Description Logics

Marco Piastra

Origini del Semantic Web

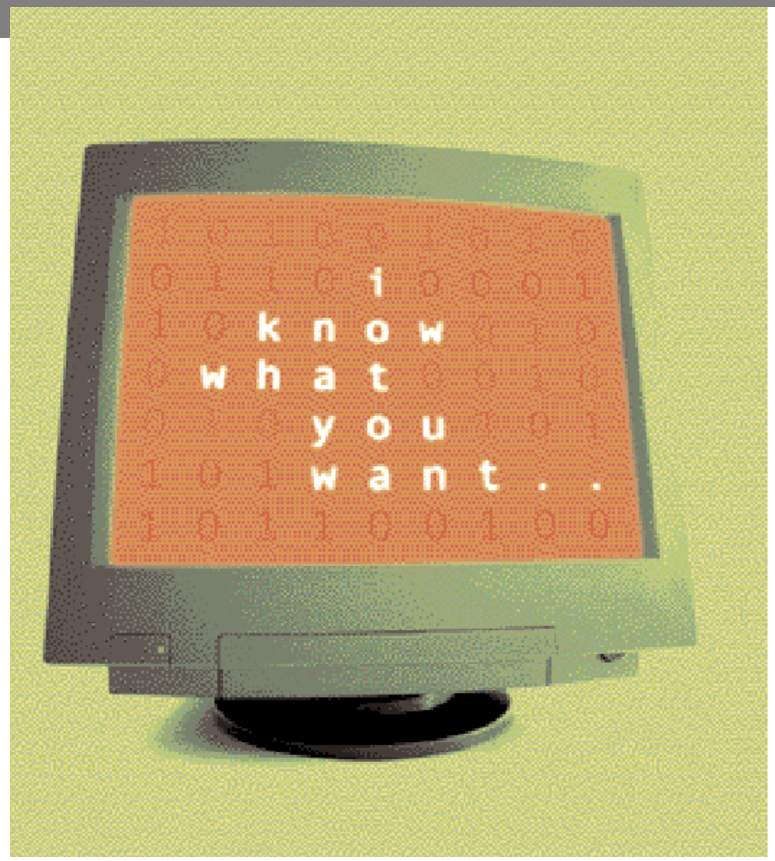
- Il Web è stato "inventato" da Tim Berners-Lee (et al.), un fisico del CERN di Ginevra
- La sua visione del Web era molto più ambiziosa di quanto poi trasformatosi in realtà: (syntactic) Web:

"… a plan for achieving a set of connected applications for data on the Web in such a way as to form a consistent logical web of data …"

"… an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation …"

- Questa visione è ora nota come **Semantic Web**

Scientific American, May 2001:



THE SEMANTIC WEB

A new form of Web content
that is meaningful to computers
will unleash a revolution of new abilities

by
TIM BERNERS-LEE,
JAMES HENDLER and
ORA LASSILA

- La realizzazione di questa visione è (probabilmente) impossibile, per ora
- Si può cominciare aggiungendo **annotazioni semantiche** (semantic annotations) alle risorse web
- Alcune applicazioni in ambito e-Science ed e-Learning già esistono

Description Logics

Una famiglia di logiche, non una logica sola

- Obiettivi
  - Definizione di ontologie
  - Ragionamento automatico
- Caratteristiche
  - Un sottoinsieme della logica del primo ordine
    - Si usano solo predicati unari (concetti) e binari (relazioni)
    - Solo forme limitate di funzioni (attributi)
    - Vincoli sull'uso dei connettivi, possibilità di introdurre vincoli numerici
- Famiglia di logiche
  - Diminuendo i vincoli, si accresce il potere espressivo (e la complessità)
- Vantaggi
  - Sono in generale decidibili
  - Trattamento automatico (abbastanza) efficace

Linguaggio

- La più semplice DL: $\mathcal{AL}$ (= Attributive Language)

  Concetti atomici: A, B, C, …

  Ruoli atomici: R, S, …

  Concetto universale: $\top$

  Concetto vuoto (*bottom concept*): $\bot$

  Intersezione: $\sqcap$

  Negazione atomica: $\neg$

  Quantificatori di ruolo: $\forall$, $\exists$

- Formule ben formate (fbf)

A	(atomic concept)
$\top$	(universal concept)
$\bot$	(bottom concept)
$\neg A$	(atomic negation)
$C \sqcap D$	(intersection)
$\forall R.C$	(value restriction)
$\exists R.\top$	(limited existential quantification).

Esempio 2

$Person \sqcap Female$

Il concetto le cui istanze sono $Person$ e $Female$ (concetti)

Tipicamente usato in una definizione:

$$Woman \equiv Person \sqcap Female$$

$Person \sqcap \neg Female$

Il concetto le cui istanze sono $Person$ e non $Female$

$Person \sqcap \exists\, hasChild.\top$

Il concetto le cui istanze sono $Person$ ed hanno almeno un figlio

Tipicamente usato in una definizione:

$$Parent \equiv Person \sqcap \exists hasChild.\top$$

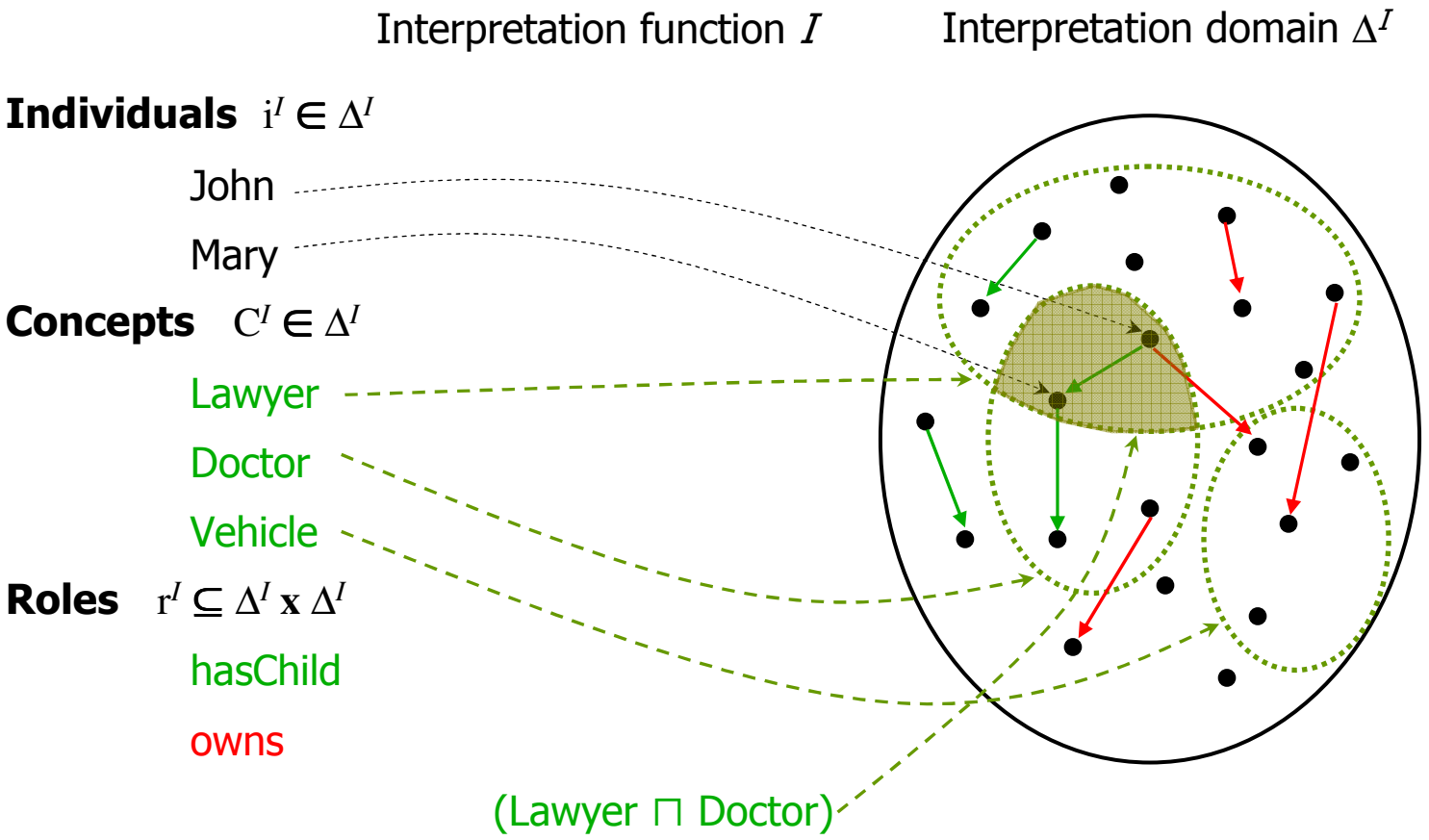
$Person \sqcap \forall\, hasChild.Female$

Il concetto le cui istanze sono $Person$ ed hanno solo figli $Female$

Famglia delle logiche $\mathcal{AL}$

- *Union* [$\mathcal{U}$]
  - Si introduce il simbolo $\sqcup$
  - Amesse fbf del tipo $A \sqcup B$
- *Full existential quantifier* [$\mathcal{E}$]
  - Ammesse fbf del tipo: $\exists R.C$
  - Esempio: $Person \sqcap \exists\ hasChild.Female$
- *Number restriction* [$\mathcal{N}$]
  - Si introducono i numeri naturali n ed i simboli $\geq$, $\leq$
  - Ammesse fbf del tipo $\geq n\ R$ e del tipo $\leq n\ R$
  - Esempio: $Person \sqcap (\geq 1\ hasChild \sqcup (\leq 3\ hasChild \sqcap \exists\ hasChild.Female))$
- *Complement* [$\mathcal{C}$]
  - Ammesse fbf del tipo $\neg C$ (C concetto qualsiasi)
  - Esempio: $Person \sqcap \neg(Female \sqcup \exists\ hasChild.Female)$
- Famiglia $\mathcal{AL}$
  - Una qualsiasi combinazione (esempio: $\mathcal{ALC}$)
  - Abbreviazioni convenzionali: $\mathcal{ALC}$ (= $\mathcal{ALUEC}$), $\mathcal{ALCN}$ (= $\mathcal{ALUENC}$)

Sematica intuitiva (da I. Horrocks)



Regole semantiche

- Interpretazione ***I*** linguaggio $\mathcal{AL}$ e sue estensioni

  in relazione ad un universo Δ^I (= $\mathbf{U}$ in $\mathbf{L_{PO}}$)

  Interpretazione base: $C^I \subseteq \Delta^I$, $R^I \subseteq \Delta^I \times \Delta^I$

  $\mathcal{AL}$

$$
\begin{aligned}
\top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} &= \emptyset \\
(\neg A)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} &= \{a \in \Delta^{\mathcal{I}} \mid \forall b.\ (a,b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\} \\
(\exists R.\top)^{\mathcal{I}} &= \{a \in \Delta^{\mathcal{I}} \mid \exists b.\ (a,b) \in R^{\mathcal{I}}\}.
\end{aligned}
$$

  $\mathcal{U}$

$$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}.$$

  $\mathcal{E}$

$$(\exists R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b.\ (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}.$$

  $\mathcal{N}$

$$(\geqslant n\, R)^{\mathcal{I}} = \left\{a \in \Delta^{\mathcal{I}} \;\middle|\; |\{b \mid (a,b) \in R^{\mathcal{I}}\}| \geq n\right\},$$

$$(\leqslant n\, R)^{\mathcal{I}} = \left\{a \in \Delta^{\mathcal{I}} \;\middle|\; |\{b \mid (a,b) \in R^{\mathcal{I}}\}| \leq n\right\},$$

  $\mathcal{C}$

Description Logics e L_{PO}

- Regole di traduzione

$$
\begin{aligned}
\pi_x(A) &= A(x), \\
\pi_x(C \sqcap D) &= \pi_x(C) \wedge \pi_x(D), \\
\pi_x(C \sqcup D) &= \pi_x(C) \vee \pi_x(D),
\end{aligned}
$$

$$
\begin{aligned}
\pi_x(\exists R.C) &= \exists y.R(x,y) \wedge \pi_y(C), \\
\pi_x(\forall R.C) &= \forall y.R(x,y) \supset \pi_y(C),
\end{aligned}
$$

$$
\begin{aligned}
\pi_x(\geqslant n\, R) &= \exists^{\geq n} y.R(x,y) = \exists y_1, \ldots, y_n . \textstyle\bigwedge_{i \neq j} y_i \neq y_j \wedge \bigwedge_i R(x, y_i) \\
\pi_x(\leqslant n\, R) &= \exists^{\leq n} y.R(x,y) = \forall y_1, \ldots, y_{n+1} . \textstyle\bigwedge_{i \neq j} y_i \neq y_j \supset \bigvee_i \neg R(x, y_i)
\end{aligned}
$$

Description Logics e L_{MP}

- Le *Description Logics* sono traducibili in logiche (multi)modali

  Si vede confrontando le regole ST di L_{MP}

- Regole di traduzione

  La funzione $f()$ definisce la traduzione

$$
\begin{aligned}
f(A) &= A, \\
f(C \sqcap D) &= f(C) \wedge f(D), \\
f(C \sqcup D) &= f(C) \vee f(D), \\
f(\neg(C)) &= \neg(f(C)), \\
\\
f(\forall R_i.C) &= \Box_i(f(C)), \\
f(\exists R_i.C) &= \Diamond_i(f(C)).
\end{aligned}
$$

Architettura generale

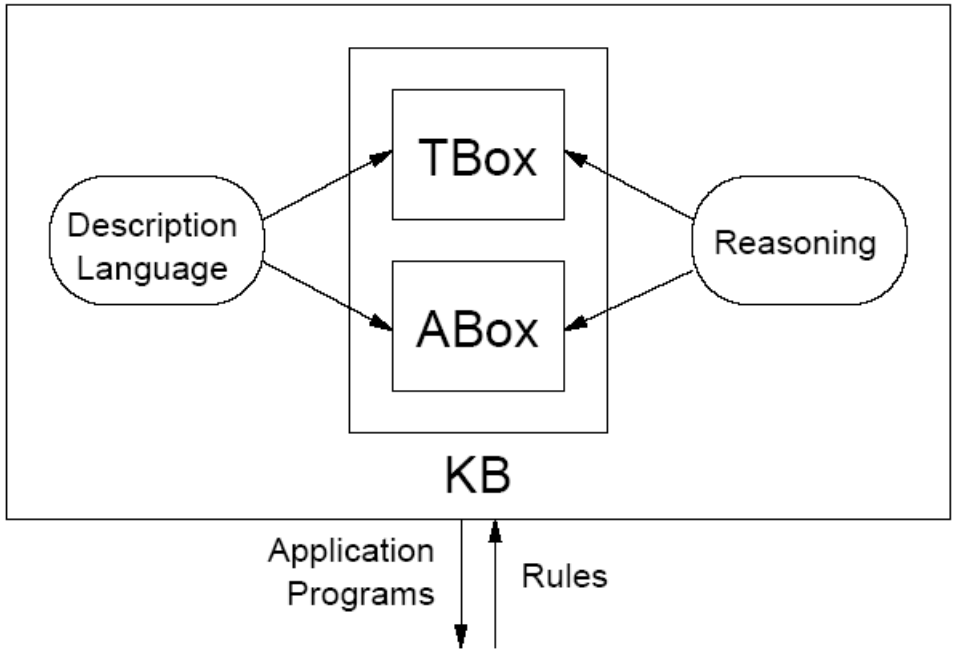


Fig. 2.1. Architecture of a knowledge representation system based on Description Logics.

- TBox e ABox sono insiemi di assiomi (di una teoria=ontologia)

  TBox (anche T) Terminology: descrizioni concettuali

  ABox (anche A) Assertion: descrizioni riguardanti istanze concrete

Definizioni

- Una TBox contiene **definizioni**

  Assiomi di una teoria espressi in un linguaggio della famiglia $\mathcal{AL}$

  Si introduce il simbolo $\equiv$

  Una definizione è una fbf del tipo $A \equiv \varphi$ (φ fbf qualsiasi)

Woman	$\equiv$	Person $\sqcap$ Female
Man	$\equiv$	Person $\sqcap \neg$Woman
Mother	$\equiv$	Woman $\sqcap \exists$hasChild.Person
Father	$\equiv$	Man $\sqcap \exists$hasChild.Person
Parent	$\equiv$	Father $\sqcup$ Mother
Grandmother	$\equiv$	Mother $\sqcap \exists$hasChild.Parent
MotherWithManyChildren	$\equiv$	Mother $\sqcap \geqslant 3$ hasChild
MotherWithoutDaughter	$\equiv$	Mother $\sqcap \forall$hasChild.$\neg$Woman
Wife	$\equiv$	Woman $\sqcap \exists$hasHusband.Man

Fig. 2.2. A terminology (TBox) with concepts about family relationships.

Terminologie

- **Nomi simbolici** (*name symbols*)

  In una definizione $A \equiv \varphi$ (φ fbf qualsiasi)
  il simbolo A viene detto **nome simbolico** (di un concetto composito)

- **Terminologia** T (o TBox)

  Un'insieme di definizioni in cui a ciascun nome simbolico
  corrisponde una sola definizione

- **Simboli base** (*base symbols*)

  In una terminologia, l'insieme dei ruoli e dei concetti che occorrono
  solo nelle parti destre delle definizioni

- Terminologia **definitoria** (*definitory*)

  Una T la cui interpretazione **estesa** (= completa) può essere costruita univocamente a partire dall'interpretazione dei simboli base

  Non sono definitorie alcune T *cicliche*

  $$Human \equiv Animal \sqcap \forall\, hasParent.Human$$

  Ma alcune T cicliche sono definitorie ...

  Qualsiasi T definitoria può essere tradotta in una T aciclica

Espansioni

- **Espansione** di una TBox

  Sostituzione esaustiva dei nomi simbolici nelle parti destre

$$
\begin{array}{rcl}
\text{Woman} & \equiv & \text{Person} \sqcap \text{Female} \\
\text{Man} & \equiv & \text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female}) \\
\text{Mother} & \equiv & (\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person} \\
\text{Father} & \equiv & (\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female})) \sqcap \exists \text{hasChild}.\text{Person} \\
\text{Parent} & \equiv & ((\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female})) \sqcap \exists \text{hasChild}.\text{Person}) \\
 & & \sqcup\, ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \\
\text{Grandmother} & \equiv & ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \\
 & & \sqcap \exists \text{hasChild}.(((\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female})) \\
 & & \qquad \sqcap \exists \text{hasChild}.\text{Person}) \\
 & & \qquad \sqcup\, ((\text{Person} \sqcap \text{Female}) \\
 & & \qquad\quad \sqcap \exists \text{hasChild}.\text{Person})) \\
\text{MotherWithManyChildren} & \equiv & ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \sqcap \geqslant 3\, \text{hasChild} \\
\text{MotherWithoutDaughter} & \equiv & ((\text{Person} \sqcap \text{Female}) \sqcap \exists \text{hasChild}.\text{Person}) \\
 & & \sqcap \forall \text{hasChild}.(\neg(\text{Person} \sqcap \text{Female})) \\
\text{Wife} & \equiv & (\text{Person} \sqcap \text{Female}) \\
 & & \sqcap \exists \text{hasHusband}.(\text{Person} \sqcap \neg(\text{Person} \sqcap \text{Female}))
\end{array}
$$

Fig. 2.3. The expansion of the Family TBox in Figure 2.2.

Terminologie generalizzate

- Uso di assiomi di **inclusione**

  Al posto di forme come

  $$Woman \equiv Person \sqcap Female$$

  è possibile usare una forma di **specializzazione**

  $$Woman \sqsubseteq Person$$

  Il concetto $Woman$ è incluso in (= è sottoclasse) $Person$

  La relazione classe/sottoclasse (p.es. UML) si traduce con $\sqsubseteq$

- **Normalizzazione**

  Una terminologia T **generalizzata** (= che contiene assiomi di inclusione) può essere tradotta in una forma $\underline{\mathrm{T}}$ **normalizzata**

  sostituendo le forme $Woman \sqsubseteq Person$ con

  $$\underline{Woman} \equiv Woman \sqcap Person$$

  dove $\underline{Woman}$ è un nuovo simbolo base (che denota qualità specifiche)

  In termini di espressività, le due forme sono equivalenti

  Ogni modello di $\underline{\mathrm{T}}$ è anche modello di T

  Qualsiasi interpretazione base di un modello di T è anche interpretazione base di un modello di $\underline{\mathrm{T}}$

Asserzioni

- Una ABox (o A) contiene solo **asserzioni**

  Fatti specifici in termini di istanze, concetti e ruoli e/o loro proprietà

  Si introducono costanti individuali: a, b, c, …

  Esempi: C(a), R(b,c)

MotherWithoutDaughter(MARY)
Father(PETER)
hasChild(MARY, PETER)
hasChild(PETER, HARRY)
hasChild(MARY, PAUL)

Fig. 2.4. A world description (ABox).

Ragionamento: TBox

- Quattro tipologie di problemi

  In relazione ad una terminologia

- ***Satisfiability*** A concept C is satisfiable with respect to $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}}$ is nonempty. In this case $\mathcal{I}$ is a model of C.
- ***Subsumption*** A concept C is subsumed by a concept D with respect to $\mathcal{T}$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$. Subsumption is denoted by $\mathcal{T} \models C \sqsubseteq D$.
- ***Equivalence*** Two concepts C and D are equivalent with respect to $\mathcal{T}$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$. Subsumption is denoted by $\mathcal{T} \models C = D$.
- ***Disjointness*** Two concepts C and D are disjoint with respect to $\mathcal{T}$ if $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$ for every model $\mathcal{I}$ of $\mathcal{T}$.

Ragionamento: ABox

- Quattro tipologie di problemi

  In relazione ad un insieme di asserzioni

- ***Consistency*** An ABox $\mathcal{A}$ is consistent with respect to a TBox $\mathcal{T}$, if there is an interpretation that is a model of both $\mathcal{T}$ and $\mathcal{A}$.
- ***Instance Checking*** An individual is an instance of a concept C with respect to an ABox $\mathcal{A}$, if it is a member of the concept set of C. Instance checking is denoted by $\mathcal{A} \models C(a)$.
- ***Retrieval***: retrieve all individuals that are instances of a given concept description
- ***Realization problem***: find the most specific concept an individual belongs to

Riducibilità dei problemi

- In generale, nella famiglia $\mathcal{AL}$

Proposition 2.12 (Reduction to Subsumption) *For concepts C, D we have*

(i) *C is unsatisfiable $\Leftrightarrow$ C is subsumed by $\bot$;*
(ii) *C and D are equivalent $\Leftrightarrow$ C is subsumed by D and D is subsumed by C;*
(iii) *C and D are disjoint $\Leftrightarrow$ $C \sqcap D$ is subsumed by $\bot$.*

The statements also hold with respect to a TBox.

Proposition 2.13 (Reduction to Unsatisfiability) *For concepts C, D we have*

(i) *C is subsumed by D $\Leftrightarrow$ $C \sqcap \neg D$ is unsatisfiable;*
(ii) *C and D are equivalent $\Leftrightarrow$ both $(C \sqcap \neg D)$ and $(\neg C \sqcap D)$ are unsatisfiable;*
(iii) *C and D are disjoint $\Leftrightarrow$ $C \sqcap D$ is unsatisfiable.*

The statements also hold with respect to a TBox.

Metodi a tableau

- Le logiche della famiglia $\mathcal{AL}$ (viste qui) sono decidibili
  - Il che significa che il problema della soddisfacibilità è decidibile
  - Inoltre:
    - Le logiche della famiglia $\mathcal{AL}$ hanno la proprietà dei modelli finiti
      - La soddisfacibilità implica l'esistenza di un modello finito
  - Nel caso peggiore (*number restriction*) la complessità è esponenziale

- Metodi a tableau
  - (sono completi rispetto alla famiglia $\mathcal{AL}$)
  - Hanno come obiettivo la dimostrazione di insoddisfacibilità
    - Quindi sono applicabili a tutte le quattro tipologie di problema, previa eventuale traduzione del problema originario

Tableaux per le Description Logics

- Caratteristiche generali

  A partire da una terminologia $\mathbf{T}$ (una TBox) ed un problema $\mathbf{P}$

  L'algoritmo cerca di costruire una $\mathbf{A}$ (una ABox) che contiene una contraddizione

  Condizioni:

  La $\mathbf{T}$ deve essere espansa

  In $\mathbf{T}$ e $\mathbf{P}$ tutti i nomi simbolici sono stati sostituiti dalle parti destre in $\mathbf{T}$ (eliminazione dei nomi simbolici)

  Le fbf in $\mathbf{T}$ e $\mathbf{P}$ devono essere trasformate in modo da contenere solo negazioni atomiche

  Ciascun nodo del tableau contiene una $\mathbf{A}$

  Il nodo radice (la $\mathbf{A}$ iniziale) contiene $\mathbf{P}$ (il problema, cioè un concetto) applicato ad un individuo x_o

  $\mathrm{A} = \{\varphi_0(x_0)\}$, dove φ_0 è la fbf che descrive il problema

Regole per le Description Logics

- Ci si limita al caso $\mathcal{ALC}$ (= $\mathcal{ALUEC}$)

  (vale a dire senza *number restriction*)

The $\rightarrow_{\sqcap}$-rule
Condition: $\mathcal{A}$ contains $(C_1 \sqcap C_2)(x)$, but it does not contain both $C_1(x)$ and $C_2(x)$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{C_1(x), C_2(x)\}$.

The $\rightarrow_{\sqcup}$-rule
Condition: $\mathcal{A}$ contains $(C_1 \sqcup C_2)(x)$, but neither $C_1(x)$ nor $C_2(x)$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{C_1(x)\}$, $\mathcal{A}'' = \mathcal{A} \cup \{C_2(x)\}$.

The $\rightarrow_{\exists}$-rule
Condition: $\mathcal{A}$ contains $(\exists R.C)(x)$, but there is no individual name z such that $C(z)$ and $R(x, z)$ are in $\mathcal{A}$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{C(y), R(x, y)\}$ where y is an individual name not occurring in $\mathcal{A}$.

The $\rightarrow_{\forall}$-rule
Condition: $\mathcal{A}$ contains $(\forall R.C)(x)$ and $R(x, y)$, but it does not contain $C(y)$.
Action: $\mathcal{A}' = \mathcal{A} \cup \{C(y)\}$.

Terminazione

- La terminazione è garantita (decidibilità)

  Quando A contiene un conflitto (*clash*) in tutti i suoi rami (successo)

(i) $\{\bot(x)\} \subseteq \mathcal{A}$ for some individual name x;
(ii) $\{A(x), \neg A(x)\} \subseteq \mathcal{A}$ for some individual name x and some concept name A;

  Quando almeno un ramo di A è completo (= non si possono più applicare regole) e non ci sono conflitti (ciò implica fallimento)

Esempio 3

- Dimostrazione di *subsumption*

Given an unfoldable terminology $\mathcal{T}$ containing the definition of vegan from Example 2.1:

$$\mathsf{vegan} \doteq \mathsf{person} \sqcap \forall \mathit{eats}.\mathsf{plant}$$

and the following definition of vegetarian:

$$\mathsf{vegetarian} \doteq \mathsf{person} \sqcap \forall \mathit{eats}.(\mathsf{plant} \sqcup \mathsf{dairy})$$

the algorithm can be used to show that $\mathsf{vegan} \sqsubseteq_{\mathcal{T}} \mathsf{vegetarian}$ by demonstrating that $(\mathsf{vegan} \sqcap \neg\mathsf{vegetarian})_{u\mathcal{T}}$ (i.e., $\mathsf{vegan} \sqcap \neg\mathsf{vegetarian}$ unfolded w.r.t. $\mathcal{T}$) is not satisfiable:

1. Unfold and normalise $\mathsf{vegan} \sqcap \neg\mathsf{vegetarian}$ to give:
$$\mathsf{person} \sqcap \forall \mathit{eats}.\mathsf{plant} \sqcap (\neg\mathsf{person} \sqcup \exists \mathit{eats}.(\neg\mathsf{plant} \sqcap \neg\mathsf{dairy}))$$
2. Initialise $\mathbf{T}$ to contain a single node x labelled:
$$\mathcal{L}(x) = \{\mathsf{person} \sqcap \forall \mathit{eats}.\mathsf{plant} \sqcap (\neg\mathsf{person} \sqcup \exists \mathit{eats}.(\neg\mathsf{plant} \sqcap \neg\mathsf{dairy}))\}$$

Esempio 3

3. Apply the $\sqcap$-rule to $\mathsf{person} \sqcap \forall \mathit{eats}.\mathsf{plant} \sqcap (\neg\mathsf{person} \sqcup \exists \mathit{eats}.(\neg\mathsf{plant} \sqcap \neg\mathsf{dairy})) \in \mathcal{L}(x)$:

$$\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{\mathsf{person}, \forall \mathit{eats}.\mathsf{plant}, \neg\mathsf{person} \sqcup \exists \mathit{eats}.(\neg\mathsf{plant} \sqcap \neg\mathsf{dairy})\}$$

4. Apply the $\sqcup$-rule to $\neg\mathsf{person} \sqcup \exists \mathit{eats}.(\neg\mathsf{plant} \sqcap \neg\mathsf{dairy}) \in \mathcal{L}(x)$:

   (a) Save $\mathbf{T}$ and try:

   $$\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{\neg\mathsf{person}\}$$

   This is an obvious contradiction as $\{\mathsf{person}, \neg\mathsf{person}\} \subseteq \mathcal{L}(x)$.

   (b) Restore $\mathbf{T}$ and try:

   $$\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{\exists \mathit{eats}.(\neg\mathsf{plant} \sqcap \neg\mathsf{dairy})\}$$

Esempio 3

5. Apply the $\exists$-rule to $\exists \mathit{eats}.(\neg\mathsf{plant} \sqcap \neg\mathsf{dairy}) \in \mathcal{L}(x)$:

$$\begin{aligned} &\text{create a new node } y \text{ and a new edge } \langle x, y\rangle \\ &\mathcal{L}(y) = \{\neg\mathsf{plant} \sqcap \neg\mathsf{dairy}\} \\ &\mathcal{L}(\langle x, y\rangle) = \mathit{eats} \end{aligned}$$

6. Apply the $\forall$-rule to $\forall \mathit{eats}.\mathsf{plant} \in \mathcal{L}(x)$ and $\mathcal{L}(\langle x, y\rangle) = \mathit{eats}$:

$$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \cup \{\mathsf{plant}\}$$

7. Apply the $\sqcap$-rule to $\neg\mathsf{plant} \sqcap \neg\mathsf{dairy} \in \mathcal{L}(y)$:

$$\mathcal{L}(y) \longrightarrow \mathcal{L}(y) \quad \cup \quad \{\neg\mathsf{plant}, \neg\mathsf{dairy}\}$$

This is an obvious contradiction as $\{\mathsf{plant}, \neg\mathsf{plant}\} \subseteq \mathcal{L}(y)$.

As all possible applications of the $\sqcup$-rule (step 4) have now been shown to lead to a contradiction, it can be concluded that, with respect to the definitions in $\mathcal{T}$, $\mathsf{vegan} \sqcap \neg\mathsf{vegetarian}$ is unsatisfiable and thus that $\mathsf{vegetarian}$ subsumes vegan.

3

OWL

La logica $\mathcal{SHIQ}$

Famiglia [$\mathcal{S}$]

$\mathcal{ALC}$ con aggiunta dei ruoli transitivi

$R \in \mathrm{Trn} \Rightarrow (\langle a,b\rangle \in R^I, \langle b,c\rangle \in R^I \Rightarrow \langle a,c\rangle \in R^I)$

Role hierarchy [$\mathcal{H}$]

Ammesse fbf del tipo: $R_2 \sqsubseteq R_1$ (Esempio: $hasDaughter \sqsubseteq hasChild$)

Inverse Roles [$\mathcal{I}$]

Ammesse fbf del tipo: $R_1 \equiv R_2^-$ (Esempio: $hasChild \equiv isChildOf^-$)

Nominals/singleton [$\mathcal{O}$]

(Esempio: $\{Italy\}$)

Qualified Number Restriction [$\mathcal{Q}$]

Ammesse fbf del tipo $\geq n\ R.C$ e $\leq n\ R.C$ (Esempio: $\geq 3\ hasChild.Female$)

Functional Restriction [$\mathcal{F}$] (una forma ristretta di $\mathcal{N}$)

Ammesse fbf del tipo $\leq 1\ R$ e $\geq 2\ R$ (Esempio: $\leq 1\ hasTutor$)

- $\mathcal{SHIQ}$ e OWL

  OWL Lite: $\mathcal{SHIF}$ – Decidibile e di complessità limitata

  OWL DL: $\mathcal{SHOIQ}$ – Decidibile ma di complessità elevata

  OWL Full: forma particolare (equivale a RDF) - Non decidibile

Formato delle ontologie OWL (XML)

- Esempio

$$\mathtt{Student} = \mathtt{Person} \sqcap \geqslant 1\,\mathtt{enrolledIn}$$

```
<owl:Class rdf:ID="Student">
   <owl:intersectionOf rdf:parsetype="Collection">
     <owl:Class rdfs:about="Person" />
     <owl:Restriction>
        <owl:onProperty rdf:resource="enrolledIn" />
        <owl:minCardinality rdfs:datatype="&xsd;Integer">
               1
        </owl:minCardinality>
     </owl:Restriction>
   <owl:intersectionOf>
</owl:Class>
```

- Riferimenti

  http://www.w3.org/TR/owl-features/

Linguaggio OWL

Abstract Syntax	DL Syntax	Semantics
Descriptions (C)		
A (URI reference)	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
`owl:Thing`	$\top$	$\texttt{owl:Thing}^{\mathcal{I}} = \Delta^{\mathcal{I}}$
`owl:Nothing`	$\bot$	$\texttt{owl:Nothing}^{\mathcal{I}} = \{\}$
`intersectionOf`(C_1 C_2 ...)	$C_1 \sqcap C_2$	$(C_1 \sqcap D_1)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap D_2^{\mathcal{I}}$
`unionOf`(C_1 C_2 ...)	$C_1 \sqcup C_2$	$(C_1 \sqcup C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
`complementOf`(C)	$\neg C$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
`oneOf`(o_1 ...)	$\{o_1, \ldots\}$	$\{o_1, \ldots\}^{\mathcal{I}} = \{o_1^{\mathcal{I}}, \ldots\}$
`restriction`(R `someValuesFrom`(C))	$\exists R.C$	$(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$
`restriction`(R `allValuesFrom`(C))	$\forall R.C$	$(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
`restriction`(R `hasValue`(o))	$R : o$	$(\forall R.o)^{\mathcal{I}} = \{x \mid \langle x, o^{\mathcal{I}}\rangle \in R^{\mathcal{I}}\}$
`restriction`(R `minCardinality`(n))	$\geqslant n\, R$	$(\geqslant n\, R)^{\mathcal{I}} = \{x \mid \sharp(\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}) \geqslant n\}$
`restriction`(R `minCardinality`(n))	$\leqslant n\, R$	$(\geqslant n\, R)^{\mathcal{I}} = \{x \mid \sharp(\{y.\langle x, y\rangle \in R^{\mathcal{I}}\}) \leqslant n\}$