# From Fuzzy to Annotated Semantic Web Languages

Umberto Straccia and Fernando Bobillo

ISTI-CNR, Pisa, Italy, straccia@isti.cnr.it
University of Zaragoza, Spain, fbobillo@unizar.es

About Vagueness

# On the Existence of Vague Concepts

What are vague concepts and do they exists?

- What are the pictures about?


Arai
rukka

- A concept is vague whenever its extension is deemed lacking in clarity
  - Aboutness of a picture or piece of text
  - Tall person
  - High temperature
  - Nice weather
  - Adventurous trip
  - Similar proof
- Vague concepts:
  - Are abundant in everyday speech and almost inevitable
  - Their meaning is often subjective and context dependent

# On the Existence of Vague Objects

What are vague objects and do they exists?

- Are there vague objects in the pictures?

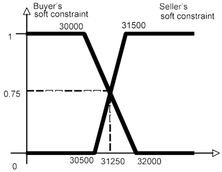

rukka

- An object is vague whenever its identity is lacking in clarity
  - Dust
  - Cloud
  - Dunes
  - Sun
- Vague objects:
  - Are not identical to anything, except to themselves (reflexivity)
  - Are characterised by a vague identity relation (e.g. a similarity relation)
- BTW: example of *uncertain object*: “habitable Earth-like planet in universe"

# Vague Statements

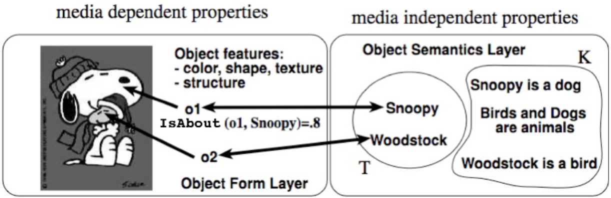
- A statement is vague whenever it involves vague concepts or vague objects
  - Heavy rain
  - Tall person
  - Hot temperature
- The truth of a vague statement is a matter of degree, as it is intrinsically difficult to establish whether the statement is entirely true or false
  - There are 33 °C. Is it hot?

# Sources of Vagueness: Matchmaking




- 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 vague constraints (fuzzy sets) (as usual in negotiation)
  - Seller would sell above 31500 €, but can go down to 30500 €
  - The buyer prefers to spend less than 30000 €, but can go up to 32000 €
  - Highest degree of matching is 0.75 . The car may be sold at 31250 €.

# Sources of Vagueness: Multimedia information retrieval

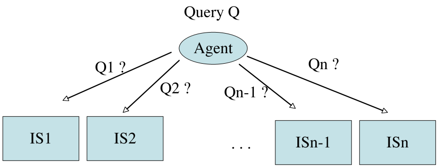



| *IsAbout* | | |
|---|---|---|
| *ImageRegion* | *Object ID* | *degree* |
| *o*1 | *snoopy* | 0.8 |
| *o*2 | *woodstock* | 0.7 |
| ⋮ | ⋮ | |

"Find top-*k* image regions about animals"

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

# Sources of Vagueness: 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)

# Sources of Vagueness: Vague database query

| *HotelID* | *hasLoc* |
|---|---|
| *h*1 | *hl*1 |
| *h*2 | *hl*2 |
| ⋮ | ⋮ |

| *ConferenceID* | *hasLoc* |
|---|---|
| *c*1 | *cl*1 |
| *c*2 | *cl*2 |
| ⋮ | ⋮ |

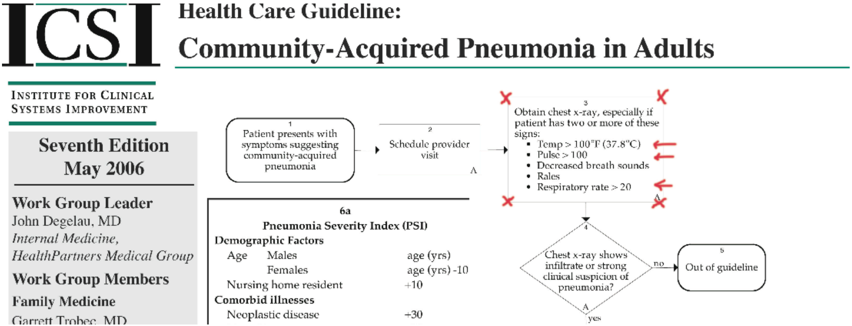
| *hasLoc* | *hasLoc* | *distance* |
|---|---|---|
| *hl*1 | *cl*1 | 300 |
| *hl*1 | *cl*2 | 500 |
| *hl*2 | *cl*1 | 750 |
| *hl*2 | *cl*2 | 800 |
| ⋮ | ⋮ | |

| *hasLoc* | *hasLoc* | *close* | *cheap* |
|---|---|---|---|
| *hl*1 | *cl*1 | 0.7 | 0.3 |
| *hl*1 | *cl*2 | 0.5 | 0.5 |
| *hl*2 | *cl*1 | 0.25 | 0.8 |
| *hl*2 | *cl*2 | 0.2 | 0.9 |
| ⋮ | ⋮ | ⋮ | |

"Find top-*k* cheapest hotels close to the train station"

$$q(h) \leftarrow \mathit{hasLocation}(h, hl) \wedge \mathit{hasLocation}(\mathtt{train}, cl) \wedge \mathit{close}(hl, cl) \wedge \mathit{cheap}(h)$$

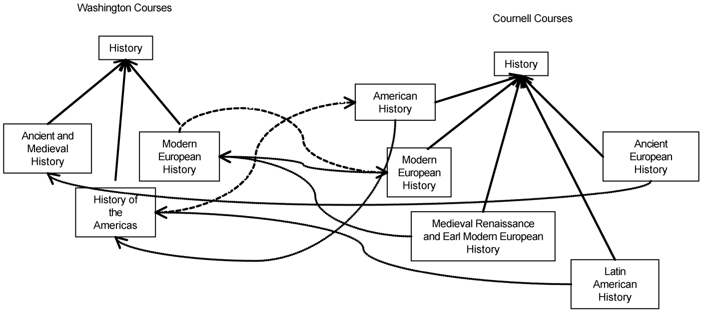
# Sources of Vagueness: Health-care: diagnosis of pneumonia




- E.g., *Temp* = 37.5, *Pulse* = 98, *RespiratoryRate* = 18 are in “danger zone” already
- Temperature, Pulse and Respiratory rate: these constraints are rather vague than crisp

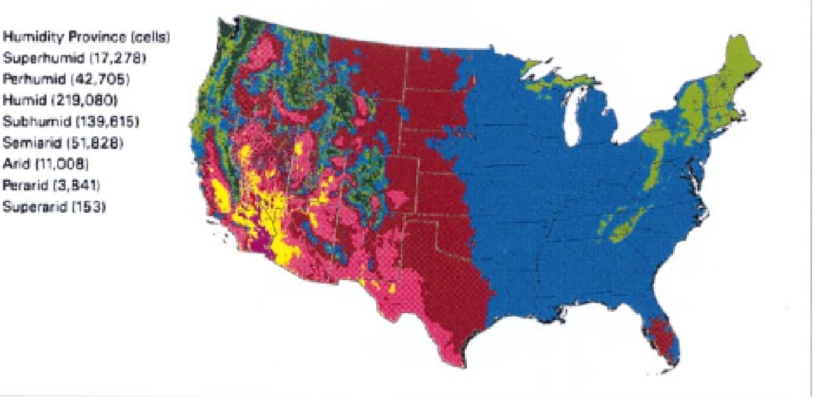
# Sources of Vagueness: Ontology alignment (schema matching)

- To which degree are two concepts of two ontologies similar?



# Sources of Vagueness: Lifezone mapping

- To which degree do certain areas have a specific bioclima




Holdridge life zones of USA

# Sources of Vagueness: ARPAT, Air quality in the province of Lucca

**I dati di domenica 21/02/2010**

**Sintesi dei dati rilevati dalle ore 0 alle ore 24 del giorno domenica 21/02/2010**

| Stazione | | Tipo stazione | $SO_2$ µg/m³ (media su 24h) | $NO_2$ µg/m³ (max oraria) | CO mg/m³ (max oraria) | $O_3$ µg/m³ (max oraria) | $PM_{10}$ µg/m³ (media su 24h) | Giudizio di qualità dell'aria |
|---|---|---|---|---|---|---|---|---|
| Lucca | P.za San Micheletto (RETE REGIONALE **) | urbana - traffico | 1 | 75 | --- | --- | 37 | Accettabile |
| Lucca | V.le Carducci | urbana - traffico | 1 | --- | 2,3 | --- | 49 | Accettabile |
| Lucca | Carignano (RETE REGIONALE **) | rurale - fondo | --- | --- | --- | 86 (h.16*) | --- | Buona |
| Viareggio | Largo Risorgimento | urbana - traffico | --- | --- | 1,8 | --- | n.d. | Buona |
| Viareggio | Via Maroncelli (RETE REGIONALE **) | urbana - fondo | 4 | 97 | --- | 61 (h.15*) | 33 | Accettabile |
| Capannori | V. di Piaggia (RETE REGIONALE **) | urbana - fondo | --- | 62 | 1,3 | --- | 25 | Accettabile |
| Porcari | V. Carrara (RETE REGIONALE **) | periferica - fondo | 1 | 51 | --- | 84 (h.15*) | 24 | Accettabile |

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

http://www.arpat.toscana.it/

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

| Stazione | | Tipo stazione | $SO_2$ µg/m$^3$ (media su 24h) | $NO_2$ µg/m$^3$ (max oraria) | CO mg/m$^3$ (max oraria) | $O_3$ µg/m$^3$ (max oraria) | $PM_{10}$ µg/m$^3$ (media su 24h) | Giudizio di qualità dell'aria |
|---|---|---|---|---|---|---|---|---|
| Lucca | P.za San Micheletto (RETE REGIONALE **) | urbana - traffico | 1 | 75 | --- | --- | 56 | Scadente |
| Lucca | V.le Carducci | urbana - traffico | 2 | --- | 2 | --- | 75 | Pessima |
| Lucca | Carignano (RETE REGIONALE **) | rurale - fondo | --- | --- | --- | 87 (h.18*) | --- | Buona |
| Viareggio | Largo Risorgimento | urbana - traffico | --- | --- | 1,7 | --- | n.d. | Buona |
| Viareggio | Via Maroncelli (RETE REGIONALE **) | urbana - fondo | 1 | 121 | --- | 60 (h.17*) | 45 | Accettabile |
| Capannori | V. di Piaggia (RETE REGIONALE **) | urbana - fondo | --- | 79 | 2 | --- | 59 | Scadente |
| Porcari | V. Carrara (RETE REGIONALE **) | periferica - fondo | 2 | 72 | --- | 82 (h.16*) | 63 | Scadente |

| Giudizio di qualità | $SO_2$ µg/m$^3$ (media su 24h) | $NO_2$ µg/m$^3$ (max oraria) | CO mg/m$^3$ (max oraria) | $O_3$ µg/m$^3$ (max oraria) | $PM_{10}$ µg/m$^3$ (media su 24h) |
|---|---|---|---|---|---|
| Buona | 0-50 | 0-50 | 0-2,5 | 0-120 | 0-25 |
| Accettabile | 51-125 | 51-200 | 2,6-15 | 121-180 | 26-50 |
| Scadente | 126-250 | 201-400 | 15,1-30 | 181-240 | 51-74 |
| Pessima | >250 | >400 | >30 | >240 | >74 |

# TripAdvisor: Hotel User Judgments

# Uncertainty vs Vagueness: a clarification

- Initial difficulty:
  - Understand the conceptual differences between uncertainty and vagueness
- Main problem:
  - Interpreting a degree as a measure of uncertainty rather than as a measure of vagueness

# Uncertain Statements

- A statement is true or false in any world/interpretation
  - We are “uncertain” about which world to consider
  - We may have e.g. a probability distribution over possible worlds
- E.g., “it will rain tomorrow”
  - We cannot exactly establish whether it will rain tomorrow or not, due to our incomplete knowledge about our world
  - We can estimate to which degree this is probable

- Consider a propositional statement (formula) $\phi$
- Interpretation (world) $\mathcal{I} \in \mathcal{W}$,

$$\mathcal{I} : \mathcal{W} \to \{0, 1\}$$

- $\mathcal{I}(\phi) = 1$ means $\phi$ is true in $\mathcal{I}$, denoted $\mathcal{I} \models \phi$
- Each interpretation $\mathcal{I}$ depicts some concrete world
- Given $n$ propositional letters, $|\mathcal{W}| = 2^n$
- In uncertainty theory, we do not know which interpretation $\mathcal{I}$ is the actual one

- One may construct a probability distribution over the worlds

$$\begin{array}{c} Pr : \mathcal{W} \to [0, 1] \\ \sum_{\mathcal{I}} Pr(\mathcal{I}) = 1 \end{array}$$

- $Pr(\mathcal{I})$ indicates the probability that $\mathcal{I}$ is the actual world
- Probability $Pr(\phi)$ of a statement $\phi$ in $Pr$

$$Pr(\phi) = \sum_{\mathcal{I} \models \phi} Pr(\mathcal{I})$$

- $Pr(\phi)$ is the probability of the event: "$\phi$ is true"

# Vague Statements

- A statement is true to some degree, which is taken from a truth space (usually $[0, 1]$)
- The convention prescribing that a proposition is either true or false is changed towards graded propositions
- E.g., “heavy rain”
  - The compatibility of “heavy” in the phrase “heavy rain” is graded and the degree depends on the amount of rain is falling
    - The intensity of precipitation is expressed in terms of a precipitation rate $R$: volume flux of precipitation through a horizontal surface, i.e. $m^3/m^2s = ms^{-1}$
    - It is usually expressed in $mm/h$

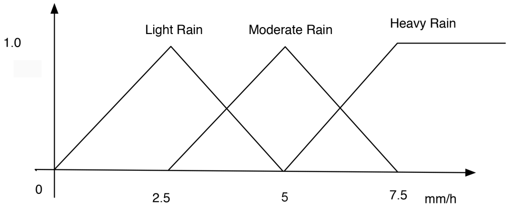
"Heavy rain" continued...E.g., in weather forecasts one may find:

- Rain intensity measured as precipitation rate $R$: volume flux of precipitation through a horizontal surface, i.e. $m^3/m^2h = mh^{-1}$

| | |
|---|---|
| Rain. | Falling drops of water larger than 0.5 mm in diameter. "Rain" usually implies that the rain will fall steadily over a period of time; |
| Light rain. | Rain falls at the rate of 2.6 mm or less an hour; |
| Moderate rain. | Rain falls at the rate of 2.7 mm to 7.6 mm an hour; |
| Heavy rain. | Rain falls at the rate of 7.7 mm an hour or more. |

- Quite harsh distinction: $\begin{array}{lcl} R = 7.7mm/h & \to & \text{heavy rain} \\ R = 7.6mm/h & \to & \text{moderate rain} \end{array}$
- This is clearly unsatisfactory, as quite naturally
  - The more rain is falling, the more the sentence "heavy rain" is true
  - Vice-versa, the less rain is falling the less the sentence is true

- In other words, that the sentence “heavy rain” is no longer either true or false, but is intrinsically graded
  - Even if we have complete knowledge about the current world, i.e. exact specification of the precipitation rate
- More fine grained approach:
  - Define the various types of rains as




  - Light rain, moderate rain and heavy rain are vague concepts

- Consider a propositional statement $\phi$
- A propositional interpretation $\mathcal{I}$ maps $\phi$ to a truth degree in $[0, 1]$

$$\mathcal{I}(\phi) \in [0, 1]$$

- I.e., we are unable to establish whether a statement is entirely true or false due the occurrence of vague concept
- Vague statements are truth-functional
  - Degree of truth of a statement can be calculated from the degrees of truth of its constituents
  - Note that this is not possible for uncertain statements
- Example of truth functional interpretation of vague statements:

$$\begin{array}{lcl} \mathcal{I}(\phi \wedge \psi) & = & \min(\mathcal{I}(\phi), \mathcal{I}(\psi)) \\ \mathcal{I}(\phi \vee \psi) & = & \max(\mathcal{I}(\phi), \mathcal{I}(\psi)) \\ \mathcal{I}(\neg \phi) & = & 1 - \mathcal{I}(\phi) \end{array}$$

# Uncertain Vague Statements

- Recap:
  - In a probabilistic setting each statement is either true or false, but there is e.g. a probability distribution telling us how probable each interpretation/sentence is

$$\mathcal{I}(\phi) \in \{0,1\}, Pr(\mathcal{I}) \in [0,1] \text{ and } \quad Pr(\phi) = \sum_{\mathcal{I} \models \phi} Pr(\mathcal{I}) \quad \in [0,1]$$

  - In vagueness theory instead, sentences are graded

$$\mathcal{I}(\phi) \in [0,1]$$

- Are there sentences combining the two orthogonal concepts of uncertainty and vagueness?
- Yes, and we use them daily !
  - E.g. “there will be heavy rain tomorrow"
- This type of sentences are called uncertain vague sentences
- Essentially, there is
  - uncertainty about the world we will have tomorrow
  - vagueness about the various types of rain

- Consider a propositional statement $\phi$
- A model for uncertain vague sentences:
  - Define probability distribution over worlds $\mathcal{I} \in \mathcal{W}$, i.e.

$$Pr(\mathcal{I}) \in [0,1], \sum_{\mathcal{I}} Pr(\mathcal{I}) = 1$$

  - Sentences are graded: each interpretation $\mathcal{I} \in W$ is truth functional and maps sentences into $[0,1]$

$$\mathcal{I}(\phi) \in [0,1]$$

  - For a sentence $\phi$, consider the expected truth of $\phi$

$$ET(\phi) = \sum_{\mathcal{I}} Pr(\mathcal{I}) \cdot \mathcal{I}(\phi) \ .$$

- Note: if $\mathcal{I}$ is bivalent (that is, $\mathcal{I}(\phi) \in \{0,1\}$) then $ET(\phi) = Pr(\phi)$

From Fuzzy Sets to Mathematical Fuzzy Logic

# Fuzzy Sets Basics

*From Crisp Sets to Fuzzy Sets.*

- Let $X$ be a universal set of objects
- The power set, denoted $2^A$, of a set $A \subset X$, is the set of subsets of $A$, i.e.,

$$2^A = \{B \mid B \subseteq A\}$$

- Often sets are defined as

$$A = \{x \mid P(x)\}$$

  - $P(x)$ is a statement "$x$ has property $P$"
  - $P(x)$ is either true or false for any $x \in X$

- Examples of universe $X$ and subsets $A, B \in 2^X$ may be

$$\begin{array}{rcl} X & = & \{x \mid x \text{ is a day}\} \\ A & = & \{x \mid x \text{ is a rainy day}\} \\ B & = & \{x \mid x \text{ is a day with precipitation rate } R \geq 7.5mm/h\} \end{array}$$

- In the above case: $B \subseteq A \subseteq X$
- The membership function of a set $A \subseteq X$:

$$\chi_A \colon X \to \{0, 1\}$$

  where $\chi_A(x) = 1$ iff $x \in A$
- Note that for sets $A, B \in 2^X$

$$A \subseteq B \text{ iff } \forall x \in X.\ \chi_A(x) \leq \chi_B(x)$$

- Complement of a set $A$, i.e. $\bar{A} = X \setminus A$: $\forall x \in X$:

$$\chi_{\bar{A}}(x) = 1 - \chi_A(x)$$

- Intersection and union: $\forall x \in X$

$$\begin{array}{rcl}\chi_{A \cap B}(x) & = & \min(\chi_A(x), \chi_B(x)) \\ \chi_{A \cup B}(x) & = & \max(\chi_A(x), \chi_B(x))\end{array}$$

- Cartesian product of two sets $A, B \in 2^X$

$$A \times B = \{\langle a, b\rangle \mid a \in A, b \in B\}$$

- A relation $R \subseteq X \times X$
  - is reflexive if for all $x \in X$
  $$\chi_R(x, x) = 1$$
  - is symmetric if for all $x, y \in X$
  $$\chi_R(x, y) = \chi_R(y, x)$$
  - Inverse of $R$, $\chi_{R^{-1}} : X \times X \to \{0, 1\}$: $\forall x, y \in X$:
  $$\chi_{R^{-1}}(y, x) = \chi_R(x, y)$$

- Fuzzy set $A$: $\chi_A \colon X \to [0,1]$, or simply

$$A \colon X \to [0,1]$$

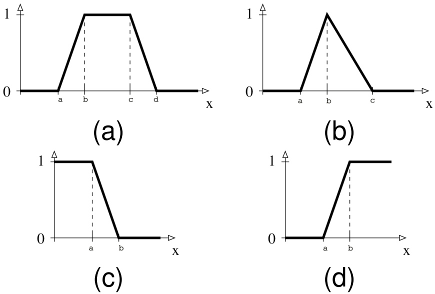
- Fuzzy power set over $X$, is denoted $\tilde{2}^X$, i.e. the set of all fuzzy sets over $X$
- Example: the fuzzy set

$$C = \{x \mid x \text{ is a day with heavy precipitation rate } R\}$$

is defined via the membership function

$$\chi_C(x) = \begin{cases} 1 & \text{if } R \geq 7.5 \\ (x-5)/2.5 & \text{if } R \in [5, 7.5) \\ 0 & \text{otherwise} \end{cases}$$

- Fuzzy membership functions may depend on the context and may be subjective
- Shape may be quite different
- Usually, it is sufficient to consider functions

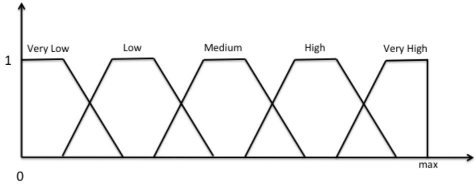



(a) Trapezoidal $trz(a, b, c, d)$; (b) Triangular $tri(a, b, c)$; (c) left-shoulder $ls(a, b)$; (d) right-shoulder $rs(a, b)$

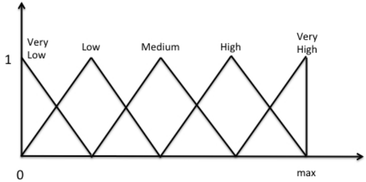
# Fuzzy Sets Construction

- The usefulness of fuzzy sets depends critically on appropriate membership functions
- Methods for fuzzy membership functions construction is largely addressed in literature

- Easy and typically satisfactory method (numerical domain)
  - uniform partitioning into 5 fuzzy sets

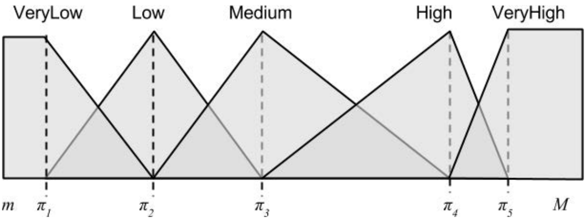



Fuzzy sets construction using trapezoidal functions




Fuzzy sets construction using triangular functions

- Another popular method is based on clustering
- Use Fuzzy C-Means to cluster data into 5 clusters
    - Fuzzy C-Means extends K-Means to accommodates graded membership
- From the clusters $c_1, \ldots, c_5$ take the centroids $\pi_1, \ldots, \pi_5$
- Build the fuzzy sets from the centroids




Fuzzy sets construction using clustering

# Norm-Based Fuzzy Set Operations

- Standard fuzzy set operations are not the only ones
- Most notable ones are triangular norms
  - t-norm $\otimes$ for set intersection
  - t-conorm $\oplus$ (also called s-norm) for set union
  - negation $\ominus$ for set complementation
  - implication $\Rightarrow$
    - set inclusion $A \sqsubseteq B$ is defined as

$$\inf_{x \in X} A(x) \Rightarrow B(x)$$

    - $\Rightarrow$ is often defined from $\otimes$ as *r-implication*

$$a \Rightarrow b = \sup \{c \mid a \otimes c \leq b\} \ .$$

- These functions satisfy some properties that one expects to hold

# Properties for t-norms and s-norms

| Axiom Name | T-norm | S-norm |
|---|---|---|
| Taututology/Contradiction | $a \otimes 0 = 0$ | $a \oplus 1 = 1$ |
| Identity | $a \otimes 1 = a$ | $a \oplus 0 = a$ |
| Commutativity | $a \otimes b = b \otimes a$ | $a \oplus b = b \oplus a$ |
| Associativity | $(a \otimes b) \otimes c = a \otimes (b \otimes c)$ | $(a \oplus b) \oplus c = a \oplus (b \oplus c)$ |
| Monotonicity | if $b \leq c$, then $a \otimes b \leq a \otimes c$ | if $b \leq c$, then $a \oplus b \leq a \oplus c$ |

# Properties for implication and negation functions

| Axiom Name | Implication Function | Negation Function |
|---|---|---|
| Tautology / Contradiction | $0 \Rightarrow b = 1,\ a \Rightarrow 1 = 1,\ 1 \Rightarrow 0 = 0$ | $\ominus 0 = 1,\ \ominus 1 = 0$ |
| Antitonicity | if $a \leq b$, then $a \Rightarrow c \geq b \Rightarrow c$ | if $a \leq b$, then $\ominus a \geq \ominus b$ |
| Monotonicity | if $b \leq c$, then $a \Rightarrow b \leq a \Rightarrow c$ | |

- By commutativity, $\otimes$ and $\oplus$ are monotone also in the first argument
- $\otimes$ is indempotent if $a \otimes a = a$, for all $a \in [0, 1]$
- Megation function $\ominus$ is involutive iff $\ominus \ominus a = a$, for all $a \in [0, 1]$.
- Salient negation functions are:

  Standard or Łukasiewicz negation: $\ominus_l a = 1 - a$;
  Gödel negation: $\ominus_g a$ is 1 if $a = 0$, else is 0.
- Łukasiewicz negation is involutive, Gödel negation is not.

- Salient t-norm functions are:

  Gödel t-norm: $a \otimes_g b = \min(a, b)$;

  Bounded difference or Łukasiewicz t-norm:

  $$a \otimes_l b = \max(0, a + b - 1);$$

  Algebraic product or product t-norm: $a \otimes_p b = a \cdot b$;

  Drastic product:

  $$a \otimes_d b = \begin{cases} 0 & \text{when } (a, b) \in [0, 1[ \times [0, 1[ \\ \min(a, b) & \text{otherwise} \end{cases}$$

- Salient s-norm functions are:

  Gödel s-norm: $a \oplus_g b = \max(a, b)$;

  Bounded sum or Łukasiewicz s-norm: $a \oplus_l b = \min(1, a + b)$;

  Algebraic sum or product s-norm: $a \oplus_p b = a + b - ab$;

  Drastic sum: $a \oplus_d b = \begin{cases} 1 & \text{when } (a, b) \in ]0, 1] \times ]0, 1] \\ \max(a, b) & \text{otherwise} \end{cases}$

Salient properties of norms:

- Ordering among t-norms ($\otimes$ is any t-norm):

$$\otimes_d \leq \otimes \leq \otimes_g$$
$$\otimes_d \leq \otimes_l \leq \otimes_p \leq \otimes_g \ .$$

- The only idempotent t-norm is $\otimes_g$.
- The only t-norm satisfying $a \otimes a = 0$ for all $a \in [0, 1[$ is $\otimes_d$.
- Ordering among s-norms ($\oplus$ is any s-norm):

$$\oplus_g \leq \oplus \leq \oplus_d$$
$$\oplus_g \leq \oplus_p \leq \oplus_l \leq \oplus_d \ .$$

- The only idempotent s-norm is $\oplus_g$.
- The only s-norm satisfying $a \oplus a = 1$ for all $a \in ]0, 1]$ is $\oplus_d$.
- The dual s-norm of $\otimes$ is defined as

$$a \oplus b = 1 - (1 - a) \otimes (1 - b) .$$

- Kleene-Dienes implication: $x \Rightarrow y = \max(1 - x, y)$ is called
- Fuzzy modus ponens: let $a \geq n$ and $a \Rightarrow b \geq m$
  - Under Kleene-Dienes implication, we infer that if $n > 1 - m$ then $b \geq m$
  - Under r-implication relative to a t-norm $\otimes$, we infer that $b \geq n \otimes m$
- composition of two fuzzy relations $R_1 : X \times X \to [0, 1]$ and $R_2 : X \times X \to [0, 1]$: for all $x, z \in X$
  - $(R_1 \circ R_2)(x, z) = \sup_{y \in X} R_1(x, y) \otimes R_2(y, z)$
- A fuzzy relation $R$ is transitive iff for all $x, z \in X$ $R(x, z) \geq (R \circ R)(x, z)$

# Łukasiewicz, Gödel, Product logic and Standard Fuzzy logic

- One distinguishes three different sets of fuzzy set operations (called fuzzy logics)
  - Łukasiewicz, Gödel, and Product logic
  - Standard Fuzzy Logic (SFL) is a sublogic of Łukasiewicz
    - $\min(a, b) = a \otimes_l (a \Rightarrow_l b)$, $\max(a, b) = 1 - \min(1 - a, 1 - b)$

| | Łukasiewicz Logic | Gödel Logic | Product Logic | SFL |
|---|---|---|---|---|
| $a \otimes b$ | $\max(a + b - 1, 0)$ | $\min(a, b)$ | $a \cdot b$ | $\min(a, b)$ |
| $a \oplus b$ | $\min(a + b, 1)$ | $\max(a, b)$ | $a + b - a \cdot b$ | $\max(a, b)$ |
| $a \Rightarrow b$ | $\min(1 - a + b, 1)$ | $\begin{cases} 1 & \text{if } a \leq b \\ b & \text{otherwise} \end{cases}$ | $\min(1, b/a)$ | $\max(1 - a, b)$ |
| $\ominus a$ | $1 - a$ | $\begin{cases} 1 & \text{if } a = 0 \\ 0 & \text{otherwise} \end{cases}$ | $\begin{cases} 1 & \text{if } a = 0 \\ 0 & \text{otherwise} \end{cases}$ | $1 - a$ |

- Mostert–Shields theorem: any continuous t-norm can be obtained as an ordinal sum of these three

# Some additional properties

| Property | Łukasiewicz Logic | Gödel Logic | Product Logic | SFL |
|---|---|---|---|---|
| $x \otimes \ominus x = 0$ | • | | | |
| $x \oplus \ominus x = 1$ | • | | | |
| $x \otimes x = x$ | | • | | • |
| $x \oplus x = x$ | | • | | • |
| $\ominus \ominus x = x$ | • | | | • |
| $x \Rightarrow y = \ominus x \oplus y$ | • | | | • |
| $\ominus (x \Rightarrow y) = x \otimes \ominus y$ | • | | | • |
| $\ominus (x \otimes y) = \ominus x \oplus \ominus y$ | • | • | • | • |
| $\ominus (x \oplus y) = \ominus x \otimes \ominus y$ | • | • | • | • |

- Note: If all conditions in the upper part of a column have to be satisfied then we collapse to classical two-valued logic

# Fuzzy Modifiers

- Fuzzy modifiers: interesting feature of fuzzy set theory
- A fuzzy modifier apply to fuzzy sets to change their membership function
  - Examples: very, more_or_less, and slightly
- A fuzzy modifier *m* represents a function

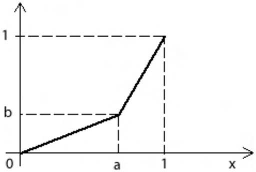
$$f_m \colon [0,1] \to [0,1]$$

Example: $f_{\texttt{very}}(x) = x^2, \quad f_{\texttt{more\_or\_less}}(x) = tri(0, x, 1), \quad f_{\texttt{slightly}}(x) = \sqrt{x}$

- Modelling the fuzzy set of very heavy rain:

$$\begin{aligned}\chi_{\texttt{very heavy rain}}(x) &= f_{\texttt{very}}(\chi_{\texttt{heavyrain}}(x)) \\ &= (\chi_{\texttt{heavyrain}}(x))^2 \\ &= (rs(5,7.5)(x))^2\end{aligned}$$

- A typical shape of modifiers: linear modifiers $lm(a,b)$




- Note: linear modifiers require one parameter $c$ only

$$lm(a,b) = lm(c)$$

where $a = c/(c+1)$ , $b = 1/(c+1)$

# Mathematical Fuzzy Logics Basics

- OWL 2 is grounded on Mathematical Logic
- Fuzzy OWL 2 is grounded on Mathematical Fuzzy Logic
- A statement is no longer either true or false, but is graded
- Truth space: set of truth values $L$ with some structure
- Given a statement $\phi$
  - Fuzzy Interpretation: a function $\mathcal{I}$ mapping $\phi$ into $L$, i.e.

$$\mathcal{I}(\varphi) \in L$$

  - Usually

$$\begin{array}{rcl} L & = & [0,1] \\ L_n & = & \{0, \frac{1}{n}, \ldots, \frac{n-2}{n-1}, \ldots, 1\} \quad (n \geq 1) \end{array}$$

- Fuzzy statement: for $r \in [0, 1]$

$$\langle \phi, r \rangle$$

*The degree of truth of $\phi$ is equal or greater than r*

- Examples:
  - Fuzzy FOL: $\langle \mathit{RainyDay}(d), 0.75 \rangle$
  - Fuzzy LPs: $\langle \mathit{RainyDay}(d) \leftarrow, 0.75 \rangle$
  - Fuzzy RDFS: $\langle \langle d, \mathit{type}, \mathit{RainyDay} \rangle, 0.75 \rangle$
  - Fuzzy DLs: $\langle d{:}\mathit{RainyDay}, 0.75 \rangle$

- Fuzzy interpretation $\mathcal{I}$:
  - Maps each basic statement $p_i$ into $[0,1]$
  - Extended inductively to all statements

$$
\begin{array}{lcl}
\mathcal{I}(\phi \land \psi) & = & \mathcal{I}(\phi) \otimes \mathcal{I}(\psi) \\
\mathcal{I}(\phi \lor \psi) & = & \mathcal{I}(\phi) \oplus \mathcal{I}(\psi) \\
\mathcal{I}(\phi \to \psi) & = & \mathcal{I}(\phi) \Rightarrow \mathcal{I}(\psi) \\
\mathcal{I}(\phi \leftrightarrow \psi) & = & \mathcal{I}(\phi \to \psi) \otimes \mathcal{I}(\psi \to \phi) \\
\mathcal{I}(\neg \phi) & = & \ominus \mathcal{I}(\phi) \\
\mathcal{I}(\exists x.\phi) & = & \sup_{a \in \Delta^{\mathcal{I}}} \mathcal{I}^a_x(\phi) \\
\mathcal{I}(\forall x.\phi) & = & \inf_{a \in \Delta^{\mathcal{I}}} \mathcal{I}^a_x(\phi) \ ,
\end{array}
$$

where

- $\Delta^{\mathcal{I}}$ is the domain of $\mathcal{I}$
- $\otimes$, $\oplus$, $\Rightarrow$, and $\ominus$ are the t-norms, t-conorms, implication functions, a negation functions
- The function $\mathcal{I}^a_x$ is as $\mathcal{I}$ except that $x$ is interpreted as $a$

# Example

- In Lukasiewicz logic:

$$\varphi = \mathit{Cold} \wedge \mathit{Cloudy}$$

| $\mathcal{I}$ | *Cold* | *Cloudy* | $\mathcal{I}(\varphi)$ |
|---|---|---|---|
| $\mathcal{I}_1$ | 0 | 0.1 | $\max(0, 0 + 0.1 - 1) = 0.0$ |
| $\mathcal{I}_2$ | 0.3 | 0.4 | $\max(0, 0.3 + 0.4 - 1) = 0.0$ |
| $\mathcal{I}_3$ | 0.7 | 0.8 | $\max(0, 0.7 + 0.9 - 1) = 0.6$ |
| $\mathcal{I}_4$ | 1 | 1 | $\max(0, 1 + 1 - 1) = 1.0$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |

- Note: given *m* propositional letters
  - Fuzzy interpretations over $L = [0, 1]$ are not recursively enumerable
  - There are $n^m$ fuzzy interpretations over $L_n$

- One may also consider the following abbreviations:

$$
\begin{array}{rcl}
\phi \wedge_g \psi & \stackrel{\mathrm{def}}{=} & \phi \wedge (\phi \to \psi) \\
\phi \vee_g \psi & \stackrel{\mathrm{def}}{=} & (\phi \to \psi) \to \phi) \wedge_g (\psi \to \phi) \to \psi) \\
\neg_{\otimes}\phi & \stackrel{\mathrm{def}}{=} & \phi \to 0 \\
\langle \phi \leq r \rangle & \stackrel{\mathrm{def}}{=} & \langle \neg_l \phi, 1 - r \rangle
\end{array}
$$

- In case $\Rightarrow$ is the r-implication based on $\otimes$, then
  - $\wedge_g$ is Gödel t-norm
  - $\vee_g$ is Gödel s-norm
  - $\neg_{\otimes}$ is interpreted as the negation function related to $\otimes$

- $\mathcal{I}$ satisfies $\langle \phi, r \rangle$, or $\mathcal{I}$ is a model of $\langle \phi, r \rangle$

$$\mathcal{I} \models \langle \phi, r \rangle \text{ iff } \mathcal{I}(\phi) \geq r$$

- $\mathcal{I}$ is a model of $\phi$ if $\mathcal{I}(\phi) = 1$
- Fuzzy knowledge base $\mathcal{K}$: finite set of fuzzy statements
- $\mathcal{I}$ satisfies (is a model of) $\mathcal{K}$: $\mathcal{I} \models \mathcal{K}$ iff it satisfies each element in it
- Best entailment degree of $\phi$ w.r.t. $\mathcal{K}$:

$$bed(\mathcal{K}, \phi) = \sup \{ r \mid \mathcal{K} \models \langle \phi, r \rangle \}$$

- Best satisfiability degree of $\phi$ w.r.t. $\mathcal{K}$:

$$bsd(\mathcal{K}, \phi) = \sup_{\mathcal{I}} \{ \mathcal{I}(\phi) \mid \mathcal{I} \models \mathcal{K} \}$$

## Proposition (Fuzzy Modus Ponens )

*For r-implication $\rightarrow$, for $r, s \in [0, 1]$:*

$$\langle \phi, r \rangle, \langle \phi \rightarrow \psi, s \rangle \models \langle \psi, r \otimes s \rangle$$

## Proposition

*Salient equivalences:*

$$\begin{array}{rcl}
\neg\neg\phi & \equiv & \phi \;\; (\text{Ł}, SFL) \\
\phi \wedge \phi & \equiv & \phi \;\; (G, SFL) \\
\neg(\phi \wedge \neg\phi) & \equiv & 1 \;\; (\text{Ł}, G, \Pi) \\
\phi \vee \neg\phi & \equiv & 1 \;\; (\text{Ł}) \\
\forall x.\phi & \equiv & \neg\exists x.\neg\phi \;\; (\text{Ł}, SFL)
\end{array}$$

## Proposition

*Salient equivalences:*

$$
\begin{array}{rcl}
\text{Ł} + G & \equiv & \textit{Boolean Logic} \\
\text{Ł} + \Pi & \equiv & \textit{Boolean Logic} \\
G + \Pi & \equiv & \textit{Boolean Logic}
\end{array}
$$

## Proposition (BED)

*bed*$(\mathcal{K}, \phi) = \min x$. *such that* $\mathcal{K} \cup \{\langle \varphi \leq x \rangle\}$ *satisfiable.*

## Proposition (BSD)

*bsd*$(\mathcal{K}, \phi) = \max x$. *such that* $\mathcal{K} \cup \{\langle \varphi, x \rangle\}$ *satisfiable.*

# On Witnessed Models

- Witnessed interpretation $\mathcal{I}$:

$$\mathcal{I}(\exists x.\phi) = \mathcal{I}_x^a(\phi), \text{ for some } a \in \Delta^{\mathcal{I}} \quad (1)$$
$$\mathcal{I}(\forall x.\phi) = \mathcal{I}_x^a(\phi), \text{ for some } a \in \Delta^{\mathcal{I}} \quad (2)$$

- The supremum (resp. infimum) are attained at some point
- Classical interpretations are witnessed
- Fuzzy interpretations may not be witnessed
- E.g., $\mathcal{I}$ is not witnessed as Eq. (1) not satisfied:

$$\begin{aligned}
\Delta^{\mathcal{I}} &= \mathbb{N} \\
\mathcal{I}_x^n(A(x)) &= 1 - 1/n < 1, \text{ for all } n \\
\mathcal{I}(\exists x.A(x)) &= \sup_n \mathcal{I}_x^n(A(x)) \\
&= \sup_n 1 - 1/n = 1
\end{aligned}$$

## Proposition (Witnessed model property)

*In Łukasiewicz logic and SFL over $L = [0, 1]$, or for all cases in which the truth space L is finite, a fuzzy KB has a witnessed fuzzy model iff it has a fuzzy model.*

- Not true for Gödel and product logic over $L = [0, 1]$
  - $\neg\forall x\, p(x) \wedge \neg\exists x\, \neg p(x)$ 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 $\mathcal{I}(p(n)) = 1/n$

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

- IMHO: non-witnessed models make little sense in KR
- We will always assume that interpretations are witnessed

# Fuzzy Propositional Logic: Reasoning

- We need to distinguish if truth space is $L = [0, 1]$ or $L_n = \{0, \frac{1}{n}, \ldots, \frac{n-2}{n-1}, \ldots, 1\}$
- Case $L_n$ easier: given $m$ propositional letters, there are $m^n$ possible interpretations
- We may use
  - Operational Research
  - Analytic Tableaux, Non-Deterministic Analytic Tableaux
  - Reduction into Classical Propositional Logic

- Basic idea: translate formulae into equational constraints about truth degrees
- For a formula $\phi$ consider a variable $x_\phi$
  - Intuition: $x_\phi$ will hold the degree of truth of statement $\phi$
  - Example: constraints under Łukasiewicz for $\langle \neg\phi, 0.6\rangle$

$$
\begin{array}{rcl}
x_{\neg\phi} & \in & [0,1] \\
x_\phi & \in & [0,1] \\
x_{\neg\phi} & = & 1 - x_\phi
\end{array}
$$

- We may use Mixed Integer Linear Programming for the encodings of constraints
  For Łukasiewicz:
  - $x_1 \otimes_l x_2 = z \mapsto \{x_1 + x_2 - 1 \leq z, x_1 + x_2 - 1 \geq z - y, z \leq 1 - y, y \in \{0,1\}\}$, where $y$ is a new variable.
  - $x_1 \oplus_l x_2 = z \mapsto \{x_1 + x_2 \leq z + y, y \leq z, x_1 + x_2 \geq z, y \in \{0,1\}\}$, where $y$ is a new variable.
  - $x_1 \Rightarrow_l x_2 = z \mapsto \{(1 - x_1) \oplus_l x_2 = z\}$.

  For SFL:
  - $x_1 \otimes_g x_2 = z \mapsto \{z \leq x_1, z \leq x_2, x_1 \leq z + y, x_2 \leq z + (1 - y), y \in \{0,1\}\}$, where $y$ is a new variable.
  - $x_1 \oplus_g x_2 = z \mapsto \{z \geq x_1, z \geq x_2, x_1 + y \geq z, x_2 + (1 - y) \geq z, y \in \{0,1\}\}$, where $y$ is a new variable.
  - $x_1 \Rightarrow_{kd} x_2 = z \mapsto (1 - x_1) \oplus_g x_2 = z$.

- Negation Normal Form, *nnf*$(\phi)$

$$
\begin{array}{rcl}
\neg\perp & = & \top \\
\neg\top & = & \perp \\
\neg\neg\phi & \mapsto & \phi \\
\neg(\phi \wedge \psi) & \mapsto & \neg\phi \vee \neg\psi \\
\neg(\phi \vee \psi) & \mapsto & \neg\phi \wedge \neg\psi \\
\neg(\phi \rightarrow \psi) & \mapsto & \phi \wedge \neg\psi\ .
\end{array}
$$

1. Transform $\mathcal{K}$ into NNF
2. Initialize the fuzzy theory $\mathcal{T}_{\mathcal{K}}$ and the initial set of constraints $\mathcal{C}_{\mathcal{K}}$ by

$$\begin{array}{rcl} \mathcal{T}_{\mathcal{K}} & = & \{\phi \mid \langle \phi, n \rangle \in \mathcal{K}\} \\ \mathcal{C}_{\mathcal{K}} & = & \{x_{\psi} \geq n \mid \langle \phi, n \rangle \in \mathcal{K}\} \end{array}$$

3. Apply the following inference rules until no more rules can be applied

   $(var)$. For variable $x_{\phi}$ occurring in $\mathcal{C}_{\mathcal{K}}$ add $x_{\phi} \in [0,1]$ to $\mathcal{C}_{\mathcal{K}}$
   $(\bar{var})$. For variable $x_{\neg\phi}$ occurring in $\mathcal{C}_{\mathcal{K}}$ add $x_{\phi} = 1 - x_{\neg\phi}$ to $\mathcal{C}_{\mathcal{K}}$
   $(\bot)$. If $\bot \in \mathcal{T}_{\mathcal{K}}$ then $\mathcal{C}_{\mathcal{K}} := \mathcal{C}_{\mathcal{K}} \cup \{x_{\bot} = 0\}$
   $(\top)$. If $\top \in \mathcal{T}_{\mathcal{K}}$ then $\mathcal{C}_{\mathcal{K}} := \mathcal{C}_{\mathcal{K}} \cup \{x_{\top} = 1\}$

   $(\wedge)$. If $\phi \wedge \psi \in \mathcal{T}_{\mathcal{K}}$, then
   1. add $\phi$ and $\psi$ to $\mathcal{T}_{\mathcal{K}}$
   2. $\mathcal{C}_{\mathcal{K}} := \mathcal{C}_{\mathcal{K}} \cup \{x_{\phi} \otimes x_{\psi} = x_{\phi \wedge \psi}\}$

   $(\vee)$. If $\phi \vee \psi \in \mathcal{T}_{\mathcal{K}}$, then
   1. add $\phi$ and $\psi$ to $\mathcal{T}_{\mathcal{K}}$
   2. $\mathcal{C}_{\mathcal{K}} := \mathcal{C}_{\mathcal{K}} \cup \{x_{\phi} \oplus x_{\psi} = x_{\phi \wedge \psi}\}$

   $(\rightarrow)$. If $\phi \rightarrow \psi \in \mathcal{T}_{\mathcal{K}}$, then
   1. add $nnf(\neg\phi)$ and $\psi$ to $\mathcal{T}_{\mathcal{K}}$
   2. $\mathcal{C}_{\mathcal{K}} := \mathcal{C}_{\mathcal{K}} \cup \{(1 - x_{nnf(\neg\phi)}) \Rightarrow x_{\psi} = x_{\phi \rightarrow \psi}\}$

*sat*($\mathcal{K}$): $\mathcal{K}$ is satisfiable iff the final set of constraints $\mathcal{C}_\mathcal{K}$ has a solution

*bed*($\mathcal{K}, \phi$):

- Add $\neg\phi$ to $\mathcal{T}_\mathcal{K}$
- Add $x_{\neg\phi} \geq 1 - x, x \in [0, 1]$ to $\mathcal{C}_\mathcal{K}$, $x$ new
- Compute final set of constraints $\mathcal{C}_\mathcal{K}$
- Then, solve the optimisation problem

$$bed(\mathcal{K}, \phi) = \min x. \text{ such that } \mathcal{C}_\mathcal{K} \text{ has a solution}$$

*bsd*($\mathcal{K}, \phi$):

- Add $\phi$ to $\mathcal{T}_\mathcal{K}$
- Add $x_{\phi} \geq x, x \in [0, 1]$ to $\mathcal{C}_\mathcal{K}$, $x$ new
- Compute final set of constraints $\mathcal{C}_\mathcal{K}$
- Then, solve the optimisation problem

$$bsd(\mathcal{K}, \phi) = \max x. \text{ such that } \mathcal{C}_\mathcal{K} \text{ has a solution}$$

# Analytical Fuzzy Tableau: Case SFL

- Main property the method is based on:
  - if $\mathcal{I}$ is model of $\langle \phi \wedge \psi, n \rangle$ then $\mathcal{I}$ is a model of both $\langle \phi, n \rangle$ and $\langle \psi, n \rangle$;
  - if $\mathcal{I}$ is model of $\langle \phi \vee \psi, n \rangle$ then $\mathcal{I}$ is a model of either $\langle \phi, n \rangle$ or $\langle \psi, n \rangle$.
  - $\mathcal{I}$ cannot be a model of both $\langle p, n \rangle$ and $\langle \neg p, m \rangle$ if $n > 1 - m$.
- A clash is either
  - a fuzzy statement $\langle \bot, n \rangle$ with $n > 0$; or
  - a pair of fuzzy statements $\langle p, n \rangle$ and $\langle \neg p, m \rangle$ with $n > 1 - m$
- Clash-free: does not contain a clash

1. Transform $\mathcal{K}$ into NNF
2. Initialize the completion $\mathcal{S}_{\mathcal{K}} = \mathcal{K}$
3. Apply the following inference rules to $\mathcal{S}_{\mathcal{K}}$ until no more rules can be applied
4. We call a set of fuzzy statements $\mathcal{S}_{\mathcal{K}}$ complete iff none of the rules below can be applied to $\mathcal{S}_{\mathcal{K}}$
5. Note that rule $(\vee)$ is non-deterministic

$(\wedge)$. If $\langle \phi \wedge \psi, n \rangle \in \mathcal{S}_{\mathcal{K}}$ and $\{\langle \phi, n \rangle, \langle \psi, n \rangle\} \not\subseteq \mathcal{S}_{\mathcal{K}}$ , then add both $\langle \phi, n \rangle$ and $\langle \psi, n \rangle$ to $\mathcal{S}_{\mathcal{K}}$

$(\vee)$. If $\langle \phi \vee \psi, n \rangle \in \mathcal{S}_{\mathcal{K}}$ and $\{\langle \phi, n \rangle, \langle \psi, n \rangle\} \cap \mathcal{S}_{\mathcal{K}} = \emptyset$, then add either $\langle \phi, n \rangle$ *or* $\langle \psi, n \rangle$ to $\mathcal{S}_{\mathcal{K}}$

$(\rightarrow)$. If $\langle \phi \rightarrow \psi, n \rangle \in \mathcal{S}_{\mathcal{K}}$ and $\langle nnf(\neg\phi) \vee \psi, n \rangle \notin \mathcal{S}_{\mathcal{K}}$, then add $\langle nnf(\neg\phi) \vee \psi, n \rangle$ to $\mathcal{S}_{\mathcal{K}}$

*sat*$(\mathcal{K})$: $\mathcal{K}$ is satisfiable iff we find a complete and clash-free completion $\mathcal{S}_{\mathcal{K}}$ of $\mathcal{K}$

- For BED and BSD we need some more work
- Given $\mathcal{K}$, define

$$
\begin{array}{lcl}
N^{\mathcal{K}} & = & \{0, 0.5, 1\} \;\cup\; \{n \mid \langle \phi, n \rangle \in \mathcal{K}\} \\
\bar{N}^{\mathcal{K}} & = & N^{\mathcal{K}} \cup \{1 - n \mid n \in N^{\mathcal{K}}\} \\
\epsilon & = & \min\{d/2 \mid n, m \in \bar{N}^{\mathcal{K}}, n \neq m, d = |n - m|\}
\end{array}
$$

**Proposition**

*Under SFL, given $\mathcal{K}$, then for $n > 0$*

$$\mathcal{K} \models \langle \phi, n \rangle \text{ iff } \mathcal{K} \cup \{\langle \neg\phi, 1 - n + \epsilon \rangle\} \text{ is not satisfiable .}$$

*Moreover, $\mathcal{K}$ is satisfiable iff it has a model over $\bar{N}^{\mathcal{K}}$.*

*bed*$(\mathcal{K}, \phi)$: Find greatest $n \in \bar{N}^{\mathcal{K}}$ such that $\mathcal{K} \models \langle \phi, n \rangle$

*bsd*$(\mathcal{K}, \phi)$: Find greatest $n \in \bar{N}^{\mathcal{K}}$ such that $\mathcal{K} \cup \{\langle \phi, n \rangle\}$ satisfiable

# Non Deterministic Analytic Fuzzy Tableau

- Works for finitely-valued fuzzy propositional logic over $L_n$
- Works also for SFL (as in place of $[0, 1]$, we may use $\bar{N}^{\mathcal{K}}$)
- Basic idea is as for fuzzy tableau, but now we guess the truth degrees

$(\wedge)$. If $\langle \phi \wedge \psi, n \rangle \in \mathcal{S}_{\mathcal{K}}$, $n_1, n_2 \in L_n$ such that $n_1 \otimes n_2 = n$ and $\{\langle \phi, n_1 \rangle, \langle \psi, n_2 \rangle\} \not\subseteq \mathcal{S}_{\mathcal{K}}$ , then add both $\langle \phi, n_1 \rangle$ and $\langle \psi, n_2 \rangle$ to $\mathcal{S}_{\mathcal{K}}$

$(\vee)$. If $\langle \phi \vee \psi, n \rangle \in \mathcal{S}_{\mathcal{K}}$, $n_1, n_2 \in L_n$ such that $n_1 \oplus n_2 = n$ and $\{\langle \phi, n_1 \rangle, \langle \psi, n_2 \rangle\} \not\subseteq \mathcal{S}_{\mathcal{K}}$ , then add both $\langle \phi, n_1 \rangle$ and $\langle \psi, n_2 \rangle$ to $\mathcal{S}_{\mathcal{K}}$

$(\rightarrow)$. If $\langle \phi \rightarrow \psi, n \rangle \in \mathcal{S}_{\mathcal{K}}$, $n_1, n_2 \in L_n$ such that $n_1 \Rightarrow n_2 = n$ and $\{\langle \phi, n_1 \rangle, \langle \psi, n_2 \rangle\} \not\subseteq \mathcal{S}_{\mathcal{K}}$ , then add both $\langle \phi, n_1 \rangle$ and $\langle \psi, n_2 \rangle$ to $\mathcal{S}_{\mathcal{K}}$

- A clash is either
  - a fuzzy statement $\langle \bot, n \rangle$ with $n > 0$; or
  - a pair of fuzzy statements $\langle p, n \rangle$ and $\langle \neg p, m \rangle$ such that

$$x_p \geq n, \quad \ominus x_p \geq m, x_p \in L_n$$

  has no solution

# Reduction to Classical Propositional Logic: Case SFL over $[0, 1]$

- Given $\mathcal{K}$, we know that we can use

$$L_n = \bar{N}^{\mathcal{K}} = \{\gamma_1, \ldots, \gamma_n\}$$

  with $\gamma_i < \gamma_{i+1}, 1 \leq i \leq n-1$
- Basic idea: use atom $A_{\geq r}$ to represent

  *The truth degree of A has to be equal or greater than r*

- Similarly for $A_{>r}$, $A_{\leq r}$ and $A_{<r}$

- To start with, build $Crisp_{L_n}$
  - For all atoms $A$, for all $1 \le i \le n-1, 2 \le j \le n-1$

$$\begin{array}{l} A_{\geq \gamma_{i+1}} \to A_{>\gamma_i} \\ A_{>\gamma_j} \to A_{\geq \gamma_j} \end{array}$$

- Build $Crisp_{\mathcal{K}}$:

$$\begin{array}{rcl} Crisp_{\mathcal{K}} & = & \{\rho(\phi, n) \mid \langle \phi, n \rangle \in \mathcal{K}\} \cup \\ & & Crisp_{L_n} \ , \end{array}$$

| $x$ | $y$ | $\rho(x, y)$ |
|---|---|---|
| $\top$ | $c$ | $\top$ |
| $\bot$ | $0$ | $\top$ |
| $\bot$ | $c$ | $\bot$ if $c > 0$ |
| $A$ | $c$ | $A_{\geq c}$ |
| $\neg A$ | $c$ | $\neg A_{>1-c}$ |
| $\phi \wedge \psi$ | $c$ | $\rho(\phi, c) \wedge \rho(\psi, c)$ |
| $\phi \vee \psi$ | $c$ | $\rho(\phi, c) \vee \rho(\psi, c)$ |

## Proposition

*Given $\mathcal{K}$ under SFL over $L_n$, then $\mathcal{K} \models \langle \phi, c \rangle$ iff $\mathcal{K} \cup \{\langle \neg\phi, 1 - c^- \rangle\}$ is not satisfiable, where $c^-$ is the next smaller value than $c$ in $L_n$*

$sat(\mathcal{K})$: $\mathcal{K}$ is satisfiable iff $Crisp_{\mathcal{K}}$ satisfiable

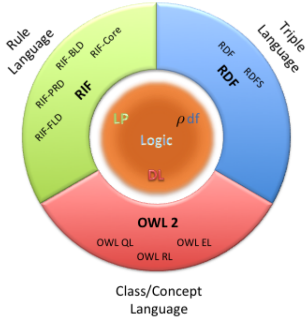
$bed(\mathcal{K}, \phi)$: Find greatest $c \in L_n$ such that $\mathcal{K} \models \langle \phi, c \rangle$

$bsd(\mathcal{K}, \phi)$: Find greatest $c \in L_n$ such that $\mathcal{K} \cup \{\langle \phi, c \rangle\}$ satisfiable

From Fuzzy to Annotated Semantic Web Languages

# The Semantic Web Family of Languages

- Wide variety of languages
  - RDFS: *Triple language*, -*Resource Description Framework*
    - The logical counterpart is $\rho$df
  - RIF: *Rule language*, -*Rule Interchange Format*,
    - Relate to the *Logic Programming* (LP) paradigm
  - OWL 2: *Conceptual language*, -*Ontology Web Language*
    - Relate to Description Logics (DLs)



# RDFS

- RDFS: the triple language

$$\langle \textit{subject}, \textit{predicate}, \textit{object} \rangle$$

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

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

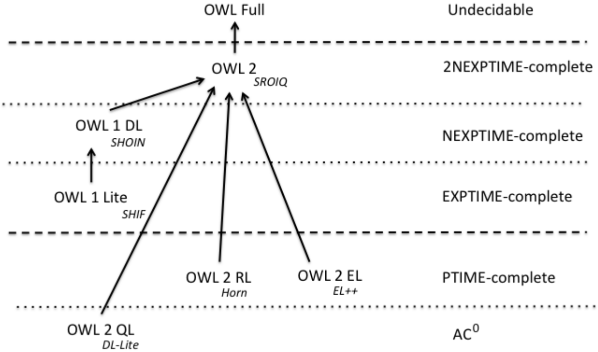
"if $A$ subclass of $B$, $X$ instance of $A$ then $X$ is instance of $B$"

- Complexity: $\mathcal{O}(|\mathcal{K}|^2)$
- Store all inferred triples into a relational database and query via SQL

- OWL 2 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 in polynomial 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 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/RuleML

- RIF/RuleML family: the rule language

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

Important point: RDFS, OWL 2 and RIF/RuleML are logical languages

- RDFS: logic with intensional semantics
- OWL 2: relates to the Description Logics family
- RIF/RuleML: relates to the Logic Programming paradigm (e.g., Datalog, Datalog$^{\pm}$)
- OWL 2 and RIF/RuleML have extensional semantics
- We will address them from a logical point of view

## The case of RDF

# 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:

$$(airplane, has, enginefault)$$

# $\rho$df (restricted RDFS)

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

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

- $(p, \text{sp}, q)$
  - property $p$ is a *sub property* of property $q$
- $(c, \text{sc}, d)$
  - class $c$ is a *sub class* of class $d$
- $(a, \text{type}, b)$
  - $a$ is of *type* $b$
- $(p, \text{dom}, c)$
  - *domain* of property $p$ is $c$
- $(p, \text{range}, c)$
  - *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. $\Delta_R$ is a nonempty set of resources, called the domain or universe of $\mathcal{I}$;
2. $\Delta_P$ is a set of property names (not necessarily disjoint from $\Delta_R$);
3. $\Delta_C \subseteq \Delta_R$ is a distinguished subset of $\Delta_R$ identifying if a resource denotes a class of resources;
4. $\Delta_L \subseteq \Delta_R$, the set of literal values, $\Delta_L$ 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 $\Delta_R$ 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 $\rho$df.

- An interpretation $\mathcal{I}$ is a model of $G$ under $\rho$df, denoted $\mathcal{I} \models G$, iff
  - $\mathcal{I}$ is an interpretation over the vocabulary $\rho\mathrm{df} \cup \mathit{universe}(G)$
  - $\mathcal{I}$ 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. $P[\![\mathrm{sp}^{\mathcal{I}}]\!]$ is transitive over $\Delta_P$;
2. if $(p, q) \in P[\![\mathrm{sp}^{\mathcal{I}}]\!]$ then $p, q \in \Delta_P$ and $P[\![p]\!] \subseteq P[\![q]\!]$;

# Models (cont.)

Subclass:

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

Typing I:

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

Typing II:

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

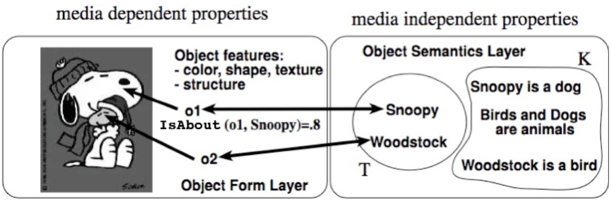
# Entailment

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

$$\begin{array}{lcl} G & \models & (a, \mathsf{sp}, a) \\ G & \models & (a, \mathsf{sc}, a) \end{array}$$

    which anyway are of marginal interest

# Example




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

# 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 $\tau_i$ 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

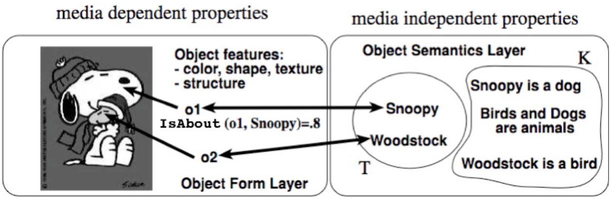
$$q(x, y) \leftarrow (x, \mathit{creates}, y), (x, \mathrm{type}, \mathit{Flemish}), (x, \mathit{paints}, y), (y, \mathit{exhibited}, \mathit{Uffizi})$$

  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




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

Consider the query

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

Then

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

# Fuzzy RDFS

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

$$\langle(\mathit{subject}, \mathit{predicate}, \mathit{object}), n\rangle$$

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

$$\langle(o1, \mathit{IsAbout}, \mathit{snoopy}), 0.8\rangle$$

# Fuzzy RDF Syntax

- Fuzzy RDF triple (or Fuzzy RDF atom):

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

  - $s \in \mathbf{UBL}$ 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:

$$\langle (\mathrm{audiTT}, \mathrm{type}, \mathrm{SportCar}), 0.8 \rangle$$

- 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. $\Delta_R$ is a nonempty set of resources, called the domain or universe of $\mathcal{I}$;
2. $\Delta_P$ is a set of property names (not necessarily disjoint from $\Delta_R$);
3. $\Delta_C \subseteq \Delta_R$ is a distinguished subset of $\Delta_R$ identifying if a resource denotes a class of resources;
4. $\Delta_L \subseteq \Delta_R$, the set of literal values, $\Delta_L$ 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 $\mathbf{UL}$ in $V$, and such that $\cdot^{\mathcal{I}}$ is the identity for plain literals and assigns an element in $\Delta_R$ to elements in $\mathbf{L}$;
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 $\mathbf{B}$.

# Models

Let $G$ be a graph over $\rho$df.

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

Simple:

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

Subproperty:

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

# 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

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

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

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

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

# Deduction System for fuzzy RDFS

- Very simple:

$$(AG) \quad \frac{\langle \tau_1, n_1 \rangle, \ldots, \langle \tau_k, n_k, \{\tau_1, \ldots \tau_k\} \vdash_{\mathrm{RDFS}} \tau \rangle}{\langle \tau, \bigotimes_i \lambda_i \rangle}$$

# Deduction System for fuzzy RDFS

1. Simple:

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

2. Subproperty:

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

3. Subclass:

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

4. Typing:

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

5. Implicit Typing:

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

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

# 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

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

  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 $f \colon ([0, 1])^{n+h} \to [0, 1]$, 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

$$\langle q(x), s\rangle \leftarrow \langle(x, \mathrm{type}, \mathrm{SportCar}), s_1\rangle, (x, \mathrm{hasPrice}, y), s = s_1 \cdot \mathrm{cheap}(y)$$

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

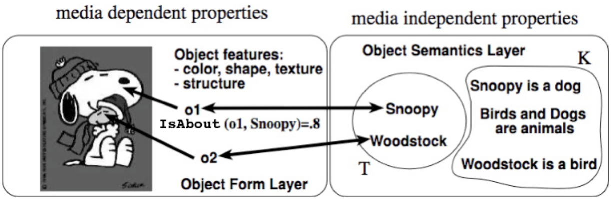
# Fuzzy RDF Query Answering (cont.)

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

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

- 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




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

Consider the query

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

Then (under any t-norm)

$$ans(G, q) \quad = \quad \{\langle o1, 0.32\rangle, \langle o2, 0.63\rangle\}, \quad ans_1(G, q) = \{\langle o2, 0.63\rangle\}$$

# Annotation domains & RDFS

- Generalisation of fuzzy RDFS
  - a triple is annotated with a value $\lambda$ 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
  - $(\mathit{umberto}, \mathit{workedFor}, \mathit{IEI})$
  - true during 1992–2001
- Fuzzyness
  - $(\mathit{WingateHotel}, \mathit{closeTo}, \mathit{RR}11\mathit{Venue})$
  - true to some degree
- Provenance
  - $(\mathit{umberto}, \mathit{knows}, \mathit{didier})$
  - 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:

$$\lambda_1 \preceq \lambda_2 \iff \lambda_1 \oplus \lambda_2 = \lambda_2$$

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

$$\langle (s, p, o), \lambda \rangle$$

- For instance,

$\langle (\mathit{umberto}, \mathit{workedFor}, \mathit{IEI}), [1992, 2001] \rangle$

$\langle (\mathit{WingateHotel}, \mathit{closeTo}, \mathit{RR11Venue}), 0.8 \rangle$

$\langle (\mathit{umberto}, \mathit{knows}, \mathit{didier}), \texttt{http://www.straccia.info/foaf.rdf} \rangle$

## 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 $\langle \tau, \lambda \rangle$ is true to a lesser extent than $\langle \tau', \lambda' \rangle$
    - 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: $\langle (\mathit{WingateHotel}, \mathit{closeTo}, \mathit{RR}11\mathit{Venue}), 0.8\rangle$
  - $L = [0, 1]$
  - $\otimes =$ any t-norm
  - $\vee = \max$
- Provenance: $\langle (\mathit{umberto}, \mathit{knows}, \mathit{didier}), p\rangle$
  - $L =$ DNF propositional formulae over URIs
  - $\otimes = \wedge$
  - $\vee = \vee$
- Multiple Domains: our frameworks allows to combine domains

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

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

- Inference rule:

$$\frac{\langle \tau_1, \lambda_1 \rangle, \; \ldots, \; \langle \tau_k, \lambda_k, \{\tau_1, \ldots, \tau_k\} \vdash_{\mathrm{RDFS}} \tau \rangle}{\langle \tau, \bigotimes_i \lambda_i \rangle}$$

- 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/`

The case of Description Logics

# Description Logics (DLs)

- Concept/Class: names are equivalent to unary predicates
  - In general, concepts equiv to formulae with one free variable
- Role or attribute: names are equivalent to binary predicates
  - In general, roles equiv to formulae with two free variables
- Taxonomy: Concept and role hierarchies can be expressed
- Individual: names are equivalent to constants
- Operators: restricted so that
  - Language is decidable and, if possible, of low complexity
  - No need for explicit use of variables
    - Restricted form of $\exists$ and $\forall$
  - Features such as counting can be succinctly expressed

- Basic ingredients: descriptions of classes, properties, and their instances, such as
  - $a{:}C$, meaning that individual $a$ is an instance of concept/class $C$

$$\mathrm{a{:}Person} \sqcap \forall \mathrm{hasChild.Femal}$$

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

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

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

$$\mathrm{Person} \sqsubseteq \forall \mathrm{hasChild.Person}$$

# The DL Family

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

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

# DL Semantics

- Semantics is given in terms of an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
  - $\Delta^{\mathcal{I}}$ is the domain (a non-empty set)
  - $\cdot^{\mathcal{I}}$ is an interpretation function that maps:
    - Concept (class) name $A$ into a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$
    - Role (property) name $R$ into a subset $R^{\mathcal{I}}$ of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
    - Individual name $a$ into an element of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ s.t. $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$ (UNA)
  - Interpretation function $\cdot^{\mathcal{I}}$ is extended to concept expressions:

$$
\begin{array}{rcl}
\top^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} & = & \emptyset \\
(C_1 \sqcap C_2)^{\mathcal{I}} & = & C_1{}^{\mathcal{I}} \cap C_2{}^{\mathcal{I}} \\
(C_1 \sqcup C_2)^{\mathcal{I}} & = & C_1{}^{\mathcal{I}} \cup C_2{}^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} & = & \{x \in \Delta^{\mathcal{I}} \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} & = & \{x \in \Delta^{\mathcal{I}} \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\}
\end{array}
$$

  - Finally, we say that
    - $\mathcal{I}$ is a model of $C \sqsubseteq D$, written $\mathcal{I} \models C \sqsubseteq D$, iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
    - $\mathcal{I}$ is a model of $a{:}C$, written $\mathcal{I} \models a{:}C$, iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$
    - $\mathcal{I}$ is a model of $(a, b){:}R$, written $\mathcal{I} \models (a, b){:}R$, iff $\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in R^{\mathcal{I}}$

# Note on DL Naming

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

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

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

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

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

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

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

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

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

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

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

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

For instance,

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

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

# Basic Inference Problems (Formally)

Consistency: Check if knowledge is meaningful

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

Subsumption: structure knowledge, compute taxonomy

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

Equivalence: check if two classes denote same set of instances

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

Instantiation: check if individual $a$ instance of class $C$

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

Retrieval: retrieve set of individuals that instantiate $C$

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

# Reduction to Satisfiability

Problems are all reducible to KB satisfiability

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

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

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

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

# Reasoning in DLs: Basics

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

# Tableaux: 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 $\mathcal{L}(a) \to \mathcal{L}(a) \cup \{C\}$
- 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 | $\mathcal{L}(y) \to \mathcal{L}(y) \cup \{C\}$ |

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D\}$$

$x$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D\}$$

$$x$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D\}$$

$$x$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$x$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

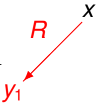
$$x$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\mathcal{L}(y_1) = \{C\}$$



# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\mathcal{L}(y_1) = \{C\} \qquad x \xrightarrow{R} y_1$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$x \xrightarrow{R} y_1$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$x \xrightarrow{R} y_1$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \\ \mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, {\color{red}\neg C}\} \quad y_1 & \end{tikzcd}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \\ y_1 & \end{tikzcd}$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg C\}$$

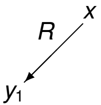
Clash

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D\}$$



# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \\ y_1 & \end{tikzcd}$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$x \xrightarrow{R} y_1$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd}
 & & x \arrow[dl, "R"'] \arrow[dr, "R"] & & \\
\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\} & y_1 & & y_2 & \mathcal{L}(y_2) = \{D\}
\end{tikzcd}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\ y_1 & & y_2 \end{tikzcd}$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\} \qquad \mathcal{L}(y_2) = \{D\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y_1 & & y_2
\end{tikzcd}$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\}$$

$$\mathcal{L}(y_2) = \{D, \neg C \sqcup \neg D\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes. w

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\ y_1 & & y_2 \end{tikzcd}$$

$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\} \qquad \mathcal{L}(y_2) = \{D, {\color{red}\neg C \sqcup \neg D}\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y_1 & & y_2
\end{tikzcd}$$

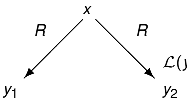
$$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\} \qquad \mathcal{L}(y_2) = \{D, \neg C \sqcup \neg D, \neg C\}$$

# Example

Is $\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D$ satisfiable? Yes.

$$\mathcal{L}(x) = \{\exists R.C, \forall R.(\neg C \sqcup \neg D), \exists R.D\}$$

$\mathcal{L}(y_1) = \{C, \neg C \sqcup \neg D, \neg D\}$




$\mathcal{L}(y_2) = \{D, \neg C \sqcup \neg D, \neg C\}$

- Finished. No more rules applicable and the tableau is complete and clash-free
- Hence, the concept is satisfiable
- The tree corresponds to a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$
  - The nodes are the elements of the domain: $\Delta^{\mathcal{I}} = \{x, y_1, y_2\}$
  - For each atomic concept $A$, set $A^{\mathcal{I}} = \{z \mid A \in \mathcal{L}(z)\}$
    - $C^{\mathcal{I}} = \{y_1\}$, $D^{\mathcal{I}} = \{y_2\}$
  - For each role $R$, set $R^{\mathcal{I}} = \{\langle x, y\rangle \mid$ there is an edge labeled $R$ from $x$ to $y\}$
    - $R^{\mathcal{I}} = \{\langle x, y_1\rangle, \langle x, y_2\rangle\}$
  - It can be shown that $x \in (\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D)^{\mathcal{I}} \neq \emptyset$

# Soundness and Completeness

## Theorem

*Let $\mathcal{A}$ be an $\mathcal{ALC}$ ABox and F a completion-forest obtained by applying the tableau rules to $\mathcal{A}$. Then*

1. *The rule application terminates;*
2. *If F is clash-free and complete, then F defines a (canonical) (tree) model for $\mathcal{A}$; and*
3. *If $\mathcal{A}$ has a model $\mathcal{I}$, then the rules can be applied such that they yield a clash-free and complete completion-forest.*

# KBs with GCIs

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable with $\mathcal{T} \neq \emptyset$?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x.\neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{rcl}
\mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\
\mathcal{A} & = & \{umberto{:}Human\}
\end{array}
$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$ $umberto$

# KBs with GCIs

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable with $\mathcal{T} \neq \emptyset$?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\ \mathcal{A} & = & \{umberto{:}Human\} \end{array}$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$ *umberto*

# KB Satisfiability

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\ \mathcal{A} & = & \{umberto{:}Human\} \end{array}$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$ *umberto*

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x.\neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\ \mathcal{A} & = & \{umberto{:}Human\} \end{array}$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \neg Human\}$ *umberto*

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x.\neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\ \mathcal{A} & = & \{umberto{:}Human\} \end{array}$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \neg Human\}$ *umberto*

Clash

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x.\neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{rcl}
\mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\
\mathcal{A} & = & \{umberto{:}Human\}
\end{array}
$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$ umberto

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{rcl}
\mathcal{T} & = & \{ \mathit{Human} \sqsubseteq \exists \mathit{hasMother}.\mathit{Human} \} \\
\mathcal{A} & = & \{ \mathit{umberto}{:}\mathit{Human} \}
\end{array}
$$

$\mathcal{L}(\mathit{umberto}) = \{ \mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human} \}$ *umberto*

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{rcl}
\mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\
\mathcal{A} & = & \{umberto{:}Human\}
\end{array}
$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$ *umberto*

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\ \mathcal{A} & = & \{umberto{:}Human\} \end{array}$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$\mathcal{L}(y_1) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$

$$\begin{tikzcd} umberto \arrow[d, "hasMother"'] \\ y_1 \end{tikzcd}$$

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
    - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
    - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{rcl}
\mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\
\mathcal{A} & = & \{umberto{:}Human\}
\end{array}
$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$\mathcal{L}(y_1) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$

$$
\begin{tikzcd}
umberto \arrow[d, "hasMother"'] \\
y_1
\end{tikzcd}
$$

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\ \mathcal{A} & = & \{umberto{:}Human\} \end{array}$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$\mathcal{L}(y_1) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$$\begin{tikzcd} umberto \arrow[d, "hasMother"'] \\ y_1 \end{tikzcd}$$

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{rcl}
\mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\
\mathcal{A} & = & \{umberto{:}Human\}
\end{array}
$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$\mathcal{L}(y_1) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$\mathcal{L}(y_2) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$

$$
\begin{tikzcd}
umberto \arrow[d, "hasMother"'] \\
y_1 \arrow[d, "hasMother"'] \\
y_2
\end{tikzcd}
$$

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{rcl}
\mathcal{T} & = & \{Human \sqsubseteq \exists hasMother.Human\} \\
\mathcal{A} & = & \{umberto{:}Human\}
\end{array}
$$

$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$\mathcal{L}(y_1) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$

$\mathcal{L}(y_2) = \{Human, \neg Human \sqcup \exists hasMother.Human\}$

$$
\begin{tikzcd}
umberto \arrow[d, "hasMother"'] \\
y_1 \arrow[d, "hasMother"'] \\
y_2
\end{tikzcd}
$$

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x. \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{\mathit{Human} \sqsubseteq \exists \mathit{hasMother}.\mathit{Human}\} \\ \mathcal{A} & = & \{\mathit{umberto}{:}\mathit{Human}\} \end{array}$$

$\mathcal{L}(\mathit{umberto}) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$\mathcal{L}(y_1) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$\mathcal{L}(y_2) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$$\begin{tikzcd} \mathit{umberto} \arrow[d, "\mathit{hasMother}"'] \\ y_1 \arrow[d, "\mathit{hasMother}"'] \\ y_2 \end{tikzcd}$$

- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x . \neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$
\begin{array}{lcl}
\mathcal{T} & = & \{\mathit{Human} \sqsubseteq \exists \mathit{hasMother}.\mathit{Human}\} \\
\mathcal{A} & = & \{\mathit{umberto}{:}\mathit{Human}\}
\end{array}
$$

$\mathcal{L}(\mathit{umberto}) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$\mathcal{L}(y_1) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$\mathcal{L}(y_2) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$$
\begin{tikzcd}
\mathit{umberto} \arrow[d, "\mathit{hasMother}"'] \\
y_1 \arrow[d, "\mathit{hasMother}"'] \\
y_2
\end{tikzcd}
$$

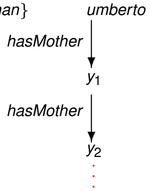
- We have seen how to test the satisfiability of an ABox $\mathcal{A}$
- But, how can we check if a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable?
- Basic idea: since $t(C \sqsubseteq D) \equiv \forall x.\neg t(C, x) \vee t(D, x)$
  - we use the rule: for each $C \sqsubseteq D \in \mathcal{T}$, add $\neg C \sqcup D$ to every node
- But, termination is not guaranteed
  - E.g., consider $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{\mathit{Human} \sqsubseteq \exists \mathit{hasMother}.\mathit{Human}\} \\ \mathcal{A} & = & \{\mathit{umberto}{:}\mathit{Human}\} \end{array}$$

$\mathcal{L}(\mathit{umberto}) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$\mathcal{L}(y_1) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$

$\mathcal{L}(y_2) = \{\mathit{Human}, \neg \mathit{Human} \sqcup \exists \mathit{hasMother}.\mathit{Human}, \exists \mathit{hasMother}.\mathit{Human}\}$



# Node Blocking in $\mathcal{ALC}$

- When creating new node, check ancestors for equal label set
- If such a node is found, new node is blocked
- No rule is applied to blocked nodes

$$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$$

$$\mathcal{L}(y_1) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$$

$$\mathcal{L}(y_2) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$$

$$\begin{tikzcd} umberto \arrow[d, "hasMother"'] \\ y_1 \arrow[d, "hasMother"'] \\ y_2 \ \text{blocked}: \mathcal{L}(y_1) = \mathcal{L}(y_2) \end{tikzcd}$$

# Node Blocking in $\mathcal{ALC}$

- When creating new node, check ancestors for equal label set
- If such a node is found, new node is blocked
- No rule is applied to blocked nodes

$$\mathcal{L}(umberto) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$$

$$\mathcal{L}(y_1) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$$

$$\mathcal{L}(y_2) = \{Human, \neg Human \sqcup \exists hasMother.Human, \exists hasMother.Human\}$$

$$\begin{tikzcd}
umberto \arrow[d, "hasMother"'] \\
y_1 \arrow[d, "hasMother"'] \\
y_2 \arrow[u, bend right, "hasMother"'] & \text{blocked: } \mathcal{L}(y_1) = \mathcal{L}(y_2)
\end{tikzcd}$$

- Block represents cyclical model
  - $\Delta^{\mathcal{I}} = \{umberto, y_1, y_2\}$
  - $Human^{\mathcal{I}} = \{umberto, y_1, y_2\}$
  - $hasMother^{\mathcal{I}} = \{\langle umberto, y_1\rangle, \langle y_1, y_2\rangle, \langle y_2, y_1\rangle\}$

# Blocking in $\mathcal{ALC}$

- A non-root node $x$ is blocked if for some ancestor $y$, $y$ is blocked or $\mathcal{L}(x) = \mathcal{L}(y)$, where $y$ is not a root node
- A blocked node $x$ is indirectly blocked if its predecessor is blocked, otherwise it is directly blocked
- If $x$ is directly blocked, it has a unique ancestor $y$ such that $\mathcal{L}(x) = \mathcal{L}(y)$
- if there existed another ancestor $z$ such that $\mathcal{L}(x) = \mathcal{L}(z)$ then either $y$ or $z$ must be blocked
- If $x$ is directly blocked and $y$ is the unique ancestor such that $\mathcal{L}(x) = \mathcal{L}(y)$, we will say that $y$ blocks $x$

# $\mathcal{ALC}$ Tableau rules with GCI's

| Rule | | Description |
|---|---|---|
| $(\sqcap)$ | if 1. | $C_1 \sqcap C_2 \in \mathcal{L}(x)$, $x$ is not indirectly blocked 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. | $C_1 \sqcup C_2 \in \mathcal{L}(x)$, $x$ is not indirectly blocked 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. | $\exists R.C \in \mathcal{L}(x)$, $x$ is not blocked 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. | $\forall R.C \in \mathcal{L}(x)$, $x$ is not indirectly blocked and |
| | 2. | $x$ has an $R$-successor $y$ with $C \notin \mathcal{L}(y)$ |
| | then | $\mathcal{L}(y) \to \mathcal{L}(y) \cup \{C\}$ |
| $(\sqsubseteq)$ | if 1. | $C \sqsubseteq D \in \mathcal{T}$, $x$ is not indirectly blocked and |
| | 2. | $\{nnf(\neg C), D\} \cap \mathcal{L}(x) = \emptyset$ |
| | then | $\mathcal{L}(x) \to \mathcal{L}(x) \cup \{E\}$ for some |
| | | $E \in \{nnf(\neg C), D\}$ ($nnf(\neg C)$ is NNF of $\neg C$) |

# Soundness and Completeness

**Theorem**

*Let $\mathcal{K}$ be an $\mathcal{ALC}$ KB and $F$ a completion-forest obtained by applying the tableau rules to $\mathcal{K}$. Then*

1. *The rule application terminates;*
2. *If $F$ is clash-free and complete, then $F$ defines a (canonical) (tree) model for $\mathcal{K}$; and*
3. *If $\mathcal{K}$ has a model $\mathcal{I}$, then the rules can be applied such that they yield a clash-free and complete completion-forest.*

# Fuzzy DLs Basics

- We have seen how to "fuzzify" classical sets and FOL
  - Fuzzy statements are of the form $\langle \phi, n \rangle$, where $\phi$ is a statement and $n \in [0, 1]$
- The natural extension to fuzzy DLs consists then in replacing $\phi$ with a DL expression
- Several fuzzy variants of DLs have been proposed: they can be classified according to
  - The DL resp. ontology language that they generalize
  - The allowed fuzzy constructs
  - The underlying fuzzy logic
  - Their reasoning algorithms and computational complexity results

- In classical DLs, a concept $C$ is interpreted by an interpretation $\mathcal{I}$ as a set of individuals
- In fuzzy DLs, a concept $C$ is interpreted by $\mathcal{I}$ as a fuzzy set of individuals
- Each individual is instance of a concept to a degree in $[0, 1]$
- Each pair of individuals is instance of a role to a degree in $[0, 1]$

- $\langle a{:}C, n\rangle$ states that $a$ is an instance of concept/class $C$ with degree at least $n$
- $\langle (a, b){:}R, n\rangle$ states that $\langle a, b\rangle$ is an instance of relation $R$ with degree at least $n$
- $\langle C_1 \sqsubseteq C_2, n\rangle$ states a vague subsumption relationship
  - The FOL statement $\forall x.C_1(x) \to C_2(x)$ is true to degree at least $n$
- Note: one may find also fuzzy DL expressions $\langle \alpha \geq n\rangle$, $\langle \alpha \leq n\rangle$, $\langle \alpha > n\rangle$, $\langle \alpha < n\rangle$, and $\langle \alpha = n\rangle$
- We use the form $\langle \alpha, n\rangle$, i.e. $\langle \alpha \geq n\rangle$ only
  - Remind that graded axioms are intended to be produced semi- or automatically
  - Hardly they may have the form $\langle \alpha \leq n\rangle$, $\langle \alpha > n\rangle$ or $\langle \alpha < n\rangle$

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

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

General Inclusion Axioms: $\langle C \sqsubseteq D, r\rangle$,

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

# Some Remarks

- Like for fuzzy FOL, $\forall$ and $\exists$ are not complementary in general: i.e. $\forall R.C \not\equiv \neg\exists R.\neg C$
- $\forall R.C \equiv \neg\exists R.\neg C$ under Łukasiewicz logic and SFL
- $\langle C \sqsubseteq D, n\rangle$ may be rewritten as $\langle \top \sqsubseteq C \rightarrow D, n\rangle$
- In early works, a fuzzy GCI is of the form $C \sqsubseteq D$ with semantics:
  - $\mathcal{I}$ is a model of $C \sqsubseteq D$ iff for every $x \in \Delta^{\mathcal{I}}$ we have that $C^{\mathcal{I}}(x) \leq D^{\mathcal{I}}(x)$
  - This is the same of fuzzy axiom $\langle \top \sqsubseteq C \rightarrow_x D, 1\rangle$, where $\rightarrow_x$ is an $r$-implication
- Disjointness: use $\langle C \sqcap D \sqsubseteq \bot, 1\rangle$ rather than $\langle C \sqsubseteq \neg D, 1\rangle$
  - they are not the same, e.g. $\langle A \sqsubseteq \neg A, 1\rangle$ says that $A^{\mathcal{I}}(x) \leq 0.5$, for all $\mathcal{I}$ and for all $x \in \Delta^{\mathcal{I}}$

# Witnessed Interpretation

- Witnessed Interpretation:
  - Infima and suprema are attained at some point

$$
\begin{array}{rcl}
(\exists R.C)^{\mathcal{I}}(x) & = & R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y) \text{ for some } y \in \Delta^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}}(x) & = & R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y) \text{ for some } y \in \Delta^{\mathcal{I}} \\
(C \sqsubseteq D)^{\mathcal{I}} & = & C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x) \text{ for some } x \in \Delta^{\mathcal{I}}
\end{array}
$$

- It is customary to stick to witnessed interpretations only

- Fuzzy knowledge base: $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$
  - $\mathcal{T}$ is a fuzzy TBox, that is a finite set of fuzzy GCI
  - $\mathcal{A}$ is a *fuzzy ABox*, that is a finite set of fuzzy assertions
- Acyclic fuzzy ontologies: TBox with axioms of the form

$$\begin{array}{lll} A & \sqsubseteq_n & C \text{ (primitive GCI)} \\ A & \tilde{\sqsubseteq} & C \text{ (primitive GCI)} \\ A & \tilde{=} & C \text{ (definitional GCI)} \end{array}$$

  - $A$ concept name
  - $A \sqsubseteq_n C$ shorthand for $\langle \top \sqsubseteq A \rightarrow C, n \rangle$
  - No nominal $\{a\}$ occurs in the TBox

- We say that
  - concept name $A$ directly uses concept name $B$ w.r.t. $\mathcal{T}$, denoted $A \to_{\mathcal{T}} B$, if $A$ is the head of some axiom $\tau \in \mathcal{T}$ such that $B$ occurs in the body of $\tau$
  - concept name $A$ uses concept name $B$ w.r.t. $\mathcal{T}$, denoted $A \rightsquigarrow_{\mathcal{T}} B$, if there exist concept names $A_1, \ldots, A_n$, such that $A_1 = A$, $A_n = B$ and, for every $1 \leq i < n$, it holds that $A_i \to_{\mathcal{T}} A_{i+1}$
- TBox $\mathcal{T}$ is cyclic (acyclic) if there is (no) $A$ such that $A \rightsquigarrow_{\mathcal{T}} A$
- TBox $\mathcal{T}$ is unfoldable if
  - $\mathcal{T}$ is acyclic
  - If $A \tilde{=} C \in \mathcal{T}$ then $A$ does not occur in the head of any other axiom

- $\mathcal{I}$ satisfies (is a model of) $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ iff it satisfies each element in $\mathcal{A}$ and $\mathcal{T}$
- A fuzzy KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ entails an axiom $E$, denoted $\mathcal{K} \models E$, iff every model of $\mathcal{K}$ satisfies $E$
- We say that two concepts $C$ and $D$ are equivalent, denoted $C \equiv_{\mathcal{K}} D$ iff in every model $\mathcal{I}$ of $\mathcal{K}$ and for all $x \in \Delta^{\mathcal{I}}$, $C^{\mathcal{I}}(x) = D^{\mathcal{I}}(x)$
- Best entailment degree: for assertion of GCI $\phi$

$$bed(\mathcal{K}, \phi) = \sup \{r \mid \mathcal{K} \models \langle \phi, r \rangle\}$$

- Best satisfiability degree: for concept $C$

$$bsd(\mathcal{K}, C) = \sup_{\mathcal{I} \models \mathcal{K}} \sup_{x \in \Delta^{\mathcal{I}}} C^{\mathcal{I}}(x) .$$

# Some Salient Fuzzy Concept Equivalences

| Property | Łukasiewicz | Gödel | Product | SFL |
|---|---|---|---|---|
| $C \sqcap \neg C \equiv \bot$ | • | • | • | |
| $C \sqcup \neg C \equiv \top$ | • | | | |
| $C \sqcap C \equiv C$ | | • | | • |
| $C \sqcup C \equiv C$ | | • | | • |
| $\neg \neg C \equiv C$ | • | | | • |
| $C \rightarrow D \equiv \neg C \sqcup D$ | • | | | • |
| $C \rightarrow D \equiv \neg D \rightarrow \neg C$ | • | | | • |
| $\neg (C \rightarrow D) \equiv C \sqcap \neg D$ | • | | | • |
| $\neg (C \sqcap D) \equiv \neg C \sqcup \neg D$ | • | • | • | • |
| $\neg (C \sqcup D) \equiv \neg C \sqcap \neg D$ | • | • | • | • |
| $C \sqcap (D \sqcup E) \equiv (C \sqcap D) \sqcup (C \sqcap E)$ | | • | | • |
| $C \sqcup (D \sqcap E) \equiv (C \sqcup D) \sqcap (C \sqcup E)$ | | • | | • |
| $\exists R.C \equiv \neg \forall R.\neg C$ | • | | | • |

# Towards Fuzzy OWL 2 and its Profiles

- Recall that OWL 2 relates to $\mathcal{SROIQ}(D)$
- We need to extend the semantics 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

# Number Restrictions, Inverse, Transitive roles, . . .

- The semantics of the concept $(\geq n\ R.C)$ is: $\wedge$ interpreted as Gödel t-norm

$$\exists y_1, \ldots, y_n. \bigwedge_{i=1}^{n} R(x, y_i) \wedge C(y_i) \wedge \bigwedge_{1 \leq i < j \leq n} y_i \neq y_j$$

- The semantics of the concept $(\leq n\ R.C)$ is: $\wedge$ interpreted as Gödel t-norm

$$(\leq n\ R)^{\mathcal{I}}(x) = \forall y_1, \ldots, y_{n+1}. \bigwedge_{i=1}^{n+1} (R(x, y_i) \wedge C(y_i)) \Rightarrow \bigvee_{1 \leq i < j \leq n+1} y_i = y_j$$

- Note: $(\geq 1\ R) \equiv \exists R.\top$
- For transitive roles $R$ we impose: for all $x, y \in \Delta^{\mathcal{I}}$

$$R^{\mathcal{I}}(x, y) \geq \sup_{z \in \Delta^{\mathcal{I}}} R^{\mathcal{I}}(x, z) \otimes R^{\mathcal{I}}(z, y)$$

- For inverse roles we have for all $x, y \in \Delta^{\mathcal{I}}$

$$R^{\mathcal{I}}(x, y) = R^{\mathcal{I}}(y, x)$$

- The semantics of fucntional roles $fun(R)$ is

$$\forall x \forall y \forall z.\ R(x, y) \wedge R(x, z) \Rightarrow y = z$$

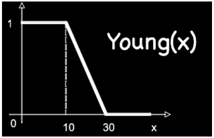
- Similar for other $\mathcal{SROIQ}$ constructs

# Fuzzy Concrete Domains

- E.g., *Small*, *Young*, *High*, *etc*. with explicit membership function
- Use the idea of concrete domains:
  - $D = \langle \Delta_D, \Phi_D \rangle$
  - $\Delta_D$ is an interpretation domain
  - $\Phi_D$ is the set of concrete unary fuzzy domain predicates $d$ and fixed interpretation $d^D \colon \Delta_D \to [0, 1]$
- Specifically,

$$\begin{array}{lcl} \mathbf{d} & \to & ls(a, b) \mid rs(a, b) \mid tri(a, b, c) \mid trz(a, b, c, d) \\ & & \mid \ \geq_v \ \mid \ \leq_v \ \mid \ =_v \\ \\ C, D & \to & \forall T.\mathbf{d} \mid \exists T.\mathbf{d} \end{array}$$

- Representation of Young Person:




$$
\begin{aligned}
Minor &= Person \sqcap \exists hasAge. \leq_{18} \\
YoungPerson &= Person \sqcap \exists hasAge.ls(10, 30)
\end{aligned}
$$

- Representation of Heavy Rain:

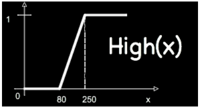
$$HeavyRain = Rain \sqcap \exists hasPrecipitationRate.rs(5, 7.5)$$

# Modifiers

- *Very*, *moreOrLess*, *slightly*, etc.
- Fuzzy modifier $m$ with function $f_m \colon [0,1] \to [0,1]$

$$C \quad \to \quad m(C) \mid \forall T.m(\mathbf{d}) \mid \exists T.m(\mathbf{d})$$

where $m$ is a linear modifier
- Representation of Sport Car




$$SportsCar \quad = \quad Car \sqcap \exists speed.very(rs(80,250))$$

- Representation of Very Heavy Rain

$$VeryHeavyRain = Rain \sqcap \exists hasPrecipitationRate.very(rs(5,7.5)) \ .$$

# Aggregation Operators

- Aggregation operators: aggregate concepts, using functions such as the mean, median, weighted sum operators
- Given an $n$-ary aggregation operator $@ : [0,1]^n \to [0,1]$
  - We fuzzy concepts by allowing to apply @ to $n$ concepts $C_1, \ldots, C_n$, i.e.

$$C \quad \to \quad @(C_1, \ldots, C_n)$$

- Semantics:

$$@(C_1, \ldots, C_n)^{\mathcal{I}}(x) \quad = \quad @(C_1^{\mathcal{I}}(x), \ldots, C_n^{\mathcal{I}}(x)) \ .$$

- Allows to express the concept

$$GoodHotel = 0.3 \cdot ExpensiveHotel + 0.7 \cdot LuxuriousHotel$$

  - The membership function of good hotels is the weighted sum of being an expensive and luxurious hotel

# Some Applications

- Information retrieval
- Recommendation systems
- Image interpretation
- Ambient intelligence
- Ontology merging
- Matchmaking
- decision making
- Summarization
- Robotics perception
- Software design
- Machine learning

# Example (Graded Entailment)

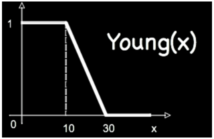
|  |  |  |
|---|---|---|
| *audi_tt* | *mg* | *ferrari_enzo* |

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

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

$$\begin{array}{lcl} \mathcal{K} & \models & \langle ferrari\_enzo{:}SportsCar, 1\rangle \\ \mathcal{K} & \models & \langle audi\_tt{:}SportsCar, 0.92\rangle \\ \mathcal{K} & \models & \langle mg{:}\neg SportsCar, 0.72\rangle \end{array}$$

# Example (Graded Subsumption)

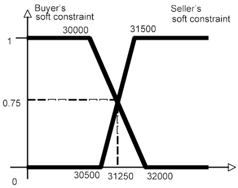



$$
\begin{array}{rcl}
Minor & = & Person \sqcap \exists hasAge. \leq_{18} \\
YoungPerson & = & Person \sqcap \exists hasAge.Young \\
 & & fun(hasAge)
\end{array}
$$

$$\mathcal{K} \models \langle Minor \sqsubseteq YoungPerson, 0.6 \rangle$$

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

# Example (Simplified Matchmaking)




- 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 sets: 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 €

$$\begin{array}{lcl} \mathit{AudiTT} & = & \mathit{SportsCar} \sqcap \exists \mathit{hasPrice}.\mathit{rs}(30500, 31500) \\ \mathit{Query} & = & \mathit{SportsCar} \sqcap \exists \mathit{hasPrice}.\mathit{ls}(30000, 32000) \end{array}$$

  - Highest degree to which the concept
    $C = \mathit{AudiTT} \sqcap \mathit{Query}$
    is satisfiable is 0.75 (the degree to which the Audi TT and the query matches is 0.75)
  - The car may be sold at 31250 €

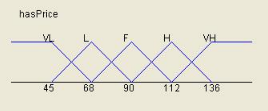
# Example: Learning fuzzy GCIs from data

- Learning of fuzzy GCIs from crisp data
- Use Case: What are Good hotels, using TripAdvisor data?
  - Given
    - OWL 2 Ontology about meaningful city entities and their descriptions
    - TripAdvisor data about hotels and user judgments
  - We may learn that in e.g., Pisa, Italy

$$\langle \exists hasAmenity.Babysitting \sqcap \exists hasPrice.fair \sqsubseteq Good\_Hotel, 0.282 \rangle$$

*"A hotel having babysitting as amenity and a fair price is a good hotel (to degree 0.282)"*

  - Real valued price attribute *hasPrice* has been automatically fuzzyfied



# Example: Multi-Criteria Decision Making

- We have to select among two sites, $A_1$ , $A_2$
- There are two criteria ($C_1$ -Transportation Issues, and $C_2$ -Public Nuisance) for judgement
- There are two experts ($E_1$ , $E_2$) that make judgments
- The decision matrix of the experts is shown below:

| $E_1$ | | Criteria | |
|---|---|---|---|
| | | 0.48 | 0.52 |
| Alter. | | $C_1$ | $C_2$ |
| $x_1$ | $A_1$ | $tri(0.6, 0.7, 0.8)$ | $tri(0.9, 0.95, 1.0)$ |
| $x_2$ | $A_2$ | $tri(0.6, 0.7, 0.8)$ | $tri(0.4, 0.5, 0.6)$ |

| $E_2$ | | Criteria | |
|---|---|---|---|
| | | 0.52 | 0.48 |
| Alter. | | $C_1$ | $C_2$ |
| $x_1$ | $A_1$ | $tri(0.55, 0.6, 0.7)$ | $tri(0.4, 0.45, 0.5)$ |
| $x_2$ | $A_2$ | $tri(0.35, 0.4, 0.45)$ | $tri(0.5, 0.55, 0.6)$ |

- For each expert $k = 1, 2$, for each alternative $i = 1, 2$ and for each criteria $j = 1, 2$, we define the concept

$$P^k_{ij} = \exists \text{hasScore}.a^k_{ij}$$

- Now, for each expert $k$ and alternative $i$, we define the weighted concept

$$A^k_i = w^k_1 \cdot P^k_{i1} + w^k_2 \cdot P^k_{i2}$$

- Finally, we combine the two experts outcome, by defining the weighted concept

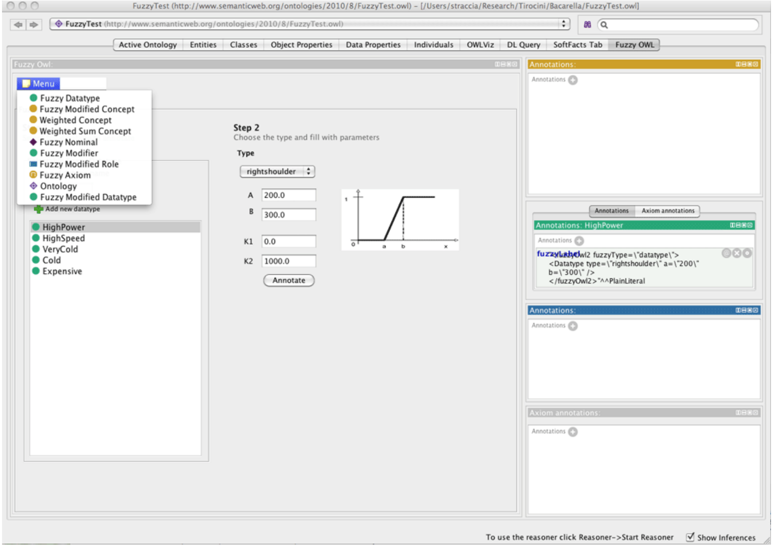
$$A_i = 0.5 \cdot A^1_i + 0.5 \cdot A^2_i$$

- It can be verified that $rv(\mathcal{K}, A_1) = bsd(\mathcal{K}, A_1) = 0.26$ and $rv(\mathcal{K}, A_2) = bsd(\mathcal{K}, A_2) = 0.37$

# Representing Fuzzy OWL Ontologies in OWL

- OWL 2 is W3C standard, with classical logic semantics
  - Hence, cannot support natively Fuzzy Logic
- However, Fuzzy OWL 2, has been defined using OWL 2
  - Uses the axiom annotation feature of OWL 2
- Any Fuzzy OWL 2 ontology is a legal OWL 2 ontology

- A java parser for Fuzzy OWL 2 exists
- Protégé plug-in exists to encode Fuzzy OWL ontologies



# Reasoning Problems and Algorithms

Consistency problem:

- Is $\mathcal{K}$ satisfiable?
- Is $C$ coherent, i.e. is $C^{\mathcal{I}}(x) > 0$ for some $\mathcal{I} \models \mathcal{K}$ and $x \in \Delta^{\mathcal{I}}$?

Instance checking problem:

- Does $\mathcal{K} \models \langle a{:}C, n \rangle$ hold?

Subsumption problem:

- Does $\mathcal{K} \models \langle C \sqsubseteq D, n \rangle$ hold?

Best entailment degree problem:

- What is $bed(\mathcal{K}, \phi)$?

Best satisfiability degree problem:

- What is $bsd(\mathcal{K}, \phi)$?

Instance retrieval problem:

- Compute the set $\{\langle a, n \rangle \mid n = bed(\mathcal{K}, a{:}C)\}$

Top-k retrieval problem:

- Compute the top-k ranked elements of $\{\langle a, n \rangle \mid n = bed(\mathcal{K}, a{:}C)\}$

# Some Reductions

- $\mathcal{K}$ is satisfiable iff *bsd*$(\mathcal{K}, a{:}\bot) > 0$, where $a$ is a new individual.
- $C$ is coherent w.r.t. $\mathcal{K}$ if one of the following holds:
  - $\mathcal{K} \cup \{\langle a{:}C > 0\rangle\}$ is satisfiable, where $a$ is a new individual
  - $\mathcal{K} \not\models \langle C \sqsubseteq \bot, 1\rangle$
  - $bsd(\mathcal{K}, C) > 0$
- $\mathcal{K} \models \langle a{:}C, n\rangle$ if one of the following holds:
  - $\mathcal{K} \cup \{\langle a{:}C < n\rangle\}$ is not satisfiable
  - $bed(\mathcal{K}, a{:}C) \geq n$
- $\mathcal{K} \models \langle C \sqsubseteq D, n\rangle$ if one of the following holds:
  - $\mathcal{K} \cup \{\langle a{:}C \to D < n\rangle\}$ is not satisfiable, where $a$ is a new individual
  - $bed(\mathcal{K}, C \sqsubseteq D) \geq n$
- We have that

$$\begin{array}{lcl} bed(\mathcal{K}, \phi) & = & \text{min } x. \text{ such that } \mathcal{K} \cup \{\langle \phi \leq x\rangle\} \text{ satisfiable} \\ bsd(\mathcal{K}, \phi) & = & \text{max } x. \text{ such that } \mathcal{K} \cup \{\langle \phi \geq x\rangle\} \text{ satisfiable} \end{array}$$

# Reasoning in Fuzzy DLs: Basics

- Algorithms for fuzzy DLs: are a mixture of classical DLs reasoning algorithms and algorithms for Mathematical Fuzzy Logic
- Fuzzy OWL 2:
  - Fuzzy tableaux based algorithms
    - Tableaux and non deterministic tableaux
    - Operational Research
  - Reduction into classical DLs
- Fuzzy OWL 2 EL: fuzzy structural based algorithms
- Fuzzy OWL 2 QL: fuzzy query rewriting based algorithms
- fuzzy OWL 2 RL: fuzzy logic programming based algorithms

# OR Fuzzy Tableaux: $\mathcal{ALC}$ under SFL over $[0, 1]$

- Works as for classical $\mathcal{ALC}$ on completion forests
  - Blocking is as for classical $\mathcal{ALC}$
  - The completion forest is expanded by repeatedly applying inference rules
  - The completion-forest is complete when none of the rules are applicable
- Additionally, at each inference step we add equational constraints that have to hold
- Eventually, the initial KB is satisfiable if the final set of equational constraints has a solution
  - For the latter case, we may use a MILP solver

| e | | **Description** |
|---|---|---|
| ) | | 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 $\mathcal{C}_{\mathcal{F}}$ |
| | if | $\neg A \in \mathcal{L}(v)$ then add $x_{v:A} = 1 - x_{v:\neg A}$ to $\mathcal{C}_{\mathcal{F}}$ |
| ) | If | $\bot \in \mathcal{L}(v)$ then add $x_{v:\bot} = 0$ to $\mathcal{C}_{\mathcal{F}}$ |
| ) | If | $\top \in \mathcal{L}(v)$ then add $x_{v:\top} = 1$ to $\mathcal{C}_{\mathcal{F}}$ |
| | if | $C_1 \sqcap C_2 \in \mathcal{L}(v)$, $v$ is not indirectly blocked |
| | then | $\mathcal{L}(v) \to \mathcal{L}(v) \cup \{C_1, C_2\}$, and add $x_{v:C_1} \otimes x_{v:C_2} \geq x_{v:C_1 \sqcap C_2}$ to $\mathcal{C}_{\mathcal{F}}$ |
| | if | $C_1 \sqcup C_2 \in \mathcal{L}(v)$, $v$ is not indirectly blocked |
| | then | $\mathcal{L}(v) \to \mathcal{L}(v) \cup \{C_1, C_2\}$, and add $x_{v:C_1} \oplus x_{v:C_2} \geq x_{v:C_1 \sqcup C_2}$ to $\mathcal{C}_{\mathcal{F}}$ |
| | if | $\forall R.C \in \mathcal{L}(v)$, $v$ is not indirectly blocked |
| | then | $\mathcal{L}(w) \to \mathcal{L}(w) \cup \{C\}$, and add $x_{w:C} \geq x_{v:\forall R.C} \otimes x_{(v, w):R}$ to $\mathcal{C}_{\mathcal{F}}$ |
| | 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 $\mathcal{C}_{\mathcal{F}}$ |
| ) | if | $\langle C \sqsubseteq D, n \rangle \in \mathcal{T}$, $v$ is not indirectly blocked |
| | then | $\mathcal{L}(v) \to \mathcal{L}(v) \cup \{C, D\}$, and add $x_{v:D} \geq x_{v:C} \otimes n$ to $\mathcal{C}_{\mathcal{F}}$ |

# Analytical Fuzzy Tableaux: $\mathcal{ALC}$ under SFL over $[0, 1]$

- Works as for classical $\mathcal{ALC}$ on completion forests
  - Node labels $\mathcal{L}(v)$ contain, rather than DL concept expressions, expressions of the form $\langle C, n\rangle$

    *"The truth degree of being v instance of C is $\geq n$"*

  - Blocking is as for classical $\mathcal{ALC}$
  - The completion forest is expanded by repeatedly applying inference rules
  - The completion-forest is complete when none of the rules are applicable
- Additionally, we adapt the notion of clash: a clash is either
  - $\langle \perp, n\rangle$ with $n > 0$; or
  - a pair $\langle C, n\rangle$ and $\langle \neg C, m\rangle$ with $n > 1 - m$
- Eventually, the initial KB is satisfiable if there is a clash-free complete completion forest

- ($\sqcap$). If *(i)* $\langle C_1 \sqcap C_2, n\rangle \in \mathcal{L}(v)$, *(ii)* $\{\langle C_1, n\rangle, \langle C_2, n\rangle\} \not\subseteq \mathcal{L}(v)$, and *(iii)* node $v$ is not indirectly blocked, then add $\langle C_1, n\rangle$ and $\langle C_2, n\rangle$ to $\mathcal{L}(v)$.
- ($\sqcup$). If *(i)* $\langle C_1 \sqcup C_2, n\rangle \in \mathcal{L}(v)$, *(ii)* $\{\langle C_1, n\rangle, \langle C_2, n\rangle\} \cap \mathcal{L}(v) = \emptyset$, and *(iii)* node $v$ is not indirectly blocked, then add some $\langle C, n\rangle \in \{\langle C_1, n\rangle, \langle C_2, n\rangle\}$ to $\mathcal{L}(v)$.
- ($\forall$). If *(i)* $\langle \forall R.C, n\rangle \in \mathcal{L}(v)$, *(ii)* $\langle R, m\rangle \in \mathcal{L}(\langle v, w\rangle)$ with $m > 1 - n$, *(iii)* $\langle C, n\rangle \notin \mathcal{L}(w)$, and *(iv)* node $v$ is not indirectly blocked, then add $\langle C, n\rangle$ to $\mathcal{L}(w)$.
- ($\exists$). If *(i)* $\langle \exists R.C, n\rangle \in \mathcal{L}(v)$, *(ii)* there is no $\langle R, n_1\rangle \in \mathcal{L}(\langle v, w\rangle)$ with $\langle C, n_2\rangle \in \mathcal{L}(w)$ such that $\min(n_1, n_2) \geq n$, and *(iii)* node $v$ is not blocked, then create a new node $w$, add $\langle R, n\rangle$ to $\mathcal{L}(\langle v, w\rangle)$ and add $\langle C, n\rangle$ to $\mathcal{L}(w)$.
- ($\sqsubseteq$). If *(i)* $\langle \top \sqsubseteq D, n\rangle \in \mathcal{T}$, *(ii)* $\langle D, n\rangle \notin \mathcal{L}(v)$, and *(iii)* node $v$ is not indirectly blocked, then add $\langle D, n\rangle$ to $\mathcal{L}(v)$.

# Non-Deterministic Analytic Fuzzy Tableaux

- It's a combination of the analogous method for fuzzy propositional logic and analytical fuzzy tableau
- Rule example:

  ($\sqcap$). If *(i)* $\langle C_1 \sqcap C_2, m\rangle \in \mathcal{L}(v)$, *(ii)* there are $m_1, m_2 \in L_n$ such that $m_1 \otimes m_2 = m$ with $\{\langle C_1, m_1\rangle, \langle C_2, m_2\rangle\} \not\subseteq \mathcal{L}(v)$, and *(iii)* node $v$ is not indirectly blocked, then add $\langle C_1, m_1\rangle$ and $\langle C_2, m_2\rangle$ to $\mathcal{L}(v)$

# Reduction to Classical DLs

- Same principle as for the reduction for propositional fuzzy logic
- Needs adaption to the DL constructs: e.g. $\exists, \forall$ and $\sqsubseteq$
- Examples of reduction rules for SFL:

$$
\begin{array}{rcl}
\rho(A, \geq \gamma) & = & A_{\geq \gamma} \\
\rho(C \sqcap D, \geq \gamma) & = & \rho(C, \geq \gamma) \sqcap \rho(D, \geq \gamma) \\
\rho(C \sqcap D, \leq \gamma) & = & \rho(C, \leq \gamma) \sqcup \rho(D, \leq \gamma) \\
\rho(\forall R.C, \geq \gamma) & = & \forall \rho(R, > 1 - \gamma).\rho(C, \geq \gamma) \\
\rho(\forall R.C, \leq \gamma) & = & \exists \rho(R, \geq 1 - \gamma).\rho(C, \leq \gamma) \\
\rho(\exists R.C, \geq \gamma) & = & \exists \rho(R, \geq \gamma).\rho(C, \geq \gamma) \\
\rho(\exists R.C, \leq \gamma) & = & \forall \rho(R, > \gamma).\rho(C, \leq \gamma) \\
\rho(R, \geq \gamma) & = & R_{\geq \gamma} \\
\rho(\langle a{:}C, \gamma \rangle) & = & \{a{:}\rho(C, \geq \gamma)\} \\
\rho(\langle C \sqsubseteq D, n \rangle) & = & \bigcup_{\alpha \in \bar{N}_{+}^{\mathcal{K}}, \alpha \leq n} \{\rho(C, \geq \alpha) \sqsubseteq \rho(D, \geq \alpha)\}
\end{array}
$$

# Computational Complexity

The bad news...undecidability!

## Proposition

*Assume that fuzzy GCIs are restricted to be classical, i.e. of the form $\langle \alpha, 1 \rangle$ only. Then for the following fuzzy DLs, the KB satisfiability problem is undecidable over $[0, 1]$:*

1. $\mathcal{ELC}$ *with classical axioms only under Łukasiewicz logic and product logic;*
2. $\mathcal{ELC}$ *under any non Gödelt-norm* $\otimes$*;*
3. $\mathcal{ELC}$ *with concept assertions of the form* $\langle \alpha = n \rangle$ *only under any non Gödelt-norm* $\otimes$*;*
4. $\mathcal{AL}$ *with concept implication operator* $\rightarrow$ *and concept assertions of the form* $\langle \alpha = n \rangle$ *only under any non Gödelt-norm* $\otimes$*.*
5. $\mathcal{ELC}$ *under SFL with weighted sum constructor.*

Some decidability results..

## Proposition

*The KB satisfiability problem is decidable for*

- $\mathcal{SROIQ}$ *under SFL over* $[0,1]$ *and Gödel logic over* $L_n$
- $\mathcal{SROIN}$ *under Łukasiewicz logic over* $L_n$
- $\mathcal{SHI}$ *under any continuous t-norm over* $L_n$ *without TBox*
- $\mathcal{ALC}$ *with concept implication operator* $\rightarrow$*, for any continuous t-norm over* $[0,1]$ *with acyclicTBox*
- $\mathcal{SHIF}$ *with concept implication operator* $\rightarrow$*, for Łukasiewicz logic over* $[0,1]$ *with acyclicTBox*
- $\mathcal{SI}$ *under any continuous t-norm over* $[0,1]$ *without TBox*

# 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

# Reasoners

Languages supported by fuzzy ontology reasoners:

| Reasoner | Fuzzy DL | Logic | Degrees | Other constructors | GUI |
|---|---|---|---|---|---|
| fuzzyDL | $\mathcal{SHIF}(\mathbf{D})$ | Z, Ł | General | Modifiers, rough, aggregation | • |
| Fire | $\mathcal{SHIN}$ | Z | Numbers | | • |
| FPLGERDS | $\mathcal{ALC}$ | Ł | Numbers | Role negatio/top/bottom | |
| YADLR | $\mathcal{ALCOQ}$ | Z, Ł | General | Local reflexivity | |
| DeLorean | $\mathcal{SROIQ}(\mathbf{D})$ | Z, G | General | Modifiers, rough DL | • |
| GURDL | $\mathcal{ALC}$ | General | Numbers | | No |
| FRESG | $\mathcal{ALC}(\mathbf{D})$ | Z | Numbers | Fuzzy datatype expressions | • |
| LiFR | *DLP* fragment | Z | Numbers | Weighted concepts | |
| SMT-based solver | $\mathcal{ALE}$ | Π | No | No | |
| DLMedia | *DLR–Lite* | Z, G | Numbers | Image similarity | • |
| SoftFacts | *DLR–Lite* | Z, G | Numbers | Fuzzy datataypes | • |
| ONTOSEARCH2 | $DL - Lite_R$ | General | Numbers | | • |

Reasoning services offered by fuzzy ontology reasoners

| Reasoner | CON | ENT | CSAT | SUB | IR | BDB | Other tasks | OPT |
|---|---|---|---|---|---|---|---|---|
| fuzzyDL | • | • | • | • | • | • | Defuzzification | • |
| Fire | • | • | • | • | | • | Classification | • |
| FPLGERDS | | • | | | | | | |
| YADLR | | Partial | | | • | Partial | Realisation | |
| DeLorean | • | • | • | • | | • | | • |
| GURDL | • | • | | • | | | | • |
| FRESG | • | • | • | | • | | Realisation | |
| LiFR | | Partial | • | • | | • | | |
| SMT-based solver | | | • | | | | | |
| DLMedia | | | | | | | Top-k Image Retrieval | • |
| SoftFacts | | | | | | | Top-k CQA | • |
| ONTOSEARCH2 | | | | | | | Retrieval | |

“CON”, “ENT”, “CSAT”, “SUB”, “IR”, “BED”, and “OPT” represent consistency, entailment, concept satisfiability, subsumption, instance retrieval, BED, and optimisations, respectively

The case of Logic Programs

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

# LPs Semantics: FOL semantics

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

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

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

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

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

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

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

- $M$ can be computed as the limit of

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

# 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

- We consider fuzzy LPs, which extends classical LPs, where
  - Truth space is [0, 1]
  - Interpretation is a mapping $I : B_{\mathcal{P}} \to [0, 1]$
  - 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 $\langle R(\mathbf{c}), n \rangle$
    - 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 $f\colon [0, 1]^{k+h} \rightarrow [0, 1]$, 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 $I : B_{\mathcal{P}} \to [0, 1]$.
- Model $I \models \mathcal{P}$ iff for all $r \in \mathcal{P}^*$ $I \models r$, where $I \models A \leftarrow \varphi$ iff $I(\varphi) \leq I(A)$
- 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 $\langle q(\mathbf{c}), s\rangle$, $s \in [0, 1]$

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

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

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

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

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

- Fuzzy LPs may be tricky:

$$\begin{array}{lll} \langle A, 0 \rangle & & \\ A & \leftarrow & (A+1)/2 \end{array}$$

- 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

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

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

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

  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 $B$.

$A\colon = Q$, $dg\colon = Q$, $in\colon = \emptyset$, **for all** $x \in V$ **do** $v(x) = 0$, $exp(x) = 0$
**while** $A \neq \emptyset$ **do**
  **select** $x_i \in A$, $A\colon = A \setminus \{x_i\}$, $dg\colon = dg \cup s(x_i)$
  $r\colon = f_i(v(x_{i_1}), ..., v(x_{i_{a_i}}))$
  **if** $r \succ v(x_i)$ **then** $v(x_i)\colon = r$, $A\colon = A \cup (p(x_i) \cap dg)$ **fi**
  **if not** $exp(x_i)$ **then** $exp(x_i) = 1$, $A\colon = A \cup (s(x_i) \setminus in)$, $in\colon = in \cup s(x_i)$ **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 $\phi$. With $\mathrm{p}(q)$ we denote the set of *parents* of $q$, i.e. the set $\mathrm{p}(q) = \{p_i\colon q \in \mathrm{s}(p_i, r)\}$ (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 $\delta$
- The threshold $\delta$ 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 $\delta$, because any successively computed answer will have a score below $\delta$

# 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 | |
| | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | |

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* | $\delta$ |
|---|---|
| — | 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 $\delta$

**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 $\texttt{rankedList}$ such that $\texttt{rankedList}(q)$ contains top-k answers of $q$
**Init:** $\delta = 1$, **for all** predicates $p$ in $\mathcal{P}$ **do**
  **if** $p$ intensional **then** $\texttt{rankedList}(p) = \emptyset$, $\texttt{Q}(p) := \emptyset$ **fi**
  **if** $p$ extensional **then** $\texttt{rankedList}(p) = T_p$ **fi**
 **endfor**

1. **loop**
2. **if** $\texttt{A} = \emptyset$ **then** $\texttt{A} := \{q\}$, $\texttt{dg} := \{q\}$, $\texttt{in} := \emptyset$, $\texttt{rL}' := \texttt{rankedList}$, initialise all pointers $ptr_i^r$ to 0
3. **for all** intensional predicates $p$ **do** $\texttt{exp}(p) = \texttt{false}$ **endfor**
   **fi**
4. **select** $p \in \texttt{A}$, $\texttt{A} := \texttt{A} \setminus \{p\}$, $\texttt{dg} := \texttt{dg} \cup \texttt{s}(p)$
5. $\langle \mathbf{t}, s \rangle$ := *getNextTuple*($p$)
6. **if** $\langle \mathbf{t}, s \rangle \neq \texttt{null}$ **then** $\texttt{rankedList}(p) := \texttt{rankedList}(p) \cup \{\langle \mathbf{t}, s \rangle\}$, $\texttt{A} := \texttt{A} \cup (\texttt{p}(p) \cap \texttt{dg})$ **fi**
7. **if not** $\texttt{exp}(p)$ **then** $\texttt{exp}(p) = \texttt{true}$, $\texttt{A} := \texttt{A} \cup (\texttt{s}(p) \setminus \texttt{in})$, $\texttt{in} := \texttt{in} \cup \texttt{s}(p)$ **fi**
8. Update threshold $\delta$
9. **until** ($\texttt{rankedList}(q)$ does contain $k$ top-ranked tuples with score above query rule threshold)
   **or** (($\texttt{rL}' = \texttt{rankedList}$) **and** $\texttt{A} = \emptyset$)
10. **return** top-k ranked tuples in $\texttt{rankedList}(q)$

**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** $\mathbb{Q}(p) \neq \emptyset$ **then**
    $\langle \mathbf{t}, s \rangle := getTop(\mathbb{Q}(p))$, remove $\langle \mathbf{t}, s \rangle$ from $\mathbb{Q}(p)$, **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 `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 $\langle \mathbf{t}, s' \rangle \in$ `rankedList`($p$) nor $\langle \mathbf{t}, s' \rangle \in \mathbb{Q}(p)$ with $s \preceq s'$ **then**
   insert $\langle \mathbf{t}, s \rangle$ into $\mathbb{Q}(p)$ **fi**
   **endfor**
   **endfor**
   **until** $\mathbb{Q}(p) \neq \emptyset$ or no new valid join tuple can be generated
7. **if** $\mathbb{Q}(p) \neq \emptyset$ **then** $\langle \mathbf{t}, s \rangle := getTop(\mathbb{Q}(p))$, remove $\langle \mathbf{t}, s \rangle$ from $\mathbb{Q}(p)$, **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 $\delta^p$
- with $r.\mathbf{t}_{p_i}^{\perp}$ ($r.\mathbf{t}_{p_i}^{\top}$) we denote the last tuple seen (the top ranked one) in `rankedList`$(p, r)$
- we define

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

- if $p_i$ is an extensional predicate, we define

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

- 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 $\delta$ of the *TopAnswers* algorithm is defined as to be

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

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

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

That's it !