Ragionamento nelle logiche descrittive

M. Simi, 2012-2013

LA KB delle logiche descrittive

- $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
- $\mathcal{T}$ *(T-BOX)*, componente terminologica
- $\mathcal{A}$ (A-BOX), componente asserzionale
- Una interpretazione $\mathcal{I}$ soddisfa $\mathcal{A}$ e $\mathcal{T}$ (quindi $\mathcal{K}$) sse soddisfa ogni asserzione in $\mathcal{A}$ e ogni definizione in T ($\mathcal{I}$ è un *modello* di $\mathcal{K}$).

Che tipo di ragionamenti?

- Progetto e gestione di ontologie
  - Controllo di consistenza dei concetti e supporto alla creazione di gerarchie
- Integrazione di ontologie
  - Relazioni tra concetti di ontologie diverse
  - Consistenza di gerarchie integrate
- Interrogazioni
  - Determinare fatti consistenti rispetto alle ontologie
  - Determinare se individui sono istanze di concetti
  - Recuperare individui che soddisfano una query
  - Verificare se un concetto è più generale di un'altro

Problemi decisionali per DL

- Problemi decisionali tipici
  - Soddisfacibilità di concetti
  - Sussunzione
- Problemi decisionali classici
  - Soddisfacilità di una KB
  - Conseguenza logica di una KB
- Altri servizi inferenziali

Soddisfacilità di concetti (CS)

- Soddisfacibilità di un concetto [CS(C)]: esiste un'interpretazione diversa dall'insieme vuoto?
- Un concetto C è *soddisfacibile* rispetto a $\mathcal{T}$ se esiste un modello $\mathcal{I}$ di $\mathcal{T}$ tale che $C^{\mathcal{I}}$ è *non vuoto*.
- Esempi

  (father), concetto primitivo, è soddisfacibile;

  (father $\sqcap$ ¬father) è insoddisfacibile

Sussunzione

- Sussunzione

  $\mathcal{K} \models C \sqsubseteq D$ (*D sussume* C)

  se per ogni modello $\mathcal{I}$ di $\mathcal{T}$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

  Es. person *sussume* (person $\sqcap$ ∃hasChild.T)
- Sussunzione strutturale e ibrida

  Ibrida se si usano anche le definizioni nella KB

  Es. Se student $\sqsubseteq$ person ∈ T-BOX

  allora person $\sqcap$ ∃hasChild.T *sussume* student $\sqcap$ ∃hasChild.T

Concetti equivalenti e disgiunti

- Equivalenza: $\mathcal{K} \models C \equiv D$
  Due concetti C e *D* sono *equivalenti* rispetto a una terminologia $\mathcal{T}$ se $C^{\mathcal{I}} = D^{\mathcal{I}}$ *per ogni modello* $\mathcal{I}$ di $\mathcal{T}$.
- Concetti disgiunti: Due concetti C e *D* sono disgiunti rispetto a $\mathcal{T}$ se $C^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$ per ogni modello $\mathcal{I}$ di $\mathcal{T}$.

Problemi decisionali classici

- Soddisfacibilità di una KB (KBS)
  Esiste un modello per $\mathcal{K} = (\mathcal{T}, \mathcal{A})$?
- Conseguenza logica di una KB:
  $\mathcal{K} \models a$:C il problema di decidere, se l'asserzione a:C è conseguenza logica di $\mathcal{K}$
  detto anche "controllo di istanza" o Instance Checking (IC)

Altre inferenze per DL

- Recupero: trovare tutti gli individui che sono istanze di C. Calcola l' insieme $\{a \mid \mathcal{K} \models a{:}C\}$
- Most Specific Concept (MSC)
  Dato un insieme di individui, trovare il concetto più specifico di cui sono istanza. Serve per la classificazione.
- Least Common Subsumer (LCS)
  Dato un insieme di concetti, trovare il concetto più specifico che li sussume tutti. Serve per la classificazione.

Riduzione tra problemi decisionali

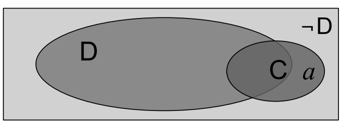
- I problemi decisionali non sono indipendenti
  - la sussunzione ibrida e strutturale coincidono se la T-BOX è vuota
  - la sussunzione strutturale può essere ricondotta alla soddisfacibilità di concetti
    $C \sqsubseteq D$ sse $C \sqcap \neg D$ è insoddisfacibile
  - C è insoddisfacibile sse C è sussunto da $\bot$
  - C e D sono disgiunti sse $C \sqcap D$ è insoddisfacibile

Riconducibilità a KBS

- Tutti i problemi possono essere ricondotti a KBS, la soddisfacibilità di una KB.

1. Consistenza di concetto
   C è soddisfacibile *sse* $\mathcal{K} \cup \{a{:}C\}$ è soddisfacibile
   con a un nuovo individuo. Nota: $\{a{:}C\}$ viene aggiunto ad $\mathcal{A}$.
2. Sussunzione
   $\mathcal{K} \models C \sqsubseteq D$ (*D sussume* C) *sse* $\mathcal{K} \cup \{a{:}\ C \sqcap \neg D\}$
   è insoddisfacibile, con a un nuovo individuo

D non sussume C



Riconducibilità a KBS (cont.)

3. Equivalenza
   $\mathcal{K} \models C \equiv D$ *sse* $\mathcal{K} \models C \sqsubseteq D$ e $\mathcal{K} \models D \sqsubseteq C$
4. Controllo di istanza
   $\mathcal{K} \models a{:}C$ *sse* $\mathcal{K} \cup \{a{:}\neg C\}$ è insoddisfacibile
5. Recupero riconducibile a controllo di istanza a sua volta riconducibile a KBS

Esempi di riduzione di problemi

1. *I ricchi sono felici?*
   - Felice *sussume* Ricco? $\mathcal{K} \models$ Ricco $\sqsubseteq$ Felice
   - $\mathcal{K} \cup \{a$: Ricco $\sqcap \neg$Felice$\}$ è insoddisfacibile?
2. *Essere ricco e sano basta per essere felice?*
   - $\mathcal{K} \models$ Ricco $\sqcap$ Sano $\sqsubseteq$ Felice
   - $\mathcal{K} \cup \{a$: Ricco $\sqcap$ Sano $\sqcap \neg$Felice$\}$ è insoddisfacibile?

Esempi di riduzione di problemi

Sapendo che:

Per essere felici bisogna essere ricchi e sani
(e non basta)

- T-BOX: Felice $\sqsubseteq$ Ricco $\sqcap$ Sano

Una persona ricca può essere infelice?

- (Ricco $\sqcap \neg$Felice) è soddisfacibile?
- $\mathcal{K} \cup \{a$: Ricco $\sqcap \neg$Felice $\}$ è soddisfacibile?

Sistemi deduttivi per DL

- Algoritmi per determinare la sussunzione strutturale
  - Per linguaggi poco espressivi (senza negazione)
- La tecnica più diffusa è una *tecnica per la soddisfacibilità* di una KB.
  - tecnica di propagazione|espansione di vincoli
  - una variante di un metodo di deduzione naturale, i *tableaux semantici*

Tecnica di propagazione di vincoli

- *L'idea di base*: ogni formula nella KB è un vincolo sulle interpretazioni affinché siano modelli di KB
- I vincoli complessi si scindono in vincoli più elementari mediante *regole di propagazione* fino ad arrivare, in un numero finito di passi, a *vincoli atomici,* non ulteriormente decomponibili
- Se l'insieme di vincoli atomici contiene una contraddizione evidente (detta *clash*) allora la KB non è soddisfacibile, altrimenti abbiamo trovato un modello.

Vantaggi della tecnica

- è semplice
- è costruttiva
- è modulare: abbiamo una regola per ogni costrutto
- è utile per progettare algoritmi di decisione e per valutarne la complessità

- Vediamo la tecnica in dettaglio per $\mathcal{ALC}$

Richiamo di $\mathcal{ALC}$

A	(concetto primitivo)
| ⊤	(top, concetto universale)
| ⊥	(bottom)
| ¬C	(negazione)
| C ⊓ *D*	(intersezione)
| C ⊔ *D*	(unione)
| ∀*R*.*C*	(restrizione di valore)
| ∃*R*.*C*	(esistenziale)

A, B concetti primitivi *R ruolo primitivo*

C, D concetti

Passi preliminari per KBS in $\mathcal{ALC}$

1. Espansione delle definizioni: passo preliminare che consiste nel ricondursi ad una $\mathcal{K} = (\{\ \}, \mathcal{A})$ con solo la parte di asserzioni. Le asserzioni sono i vincoli iniziali
2. Normalizzazione: portare le asserzioni in *forma normale negativa*

- A questo punto possiamo applicare le regole di propagazione di vincoli

Normalizzazione

- Un insieme di vincoli si dice in forma *normale negativa se* ogni occorrenza dell'operatore $\neg$ è davanti a un concetto primitivo.
- *Regole di normalizzazione*:

$$
\begin{aligned}
\neg\top &\mapsto \bot \\
\neg\bot &\mapsto \top \\
\neg\neg C &\mapsto C \\
\neg(C_1 \sqcap C_2) &\mapsto \neg C_1 \sqcup \neg C_2 \\
\neg(C_1 \sqcup C_2) &\mapsto \neg C_1 \sqcap \neg C_2 \\
\neg(\exists R.C) &\mapsto \forall R.\neg C \\
\neg(\forall R.C) &\mapsto \exists R.\neg C
\end{aligned}
$$

Clash per $\mathcal{ALC}$

- Un *clash per* $\mathcal{ALC}$ è un insieme di vincoli di uno dei seguenti tipi:
  - $\{a : C,\ a : \neg C\}$
  - $\{a : \bot\}$

Propagazione di vincoli per DL

- Un *vincolo* è una asserzione della forma a:C o (b, c):R, dove a, b e c sono costanti (individui distinti) o variabili (x, y … individui non necessariamente distinti).
- Un insieme di vincoli $\mathcal{A}$ è *soddisfacibile* sse esiste una interpretazione che soddisfa ogni vincolo in $\mathcal{A}$.

Alberi di completamento

- *Foresta di completamento*: struttura dati che serve per l' esecuzione dell'algoritmo
- Per ogni asserzione x:C in $\mathcal{A}$ si inizializza un albero

  x $\quad$ $\mathcal{L}(x)=\{C\}$ *label di x*
- Ad ogni passo si espande un nodo dell' albero o si creano nuovi nodi con le seguenti regole.

Regole per $\mathcal{ALC}$

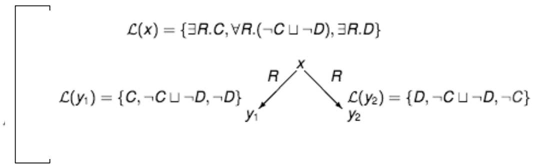
Rule		Description
($\sqcap$)	if 1.	$C_1 \sqcap C_2 \in \mathcal{L}(x)$ and
	2.	$\{C_1, C_2\} \not\subseteq \mathcal{L}(x)$
	then	$\mathcal{L}(x) \to \mathcal{L}(x) \cup \{C_1, C_2\}$
($\sqcup$)	if 1.	$C_1 \sqcup C_2 \in \mathcal{L}(x)$ and
	2.	$\{C_1, C_2\} \cap \mathcal{L}(x) = \emptyset$ *nessuno dei due sta in* $\mathcal{L}(x)$
	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	$\mathcal{L}(y) \to \mathcal{L}(y) \cup \{C\}$

Non determinismo

- Le regole per la congiunzione, per $\mathcal{ALC}$ sono *deterministiche*
- La regola per la disgiunzione, è non deterministica: la sua applicazione risulta in insiemi di vincoli alternativi
- $\mathcal{A}$ è soddisfacibile sse almeno uno degli insiemi di vincoli ottenuti lo è.
- $\mathcal{A}$ è insoddisfacibile sse tutte le alternative si concludono con una contraddizione evidente (clash)

Esempio 1

- $\mathcal{A}=\{x{:}\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D\}$ soddisfacibile?

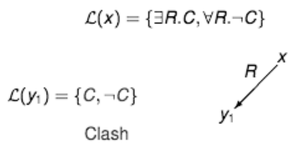


- $\mathcal{A}=\{x{:}\exists R.C \sqcap \forall R.(\neg C \sqcup \neg D) \sqcap \exists R.D\}$ è soddisfacibile
- Modello trovato: $\Delta^\mathcal{I} = \{x, y_1, y_2\}$

$C^\mathcal{I} = \{y_1\}$ $D^\mathcal{I} = \{y_2\}$ $R^\mathcal{I} = \{(x, y_1),(x, y_2)\}$

Esempio 2

- $\mathcal{A}=\{x{:}\exists R.C \sqcap \forall R.\neg C\}$ soddisfacibile?



- $\mathcal{A}=\{x{:}\exists R.C \sqcap \forall R.\neg C\}$ non è soddisfacibile
- Non ha modelli

Esempio 3

Tutti i figli di John sono femmine. Mary è una figlia di John. Tim è un amico del professor Blake. Dimostra che Mary è femmina.

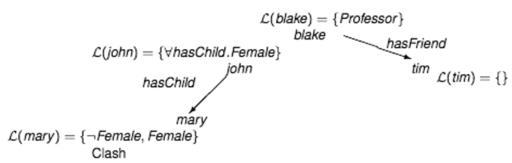
$\mathcal{A}$ = {john:∀hasChild.Female, (john, mary):hasChild, (blake, tim):hasFriend, blake:Professor}

Dimostrare:

$\mathcal{A} \models$ mary: Female ovvero che

$\mathcal{A} \cup$ mary:¬Female insoddisfacile

Esempio 3



Correttezza e completezza di KBS

1. Il risultato è dimostrabilmente invariante rispetto all'ordine di applicazione delle regole.
2. Correttezza: se l'algoritmo termina con un sistema di vincoli completo e senza *clash*, allora $\mathcal{A}$ è soddisfacibile e dai vincoli si può ricavare un modello
3. Completezza: se una base di conoscenza $\mathcal{A}$ è soddisfacibile, allora l'algoritmo termina producendo almeno un modello finito senza *clash*.
4. KBS è decidibile per $\mathcal{ALC}$ e anche per $\mathcal{ALCN}$.

Altri costrutti

$\mathcal{H}$: assiomi di inclusione tra ruoli

R $\sqsubseteq$ S sse $R^I \subseteq S^I$

$\mathcal{Q}$: restrizioni numeriche qualificate

$(\geq n\ R.C)^I = \{a \in \Delta^I \mid |\{ b \mid (a, b) \in R^I \wedge b \in C^I \}| \geq n\}$

$(\leq n\ R.C)^I = \{a \in \Delta^I \mid |\{ b \mid (a, b) \in R^I \wedge b \in C^I \}| \leq n\}$

$\mathcal{O}$: nominali (singoletti); $\{a\}^I = \{a^I\}$

$\mathcal{I}$: ruolo inverso, $(R^-)^I = \{(a, b) \mid (b, a) \in R^I\}$

$\mathcal{F}$: ruolo funzionale

fun(*F*) sse $\forall x,y,z\ (x,y) \in F^I \wedge (x,z) \in F^I \Rightarrow y=z$

$\mathcal{R}_+$: ruolo transitivo

$(R_+)^I = \{(a, b) \mid \exists c$ tale che $(a, c) \in R^I \wedge (c, b) \in R^I\}$

$\mathcal{S}$: $\mathcal{ALC} + \mathcal{R}_+$

OWL-DL

OWL-DL equivalente a $\mathcal{SHOIN}$ =

$\mathcal{S}$: $\mathcal{ALC}$ + *ruoli transitivi* $\mathcal{R}_+$

$\mathcal{H}$: *specializzazione di ruoli*

$\mathcal{O}$: *nominali o singoletti*

$\mathcal{I}$: *ruoli inversi*

$\mathcal{N}$: *restrizioni numeriche*

OWL-Lite

OWL-Lite equivalente a $\mathcal{SHIF}$ =

$\mathcal{S}$: $\mathcal{ALC}$ + *ruoli transitivi* $\mathcal{R}_+$

$\mathcal{H}$: *specializzazione di ruoli*

$\mathcal{I}$: *ruoli inversi*

$\mathcal{F}$: *ruoli funzionali*

Costruttori di OWL

Costruttore	Sintassi DL	Esempio
A (URI)	A	Conference
thing	$\top$	
nothing	$\bot$	
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Reference $\sqcap$ Journal
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Organization $\sqcup$ Institution
complementOf	$\neg C$	$\neg$MasterThesis
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{WISE, ISWC, …}
allValuesFrom	$\forall P.C$	$\forall$date.Date
someValuesFrom	$\exists P.C$	$\exists$date.{2005}
maxCardinality	$\leqslant nP$	($\leq$ 1 location)
minCardinality	$\geqslant nP$	($\geq$ 1 publisher)

Assiomi OWL

Axiom	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Man $\equiv$ Human $\sqcap$ Male
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$Female
sameIndividualAs	$\{x_1\} \equiv \{x_2\}$	{President_Bush} $\equiv$ {G_W_Bush}
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	{john} $\sqsubseteq \neg${peter}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
inverseOf	$P_1 \equiv P_2^-$	hasChild $\equiv$ hasParent$^-$
transitiveProperty	$P^+ \sqsubseteq P$	ancestor$^+ \sqsubseteq$ ancestor
functionalProperty	$\top \sqsubseteq \leqslant 1P$	$\top \sqsubseteq \leqslant 1$hasMother
inverseFunctionalProperty	$\top \sqsubseteq \leqslant 1P^-$	$\top \sqsubseteq \leqslant 1$hasSSN$^-$

Un esempio: sintassi XML

E.g., Person $\sqcap$ $\forall$hasChild.Doctor $\sqcup$ $\exists$hasChild.Doctor:

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:toClass>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:hasClass rdf:resource="#Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:toClass>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Complessità e decidibilità per DL
(con terminologie acicliche)

Trattabile	Decidibile	Indecidibile
$\mathcal{ALN}$	$\mathcal{ALCNR}$	FOL
$\mathcal{AL}$	$\mathcal{ALC}$	KL-One
	$\mathcal{ALR}$	NIKL
$\mathcal{ALNO}$	$\mathcal{ALE}$	
	PROP OWL-Lite OWL-DL	

P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ EXPTIME $\subseteq$ NEXPTIME

soglia della trattabilità *soglia della decidibilità*

Considerazioni: DL trattabili

- Logiche con sussunzione decidibili in tempo polinomiale
  - $\mathcal{AL}$: intersezione di concetti, negazione limitata, esistenziale semplice, restrizione universale di ruolo
  - $\mathcal{ALN}$: $\mathcal{AL}$ + restrizioni numeriche
  - $\mathcal{ALNO}$: $\mathcal{ALN}$ + concetti individuali
- PROP è NP-completo, ma $\mathcal{ALNO}$ e PROP non sono confrontabili dal punto di vista espressivo.

DL espressive e decidibili

- $\mathcal{ALC}$ è PSPACE con espansione incrementale di $\mathcal{T}$
  - $\mathcal{ALC}$ = $\mathcal{ALUE}$ (unione è fonte di complessità)
- $\mathcal{ALCNR}$ = $\mathcal{ALC}$ + restrizioni numeriche e congiunzione di ruoli è decidibile in PSPACE
- $\mathcal{ALC}$ è EXPTIME nel caso di $\mathcal{T}$ ciclica.
- $\mathcal{SHIF}$ = OWL-Lite, $\mathcal{AL}$ + specializzazione di ruoli, ruoli transitivi, inversi e funzionali è decidibile in EXPTIME
- $\mathcal{SHOIN}$ = OWL-DL, $\mathcal{AL}$ + specializzazione di ruoli, ruoli transitivi e inversi, singoletti e restrizioni numeriche è decidibile in EXPTIME

Conclusioni

- Gli studi di complessità sulle DL hanno messo in luce un ampio spettro di possibilità rispetto al *trade-off* tra espressività e complessità
- Hanno consentito di progettare sistemi espressivi ed efficienti (anche se di complessità esponenziale nel caso peggiore).
- Il web semantico ha solidi fondamenti teorici.