Fuzzy Logic, Annotation Domains and Semantic Web Languages

Umberto Straccia

ISTI-CNR, Pisa, Italy
http://www.straccia.info
straccia@isti.cnr.it

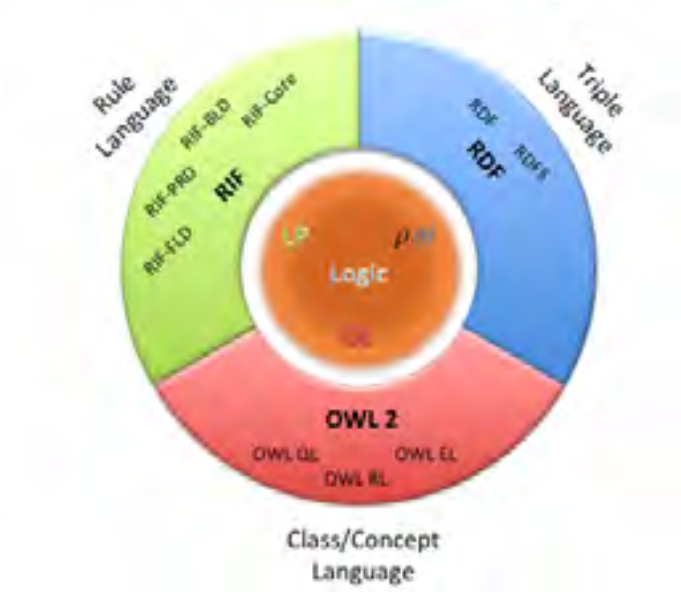
Semantic Web Languages Basics

Semantic Web Initiative: From a Knowledge Representation and Reasoning point of view

- Standards to represent Knowledge/Information at various level of granularity
- Semantics of these languages is well defined, based on some logical language
- Several tools have been developed in terms of Editors, Reasoners, ...
- The computational complexity analysis and the development of optimised/customised algorithms is very important

The Semantic Web Family of Languages

- Wide variety of languages
  - RDFS: *Triple language*, -*Resource Description Framework*
    - logic having intensional semantics and the logical counterpart is ρdf
  - OWL 2: *Conceptual language*, -*Ontology Web Language*
    - family of languages that relate to *Description Logics* (DLs)
  - RIF: *Rule language*, -*Rule Interchange Format*,
    - family of languages that relate to the *Logic Programming* (LP) paradigm
- OWL 2 and RIF have an extensional semantics



RDFS

- RDFS: the triple language

$$\langle subject, predicate, object \rangle$$

  e.g. $\langle umberto, born, zurich \rangle$
- Computationally: compute *closure*, $cl(KB)$,
  - Infer all possible triples using inference rules, e.g.

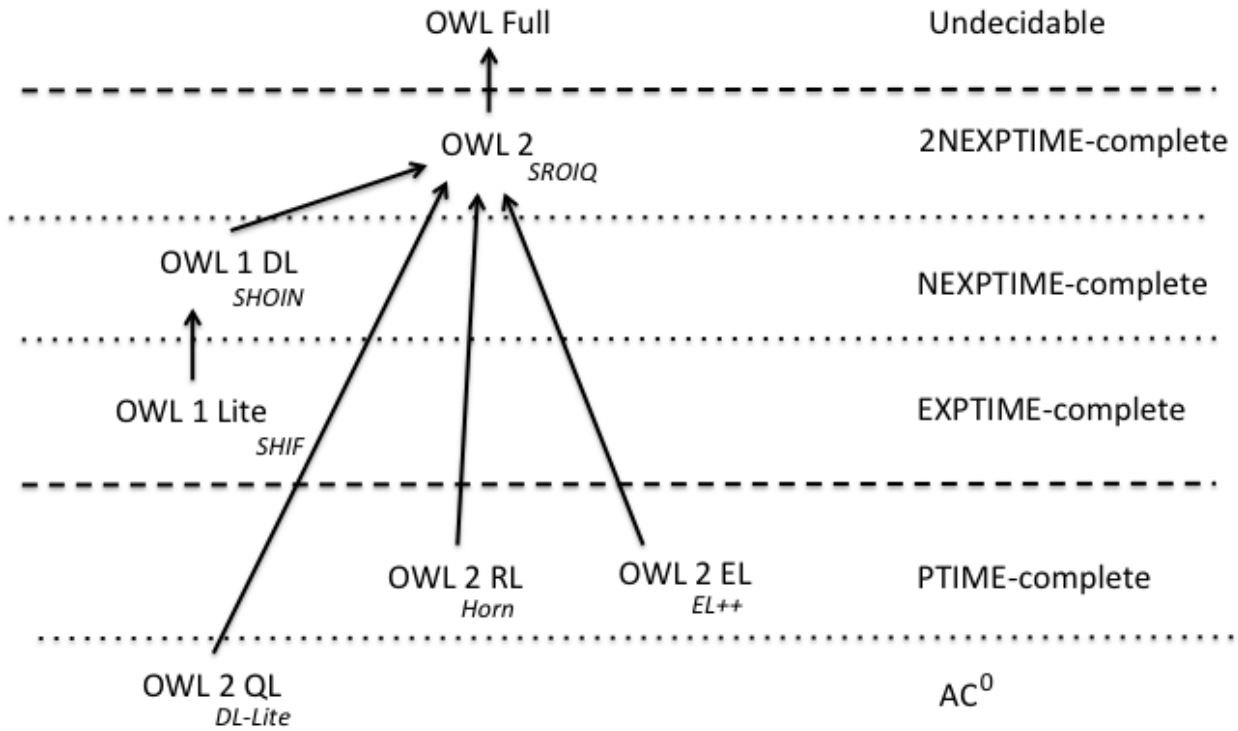
$$\frac{(A, \text{sc}, B), (X, \text{type}, A)}{(X, \text{type}, B)}$$

    "if A subclass of B, X instance of A then X is instance of B"
- Complexity: $\mathcal{O}(|KB|^2)$
- Store all inferred triples into a relational database and query via SQL

- OWL family: the object oriented language

```
class      PERSON partial
           restriction (hasName someValuesFrom String)
           restriction (hasBirthPlace someValuesFrom GEOPLACE)
           ...
```

- Computationally: tableaux like algorithms



OWL 2 Profiles

OWL 2 EL

- useful for large size of properties and/or classes
- basic reasoning problems solved inpolynomial time
- 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 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
- Computational complexity: same as for Datalog, polynomial in size of the data, EXPTIME w.r.t. size of knowledge base

RIF

- RIF family: the rule language

  `Forall` ?Buyer ?Item ?Seller

  buy(?Buyer ?Item ?Seller) :- sell(?Seller ?Item ?Buyer)

RIF Dialects

RIF-BLD: *Basic Logic Dialect*

- Horn logic with various syntactic and semantic extensions.
- Frame syntax, with datatypes and externally defined predicates

RIF-PRD: *Production Rule Dialect*

- aims at capturing the main aspects of various production rule systems
- defined using ad hoc computational mechanism, and not based on a logic
- RIF-PRD is not part of the suite of logical RIF dialects and stands apart from them

RIF-Core: *Core Dialect*

- subset of both RIF-BLD and RIF-PRD
- RIF-Core corresponds to Horn logic without function symbols (i.e., Datalog)
- number of extensions to support features such as objects and frames as in F-logic

The Quest for Fuzzy Semantic Web Languages

Fuzzyness

- Fuzzy statements involve context sensitive concepts with no exact definition (binary decision/membership function)

  tall, small, close, far, cheap, expensive, is about, similar to, warm, cold, . . .

- Fuzzy concepts may occur in information need formulation

  "Find me the a good hotel close to the conference venue"

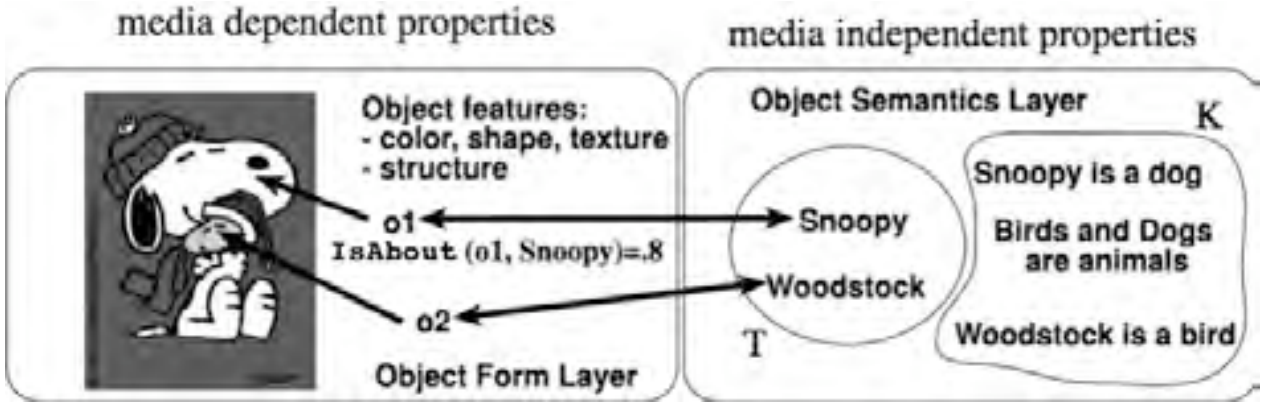
- Fuzzy concepts may occur also in informative statements

  "If a hotel is close to the leaning tower of Pisa then it is a touristic hotel"

- Fuzzy statements are true to some degree, which is taken from a truth space (usually $[0, 1]$)

Other examples

- (Multimedia) Information Retrieval:
  - To which degree is a Web site, a Web page, a text passage, an image region, a video segment, . . . relevant to my information need?



IsAbout		
ImageRegion	*Object ID*	*degree*
*o*1	*snoopy*	0.8
*o*2	*woodstock*	0.7
⋮	⋮	

"Find top-k image regions about animals"

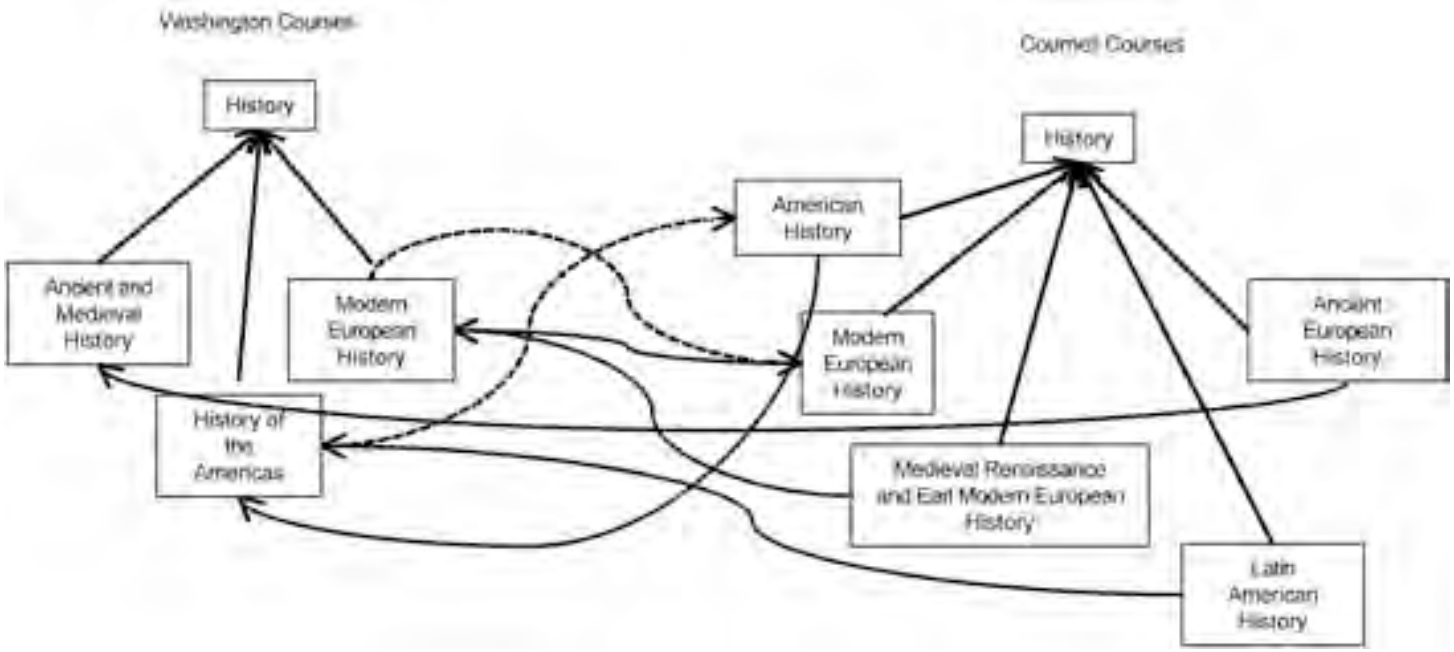
$$Query(x) \leftarrow ImageRegion(x) \wedge isAbout(x, y) \wedge Animal(y)$$

- Matchmaking
  - To which degree does an object match my requirements?
    - if I'm looking for a car and my budget is *about* 30.000 €, to which degree does a car's price match my budget?

- Semantic annotation / classification
  - To which degree does e.g., an image object represent or is about a dog?

“White Dog Cafe”

- Ontology alignment (schema mapping)
  - To which degree do two concepts of two ontologies represent the same, or are disjoint, or are overlapping?

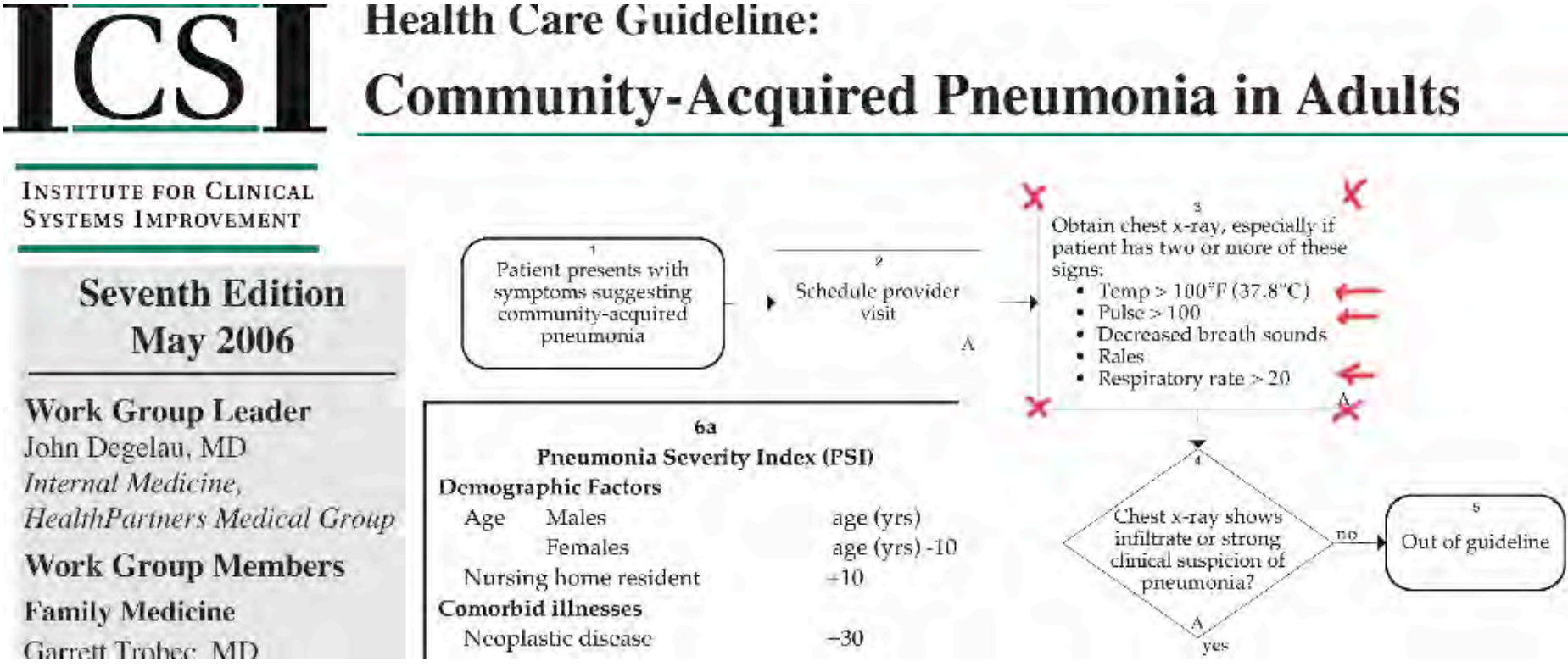


Example (Distributed Information Retrieval)

Then the agent has to perform automatically the following steps:

1. The agent has to select a subset of relevant resources $\mathscr{S}' \subseteq \mathscr{S}$, as it is not reasonable to assume to access to and query all resources (resource selection/resource discovery);
2. For every selected source $\mathcal{S}_i \in \mathscr{S}'$ the agent has to reformulate its information need Q_A into the query language $\mathcal{L}_i$ provided by the resource (schema mapping/ontology alignment);
3. The results from the selected resources have to be merged together (data fusion/rank aggregation)

Example (Health-care: diagnosis of pneumonia)



ICSI

INSTITUTE FOR CLINICAL SYSTEMS IMPROVEMENT

Health Care Guideline:

Community-Acquired Pneumonia in Adults

Seventh Edition
May 2006

Work Group Leader
John Degelau, MD
Internal Medicine,
HealthPartners Medical Group

Work Group Members
Family Medicine
Garrett Trobec, MD

1
Patient presents with symptoms suggesting community-acquired pneumonia

2
Schedule provider visit
A

3
Obtain chest x-ray, especially if patient has two or more of these signs:
- Temp > 100°F (37.8°C)
- Pulse > 100
- Decreased breath sounds
- Rales
- Respiratory rate > 20

A

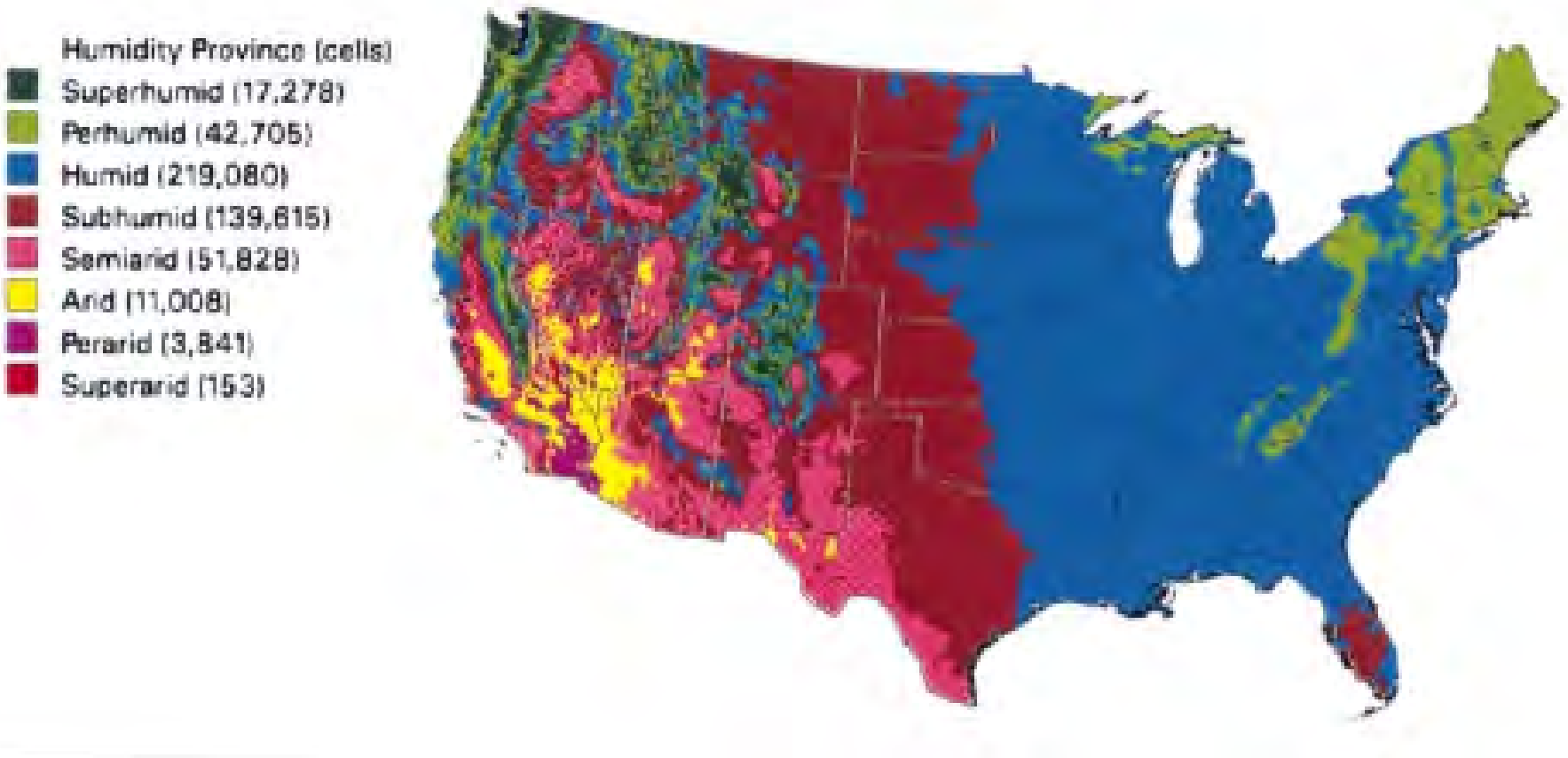
4
Chest x-ray shows infiltrate or strong clinical suspicion of pneumonia?
yes
no

5
Out of guideline

6a
Pneumonia Severity Index (PSI)

Demographic Factors		
Age	Males	age (yrs)
	Females	age (yrs) -10
Nursing home resident		+10
Comorbid illnesses		
Neoplastic disease		−30

- Lifezone mapping
  - To which degree do certain areas have a specific bioclima



Holdridge life zones of USA

- Allowing to deal with fuzzy concepts in SWLs seems a nice feature
- To be able to to do it rigorously and computationally in an attractive way is not so easy

Fuzzy Logic Basics

Fuzzyness & Logic

- A fuzzy statement is true to some degree, which is taken from a truth space
- Truth space: set of truth values L and an partial order $\leq$
- Many-valued Interpretation: a function I mapping formulae into L, i.e. $I(\varphi) \in L$
- Mathematical Fuzzy Logic: $L = [0, 1]$, but also $L_n = \{0, \frac{1}{n}, \ldots, \frac{n-1}{n}, 1\}$ for an integer $n \geq 1$

- Problem: what is the interpretation of e.g.

$$\varphi \wedge \psi \qquad ?$$

  - E.g., if $I(\varphi) = 0.83$ and $I(\psi) = 0.2$, what is the result of $0.83 \wedge 0.2$?
- More generally, what is the result of $n \wedge m$, for $n, m \in [0, 1]$?
- The choice cannot be any arbitrary computable function, but has to reflect some basic properties that one expects to hold for a "conjunction"
- Norms: functions that are used to interpret connectives such as $\wedge, \vee, \neg, \rightarrow$
  - t-norm: interprets conjunction
  - s-norm: interprets disjunction
- Norms are compatible with classical two-valued logic

Axioms for t-norms and s-norms

Axiom Name	T-norm	S-norm
Tautology / Contradiction	$a \wedge 0 = 0$	$a \vee 1 = 1$
Identity	$a \wedge 1 = a$	$a \vee 0 = a$
Commutativity	$a \wedge b = b \wedge a$	$a \vee b = b \vee a$
Associativity	$(a \wedge b) \wedge c = a \wedge (b \wedge c)$	$(a \vee b) \vee c = a \vee (b \vee c)$
Monotonicity	if $b \leq c$, then $a \wedge b \leq a \wedge c$	if $b \leq c$, then $a \vee b \leq a \vee c$

Axioms for implication and negation functions

Axiom Name	Implication Function	Negation Function
Tautology / Contradiction	$0 \to b = 1$ $a \to 1 = 1$	$\neg 0 = 1,\ \neg 1 = 0$
Antitonicity	if $a \leq b$, then $a \to c \geq b \to c$	if $a \leq b$, then $\neg a \geq \neg b$
Monotonicity	if $b \leq c$, then $a \to b \leq a \to c$	

Usually,

$$a \to b = \sup\{c\colon a \wedge c \leq b\}$$

is used and is called r-implication and depends on the t-norm only

Typical norms

	Lukasiewicz Logic	Gödel Logic	Product Logic	Zadeh
$\neg x$	$1 - x$	if $x = 0$ then 1 else 0	if $x = 0$ then 1 else 0	$1 - x$
$x \wedge y$	$\max(x + y - 1, 0)$	$\min(x, y)$	$x \cdot y$	$\min(x, y)$
$x \vee y$	$\min(x + y, 1)$	$\max(x, y)$	$x + y - x \cdot y$	$\max(x, y)$
$x \Rightarrow y$	if $x \leq y$ then 1 else $1 - x + y$	if $x \leq y$ then 1 else y	if $x \leq y$ then 1 else y/x	$\max(1 - x, y)$

Note: for Lukasiewicz Logic and Zadeh, $x \Rightarrow y \equiv \neg x \vee y$

- Any other t-norm can be obtained as a combination of Lukasiewicz, Gödel and Product t-norm
- Zadeh: not interesting for mathematical fuzzy logicians: its a sub-logic of Łukasiewicz and, thus, rarely considered by fuzzy logicians

$$\begin{aligned} \neg_Z x &= \neg_Ł x \\ x \wedge_Z y &= x \wedge_Ł (x \rightarrow_Ł y) \\ x \rightarrow_Z y &= \neg_Ł x \vee_Ł y \end{aligned}$$

Some additional properties of t-norms, s-norms, implication functions, and negation functions of various fuzzy logics.

Property	Łukasiewicz Logic	Gödel Logic	Product Logic	Zadeh Logic
$x \wedge \neg x = 0$	•	•	•	
$x \vee \neg x = 1$	•			
$x \wedge x = x$		•		•
$x \vee x = x$		•		•
$\neg \neg x = x$	•			•
$x \Rightarrow y = \neg x \vee y$	•			•
$\neg (x \Rightarrow y) = x \wedge \neg y$	•			•
$\neg (x \wedge y) = \neg x \vee \neg y$	•	•	•	•
$\neg (x \vee y) = \neg x \wedge \neg y$	•	•	•	•
$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$		•		•
$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$		•		•

- Note: If all conditions in the upper part of a column have to be satisfied then we collapse to classical two-valued logic, i.e. $L = \{0, 1\}$

Propositional Fuzzy Logic

- Formulae: propositional formulae
- Truth space is $[0, 1]$
- Formulae have a a degree of truth in $[0, 1]$
- Interpretation: is a mapping $\mathcal{I} : \mathit{Atoms} \to [0, 1]$
- Interpretations are extended to formulae using norms to interpret connectives $\wedge, \vee, \neg, \to$

$$\begin{array}{lcl} \mathcal{I}(\varphi \wedge \psi) & = & \mathcal{I}(\varphi) \wedge \mathcal{I}(\psi) \\ \mathcal{I}(\varphi \vee \psi) & = & \mathcal{I}(\varphi) \vee \mathcal{I}(\psi) \\ \mathcal{I}(\varphi \to \psi) & = & \mathcal{I}(\varphi) \to \mathcal{I}(\psi) \\ \mathcal{I}(\neg\varphi) & = & \neg\mathcal{I}(\varphi) \end{array}$$

- Rational $r \in [0, 1]$ may appear as atom in formula, where $\mathcal{I}(r) = r$

- Note:

$$\mathcal{I}(r \to \varphi) = 1 \quad \textit{iff} \quad \mathcal{I}(\varphi) \geq r$$
$$\mathcal{I}(\varphi \to r) = 1 \quad \textit{iff} \quad \mathcal{I}(\varphi) \leq r$$

- We use $\varphi \geq r$ as an abbreviation of $r \to \varphi$ and $\varphi \leq r$ as an abbreviation of $\varphi \to r$
- Semantics:

$$\begin{array}{rll} I \models \varphi & \textit{iff} & \mathcal{I}(\varphi) = 1 \\ \mathcal{I} \models KB & \text{iff} & \mathcal{I} \models \varphi \text{ for all } \varphi \in KB \\ KB \models \varphi & \text{iff} & \text{for all } \mathcal{I}. \text{ if } \mathcal{I} \models KB \text{ then } \mathcal{I} \models \varphi \end{array}$$

- Deduction rule is valid: for $r, s \in [0, 1]$:

$$r \to \varphi, s \to (\varphi \to \psi) \models (r \wedge s) \to \psi$$

Informally,

From $\varphi \geq r$ and $(\varphi \to \psi) \geq s$ infer $\psi \geq r \wedge s$

- Let

$$bsd(KB, \phi) = \sup\{\mathcal{I}(\phi) \mid \mathcal{I} \models KB\} \text{ (Best Satisfiability Degree (BSD))}$$

$$bed(KB, \phi) = \sup\{r \mid KB \models \varphi \geq r\} \text{ (Best Entailment Degree (BED))}$$

- Then

$$bed(KB, \phi) = \min x. \text{ such that } KB \cup \{\phi \leq x\} \text{ satisfiable.}$$

- Assume KB is a set of formulae $\phi \geq n$ or $\phi \leq n$
- For a formula ϕ consider a variable x_ϕ (that the degree of truth of ϕ is greater or equal to x_ϕ)
- E.g., for Łukasiewicz logic, use Mixed Integer Linear Programming

$$\begin{aligned} bed(KB, \phi) = \min x. \text{ such that } & x \in [0,1], x_\phi \leq x, \sigma(\phi), \\ & \text{for all } \phi' \geq n \in KB, x_{\phi'} \geq n, \sigma(\phi'), \\ & \text{for all } \phi' \leq n \in KB, x_{\phi'} \leq n, \sigma(\phi') \end{aligned}$$

$$\sigma(\phi) = \begin{cases} x_p \in [0,1] & \text{if} \quad \phi = p \\ x_r = r & \text{if} \quad \phi = r, r \in [0,1] \\ x_{\phi'} = \ominus x_\phi, x_\phi \in [0,1] & \text{if} \quad \phi = \neg\phi' \\ x_{\phi_1} \otimes x_{\phi_2} = x_\phi, \\ \sigma(\phi_1), \sigma(\phi_2), x_\phi \in [0,1] & \text{if} \quad \phi = \phi_1 \wedge \phi_2 \\ x_{\phi_1} \oplus x_{\phi_2} = x_\phi & \text{if} \quad \phi = \phi_1 \vee \phi_2 \\ \sigma(\neg\phi_1 \vee \phi_2) & \text{if} \quad \phi = \phi_1 \rightarrow \phi_2 \,. \end{cases}$$

where

$$\begin{aligned} x_1 = \ominus x_2 \quad &\mapsto \quad x_1 = 1 - x_2 \\ x_1 \oplus x_2 = z \quad &\mapsto \quad \{y \leq z, x_1 + x_2 \geq y, z \leq x_1 + x_2 \leq z + y, y \in \{0,1\}\} \\ x_1 \otimes x_2 = z \quad &\mapsto \quad \{z \leq y, x_1 + x_2 - 1 \geq y, z - y \leq x_1 + x_2 - 1 \leq z, y \in \{0,1\}\} \end{aligned}$$

- In a similar way, we may determine $bsd(KB, \phi)$ as

$$\begin{aligned} &\min -x. \text{ such that } x \in [0,1], x_\phi \geq x, \sigma(\phi), \\ &\text{for all } \phi' \geq n \in KB, x_{\phi'} \geq n, \sigma(\phi'), \\ &\text{for all } \phi' \leq n \in KB, x_{\phi'} \leq n, \sigma(\phi') \end{aligned}$$

Example

- Consider $KB = \{p \geq 0.6, p \rightarrow q \geq 0.7\}$
- Let us show that $bed(q, KB) = 0.3$
- Recall that $bed(q, KB)$ is

$$\begin{array}{l} \min x. \text{ such that } x \in [0,1], x_q \leq x, \sigma(q), \\ \quad \text{for all } \phi' \geq n \in KB, x_{\phi'} \geq n, \sigma(\phi'), \\ \quad \text{for all } \phi' \leq n \in KB, x_{\phi'} \leq n, \sigma(\phi') \end{array}$$

$$\begin{array}{lcl} p \geq 0.6 & \mapsto & x_p \geq 0.6, x_p \in [0,1] \\ p \rightarrow q \geq 0.7 & \mapsto & x_{p \rightarrow q} \geq 0.7, x_{p \rightarrow q} \in [0,1], \sigma(p \rightarrow q) \\ \sigma(q) & \mapsto & x_q \in [0,1] \\ \sigma(p \rightarrow q) & \mapsto & x_{\neg p \vee q} = x_{p \rightarrow q}, \sigma(\neg p \vee q) \\ \sigma(\neg p \vee q) & \mapsto & x_{\neg p} + x_q = x_{\neg p \vee q}, \sigma(\neg p), \sigma(q), x_{\neg p \vee q} \in [0,1] \\ \sigma(\neg p) & \mapsto & x_p = 1 - x_{\neg p}, x_p \in [0,1] \end{array}$$

It follows that $0.3 = \min x. \ldots$

Fuzzy Concrete Domains

- Allows us to define concepts such as young, cheap, cold, etc.
- We allow also crisp constraints such as AlarmSystem $\wedge$ (price $>$ 26,000), AlarmSystem $\rightarrow$ (deliverytime $\geq$ 30)
- Fuzzy membership functions: usually of the form

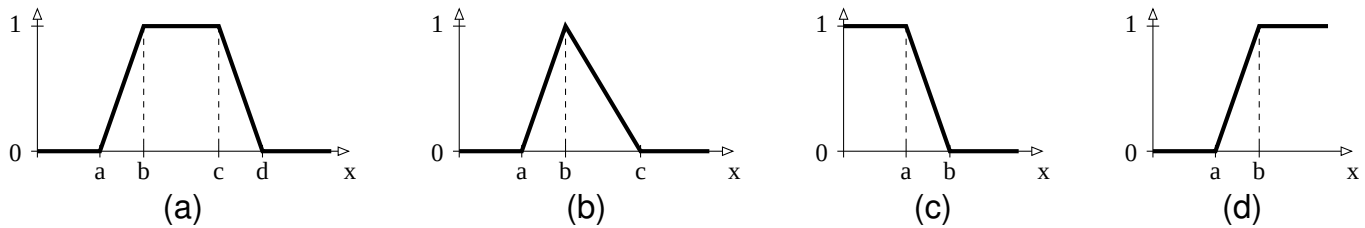
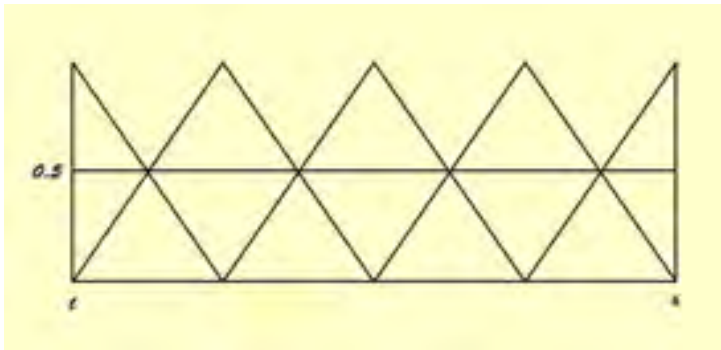


Figure: (a) Trapezoidal function *trz*(a, b, c, d), (b) triangular function *tri*(a, b, c), (c) left shoulder function *ls*(a, b), and (d) right shoulder function *rs*(a, b).

- For instance, AlarmSystem $\wedge$ (price *ls*(18000, 22000))
- Usually, domain is partitioned uniformly into 5 or 7 parts (there are methods to learn them)



Fuzzy Concrete Domains (cont.)

Definition (The language $\mathcal{P}(\mathcal{N})$)

Let $\mathcal{A}$ be a set of propositional atoms, and $\mathcal{F}$ a set of pairs $\langle f, D_f \rangle$ each made of a feature name and an associated concrete domain D_f, and let k be a value in D_f. Then the following formulae are in $\mathcal{P}(\mathcal{N})$:

1. every atom $A \in \mathcal{A}$ is a formula
2. if $\langle f, D_f \rangle \in \mathcal{F}$, $k \in D_f$, and $c \in \{\geq, \leq, =\}$ then $(f\ c\ k)$ is a formula
3. if $\langle f, D_f \rangle \in \mathcal{F}$ and c is of the form $ls(a, b)$, $rs(a, b)$, $tri(a, b, c)$, $trz(a, b, c, d)$ then $(f\ c)$ is a formula
4. if ψ and φ are formulae and $n \in [0, 1]$ then so are $\neg\psi$, $\psi \wedge \varphi$, $\psi \vee \varphi$, $\psi \rightarrow \varphi$. We use $\psi \leftrightarrow \varphi$ in place of $(\psi \rightarrow \varphi) \wedge (\varphi \rightarrow \psi)$,
5. if $\psi_1, \ldots, \psi_n$ are formulae, then $w_1 \cdot \psi_1 + \ldots + w_n \cdot \psi_n$ is a formula, where $w_i \in [0, 1]$ and $\sum_i w_i \leq 1$
6. if ψ is a formula and $n \in [0, 1]$ then $\psi : n$ is a formula in $\mathcal{P}(\mathcal{N})$. If n is omitted, then $\psi : 1$ is assumed

Definition (Interpretation and models)

An interpretation $\mathcal{I}$ for $\mathcal{P}(\mathcal{N})$ is a function (denoted as a superscript $\cdot^{\mathcal{I}}$ on its argument) that maps each atom in $\mathcal{A}$ into a truth value $A^{\mathcal{I}} \in [0, 1]$, each feature name f into a value $f^{\mathcal{I}} \in D_f$, and assigns truth values in $[0, 1]$ to formulas as follows:

- for hard constraints, $(f\ c\ k)^{\mathcal{I}} = 1$ iff the relation $f^{\mathcal{I}}\ c\ k$ is true in D_f, $(f\ c\ k)^{\mathcal{I}} = 0$ otherwise
- for soft constraints, $(f\ c)^{\mathcal{I}} = c(f^{\mathcal{I}})$, i.e., the result of evaluating the fuzzy membership function c on the value $f^{\mathcal{I}}$
- $(\neg\psi)^{\mathcal{I}} = \neg\psi^{\mathcal{I}}$, $(\psi \wedge \varphi)^{\mathcal{I}} = \psi^{\mathcal{I}} \wedge \varphi^{\mathcal{I}}$, $(\psi \vee \varphi)^{\mathcal{I}} = \psi^{\mathcal{I}} \vee \varphi^{\mathcal{I}}$, $(\psi \rightarrow \varphi)^{\mathcal{I}} = \psi^{\mathcal{I}} \Rightarrow \varphi^{\mathcal{I}}$ and $(w_1 \cdot \psi_1 + \ldots + w_n \cdot \psi_n)^{\mathcal{I}} = \sum_i w_i \cdot \psi_i^{\mathcal{I}}$
- $\mathcal{I} \models \psi : n$ iff $\psi^{\mathcal{I}} \geq n$.

Example: Matchmaking

- Suppose we have a buyer and a seller (agents)
  - A car seller sells a sedan car
  - A buyer is looking for a second hand passenger car
  - Both the buyer as well as the seller have preferences (restrictions)
  - There is some background knowledge
- The objective is determine “an optimal” (Pareto optimal) agreement among the two

Matchmaking Example: the Background Knowledge

1. A sedan is a passenger car
2. A satellite alarm system is an alarm system
3. The navigator pack is a satellite alarm system with a GPS system
4. The Insurance Plus package is a driver insurance together with a theft insurance
5. The car colours are black or grey

Matchmaking Example: Buyer's preferences

1. He does not want to pay more than 26000 euro (buyer reservation value)
2. He wants an alarm system in the car and he is completely satisfied with paying no more than 23000 euro, but he can go up to 26000 euro to a lesser degree of satisfaction
3. He wants a driver insurance and either a theft insurance or a fire insurance
4. He wants air conditioning and the external colour should be either black or grey
5. Preferably the price is no more than 22000 euro, but he can go up to 24000 euro to a lesser degree of satisfaction
6. The kilometer warranty is preferrably at least 140000, but he may go down to 160000 to a lesser degree of satisfaction
7. The weights of the preferences 2-6 are, (0.1, 0.2, 0.1, 0.2, 0.4). The higher the value the more important is the preference

Matchmaking Example: Seller's preferences

1. He wants to sell no less than 24000 euro (seller reservation value)
2. If there is an navigator pack system in the car then he is completely satisfied with selling no less than 26000 euro, but he can go down to 24000 euro to a lesser degree of satisfaction
3. Preferably the seller sells the Insurance Plus package
4. The kilometer warranty is preferrably at most 150000, but he may go up to 170000 to a lesser degree of satisfaction
5. If the color is black then the car has air conditioning
6. The weights of the preferences 2-5 are, (0.3, 0.1, 0.4, 0.2). The higher the value the more important is the preference

Matchmaking Example: Encoding

$$\mathcal{T} = \begin{cases} \text{Sedan} \rightarrow \text{PassengerCar} \\ \text{ExternalColorBlack} \rightarrow \neg\text{ExternalColorGray} \\ \text{SatelliteAlarm} \rightarrow \text{AlarmSystem} \\ \text{InsurancePlus} \leftrightarrow \text{DriverInsurance} \wedge \text{TheftInsurance} \\ \text{NavigatorPack} \leftrightarrow \text{SatelliteAlarm} \wedge \text{GPS\_system} \end{cases}$$

Buyer's request:

β = PassengerCar $\wedge$ price $\leq$ 26000
β_1 = AlarmSystem $\Rightarrow$ (price , ls(23000, 26000))
β_2 = DriverInsurance $\wedge$ (TheftInsurance $\vee$ FireInsurance)
β_3 = AirConditioning $\wedge$ (ExternalColorBlack $\vee$ ExternalColorGray)
β_4 = (price , ls(22000, 24000))
β_5 = (km_warranty , rs(140000, 160000))
$\mathcal{B} = 0.1 \cdot \beta_1 + 0.2 \cdot \beta_2 + 0.1 \cdot \beta_3 + 0.2 \cdot \beta_4 + 0.2 \cdot \beta_5$

Seller's request:

σ = Sedan $\wedge$ price $\geq$ 24000
σ_1 = NavigatorPack $\wedge$ (price , rs(24000, 26000))
σ_2 = InsurancePlus
σ_3 = (km_warranty , ls(150000, 170000))
σ_4 = ExternalColorBlack $\wedge$ AirConditioning
$\mathcal{S} = 0.3 \cdot \sigma_1 + 0.1 \cdot \sigma_2 + 0.4 \cdot \sigma_3 + 0.2 \cdot \sigma_4$

Let

$$KB = \mathcal{T} \cup \{\beta, \sigma\} \cup \{\text{buy} \leftrightarrow \mathcal{B}, \text{sell} \leftrightarrow \mathcal{S}\}$$

Pareto optimal solution:

$$bsd(KB, \text{buy} \wedge_{\Pi} \text{sell}) = 0.651$$

In particular, the final agreement is:

$\text{Sedan}^{\bar{\mathcal{I}}} = 1.0$, $\text{PassengerCar}^{\bar{\mathcal{I}}} = 1.0$, $\text{InsurancePlus}^{\bar{\mathcal{I}}} = 1.0$, $\text{AlarmSystem}^{\bar{\mathcal{I}}} = 1.0$,
$\text{DriverInsurance}^{\bar{\mathcal{I}}} = 1.0$, $\text{AirConditioning}^{\bar{\mathcal{I}}} = 1.0$, $\text{NavigatorPack}^{\bar{\mathcal{I}}} = 1.0$,
$(\text{km\_warranty ls}(150000, 170000))^{\bar{\mathcal{I}}} = 0.5$, i.e. $\text{km\_warranty}^{\bar{\mathcal{I}}} = 160000$,
$(\text{price}, \text{ls}(23000, 26000))^{\bar{\mathcal{I}}} = 0.33$, i.e. $\text{price}^{\bar{\mathcal{I}}} = 24000$,
$\text{TheftInsurance}^{\bar{\mathcal{I}}} = 1.0$, $\text{FireInsurance}^{\bar{\mathcal{I}}} = 1.0$, $\text{ExternalColorBlack}^{\bar{\mathcal{I}}} = 1.0$, $\text{ExternalColorGray}^{\bar{\mathcal{I}}} = 0.0$.

Example: (Fuzzy) Multi-Criteria Decision Making

- We have to decide which offer to choose for the development of a Public School
- There are 3 offers (Alternatives), which have been evaluated by an expert according to 3 Criteria
  - Cost, DeliveryTime, Quality

Preliminaries: MCDM Basics

- Alternatives A_i: different choices of action available to the decision maker to be ranked
- Decision criteria C_j: different dimensions from which the alternatives can be viewed and evaluated
- Decision weights w_j: importance of a criteria
- Performance weights a_{ij}: performance of alternative w.r.t. a decision criteria

		Criteria				
		w_1	w_2	·	·	w_m
Alternatives		C_1	C_2	·	·	C_m
x_1	A_1	a_{11}	a_{12}	·	·	a_{1m}
x_2	A_2	a_{21}	a_{22}	·	·	a_{2m}
·	·	·	·	·	·	·
·	·	·	·	·	·	·
x_n	A_n	a_{n1}	a_{n2}	·	·	a_{nm}

(1)

- Final ranking value x_i:

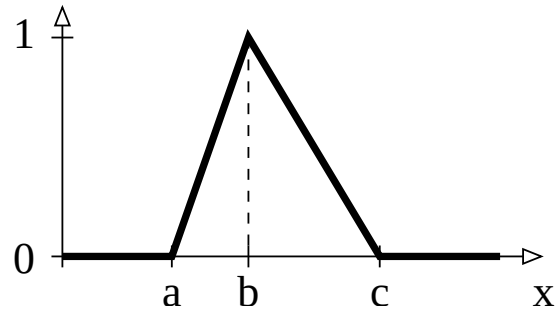
$$x_i = \sum_{j=1}^{m} a_{ij} w_j$$

- Optimal alternative A^*:

$$A^* = \arg\max_{A_i} x_i$$

Preliminaries: Fuzzy MCDM Basics

- Principal difference: weights w_i and performance a_{ij} are fuzzy numbers
- Fuzzy number $\tilde{n}$: fuzzy set over relas with triangular membership function $tri(a, b, c)$. Intended being an approximation of the number b



- Any real value n is seen as the fuzzy number $tri(n, n, n)$
- Arithmetic operators $+, -, \cdot$ and $\div$ are extended to fuzzy numbers
  - For $* \in \{+, \cdot\}$, $\tilde{n}_1 * \tilde{n}_2 = tri(a_1 * a_2, b_1 * b_2, c_1 * c_2)$
  - For $* \in \{-, \div\}$, $\tilde{n}_1 * \tilde{n}_2 = tri(a_1 * c_2, b_1 * b_2, c_1 * a_2)$
- Final ranking value x_i: fuzzy number

$$\tilde{x}_i = \sum_{j=1}^{m} \tilde{a}_{ij} \cdot \tilde{w}_j$$

- Optimal alternative A^*:

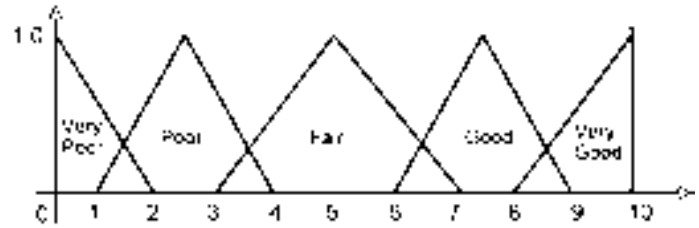
$$A^* = \arg\max_{A_i} x_i$$

using some fuzzy number ranking method. E.g., Best Non-Fuzzy Performance (BNP): $(a + b + c)/3$

Example: (Fuzzy) Multi-Criteria Decision Making

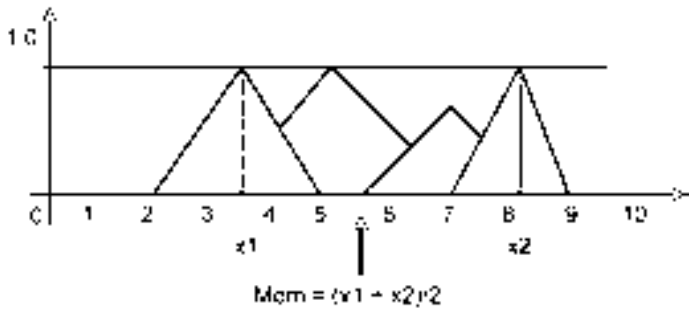
- We have to decide which offer to choose for the development of a Public School
- There are 3 offers (Alternatives), which have been evaluated by an expert according to 3 Criteria
- The importance of alternative A_i against criteria C_j is $a_{ij} \in \{\text{VeryPoor}, \text{Poor}, \text{Fair}, \text{Good}, \text{VeryGood}\}$
- The importance of the criteria is weighted $w_{ij} \in [0, 1]$, $\sum_i w_{ij} = 1$ ($w_1 = 0.3$, $w_2 = 0.2$, $w_3 = 0.5$)

Offer	Cost 0.3	DeliveryTime 0.2	Quality 0.5
A_1	VeryPoor	Fair	Good
A_2	Good	VeryGood	Poor
A_3	Fair	Fair	Poor



$$KB = \{A_1, A_2, A_3\} \text{ where}$$
$$A_i \leftrightarrow w_1 \cdot (\text{hasScore } a_{i1}) + w_2 \cdot (\text{hasScore } a_{i2}) + w_3 \cdot (\text{hasScore } a_{i3})$$

- The Final Rank Value, $rn(KB, A_i)$, of alternative A_i is defined as the Middle of Maxima (MOM) de-fuzzification method



$$rn(KB, A_1) = 0.75,\ rn(KB, A_2) = 0.25, rn(KB, A_3) = 0.375$$

- So, we may choose offer A_1

Note: Computing Middle of Maxima (MOM)

- Middle of Maxima (MOM) = (Largest of Maxima (LOM) + Smallest of Maxima (SOM))/2
- LOM is implemented in the following steps
  1. Compute $n = bsd(A_i, KB)$
  2. Maximise the value of the (internal) variable representing the value of hasScore, i.e. the variable x_{hasScore}, given $KB \cup \{A_i \geq n\}$
- SOM is implemented in the following steps
  1. Compute $n = bsd(A_i, KB)$
  2. Minimise the variable x_{hasScore}, given $KB \cup \{A_i \geq n\}$
- MOM is implemented in the following steps
  1. Compute $n = bsd(A_i, KB)$
  2. Maximise the variable x_{hasScore}, given $KB \cup \{A_i \geq n\}$
  3. Minimise the variable x_{hasScore}, given $KB \cup \{A_i \geq n\}$
  4. Take the average of the two values obtained from the two maximisation and minimisation problems

Predicate Fuzzy Logics Basics

- Formulae: First-Order Logic formulae, *terms* are either variables or constants
  - we may introduce functions symbols as well, with crisp semantics (but uninteresting), or we need to discuss also fuzzy equality (which we leave out here)
- Truth space is $[0, 1]$
- Formulae have a a degree of truth in $[0, 1]$
- Interpretation: is a mapping $\mathcal{I} : \mathit{Atoms} \to [0, 1]$
- Interpretations are extended to formulae as follows:

$$\begin{aligned}
\mathcal{I}(\neg\phi) &= \mathcal{I}(\phi) \to 0 \\
\mathcal{I}(\phi \wedge \psi) &= \mathcal{I}(\phi) \wedge \mathcal{I}(\psi) \\
\mathcal{I}(\phi \to \psi) &= \mathcal{I}(\phi) \to \mathcal{I}(\psi) \\
\mathcal{I}(\exists x \phi) &= \sup_{c \in \Delta^{\mathcal{I}}} \mathcal{I}_x^c(\phi) \\
\mathcal{I}(\forall x \phi) &= \inf_{c \in \Delta^{\mathcal{I}}} \mathcal{I}_x^c(\phi)
\end{aligned}$$

  where $\mathcal{I}_x^c$ is as $\mathcal{I}$, except that variable x is mapped into individual c
- Definitions of $\mathcal{I} \models \phi : n$, $\mathcal{I} \models \mathcal{T}$, $\mathcal{T} \models \phi : n$, $\mathit{bed}(KB, \phi)$ and $\mathit{bsd}(KB, \phi)$ are as for the propositional case

- $\neg\forall \mathbf{x}\, \varphi(\mathbf{x}) \equiv \exists \mathbf{x}\, \neg\varphi(\mathbf{x})$ true in Ł, but does not hold for logic G and Π
- $(\neg\forall x\, p(x)) \wedge (\neg\exists x\, \neg p(x))$ has no classical model. In Gödel logic it has no finite model, but has an infinite model: for integer $n \geq 1$, let $\mathcal{I}$ such that $\mathcal{I}(p(n)) = 1/n$

$$\begin{aligned} \mathcal{I}(\forall x\, p(x)) &= \inf_n 1/n = 0 \\ \mathcal{I}(\exists x\, \neg p(x)) &= \sup_n \neg 1/n = \sup 0 = 0 \end{aligned}$$

- Note: If $\mathcal{I} \models \exists x\, \phi(x)$ then not necessarily there is $c \in \Delta^{\mathcal{I}}$ such that $\mathcal{I} \models \phi(c)$.

$$\begin{aligned} \Delta_{\mathcal{I}} &= \{n \mid \text{ integer } n \geq 1\} \\ \mathcal{I}(p(n)) &= 1 - 1/n < 1, \text{ for all } n \\ \mathcal{I}(\exists x\, p(x)) &= \sup_n 1 - 1/n = 1 \end{aligned}$$

- Witnessed formula: $\exists x\, \phi(x)$ is witnessed in $\mathcal{I}$ iff there is $c \in \Delta_{\mathcal{I}}$ such that $\mathcal{I}(\exists x\, \phi(x)) = \mathcal{I}(\phi(c))$ (similarly for $\forall x\, \phi(x)$)
- Witnessed interpretation: $\mathcal{I}$ witnessed if all quantified formulae are witnessed in $\mathcal{I}$

Proposition

In Ł, ϕ is satisfiable iff there is a witnessed model of ϕ.

The proposition does not hold for logic G and Π

Fuzzy Concrete Domains

- Allows us to deal with concepts such as young, cheap, cold, etc.
- Fuzzy membership functions: usually of the form

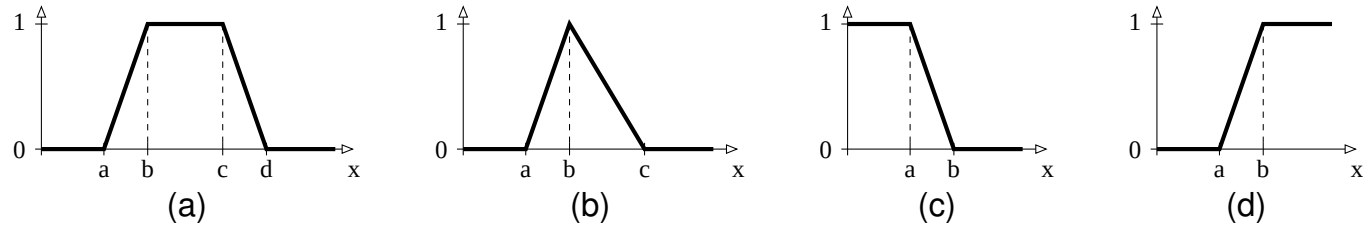


Figure: (a) Trapezoidal function $trz(a, b, c, d)$, (b) triangular function $tri(a, b, c)$, (c) left shoulder function $ls(a, b)$, and (d) right shoulder function $rs(a, b)$.

- Works similarly as for propositional case:
  - We consider a concrete domain over rational numbers with concrete predicates:

$$\geq (x, y), \leq (x, y), = (x, y), ls(a, b)(x), rs(a, b)(x), tri(a, b, c)(x), trz(a, b, c, d)(x)$$

  - Formulae may contain concrete predicates as atom
  - There are variables and constants for rational numbers
  - Formula example

$$\exists r.\text{AlarmSystem}(avs) \wedge \text{price}(avs, r) \wedge ls(350, 500)(r) : n$$

- The semantics is an obvious extension of the fuzzy FOL case

Fuzzy RDFS Basics

Crisp RDFS Syntax

- Pairwise disjoint alphabets
  - **U** (RDFS URI references)
  - **B** (Blank nodes)
  - **L** (Literals)
- For simplicity we will denote unions of these sets simply concatenating their names
- We call elements in **UBL** terms (denoted t)
- We call elements in **B** variables (denoted x)

- RDFS triple (or RDFS atom):

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

  - s is the subject
  - p is the predicate
  - o is the object
- Example:

$$(airplane, has, enginefault)$$

ρdf (restricted RDFS)

- ρdf (read rho-df, the ρ from restricted rdfs)
- ρdf is defined as the following subset of the RDFS vocabulary:

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

- (p, sp, q)
  - property p is a *sub property* of property q
- (c, sc, d)
  - class 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, range, c)
  - *range* of property p is c

RDF Semantics

- RDF interpretation $\mathcal{I}$ over a vocabulary V is a tuple

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle \ ,$$

  where
  - $\Delta_R, \Delta_P, \Delta_C, \Delta_L$ are the interpretations domains of $\mathcal{I}$
  - $P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}}$ are the interpretation functions of $\mathcal{I}$

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle$$

1. Δ_R is a nonempty set of resources, called the domain or universe of $\mathcal{I}$;
2. Δ_P is a set of property names (not necessarily disjoint from Δ_R);
3. $\Delta_C \subseteq \Delta_R$ is a distinguished subset of Δ_R identifying if a resource denotes a class of resources;
4. $\Delta_L \subseteq \Delta_R$, the set of literal values, Δ_L contains all plain literals in $\mathbf{L} \cap V$;
5. $P[\![\cdot]\!]$ maps each property name $p \in \Delta_P$ into a subset $P[\![p]\!] \subseteq \Delta_R \times \Delta_R$, i.e. assigns an extension to each property name;
6. $C[\![\cdot]\!]$ maps each class $c \in \Delta_C$ into a subset $C[\![c]\!] \subseteq \Delta_R$, i.e. assigns a set of resources to every resource denoting a class;
7. $\cdot^{\mathcal{I}}$ maps each $t \in \mathbf{UL} \cap V$ into a value $t^{\mathcal{I}} \in \Delta_R \cup \Delta_P$, i.e. assigns a resource or a property name to each element of **UL** in V, and such that $\cdot^{\mathcal{I}}$ is the identity for plain literals and assigns an element in Δ_R to elements in **L**;
8. $\cdot^{\mathcal{I}}$ maps each variable $x \in \mathbf{B}$ into a value $x^{\mathcal{I}} \in \Delta_R$, i.e. assigns a resource to each variable in **B**.

Models

Intuitively,

- A ground triple (s, p, o) in an RDF graph G will be true under the interpretation $\mathcal{I}$ if
  - p is interpreted as a property name
  - s and o are interpreted as resources
  - the interpretation of the pair (s, o) belongs to the extension of the property assigned to p
- Blank nodes, i.e. variables, work as existential variables: a triple $((x, p, o)$ with $x \in \mathbf{B}$ would be true under $\mathcal{I}$ if
  - there exists a resource s such that (s, p, o) is true under $\mathcal{I}$

Models (cont.)

Let G be a graph over ρdf.

- An interpretation $\mathcal{I}$ is a model of G under ρdf, denoted $\mathcal{I} \models G$, iff
  - $\mathcal{I}$ is an interpretation over the vocabulary $\rho\text{df} \cup \mathit{universe}(G)$
  - $\mathcal{I}$ satisfies the following conditions:

Simple:

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

Subproperty:

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

Models (cont.)

Subclass:

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

Typing I:

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

Typing II:

1. For each e $\in \rho$df, $\text{e}^{\mathcal{I}} \in \Delta_P$
2. if $(p, c) \in P[\![\text{dom}^{\mathcal{I}}]\!]$ then $p \in \Delta_P$ and $c \in \Delta_C$
3. if $(p, c) \in P[\![\text{range}^{\mathcal{I}}]\!]$ then $p \in \Delta_P$ and $c \in \Delta_C$
4. if $(x, c) \in P[\![\text{type}^{\mathcal{I}}]\!]$ then $c \in \Delta_C$

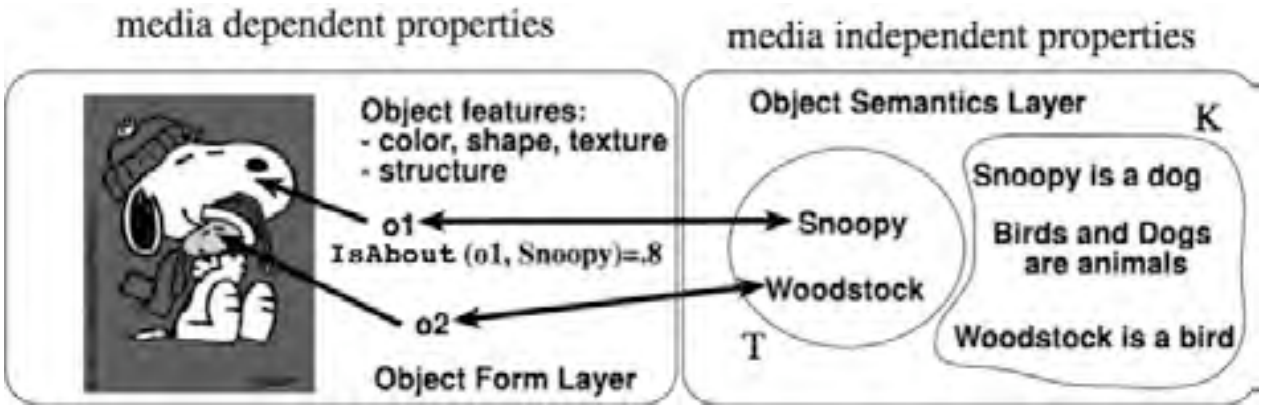
Entailment

- G entails H under ρdf, denoted $G \models H$, iff
  - every model under ρdf of G is also a model under ρdf of H
- Note: often $P[\![\text{sp}^{\mathcal{I}}]\!]$ (resp. $C[\![\text{sc}^{\mathcal{I}}]\!]$) is also *reflexive* over Δ_P (resp. Δ_C)
  - We omit this requirement and, thus, do NOT support inferences such as

$$\begin{aligned} G &\models (a, \text{sp}, a) \\ G &\models (a, \text{sc}, a) \end{aligned}$$

  which anyway are of marginal interest

Example



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

Deduction System for RDF

1. Simple:

$$\frac{G}{G'} \text{ for } G' \subseteq G$$

2. Subproperty:

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

3. Subclass:

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

4. Typing:

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

5. Implicit Typing:

$$(a) \quad \frac{(A,\text{dom},B),(C,\text{sp},A),(X,C,Y)}{(X,\text{type},B)} \qquad (b) \quad \frac{(A,\text{range},B),(C,\text{sp},A),(X,C,Y)}{(Y,\text{type},B)}$$

RDFS Query Answering

- We assume that a RDF graph G is *ground* and *closed*, i.e., G is closed under the application of the rules (2)-(5)
- Conjunctive query: is a Datalog-like rule of the form

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

where

  - $n \geq 1$, $\tau_1, \ldots, t_n$ are triples
  - $\mathbf{x}$ is a vector of variables occurring in $\tau_1, \ldots, \tau_n$, called the *distinguished variables*
  - $\mathbf{y}$ are so-called *non-distinguished variables* and are distinct from the variables in $\mathbf{x}$
  - each variable occurring in τ_i is either a distinguished variable or a non-distinguished variable

- If clear from the context, we may omit the exitential quantification $\exists \mathbf{y}$
- For instance, the query

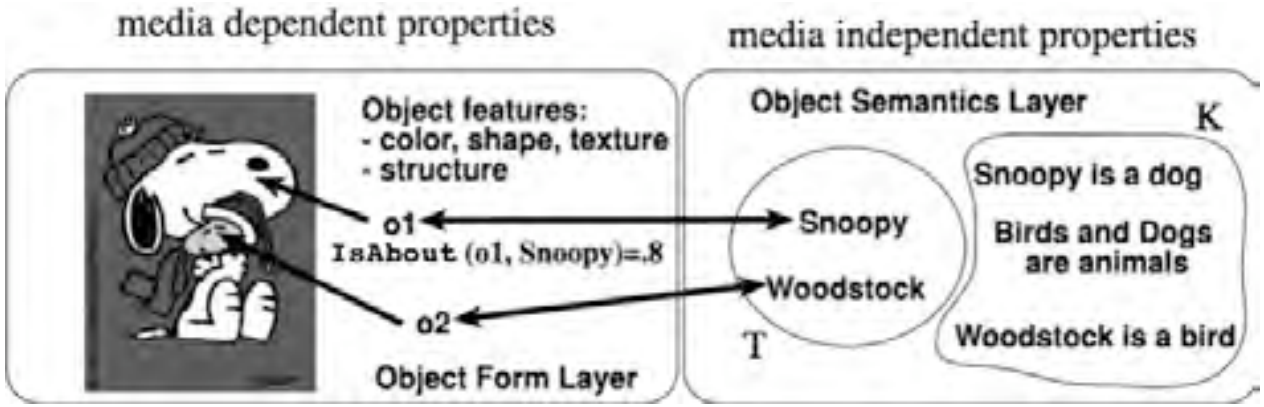
$$q(x, y) \leftarrow (x, \mathit{creates}, y), (x, \text{type}, \mathit{Flemish}), (x, \mathit{paints}, y), (y, \mathit{exhibited}, \mathit{Uffizi})$$

has intended meaning to retrieve all the artifacts x created by Flemish artists y, being exhibited at Uffizi Gallery

RDF Query Answering (cont.)

- A simple query answering procedure 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

Example



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

Consider the query

$$q(x) \quad \leftarrow \quad (x, \mathit{IsAbout}, y), (y, \text{type}, \mathit{Animal})$$

Then

$$\mathit{answer}(G, q) \quad = \quad \{o1, o2\}$$

Fuzzy RDFS

- Triples may have attached a degree in $[0, 1]$: for $n \in [0, 1]$

$$(subject, predicate, object) : n$$

- Meaning: the degree of truth of the statement is at least n
- For instance,

$$(o1, IsAbout, snoopy) : 0.8$$

Fuzzy RDF Syntax

- Fuzzy RDF triple (or Fuzzy RDF atom):

$$\tau : n \in (\mathbf{UBL} \times \mathbf{U} \times \mathbf{UBL}) \times [0, 1]$$

  - $s \in \mathbf{UBL}$ is the subject
  - $p \in \mathbf{U}$ is the predicate
  - $o \in \mathbf{UBL}$ is the object
  - $n \in (0, 1]$ is the degree of truth

- Example:

$$(\text{audiTT}, \text{type}, \text{SportCar}) : 0.8$$

Fuzzy RDF Semantics

- Fix a t-norm $\otimes$
- Fuzzy RDF interpretation $\mathcal{I}$ over a vocabulary V is a tuple

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle \ ,$$

where

- $\Delta_R, \Delta_P, \Delta_C, \Delta_L$ are the interpretations domains of $\mathcal{I}$
- $P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}}$ are the interpretation functions of $\mathcal{I}$

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle$$

1. Δ_R is a nonempty set of resources, called the domain or universe of $\mathcal{I}$;
2. Δ_P is a set of property names (not necessarily disjoint from Δ_R);
3. $\Delta_C \subseteq \Delta_R$ is a distinguished subset of Δ_R identifying if a resource denotes a class of resources;
4. $\Delta_L \subseteq \Delta_R$, the set of literal values, Δ_L contains all plain literals in $\mathbf{L} \cap V$;
5. $P[\![\cdot]\!]$ maps each property name $p \in \Delta_P$ into a function $P[\![p]\!] : \Delta_R \times \Delta_R \to [0, 1]$, i.e. assigns a degree to each pair of resources, denoting the degree of being the pair an instance of the property p;
6. $C[\![\cdot]\!]$ maps each class $c \in \Delta_C$ into a function $C[\![c]\!] : \Delta_R \to [0, 1]$, i.e. assigns a degree to every resource, denoting the degree of being the resource an instance of the class c;
7. $\cdot^{\mathcal{I}}$ maps each $t \in \mathbf{UL} \cap V$ into a value $t^{\mathcal{I}} \in \Delta_R \cup \Delta_P$, i.e. assigns a resource or a property name to each element of **UL** in V, and such that $\cdot^{\mathcal{I}}$ is the identity for plain literals and assigns an element in Δ_R to elements in **L**;
8. $\cdot^{\mathcal{I}}$ maps each variable $x \in \mathbf{B}$ into a value $x^{\mathcal{I}} \in \Delta_R$, i.e. assigns a resource to each variable in **B**.

Models

Let G be a graph over ρdf.

- An interpretation $\mathcal{I}$ is a model of G under ρdf, denoted $\mathcal{I} \models G$, iff
  - $\mathcal{I}$ is an interpretation over the vocabulary $\rho\text{df} \cup \mathit{universe}(G)$
  - $\mathcal{I}$ satisfies the following conditions:

Simple:

1. for each (s, p, o): $n \in G$, $p^{\mathcal{I}} \in \Delta_P$ and $P[\![p^{\mathcal{I}}]\!](s^{\mathcal{I}}, o^{\mathcal{I}}) \geq n$;

Subproperty:

1. $P[\![\text{sp}^{\mathcal{I}}]\!](p, q) \otimes P[\![\text{sp}^{\mathcal{I}}]\!](q, r) \leq P[\![\text{sp}^{\mathcal{I}}]\!](p, r)$;
2. $P[\![p^{\mathcal{I}}]\!](x, y) \otimes P[\![\text{sp}^{\mathcal{I}}]\!](p, q) \leq P[\![q^{\mathcal{I}}]\!](x, y)$;

Models (cont.)

Subclass:

1. $P[\![\mathrm{sc}^{\mathcal{I}}]\!](c,d) \otimes P[\![\mathrm{sc}^{\mathcal{I}}]\!](d,e) \leq P[\![\mathrm{sc}^{\mathcal{I}}]\!](c,e)$;
2. $C[\![c^{\mathcal{I}}]\!](x) \otimes P[\![\mathrm{sc}^{\mathcal{I}}]\!](c,d) \leq P[\![d^{\mathcal{I}}]\!](x)$;

Typing I:

1. $C[\![c]\!](x) = P[\![\mathrm{type}^{\mathcal{I}}]\!](x,c)$;
2. $P[\![\mathrm{dom}^{\mathcal{I}}]\!](p,c) \otimes P[\![p]\!](x,y) \leq C[\![c]\!](x)$;
3. $P[\![\mathrm{range}^{\mathcal{I}}]\!](p,c) \otimes P[\![p]\!](x,y) \leq C[\![c]\!](y)$;

Typing II:

1. For each e $\in \rho$df, $\mathrm{e}^{\mathcal{I}} \in \Delta_P$;
2. $P[\![\mathrm{sp}^{\mathcal{I}}]\!](p,q)$ is defined only for $p,q \in \Delta_P$;
3. $C[\![\mathrm{sc}^{\mathcal{I}}]\!](c,d)$ is defined only for $c,d \in \Delta_C$;
4. $P[\![\mathrm{dom}^{\mathcal{I}}]\!](p,c)$ is defined only for $p \in \Delta_P$ and $c \in \Delta_C$;
5. $P[\![\mathrm{range}^{\mathcal{I}}]\!](p,c)$ is defined only for $p \in \Delta_P$ and $c \in \Delta_C$;
6. $P[\![\mathrm{type}^{\mathcal{I}}]\!](s,c)$ is defined only for $c \in \Delta_C$.

Example & Model

$G = \{(\mathit{audiTT}, \text{type}, \mathit{SportsCar}) : 0.8, (\mathit{SportsCar}, \text{sc}, \mathit{PassengerCar}) : 0.9\}$ t-norm: Product

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle$$

$$\begin{aligned}
\Delta_R &= \{\mathit{audiTT}, \mathit{SportsCar}, \mathit{PassengerCar}\} \\
\Delta_P &= \{\text{type}, \text{sc}\} \\
\Delta_C &= \{\mathit{SportsCar}, \mathit{PassengerCar}\} \\
P[\![\text{type}]\!] &= \{\langle \mathit{audiTT}, \mathit{SportsCar} \rangle : 0.8, \langle \mathit{audiTT}, \mathit{PassengerCar} \rangle : 0.72\} \\
P[\![\text{sc}]\!] &= \{\langle \mathit{SportsCar}, \mathit{PassengerCar} \rangle : 0.9\} \\
C[\![\mathit{SportsCar}]\!] &= \{\mathit{audiTT} : 0.8\} \\
C[\![\mathit{PassengerCar}]\!] &= \{\mathit{audiTT} : 0.72\} \\
t^{\mathcal{I}} &= t \text{ for all } t \in \mathbf{UL}
\end{aligned}$$

$\mathcal{I} \models G$ $\mathcal{I}$ is a model of G

Deduction System for fuzzy RDFS

- Very simple:

$$(AG) \quad \frac{\tau_1 : n_1, ..., \tau_k : n_k, \{\tau_1, \ldots \tau_k\} \vdash_{\mathsf{RDFS}} \tau}{\tau : \bigotimes_i \lambda_i}$$

Deduction System for fuzzy RDFS

1. Simple:

$$\frac{G}{G'} \text{ for } G' \subseteq G$$

2. Subproperty:

$$(a) \quad \frac{(A, \text{sp}, B) : n, (B, \text{sp}, C) : m}{(A, \text{sp}, C) : n \otimes m} \qquad (b) \quad \frac{(A, \text{sp}, B) : n, (X, A, Y) : m}{(X, B, Y) : n \otimes m}$$

3. Subclass:

$$(a) \quad \frac{(A, \text{sc}, B) : n, (B, \text{sc}, C) : m}{(A, \text{sc}, C) : n \otimes m} \qquad (b) \quad \frac{(A, \text{sc}, B) : n, (X, \text{type}, A) : m}{(X, \text{type}, B) : n \otimes m}$$

4. Typing:

$$(a) \quad \frac{(A, \text{dom}, B) : n, (X, A, Y) : m}{(X, \text{type}, B) : n \otimes m} \qquad (b) \quad \frac{(A, \text{range}, B) : n, (X, A, Y) : m}{(Y, \text{type}, B) : n \otimes m}$$

5. Implicit Typing:

$$(a) \quad \frac{(A, \text{dom}, B) : n, (C, \text{sp}, A) : m, (X, C, Y) : r}{(X, \text{type}, B) : n \otimes m \otimes r}$$

$$(b) \quad \frac{(A, \text{range}, B) : n, (C, \text{sp}, A) : m, (X, C, Y) : r}{(Y, \text{type}, B) : n \otimes m \otimes r}$$

Fuzzy RDFS Query Answering

- We assume that a fuzzy RDF graph G is *ground* and *closed*, i.e., G is closed under the application of the rules (2)-(5)
- Conjunctive query: extends a crisp RDF query and is of the form

$$q(\mathbf{x}) : s \leftarrow \exists \mathbf{y}.\tau_1 : s_1, \ldots, \tau_n : s_n, s = f(s_1, \ldots, s_n, p_1(\mathbf{z}_1), \ldots, p_h(\mathbf{z}_h))$$

where additionally

  - $\mathbf{z_i}$ are tuples of terms in **UL** or variables in $\mathbf{x}$ or $\mathbf{y}$;
  - p_j is an n_j-ary *fuzzy predicate* assigning to each n_j-ary tuple $\mathbf{t}_j$ in **UL** a *score* $p_j(\mathbf{t}_j) \in [0, 1]$. Such predicates are called *expensive predicates* as the score is not pre-computed off-line, but is computed on query execution. We require that an n-ary fuzzy predicate p is *safe*, that is, there is not an m-ary fuzzy predicate p' such that $m < n$ and $p = p'$. Informally, all parameters are needed in the definition of p;
  - f is a *scoring* function $f : ([0, 1])^{n+h} \rightarrow [0, 1]$, which combines the scores s_i of the n triples and the h fuzzy predicates into an overall *score* to be assigned to the rule head. We assume that f is *monotone*, that is, for each $\mathbf{v}, \mathbf{v}' \in ([0, 1])^{n+h}$ such that $\mathbf{v} \leq \mathbf{v}'$, it holds $f(\mathbf{v}) \leq f(\mathbf{v}')$, where $(v_1, \ldots, v_{n+h}) \leq (v'_1, \ldots, v'_{n+h})$ iff $v_i \leq v'_i$ for all 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
- If clear from the context, we may omit the exitential quantification $\exists \mathbf{y}$
- We may omit s_i and in that case $s_i = 1$ is assumed
- $s = f(s_1, \ldots, s_n, p_1(\mathbf{z}_1), \ldots, p_h(\mathbf{z}_h))$ is called the *scoring atom*. We may also omit the scoring atom and in that case $s = 1$ is assumed.
- For instance, the query

$$q(x) : s \leftarrow (x, \text{type}, \text{SportCar}) : s_1, (x, \text{hasPrice}, y), s = s_1 \cdot \text{cheap}(y)$$

where e.g. $\text{cheap}(p) = \text{ls}(0, 10000, 12000)$, has intended meaning to retrieve all cheap sports car. Any answer is scored according to the product of being cheap and a sports car

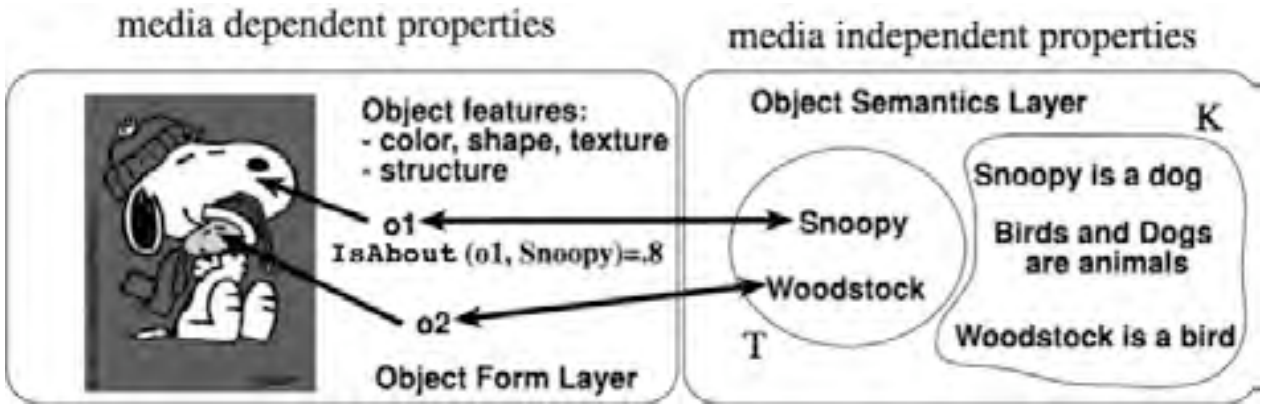
Fuzzy RDF Query Answering (cont.)

Top-k Retrieval: Given a fuzzy graph G, and a query q, retrieve k answers $\mathbf{t}: s$ with maximal scores and rank them in decreasing order relative to the score s, denoted

$$ans_k(G, q) = \text{Top}_k\ answer(G, q)$$

- A simple query answering procedure is the following:
  - Compute the closure of a graph off-line
  - Store the fuzzy RDF triples into a relational database supporting Top-k retrieval (e.g., RankSQL, Postgres)
  - Translate the fuzzy query into a top-k SQL statement
  - Execute the SQL statement over the relational database
  - Few systems exists, e.g. FuzzyRDF, AnQL (http://anql.deri.org/)

Example



$$G = \left\{ \begin{array}{ll} (o1, \mathit{IsAbout}, \mathit{snoopy}) : 0.8 & (o2, \mathit{IsAbout}, \mathit{woodstock}) : 0.9 \\ (\mathit{snoopy}, \text{type}, \mathit{dog}) & (\mathit{woodstock}, \text{type}, \mathit{bird}) \\ (\mathit{Bird}, \text{sc}, \mathit{SmallAnimal}) : 0.7 & (\mathit{Dog}, \text{sc}, \mathit{SmallAnimal}) : 0.4 \\ (\mathit{dog}, \text{sc}, \mathit{Animal}) & (\mathit{bird}, \text{sc}, \mathit{Animal}) \\ (\mathit{SmallAnimal}, \text{sc}, \mathit{Animal}) & \end{array} \right\}$$

Consider the query

$$q(x) : s \quad \leftarrow \quad (x, \mathit{IsAbout}, y) : s_1, (y, \text{type}, \mathit{Animal}) : s_2, s = s_1 \cdot s_2$$

Then (under any t-norm)

$$\mathit{ans}(G, q) \quad = \quad \{o1 : 0.32, o2 : 0.63\}, \quad \mathit{ans}_1(G, q) = \{o2 : 0.63\}$$

Annotation domains & RDFS

- Generalisation of fuzzy RDFS
  - a triple is annotated with a value λ taken from a so-called *annotation domain*, rather than with a value in [0,1]
  - allows to deal with several domains (such as, fuzzy, temporal, provenance) and their combination, in a uniform way
- Time
  - (*umberto*, *workedFor*, *IEI*)
  - true during 1992–2001
- Fuzzyness
  - (*WingateHotel*, *closeTo*, *RR*11 *Venue*)
  - true to some degree
- Provenance
  - (*umberto*, *knows*, *didier*)
  - true in `http://www.straccia.info/foaf.rdf`

- Annotation Domain: idempotent, commutative semi-ring

$$D = \langle L, \oplus, \otimes, \bot, \top \rangle$$

where $\oplus$ is $\top$-annihilating, i.e.

1. $\oplus$ is idempotent, commutative, associative;
2. $\otimes$ is commutative and associative;
3. $\bot \oplus \lambda = \lambda$, $\top \otimes \lambda = \lambda$, $\bot \otimes \lambda = \bot$, and $\top \oplus \lambda = \top$;
4. $\otimes$ is distributive over $\oplus$,
   i.e. $\lambda_1 \otimes (\lambda_2 \oplus \lambda_3) = (\lambda_1 \otimes \lambda_2) \oplus (\lambda_1 \otimes \lambda_3)$;

- Induced partial order:

$$\lambda_1 \preceq \lambda_2 \text{ if and only if } \lambda_1 \oplus \lambda_2 = \lambda_2$$

- Annotated triple: for $\lambda \in L$

$$(s, p, o) : \lambda$$

- For instance,

  $(umberto, workedFor, IEI) : [1992, 2001]$

  $(WingateHotel, closeTo, RR11\,Venue) : 0.8$

  $(umberto, knows, didier) :$ `http://www.straccia.info/foaf.rdf`

Annotation Domains: Examples

Illustration by Example: Time

- An *Annotation Domain* consists of
  - A set L of annotation values
    - e.g. $[1968, 2000]$ and $\{[1968, 2000], [2003, 2004]\}$
  - An order between elements:
    - if $\lambda \preceq \lambda'$, then $\tau : \lambda$ is true to a lesser extent than $\tau' : \lambda'$
    - e.g. $[1968, 2000] \preceq [1952, 2007]$ ($\preceq$ is $\subseteq$)
  - Top and bottom ellements:
    - $\top = [-\infty, +\infty], \bot = \emptyset$
  - "Conjunction" function $\otimes$
    - $[1992, 2001] \otimes [1968, 2000] = [1992, 2000]$ ($\otimes$ is $\cap$)
  - "Combination" function $\oplus$
    - $[1992, 2001] \oplus [1995, 2003] = [1992, 2003]$
    - $[1992, 1996] \oplus [2001, 2009] = \{[1992, 1996], [2001, 2009]\}$

Examples

- Fuzzy: $(\mathit{WingateHotel}, \mathit{closeTo}, \mathit{RR11Venue}) : 0.8$
  - $L = [0, 1]$
  - $\otimes =$ any t-norm
  - $\vee = \max$
- Provenance: $(\mathit{umberto}, \mathit{knows}, \mathit{didier}) : p$
  - $L =$ DNF propositional formulae over URIs
  - $\otimes = \wedge$
  - $\vee = \vee$
- Multiple Domains: our frameworks allows to combine domains

$$(\mathit{CountryXXX}, \text{type}, \mathit{Dangerous}) : \langle [1975, 1983], 0.8, 0.6 \rangle$$

$\mathit{Time} \times \mathit{Fuzzy} \times \mathit{Trust}$

- Inference rule:

$$\frac{\tau_1 : \lambda_1,\ \ldots,\ \tau_k : \lambda_k, \{\tau_1, \ldots, \tau_k\} \vdash_{\mathsf{RDFS}} \tau}{\tau :\ \bigotimes_i \lambda_i}$$

- Annotated conjunctive queries are as fuzzy queries, except that now variables s and s_i range over L in place of $[0, 1]$;
- A query answering procedure is similar as for the fuzzy case: compute the closure, store it on a relation database and transform an annotated CQ into a SQL query
- Computational complexity: same as for crisp RDFS plus the cost of $\otimes$, $\oplus$ and the scoring function f in the body of a query
- A prototype Prolog implementation is available

`http://anql.deri.org/`

Fuzzy OWL Basics

OWL

- OWL Family
  - OWL full is union of OWL syntax and RDF (Undecidable)
  - OWL DL restricted to Description Logics fragment (decidable in NEXPTIME)
  - OWL Lite is “easier to implement” subset of OWL DL (decidable in EXPTIME)
  - OWL 2 new OWL standard (decidable, NEXPTIME-hard)
  - OWL 2 profiles
    - OWL QL (query answering LOGSPACE)
    - OWL EL (classification in polynomial time)
    - OWL RL (EXPTIME, intersection of Description Logics and Logic Programming)
- Semantic layering
  - OWL 2 within Description Logic (DL) fragment
- OWL DL is based on $\mathcal{SHOIN}(D)$
- OWL Lite is based on $\mathcal{SHIF}(D)$
- OWL 2 is based on $\mathcal{SROIQ}(D)$
- OWL QL is based on $DL-Lite$
- OWL EL is based on $\mathcal{EL}$
- OWL RL is a Datalog fragment

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	$\rightarrow$	$\top$	$\mid$	$\top(x)$	
		$\bot$	$\mid$	$\bot(x)$	
		A	$\mid$	$A(x)$	*Human*
		$C \sqcap D$	$\mid$	$C(x) \wedge D(x)$	*Human* $\sqcap$ *Male*
		$C \sqcup D$	$\mid$	$C(x) \vee D(x)$	*Nice* $\sqcup$ *Rich*
		$\neg C$	$\mid$	$\neg C(x)$	$\neg$*Meat*
		$\exists R.C$	$\mid$	$\exists y.R(x, y) \wedge C(y)$	$\exists$*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,

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

Semantics of Additional Constructs

$\mathcal{H}$: Role inclusion axioms, $\mathcal{I} \models R_1 \sqsubseteq R_2$ iff ${R_1}^{\mathcal{I}} \subseteq {R_1}^{\mathcal{I}}$

$\mathcal{N}$: Number restrictions,
$(\geq n\ R)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} : |\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\}| \geq n\}$,
$(\leq n\ R)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} : |\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\}| \leq n\}$

$\mathcal{Q}$: Qualified number restrictions,
$(\geq n\ R.C)^{\mathcal{I}} = \{x \in |\{y \mid \langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}| \geq n\}$,
$(\leq n\ R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} : |\{y \mid \langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}| \leq n\}$

$\mathcal{O}$: Nominals (singleton class), $\{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$

$\mathcal{I}$: Inverse role, $(R^-)^{\mathcal{I}} = \{\langle x, y\rangle \mid \langle y, x\rangle \in R^{\mathcal{I}}\}$

$\mathcal{F}$: Functional role, $I \models \mathit{fun}(f)$ iff $\forall z \forall y \forall z$ if $\langle x, y\rangle \in f^{\mathcal{I}}$ and $\langle x, z\rangle \in f^{\mathcal{I}}$ the $y = z$

$\mathcal{R}_+$: transitive role,
$(R_+)^{\mathcal{I}} = \{\langle x, y\rangle \mid \exists z$ such that $\langle x, z\rangle \in R^{\mathcal{I}} \wedge \langle z, y\rangle \in R^{\mathcal{I}}\}$

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*
  $Minor = Person \sqcap \exists hasAge.\ \leq_{18}$

- Semantics: a clean separation between “object” classes and concrete domains
  - $D = \langle \Delta_D, \Phi_D \rangle$
  - Δ_D is an interpretation domain
  - Φ_D is the set of concrete domain predicates d with a predefined arity n and fixed interpretation $d^D \subseteq \Delta_D^n$
  - Concrete properties: $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta_D$
- Notation: (D). E.g., $\mathcal{ALC}(D)$ is $\mathcal{ALC}$ + concrete domains

DL Knowledge Base

- A DL Knowledge Base is a pair $KB = \langle \mathcal{T}, \mathcal{A} \rangle$, 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 $(a, b){:}R$
    - containing (in) equality Axioms of the form $a = b$ and $a \neq b$
- An interpretation $\mathcal{I}$ is a model of KB, written $\mathcal{I} \models KB$ 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 KB satisfiability? $\mapsto$ Is there some model $\mathcal{I}$ of KB ?
- Is C satisfiability? $\mapsto$ $C^{\mathcal{I}} \neq \emptyset$ for some some model $\mathcal{I}$ of KB ?

Subsumption: structure knowledge, compute taxonomy

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

Equivalence: check if two classes denote same set of instances

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

Instantiation: check if individual a instance of class C

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

Retrieval: retrieve set of individuals that instantiate C

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

Reduction to Satisfiability

Problems are all reducible to KB satisfiability

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

Equivalence: $KB \models C = D$ iff $KB \models C \sqsubseteq D$ and $KB \models D \sqsubseteq C$

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

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

Reasoning in DLs: Basics

- Tableaux algorithm deciding satisfiability
- Try to build a tree-like model $\mathcal{I}$ of the KB
- Decompose concepts C syntactically
  - Apply tableau expansion rules
  - Infer constraints on elements of model
- Tableau rules correspond to constructors in logic ($\sqcap, \sqcup, \ldots$)
  - Some rules are nondeterministic (e.g., $\sqcup, \leq$)
  - In practice, this means search
- Stop when no more rules applicable or clash occurs
  - Clash is an obvious contradiction, e.g., $A(x), \neg A(x)$
- Cycle check (blocking) may be needed for termination

Negation Normal Form (NNF)

- We have to transform concepts into Negation Normal Form: push negation inside using de Morgan' laws

$$
\begin{array}{rcl}
\neg\top & \mapsto & \bot \\
\neg\,\bot & \mapsto & \top \\
\neg\neg C & \mapsto & C \\
\neg(C_1 \sqcap C_2) & \mapsto & \neg C_1 \sqcup \neg C_2 \\
\neg(C_1 \sqcup C_2) & \mapsto & \neg C_1 \sqcap \neg C_2
\end{array}
$$

and

$$
\begin{array}{rcl}
\neg(\exists R.C) & \mapsto & \forall R.\neg C \\
\neg(\forall R.C) & \mapsto & \exists R.\neg C
\end{array}
$$

Completion-Forest

- This is a forest of trees, where
  - each node x is labelled with a set $\mathcal{L}(x)$ of concepts
  - each edge $\langle x, y \rangle$ is labelled with $\mathcal{L}(\langle x, y \rangle) = \{R\}$ for some role R (edges correspond to relationships between pairs of individuals)
- The forest is initialized with
  - a root node a, labelled $\mathcal{L}(x) = \emptyset$ for each individual a occurring in the KB
  - an edge $\langle a, b \rangle$ labelled $\mathcal{L}(\langle a, b \rangle) = \{R\}$ for each (a, b):R occurring in the KB
- Then, for each a:C occurring in the KB, set $\mathcal{L}(a) \rightarrow \mathcal{L}(a) \cup \{C\}$
- The algorithm expands the tree either by extending $\mathcal{L}(x)$ for some node x or by adding new leaf nodes.
- Edges are added when expanding $\exists R.C$
- A completion-forest contains a clash if, for a node x, $\{C, \neg C\} \subseteq \mathcal{L}(x)$
- If nodes x and y are connected by an edge$\langle x, y \rangle$, then y is called a successor of x and x is called a predecessor of y. Ancestor is the transitive closure of predecessor.
- A node y is called an R-successor of a node x if y is a successor of x and $\mathcal{L}(\langle x, y \rangle) = \{R\}$.
- The algorithm returns “satisfiable" if rules can be applied s.t. they yield a clash-free, complete (no more rules can be applied) completion forest

$\mathcal{ALC}$ Tableau rules without GCI's

Rule		Description
$(\sqcap)$	if 1.	$C_1 \sqcap C_2 \in \mathcal{L}(x)$ and
	2.	$\{C_1, C_2\} \not\subseteq \mathcal{L}(x)$
	then	$\mathcal{L}(x) \to \mathcal{L}(x) \cup \{C_1, C_2\}$
$(\sqcup)$	if 1.	$C_1 \sqcup C_2 \in \mathcal{L}(x)$ and
	2.	$\{C_1, C_2\} \cap \mathcal{L}(x) = \emptyset$
	then	$\mathcal{L}(x) \to \mathcal{L}(x) \cup \{C\}$ for some $C \in \{C_1, C_2\}$
$(\exists)$	if 1.	$\exists R.C \in \mathcal{L}(x)$ and
	2.	x has no R-successor y with $C \in \mathcal{L}(y)$
	then	create a new node y with $\mathcal{L}(\langle x, y\rangle) = \{R\}$ and $\mathcal{L}(y) = \{C\}$
$(\forall)$	if 1.	$\forall R.C \in \mathcal{L}(x)$ and
	2.	x has an R-successor y with $C \notin \mathcal{L}(y)$
	then	$\mathcal{L}(y) \to \mathcal{L}(y) \cup \{C\}$

Fuzzy DLs

The semantics is an immediate consequence of the First-Order-Logic translation of DLs expressions

Interpretation:

$$\begin{array}{lcl} \mathcal{I} & = & \Delta^{\mathcal{I}} \\ C^{\mathcal{I}} & : & \Delta^{\mathcal{I}} \to [0,1] \\ R^{\mathcal{I}} & : & \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0,1] \end{array} \qquad \begin{array}{lcl} \otimes & = & \text{t-norm} \\ \oplus & = & \text{s-norm} \\ \neg & = & \text{negation} \\ \Rightarrow & = & \text{implication} \end{array}$$

Concepts:

Syntax			*Semantics*		
C, D	$\longrightarrow$	$\top \mid$	$\top^{\mathcal{I}}(x)$	$=$	1
		$\bot \mid$	$\bot^{\mathcal{I}}(x)$	$=$	0
		$A \mid$	$A^{\mathcal{I}}(x)$	$\in$	$[0,1]$
		$C \sqcap D \mid$	$(C_1 \sqcap C_2)^{\mathcal{I}}(x)$	$=$	${C_1}^{\mathcal{I}}(x) \otimes {C_2}^{\mathcal{I}}(x)$
		$C \sqcup D \mid$	$(C_1 \sqcup C_2)^{\mathcal{I}}(x)$	$=$	${C_1}^{\mathcal{I}}(x) \oplus {C_2}^{\mathcal{I}}(x)$
		$\neg C \mid$	$(\neg C)^{\mathcal{I}}(x)$	$=$	$\neg C^{\mathcal{I}}(x)$
		$\exists R.C \mid$	$(\exists R.C)^{\mathcal{I}}(x)$	$=$	$\sup_{y \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)$
		$\forall R.C$	$(\forall R.C)^{\mathcal{I}}(x)$	$=$	$\inf_{y \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)\}$

Assertions: $a{:}C : r$, $\mathcal{I} \models a{:}C : r$ iff $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq r$ (similarly for roles)

- individual a is instance of concept C at least to degree r, $r \in [0,1] \cap \mathbb{Q}$

Inclusion axioms: $C \sqsubseteq D : r$,

- $\mathcal{I} \models C \sqsubseteq D : r$ iff $\inf_{x \in \Delta^{\mathcal{I}}} C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x) \geq r$

Main Inference Problems

Graded entailment: Check if DL axiom α is entailed to degree at least r

- $KB \models \alpha : r$?

BED: Best Entailment Degree problem

- $bed(KB, \alpha) = \sup\{r \mid KB \models \alpha : r\}$

BSD: Best Satisfiability Degree problem

- $bsd(KB, C) = \sup_{\mathcal{I} \models KB}\{C^{\mathcal{I}}(a^{\mathcal{I}})\}$, for new individual a

Top-k retrieval: Retrieve the top-k individuals that instantiate C w.r.t. best truth value bound

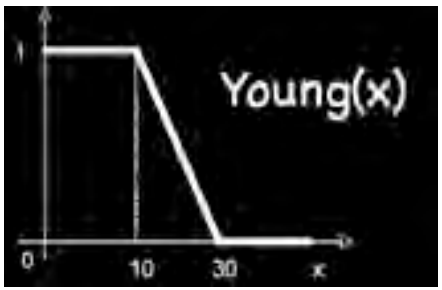
- $ans_k(KB, C) = Top_k\{\langle a, r\rangle \mid r = bed(KB, a{:}C)\}$

Towards fuzzy OWL 2 and its Profiles

- Recall that OWL 2 relates to $\mathcal{SROIQ}(D)$
- We need to extend the semantics of fuzzy $\mathcal{ALC}$ to fuzzy $\mathcal{SROIQ}(D)$
- Additionally, we add
  - modifiers (e.g., *very*)
  - concrete fuzzy concepts (e.g., *Young*)
  - both additions have explicit membership functions
  - other extensions:
    - aggregation functions: weighted sum, OWA, fuzzy integrals
    - fuzzy rough sets, fuzzy spatial, fuzzy numbers

Concrete fuzzy concepts

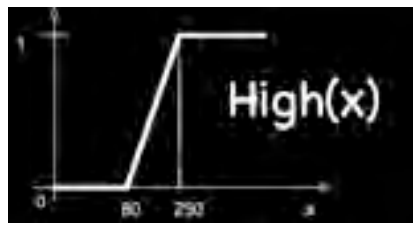
- E.g., *Small*, *Young*, *High*, *etc*. with explicit membership function
- Use the idea of concrete domains:
  - $D = \langle \Delta_\beta , \Phi_\beta \rangle$
  - Δ_β is an interpretation domain
  - Φ_β is the set of concrete fuzzy domain predicates d with a predefined arity $n = 1, 2$ and fixed interpretation $d^\beta : \Delta_\beta^n \rightarrow [0, 1]$
  - For instance,



$$\begin{array}{rcl} \textit{Minor} & = & \textit{Person} \sqcap \exists \textit{hasAge}. \leq_{18} \\ \textit{YoungPerson} & = & \textit{Person} \sqcap \exists \textit{hasAge}. \textit{Young} \\ & & \textit{functional}(\textit{hasAge}) \end{array}$$

Modifiers

- *Very*, *moreOrLess*, *slightly*, etc.
- Apply to fuzzy sets to change their membership function
  - $very(x) = x^2$
  - $slightly(x) = \sqrt{x}$
- For instance,



$$SportsCar = Car \sqcap \exists speed.very(High)$$

Fuzzy $\mathcal{SHOIN}(D)$

Concepts:

Syntax			Semantics
C, D	$\longrightarrow$	$\top$	$\top(x)$
		$\bot$	$\bot(x)$
		A	$A(x)$
		$(C \sqcap D)$	$C_1(x) \otimes C_2(x)$
		$(C \sqcup D)$	$C_1(x) \oplus C_2(x)$
		$(\neg C)$	$\neg C(x)$
		$(\exists R.C)$	$\exists x\ R(x,y) \otimes C(y)$
		$(\forall R.C)$	$\forall x\ R(x,y) \Rightarrow C(y)$
		$\{a\}$	$x = a$
		$(\geq n\ R)$	$\exists y_1, \ldots, y_n . \otimes_{i=1}^{n} R(x,y_i) \otimes \otimes_{1 \leq i < j \leq n} y_i \neq y_j$
		$(\leq n\ R)$	$\forall y_1, \ldots, y_{n+1} . \otimes_{i=1}^{n+1} R(x,y_i) \Rightarrow \oplus_{1 \leq i < j \leq n+1} y_i = y_j$
		FCC	$\mu_{FCC}(x)$
		$M(C)$	$\mu_M(C(x))$
		$\sum_i w_i \cdot C_i$	$w_1 \cdot C_1(x) + \cdots w_n \cdot C_n(x) \quad (\sum_i w_i = 1)$
R	$\longrightarrow$	P	$P(x,y)$
		P^-	$P(y,x)$

Assertions:

Syntax			Semantics
α	$\longrightarrow$	$a{:}C : r$	$C(a) \geq r$
		$(a,b){:}R : r$	$R(a,b) \geq r$

Axioms:

Syntax			Semantics
τ	$\longrightarrow$	$C \sqsubseteq D : r$	$\forall x.C(x) \Rightarrow D(x) \geq r$
		$fun(R)$	$\forall x \forall y \forall z\ R(x,y) \wedge R(x,z) \Rightarrow y = z$
		$trans(R)$	$(\exists z\ R(x,z) \wedge R(z,y)) \Rightarrow R(x,y)$

Example (Graded Entailment)

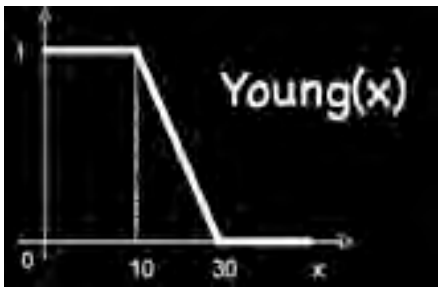
		
audi_tt	*mg*	*ferrari_enzo*

Car	*speed*
audi_tt	243
mg	$\leq$ 170
ferrari_enzo	$\geq$ 350

$$SportsCar \quad = \quad Car \sqcap \exists hasSpeed.very(High)$$

$$\begin{array}{lll} KB & \models & ferrari\_enzo{:}SportsCar : 1 \\ KB & \models & audi\_tt{:}SportsCar : 0.92 \\ KB & \models & mg{:}\neg SportsCar : 0.72 \end{array}$$

Example (Graded Subsumption)



$$\begin{aligned} \mathit{Minor} &= \mathit{Person} \sqcap \exists \mathit{hasAge}.\leq_{18} \\ \mathit{YoungPerson} &= \mathit{Person} \sqcap \exists \mathit{hasAge}.\mathit{Young} \end{aligned}$$

$$\mathit{KB} \models \mathit{Minor} \sqsubseteq \mathit{YoungPerson}: 0.6$$

Note: without an explicit membership function of *Young*, this inference cannot be drawn

Example (Simplified Negotiation)

- 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 €
- classical DLs: the problem relies on the crisp conditions on price
- more fine grained approach: to consider prices as fuzzy sets (as usual in negotiation)
  - seller may consider optimal to sell above 31500 €, but can go down to 30500 €
  - the buyer prefers to spend less than 30000 €, but can go up to 32000 €

    $AudiTT \quad = \quad SportsCar \sqcap \exists hasPrice.R(x; 30500, 31500)$

    $Query \quad = \quad SportsCar \sqcap \exists hasPrice.L(x; 30000, 32000)$
  - highest degree to which the concept
    $C = AudiTT \sqcap Query$
    is satisfiable is 0.75 (the possibility that the Audi TT and the query matches is 0.75)
  - the car may be sold at 31250 €

Reasoning in Fuzzy $\mathcal{ALC}$, under Zadeh Semantics

- Applies technique based on Mixed Integer Programming (MILP) for fuzzy propositional logic to $\mathcal{ALC}$ calculus
- For each concept assertion α of the form $a{:}C$, we use variable x_α, which holds the degree of truth of α
- It can be shown that

$$
\begin{array}{lcl}
bed(KB, (a,b){:}R) & = & bed(KB \cup \{b{:}B : 1\}, a{:}\exists R.B) \\
bed(KB, C \sqsubseteq D) & = & \min x \text{ such that } KB \cup \{b{:}C \sqcap \neg D : 1 - x\} \text{ satisfiable} \\
bed(KB, a{:}C) & = & \min x \text{ such that } KB \cup \{a{:}\neg C : 1 - x\} \text{ satisfiable} \\
bsd(KB, C) & = & \min -x \text{ such that } KB \cup \{b{:}C : x\} \text{ satisfiable}
\end{array}
$$

where b is a new individual and B is a new concept

Satisfiability Testing

- The notion of completion forest $\mathcal{F}$ is similar to the case of $\mathcal{ALC}$
  - $\mathcal{F}$ contains a root node a_i for each individual a_i occurring in $\mathcal{A}$
  - $\mathcal{F}$ contains an edge $\langle a, b \rangle$ for each (a, b):R: $n \in \mathcal{A}$
  - for each a:C: $n \in \mathcal{A}$, we add both C to $\mathcal{L}(a)$ and $x_{a:C} \geq n$ to $\mathcal{C}_{\mathcal{F}}$
  - for each (a, b):R: $n \in \mathcal{A}$, we add both R to $\mathcal{L}(\langle a, b \rangle)$ and $x_{(a,b):R} \geq n$ to $\mathcal{C}_{\mathcal{F}}$
- The notion of blocking is as for crisp $\mathcal{ALC}$
- $\mathcal{F}$ is then expanded by repeatedly applying the rules described below
- The completion-forest is complete when none of the rules are applicable
- Then, the bMILP problem on $\mathcal{C}_{\mathcal{F}}$ is solved

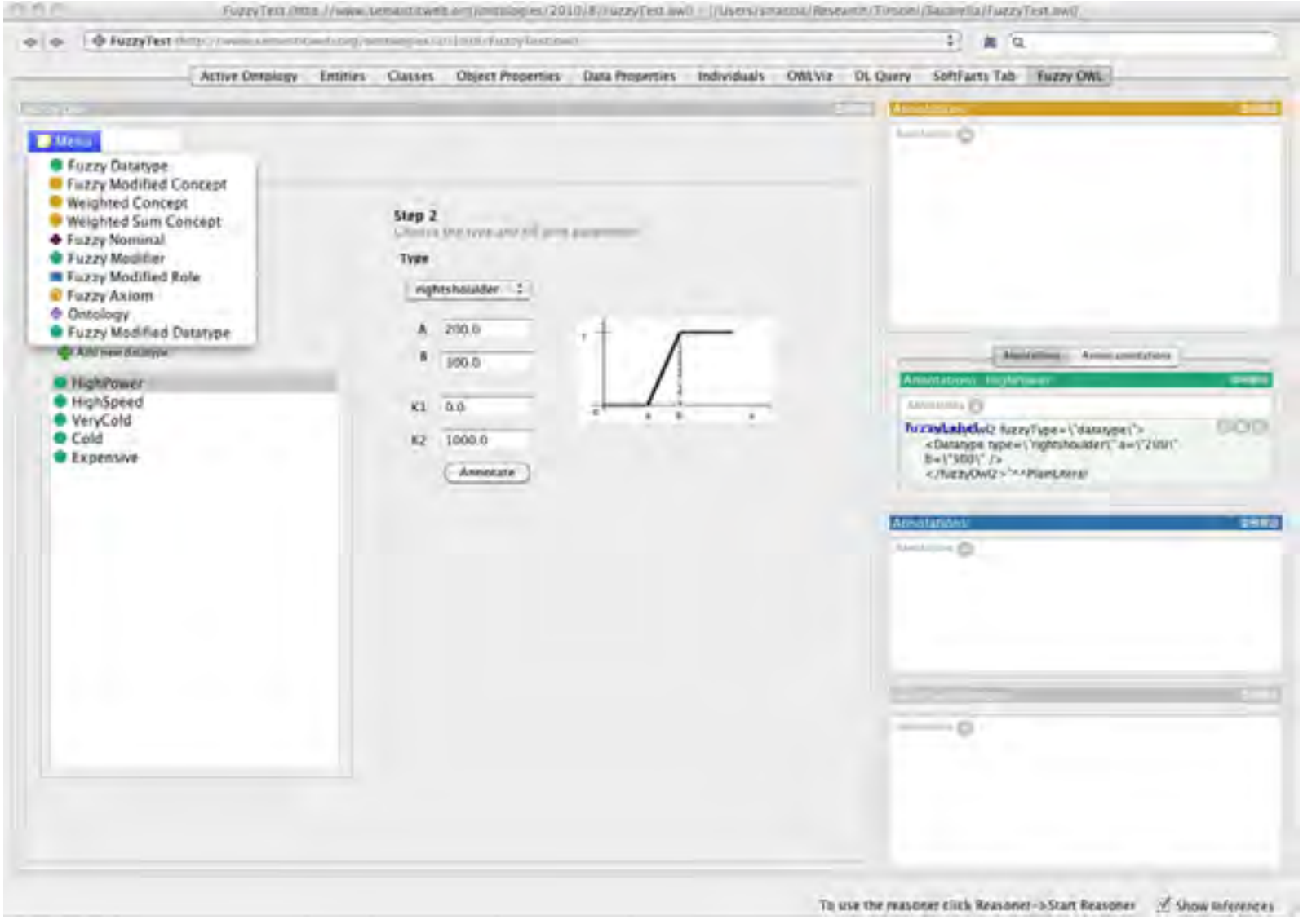
Fuzzy $\mathcal{ALC}$ Tableau rules with GCIs (Zadeh semantics)

Rule		Description
(var)		For variable $x_{v:C}$ add $x_{v:C} \in [0,1]$ to $\mathcal{C}_{\mathcal{F}}$. For variable $x_{(v,w):R}$, add $x_{(v,w):R} \in [0,1]$ to $\mathcal{C}_{\mathcal{F}}$
$(\bar{A})$	if	$\neg A \in \mathcal{L}(v)$ then add $x_{v:A} = 1 - x_{v:\neg A}$ to $\mathcal{C}_{\mathcal{F}}$
$(\bot)$	If	$\bot \in \mathcal{L}(v)$ then add $x_{v:\bot} = 0$ to $\mathcal{C}_{\mathcal{F}}$
$(\top)$	If	$\top \in \mathcal{L}(v)$ then add $x_{v:\top} = 1$ to $\mathcal{C}_{\mathcal{F}}$
$(\sqcap)$	if	$C_1 \sqcap C_2 \in \mathcal{L}(v)$, v is not indirectly blocked
	then	$\mathcal{L}(v) \rightarrow \mathcal{L}(v) \cup \{C_1, C_2\}$, and add $x_{v:C_1} \otimes x_{v:C_2} \geq x_{v:C_1 \sqcap C_2}$ to $\mathcal{C}_{\mathcal{F}}$
$(\sqcup)$	if	$C_1 \sqcup C_2 \in \mathcal{L}(v)$, v is not indirectly blocked
	then	$\mathcal{L}(v) \rightarrow \mathcal{L}(v) \cup \{C_1, C_2\}$, and add $x_{v:C_1} \oplus x_{v:C_2} \geq x_{v:C_1 \sqcup C_2}$ to $\mathcal{C}_{\mathcal{F}}$
$(\forall)$	if	$\forall R.C \in \mathcal{L}(v)$, v is not indirectly blocked
	then	$\mathcal{L}(w) \rightarrow \mathcal{L}(w) \cup \{C\}$, and add $x_{w:C} \geq x_{v:\forall R.C} \otimes x_{(v,w):R}$ to $\mathcal{C}_{\mathcal{F}}$
$(\exists)$	if	$\exists R.C \in \mathcal{L}(v)$, v is not blocked
	then	create new node w with $\mathcal{L}(\langle v, w \rangle) = \{R\}$ and $\mathcal{L}(w) = \{C\}$, and add $x_{w:C} \otimes x_{(v,w):R} \geq x_{v:\exists R.C}$ to $\mathcal{C}_{\mathcal{F}}$
$(\sqsubseteq)$	if	$C \sqsubseteq D : n \in \mathcal{T}$, v is not indirectly blocked
	then	$\mathcal{L}(v) \rightarrow \mathcal{L}(v) \cup \{C, D\}$, and add $x_{v:D} \geq x_{v:C} \otimes n$ to $\mathcal{C}_{\mathcal{F}}$

- As fuzzy DL extensions involving
  - concrete domains
  - modifiers

  are combinations of linear functions, they can be translated into MILP, MIQCP, and MINLP equations
- Therefore, the algorithm can be extended to fuzzy DLs with concrete domains
- For a fuzzy DL system, see fuzzyDL, Fire, DeLorean
- Systems supporting top-k retrieval: based on DL-Lite/DLR-Lite
  - DLMedia (Ontology mediated Multimedia data retrieval)
  - Softfacts (Ontology mediated Database data retrieval)

- Protege plug-in to encode Fuzzy OWL ontologies using OWL 2 Editor Protege exists

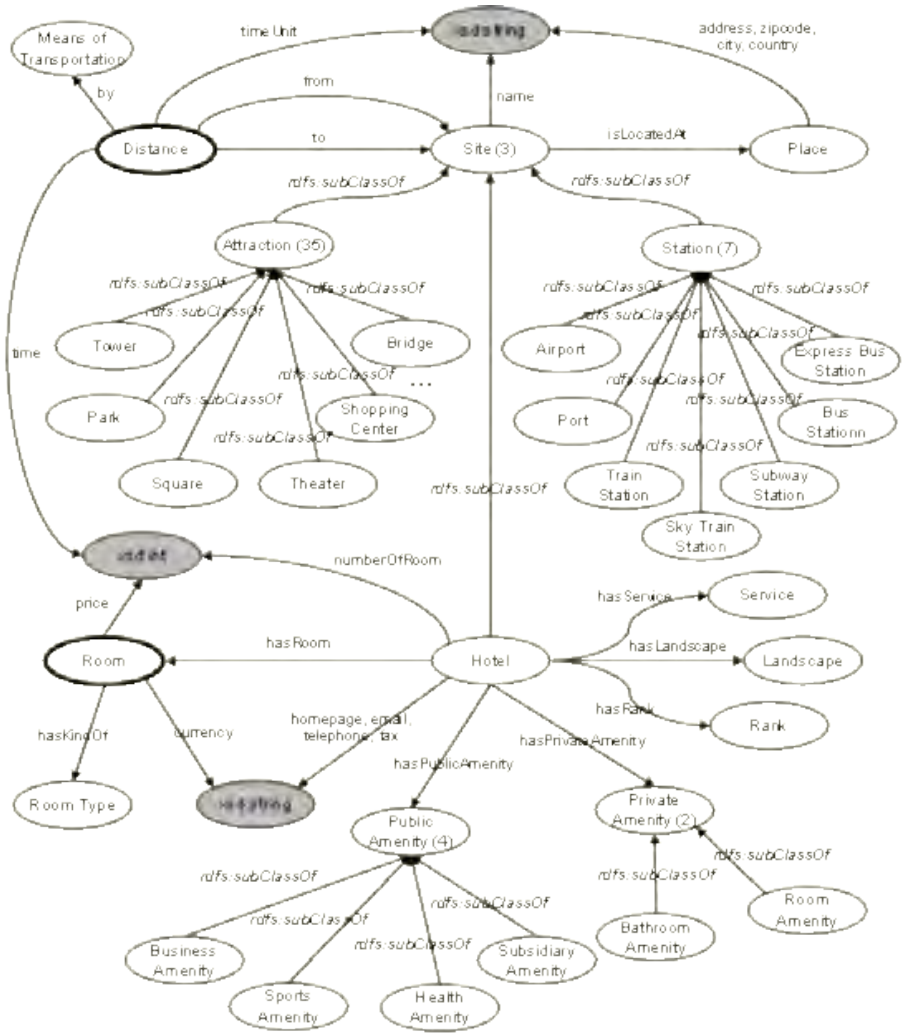


- Computational complexity of satisfiability problem:
  - EXPTIME for $\mathcal{ALC}$ with GCIs w.r.t. $\{0, \frac{1}{n}, \ldots, \frac{n-1}{n}, 1\}$
  - EXPTIME for $\mathcal{ALC}$ with GCIs w.r.t. $[0, 1]$ under Zadeh semantics
  - UNDECIDABLE (!!) for $\mathcal{ALC}$ with GCIs w.r.t. $[0, 1]$ under Product or Łukasiewicz semantics
  - For $\mathcal{EL}$ and DL-Lite family, same complexity as for the crisp case

Novel Application: Learning fuzzy GCIs from data

- A FOIL-like algorithm to learn fuzzy GCIs from data
- Example: Hotel ontology

  $Park \sqsubseteq Attraction$ $Tower \sqsubseteq Attraction$
  $Attraction \sqsubseteq Site$ $Hotel \sqsubseteq Site$



Example: Learning

- $H = \mathit{GoodHotel}$
- $\mathcal{E}^+ = \{\mathit{GoodHotel}(h1)[0.6], \mathit{GoodHotel}(h2)[0.8]\}$
- $\mathcal{E}^- = \{\mathit{GoodHotel}(h3)[0.4]\}$.
- $r_0 : \top \sqsubseteq \mathit{GoodHotel}$
  $r_1 : \mathit{Hotel} \sqsubseteq \mathit{GoodHotel}$
  $r_2 : \mathit{Hotel} \sqcap \exists \mathit{cheapPrice}.\top \sqsubseteq \mathit{GoodHotel}$
  $r_3 : \mathit{Hotel} \sqcap \exists \mathit{cheapPrice}.\top \sqcap \exists \mathit{closeTo}.\mathit{Attraction} \sqsubseteq \mathit{GoodHotel}$
  $r_4 : \mathit{Hotel} \sqcap \exists \mathit{cheapPrice}.\top \sqcap \exists \mathit{closeTo}.\mathit{Park} \sqsubseteq \mathit{GoodHotel}$
  $r_5 : \mathit{Hotel} \sqcap \exists \mathit{cheapPrice}.\top \sqcap \exists \mathit{closeTo}.\mathit{Tower} \sqsubseteq \mathit{GoodHotel}$
- Consequence: r_5 becomes part of $\mathcal{H}$

Annotation domains & OWL

- For OWL 2, it it is like for RDFS, but annotation domain has to be a complete lattice
  - satisfiability problem is inherited from crisp variant if lattice is finite, else UNDECIDABLE (even for $\mathcal{ALC}$ with GCIs)
- Exception for OWL profiles OWL EL, OWL QL and OWL RL: annotation domains may be as for RDFS
  - the complexity is inherited from their crisp variants, plus complexity of domain operators

Fuzzy RIF Basics

Crisp RIF

- RIF is a family of rule languages
  - RIF-Core: this corresponds to Datalog
  - RIF-BLD, Basic Logic Dialect: this corresponds to definite Horn rules with equality and a standard first-order semantics, and subsumes RIF-Core
  - RIF-PRD, Production Rule Dialect: subsumes RIF-Core.
    - Production rules have an if part, or condition, and a then part, or action. The “condition” is as usual, but the “then” part contains actions. An action can assert facts, modify facts, retract facts, and have other side-effects

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 $\wedge, \vee, 0, 1$
  For instance,
  $$has\_father(x, y) \quad \leftarrow \quad has\_parent(x, y) \wedge Male(y)$$

- Extensional database (EDB): set of facts
- Intentional database (IDB): set of rules
- Logic Program $\mathcal{P}$:
  - $\mathcal{P} = EDB \cup IDB$
  - 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

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

- 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 $I : B_{\mathcal{P}} \rightarrow \{0, 1\}$.
- 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

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

LP Query Answering

- Query: is a rule of the form

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

- If $\mathcal{P} \models q(\mathbf{c})$ then $\mathbf{c}$ is called an answer to q
- The answer set of q w.r.t. $\mathcal{P}$ is defined as

$$ans(\mathcal{P}, q) = \{\mathbf{c} \mid \mathcal{P} \models q(\mathbf{c})\}$$

- Efficient query answering algorithms exists

Fuzzy LPs Basics

- We consider fuzzy LPs, which extends classical LPs, where
  - Truth space is $[0, 1]$
  - Interpretation is a mapping $I : B_{\mathcal{P}} \rightarrow [0, 1]$
  - Generalized LP rules are of the form

$$R(\mathbf{x}) \leftarrow \exists \mathbf{y}.f(R_1(\mathbf{z}_1), \ldots, R_k(\mathbf{z}_k), p_1(\mathbf{z}'_1), \ldots, p_h(\mathbf{z}'_h))$$

    - Meaning of rules: "take the truth-values of all $R_i(\mathbf{z}_i), p_j(\mathbf{z}'_j)$, combine them using the truth combination function f, and assign the result to $R(\mathbf{x})$"
  - Facts: ground expressions of the form $R(\mathbf{c}) : n$
    - Meaning of facts: "the degree of truth of $R(\mathbf{c})$ is at least n"
  - Fuzzy LP: a set of facts (extensional database) and a set of rules (intentional database). No extensional relation may occur in the head of a rule

- Rules:

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

1. $\mathbf{x}$ are the *distinguished variables*;
2. s is the *score variable*, taking values in $[0, 1]$;
3. $\mathbf{y}$ are existentially quantified variables, called *non-distinguished variables*;
4. $\varphi(\mathbf{x}, \mathbf{y})$ is $f(\mathbf{R}(\mathbf{z}), \mathbf{p}(\mathbf{z}'))$, where $\mathbf{R}$ is a vector of predicates R_i and $\mathbf{p}$ is a vector of fuzzy predicates p_j;
5. $\mathbf{z}, \mathbf{z}'$ are tuples of constants in *KB* or variables in $\mathbf{x}$ or $\mathbf{y}$;
6. p_j is an n_j-ary *fuzzy predicate* assigning to each n_j-ary tuple $\mathbf{c}_j$ the *score* $p_j(\mathbf{c}_j) \in [0, 1]$;
7. f is a monotone *scoring* function $f : [0, 1]^{k+h} \rightarrow [0, 1]$, which combines the scores of the h fuzzy predicates $p_j(\mathbf{c}_j)$ with the k scores $R_i(\mathbf{c}_i)$

Semantics of fuzzy LPs

- $\mathcal{P}^*$ is constructed as follows (as for the classical case):
  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

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

- 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 $I : B_\mathcal{P} \rightarrow [0, 1]$.
- Model $I \models \mathcal{P}$ iff for all $r \in \mathcal{P}^*$ $I \models r$, where $I \models A \leftarrow \varphi$ iff $I(\varphi) \leq I(A)$
- Note:

$$I(f(R_1(\mathbf{c}_1), \ldots, R_k(\mathbf{c}_k), p_1(\mathbf{c}'_1), \ldots, p_h(\mathbf{c}'_h))) = f(I(R_1(\mathbf{c}_1)), \ldots, I(R_k(\mathbf{c}_k)), p_1(\mathbf{c}'_1), \ldots, p_h(\mathbf{c}'_h)))$$

Fuzzy LP Query Answering

- 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

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

- Entailment: for a ground expression $q(\mathbf{c})\colon s$, $s \in [0,1]$

$$\mathcal{P} \models q(\mathbf{c})\colon s \text{ iff least model of } \mathcal{P} \text{ satisfies } I(q(\mathbf{c})) \geq s$$

- We say that s is *tight* iff $s = \sup\{s' \mid \mathcal{P} \models q(\mathbf{c})\colon s'\}$
- If $\mathcal{P} \models q(\mathbf{c})\colon s$ and s is tight then $\mathbf{c}\colon s$ is called an *answer* to q
- The answer set of q w.r.t. $\mathcal{P}$ is defined as

$$ans(\mathcal{P}, q) = \{\mathbf{c}\colon s \mid \mathcal{P} \models q(\mathbf{c})\colon s,\ s \text{ is tight}\}$$

Top-k Retrieval: Given a fuzzy LP $\mathcal{P}$, and a query q, retrieve k answers $\mathbf{c}\colon s$ with maximal scores and rank them in decreasing order relative to the score s, denoted

$$ans_k(\mathcal{P}, q) = \text{Top}_k\ ans(\mathcal{P}, q)\ .$$

- Fuzzy LPs may be tricky:

$$\begin{aligned} A &: 0 \\ A &\leftarrow (A+1)/2 \end{aligned}$$

- In the minimal model the truth of A is 1 (requires infinitely many $T_{\mathcal{P}}$ iterations)!
- There are several ways to avoid this pathological behavior:
  - We may consider $L = \{0, \frac{1}{n}, \frac{2}{n} \ldots, \frac{n-1}{n}, 1\}$, n natural number, e.g. $n = 100$
  - In $A \leftarrow f(B_1, \ldots, B_n)$, f is bounded, i.e. $f(x_1, \ldots, x_n) \leq x_i$

Example: Soft shopping agent

- I may represent my preferences in Logic Programming with the rules

$$
\begin{aligned}
Pref_1(x,p) &\leftarrow HasPrice(x,p) \wedge LS(10000,14000)(p)\\
Pref_2(x) &\leftarrow HasKM(x,k) \wedge LS(13000,17000)(k)\\
Buy(x,p) &\leftarrow 0.7 \cdot Pref_1(x,p) + 0.3 \cdot Pref_2(x)
\end{aligned}
$$

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

General top-down query procedure for Many-valued LPs

- Idea: use theory of fixed-point computation of equational systems over truth space (complete lattice or complete partial order)
- Assign a variable x_i to an atom $A_i \in B_{\mathcal{P}}$
- Map a rule $A \leftarrow f(A_1, \ldots, A_n) \in \mathcal{P}^*$ into the equation $x_A = f(x_{A_1}, \ldots, x_{A_n})$
- A LP $\mathcal{P}$ is thus mapped into the equational system

$$\begin{cases} x_1 & = & f_1(x_{1_1}, \ldots, x_{1_{a_1}}) \\ & \vdots & \\ x_n & = & f_n(x_{n_1}, \ldots, x_{n_{an}}) \end{cases}$$

- f_i is monotone and, thus, the system has least fixed-point, which is the limit of

$$\begin{array}{lcl} \mathbf{y}_0 & = & \mathbf{0} \\ \mathbf{y}_{i+1} & = & \mathbf{f}(\mathbf{y}_i)\ . \end{array}$$

  where $\mathbf{f} = \langle f_1, \ldots, f_n \rangle$ and $\mathbf{f}(\mathbf{x}) = \langle f_1(x_1), \ldots, f_n(x_n) \rangle$
- The least-fixed point is the least model of $\mathcal{P}$
- Consequence: If top-down procedure exists for equational systems then it works for fuzzy LPs too!

Procedure *Solve*$(\mathcal{S}, Q)$
Input: monotonic system $\mathcal{S} = \langle \mathcal{L}, V, \mathbf{f} \rangle$, where $Q \subseteq V$ is the set of query variables;
Output: A set $B \subseteq V$, with $Q \subseteq B$ such that the mapping v equals lfp(f) on B.

1. $A := Q$, $dg := Q$, $in := \emptyset$, **for all** $x \in V$ **do** $v(x) = 0$, $exp(x) = 0$
2. **while** $A \neq \emptyset$ **do**
3. **select** $x_i \in A$, $A := A \setminus \{x_i\}$, $dg := dg \cup s(x_i)$
4. $r := f_i(v(x_{i_1}), ..., v(x_{i_{a_i}}))$
5. **if** $r \succ v(x_i)$ **then** $v(x_i) := r$, $A := A \cup (p(x_i) \cap dg)$ **fi**
6. **if not** $exp(x_i)$ **then** $exp(x_i) = 1$, $A := A \cup (s(x_i) \setminus in)$, $in := in \cup s(x_i)$ **fi**

od

For $q(\mathbf{x}) \leftarrow \phi \in \mathcal{P}$, with $\mathrm{s}(q)$ we denote the set of *sons* of q w.r.t. r, i.e. the set of intentional predicate symbols occurring in ϕ. With $\mathrm{p}(q)$ we denote the set of *parents* of q, i.e. the set $\mathrm{p}(q) = \{p_i : q \in \mathrm{s}(p_i, r)\}$ (the set of predicate symbols directly depending on q).

- The top-down procedure can be extended to
  - fuzzy Normal Logic Programs (Logic programs with non-monotone negation)
  - Many-valued Normal Logic Programs under Any-world Assumption
  - Logic Programs, without requiring the grounding of the program
- Other approaches for top-down methods for monotone fuzzy LPs:
- Magics sets like methods: yet to investigate ...
- There are also extensions to Fuzzy Disjunctive Logic Programs with or without default negation

Top-k retrieval in LPs

- If the database contains a huge amount of facts, a brute force approach fails:
  - one cannot anymore compute the score of all tuples, rank all of them and only then return the top-k
- Better solutions exists for restricted fuzzy LP languages: Datalog + restriction on the score combination functions appearing in the body

Basic Idea

- We do not compute all answers, but determine answers incrementally
- At each step i, from the tuples seen so far in the database, we compute a threshold δ
- The threshold δ has the property that any successively retrieved answer will have a score $s \leq \delta$
- Therefore, we can stop as soon as we have gathered k answers above δ, because any successively computed answer will have a score below δ

Example

Logic Program $\mathcal{P}$ is

$$q(x) \leftarrow p(x)$$
$$p(x) \leftarrow \min(r_1(x, y), r_2(y, z))$$

RecordID	r_1			r_2		
1	a	b	1.0	m	h	0.95
2	c	d	0.9	m	j	0.85
3	e	f	0.8	f	k	0.75
4	l	m	0.7	m	n	0.65
5	o	p	0.6	p	q	0.55
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

What is

$$Top_1(\mathcal{P}, q) = Top_1\{\langle c, s\rangle \mid \mathcal{P} \models q(c, s)\} \ ?$$

$$q(x) \leftarrow p(x)$$
$$p(x) \leftarrow \min(r_1(x, y), r_2(y, z))$$

	RecordID	r_1			r_2			
	1	*a*	*b*	1.0	*m*	*h*	0.95	
	2	*c*	*d*	0.9	*m*	*j*	0.85	
	3	*e*	*f*	0.8	*f*	*k*	0.75	$\leftarrow$
$\rightarrow$	4	*l*	*m*	0.7	*m*	*n*	0.65	
	5	*o*	*p*	0.6	*p*	*q*	0.55	
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	

Action: STOP, top-1 tuple score is equal or above threshold $0.75 = \max(\min(1.0, 0.75), \min(0.7, 0.95))$

Queue	δ
–	0.75

Predicate	*Answers*
q	$\langle e, 0.75\rangle \langle l, 0.7\rangle$
p	$\langle e, 0.75\rangle, \langle l, 0.7\rangle$

$$Top_1(\mathcal{P}, q) = \{\langle e, 0.75\rangle\}$$

Note: no further answer will have score above threshold δ

Procedure *TopAnswers*($\mathcal{L}$, $\mathcal{K}$, q, k)
Input: Truth space $\mathcal{L}$, KB $\mathcal{K} = \langle \mathcal{F}, P \rangle$, query relation q, $k \geq 1$
Output: Mapping `rankedList` such that `rankedList`(q) contains top-k answers of q
Init: $\delta = 1$, **for all** predicates p in $\mathcal{P}$ **do**
if p intensional **then** `rankedList`(p) $= \emptyset$, `Q`(p) $:= \emptyset$ **fi**
if p extensional **then** `rankedList`(p) $= T_p$ **fi**
endfor
1. **loop**
2. **if** `A` $= \emptyset$ **then** `A` $:=$ $\{q\}$, `dg` $:=$ $\{q\}$, `in` $:= \emptyset$, `rL`$'$ $:=$ `rankedList`, initialise all pointers ptr_i^r to 0
3. **for all** intensional predicates p **do** `exp`(p) $=$ `false` **endfor**
fi
4. **select** $p \in$ `A`, `A` $:=$ `A` $\setminus \{p\}$, `dg` $:=$ `dg` $\cup$ `s`(p)
5. $\langle \mathbf{t}, s \rangle :=$ *getNextTuple*(p)
6. **if** $\langle \mathbf{t}, s \rangle \neq$ `null` **then** `rankedList`(p) $:=$ `rankedList`(p) $\cup \{\langle \mathbf{t}, s \rangle\}$, `A` $:=$ `A` $\cup$ (`p`(p) $\cap$ `dg`) **fi**
7. **if not** `exp`(p) **then** `exp`(p) $=$ `true`, `A` $:=$ `A` $\cup$ (`s`(p) $\setminus$ `in`), `in` $:=$ `in` $\cup$ `s`(p) **fi**
8. Update threshold δ
9. **until** (`rankedList`(q) does contain k top-ranked tuples with score above query rule threshold)
or ((`rL`$'$ $=$ `rankedList`) **and** `A` $= \emptyset$)
10. **return** top-k ranked tuples in `rankedList`(q)

Procedure *getNextTuple*(p)
Input: intensional relation symbol p. Consider set of rules $\mathcal{R} = \{r \mid r : p(\mathbf{x}) \leftarrow f(A_1, \ldots, A_n) \in \mathcal{P}\}$
Output: Next instance of p together with the score
Init: Let p_i be the relation symbol occurring in A_i

1. **if** $\mathtt{Q}(p) \neq \emptyset$ **then**
   $\langle \mathbf{t}, s \rangle := \mathit{getTop}(\mathtt{Q}(p))$, remove $\langle \mathbf{t}, s \rangle$ from $\mathtt{Q}(p)$, **return** $\{\langle \mathbf{t}, s \rangle\}$ **fi**
   loop
2. **for all** $r \in \mathcal{R}$ **do**
3. Generate the set T of all new valid join tuples $\mathbf{t}$ for rule r,
   using tuples in $\mathtt{rankedList}(p_i)$ and pointers prt_i^r
4. **for all** $\mathbf{t} \in T$ **do**
5. $s :=$ compute the score of $p(\mathbf{t})$ using f;
6. **if** neither $\langle \mathbf{t}, s' \rangle \in \mathtt{rankedList}(p)$ nor $\langle \mathbf{t}, s' \rangle \in \mathtt{Q}(p)$ with $s \preceq s'$ **then**
   insert $\langle \mathbf{t}, s \rangle$ into $\mathtt{Q}(p)$ **fi**
   endfor
   endfor
   until $\mathtt{Q}(p) \neq \emptyset$ or no new valid join tuple can be generated
7. **if** $\mathtt{Q}(p) \neq \emptyset$ **then** $\langle \mathbf{t}, s \rangle := \mathit{getTop}(\mathtt{Q}(p))$, remove $\langle \mathbf{t}, s \rangle$ from $\mathtt{Q}(p)$, **return** $\langle \mathbf{t}, s \rangle$
   else return `null` **fi**

Threshold computation

For an intentional predicate p, head of a rule $r : p(\mathbf{x}) \leftarrow f(p_1, p_2, \ldots, p_n)$.

- consider a threshold variable δ^p
- with $r.\mathbf{t}^{\perp}_{p_i}$ ($r.\mathbf{t}^{\top}_{p_i}$) we denote the last tuple seen (the top ranked one) in `rankedList`(p, r)
- we define

$$\begin{aligned} p_i^{\top} &= \max(\delta^{p_i}, r.\mathbf{t}^{\top}_{p_i}.score) \\ p_i^{\perp} &= \delta^{p_i} \end{aligned}$$

- if p_i is an extensional predicate, we define

$$\begin{aligned} p_i^{\top} &= r.\mathbf{t}^{\top}_{p_i}.score \\ p_i^{\perp} &= r.\mathbf{t}^{\perp}_{p_i}.score \end{aligned}$$

- for rule r we consider the equation $\delta(r)$

$$\delta^p = \max(f(p_1^{\perp}, p_2^{\top}, \ldots, p_n^{\top}), f(p_1^{\top}, p_2^{\perp}, \ldots, p_n^{\top}), \ldots, f(p^{\top}, p^{\top}, \ldots, p_n^{\perp}))$$

- consider the set of equations of all equations involving intentional predicates, i.e.

$$\Delta = \bigcup_{r \in P} \{\delta(r)\} .$$

- for a query $q(\mathbf{x})$, the threshold δ of the *TopAnswers* algorithm is defined as to be

$$\delta = \bar{\delta}^q ,$$

where $\bar{\delta}^q$ is the solution to δ^q in the minimal solution $\bar{\Delta}$ of the set of equations Δ.

- note that $\bar{\delta}^q$, can be computed iteratively as least fixed-point

Complexity

- The problem of determining the truth of ground q in least model of $\mathcal{P}$ is

$$O(|\mathcal{P}^*|h(a+p))$$

where h is the cardinality of the truth space, a is max arity of functions, p is max numbers of predecessors of an atom

- The problem of determining top-k answers to q is

$$O(|\mathcal{P}^*|h(a\log|H| + |\mathcal{P}|h(\bar{a}+|D_q|)))$$

  - H is Herbrand universe
  - D_q is set of intentional relation symbols that *depend* on q
  - $\bar{a} = \max(a, r)$, where r is the number of rules

Annotation domains & RIF

- For RIF, it it is like for RDFS
- The complexity is inherited from their fuzzy variants if lattice is finite, else conjectured undecidable in general

Some open issues for Fuzzy/Annotated SWLs

- Fuzzyfication of these languages is an emerging demand along several directions such as
  1. Fuzzyfying predicates
  2. Several t-norms
  3. Allowing modifiers
  4. Explicit representation of membership functions
  5. Fuzzy quantifiers
  6. Fuzzy spatial, fuzzy time
  7. Standardisation of Syntax & Semantics
- Fuzzy RDFS & SPARQL.
  - AnQL (Annotated RDFS and SPARQL) is the most advanced framework so far
  - Can easily be prototyped
  - But, no efficient system exists yet
- Fuzzy OWL 2
  - Extends OWL 2 with many “fuzzy features”
  - Reasoning procedures very tricky, decidability and computational complexity not simple to figure out
  - Some reasoners exists, fuzzyDL, Fire, DeLorean
  - Expandable Fuzzy OWL 2 standard encoding proposed and Protege plug-in available
  - top-k retrieval algorithm known only for OWL QL and OWL RL
  - integration of RIF and OWL 2 relatively unexplored
- Fuzzy RIF
  - Fuzzy RIF can be obtained from the various Fuzzy LPs framework developed so far
  - No system exists yet supporting Fuzzy RIF
  - More problematic for large scale databases, no efficient support for top-k retrieval