal{G}$, which may be unstructured, based on RDFS, OWL QL, OWL RL or LPs
- Sample the LIRs before hand
- For query q over global schema $\mathcal{G}$
  1. Rewrite the query q into a set $\{q_i\}$ of queries over of the local schemas $\mathcal{S}$
  2. Use the samples to determine which of the q_i are the top-s most relevant queries
  3. Submit the queries to the LIRs accessed through wrappers
  4. Merge all the ranked lists, using score normalisation and the DTA, and provide the result back to the user

Some small size mediator implementations

- Ontop
  - `https://ontop-vkg.org`
  - OWL-QL, SPARQL queries
- OBDA solutions, Mastro, Monolith, Eddy
  - `https://obdm.obdasystems.com`
  - OWL-QL, SPARQL queries
- SoftFacts
  - `https://www.umbertostraccia.it/cs/software/SoftFacts/SoftFacts.html`
  - DLR-Lite (n-ary DL-Lite), CQ with scoring atoms, top-k retrieval

Thanks

Bibliography

Arguello, J., Callan, J., and Diaz, F. (2009).
Classification-based resource selection.
In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 1277–1286, New York, NY, USA. ACM.

Artale, A., Calvanese, D., Kontchakov, R., and Zakharyaschev, M. (2009).
The DL-Lite family and relations.
Journal of Artificial Intelligence Research, 36:1–69.

Calì, A. and Straccia, U. (2015).
A framework for conjunctive query answering over distributed deep web information resources.
In *Proceedings of the 23rd Italian Symposium on Advanced Database Systems (SEBD-15)*, pages 358–365. Curran Associates, Inc.

Calì, A. and Straccia, U. (2017).
Integration of deep web sources: A distributed information retrieval approach.
In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics (WIMS-17)*, WIMS '17, pages 33:1–33:4, New York, NY, USA. ACM.

Callan, J. and Connell, M. (2001).
Query-based sampling of text databases.
ACM Transactions on Information Systems, 19(2):97–130.

Cardillo, F. A., Debole, F., and Straccia, U. (2023).
Pn-owl: A two stage algorithm to learn fuzzy concept inclusions from owl ontologies.
Technical report, Computing Research Repository.
Available as CoRR technical report at http://arxiv.org/abs/2303.07192.

Cardillo, F. A. and Straccia, U. (2022).
Fuzzy owl-boost: Learning fuzzy concept inclusions via real-valued boosting.
Fuzzy Sets and Systems, 438:164–186.

Caverlee, J., Liu, L., and Bae, J. (2006).

Distributed query sampling: A quality-conscious approach.
In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 340–347, New York, NY, USA. ACM.

Damásio, C. V., Medina, J., and Ojeda Aciego, M. (2004).
A tabulation proof procedure for residuated logic programming.
In *Proceedings of the 6th European Conference on Artificial Intelligence (ECAI-04)*.

Grosof, B. N., Horrocks, I., Volz, R., and Decker, S. (2003).
Description logic programs: combining logic programs with description logic.
In *Proceedings of the 12th International Conference on World Wide Web*, pages 48–57. ACM Press.

Hankerson, D., Johnson, P. D., and Harris, G. A. (1998).
Introduction to Information Theory and Data Compression.
CRC Press, Inc., Boca Raton, FL, USA, 1st edition.

Kifer, M. and Subrahmanian, V. (1992).
Theory of generalized annotated logic programming and its applications.
Journal of Logic Programming, 12:335–367.

Klir, G. J. and Yuan, B. (1995).
Fuzzy sets and fuzzy logic: theory and applications.
Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Lenzerini, M. (2002).
Data integration: a theoretical perspective.
In *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS-02)*, pages 233–246. ACM Press.

Madrid, N. and Straccia, U. (2013).
Towards a top-k retrieval for non-monotonic ranking functions.
In *In Proceedings of the 10th International Conference on Flexible Query Answering Systems (FQAS-13)*, volume 8132 of *Lecture Notes in Artificial Intelligence*, pages 507–518. Springer Verlag.

Markov, I., Arampatzis, A., and Crestani, F. (2013).

On CORI results merging.
In *Proceedings of the 35th European Conference on Advances in Information Retrieval*, ECIR'13, pages 752–755, Berlin, Heidelberg. Springer-Verlag.

Meghini, C., Sebastiani, F., and Straccia, U. (2001).
A model of multimedia information retrieval.
Journal of the ACM, 48(5):909–970.

Ragone, A., Straccia, U., Noia, T. D., Sciascio, E. D., and Donini, F. M. (2009).
Fuzzy matchmaking in e-marketplaces of peer entities using Datalog.
Fuzzy Sets and Systems, 160(2):251–268.

Renda, M. E. and Straccia, U. (2002).
Metasearch: Rank vs. score based rank list fusion methods (without training data).
Technical Report 2002-TR-07, Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy.

Renda, M. E. and Straccia, U. (2003).
Web metasearch: Rank vs. score based rank aggregation methods.
In *Proceedings of the 18th Annual ACM Symposium on Applied Computing (SAC-03)*, pages 841–846, Melbourne, Florida, USA. ACM.

Shokouhi, M. and Si, L. (2011).
Federated search.
Found. Trends Inf. Retr., 5(1):1–102.

Shokouhi, M. and Zobel, J. (2009).
Robust result merging using sample-based score estimates.
ACM Trans. Inf. Syst., 27(3):14:1–14:29.

Si, L. and Callan, J. (2004).
Unified utility maximization framework for resource selection.
In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, pages 32–41, New York, NY, USA. ACM.

Straccia, U. (2006).

Towards top-k query answering in deductive databases.
In *Proceedings of the 2006 IEEE International Conference on Systems, Man and Cybernetics (SMC-06)*, pages 4873–4879. IEEE.

Straccia, U. (2012).
Top-k retrieval for ontology mediated access to relational databases.
Information Sciences, 198:1–23.

Straccia, U. (2013).
Foundations of Fuzzy Logic and Semantic Web Languages.
CRC Studies in Informatics Series. Chapman & Hall.

Straccia, U. (2014).
On the top-k retrieval problem for ontology-based access to databases.
In Pivert, Olivier; Zadrożny, S., editor, *Flexible Approaches in Data, Information and Knowledge Management*, volume 497 of *Studies in Computational Intelligence*, chapter 5, pages 95–114. Springer Verlag.

Straccia, U. and Casini, G. (2022).
A Minimal Deductive System for RDFS with Negative Statements.
In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, pages 351–361.

Straccia, U. and Madrid, N. (2012).
A top-k query answering procedure for fuzzy logic programming.
Fuzzy Sets and Systems, 205:1–29.

Straccia, U. and Troncy, R. (2006).
Towards distributed information retrieval in the semantic web: Query reformulation using the omap framework.
In *3rd European Semantic Web Conference (ESWC-06)*, volume 4011 of *Lecture Notes in Computer Science*, pages 378–392. Springer Verlag.

Sutherland, W. J., editor (2006).
Ecological Census Techniques: A Handbook.
Cambridge University Press.

ter Horst, H. J. (2005).

Completeness, decidability and complexity of entailment for rdf schema and a semantic extension involving the owl vocabulary.
Journal of Web Semantics, 3(2-3):79–115.

Thomas, P. (2012).
To what problem is distributed information retrieval the solution?
J. Am. Soc. Inf. Sci. Technol., 63(7):1471–1476.

Thomas, P. and Hawking, D. (2009).
Server selection methods in personal metasearch: A comparative empirical study.
Inf. Retr., 12(5):581–604.

Vojtás, P. (2001).
Fuzzy logic programming.
Fuzzy Sets and Systems, 124:361–370.

Yu, C., Liu, K.-L., Meng, W., Wu, Z., and Rishe, N. (2002).
A methodology to retrieve text documents from multiple databases.
IEEE Trans. on Knowl. and Data Eng., 14(6):1347–1361.

Zadeh, L. A. (1965).
Fuzzy sets.
Information and Control, 8(3):338–353.

Zimmermann, A., Lopes, N., Polleres, A., and Straccia, U. (2012).
A general framework for representing, reasoning and querying with annotated semantic web data.
Journal of Web Semantics, 11:72–95.